



(51) International Patent Classification:

H04L 1/00 (2006.01) *H03M 13/11* (2006.01)

(21) International Application Number:

PCT/US2020/060483

(22) International Filing Date:

13 November 2020 (13.11.2020)

(25) Filing Language:

English

(26) Publication Language:

English

(30) Priority Data:

16/685,641 15 November 2019 (15.11.2019) US

(71) Applicant: **NVIDIA CORPORATION** [US/US]; 2788 San Tomas Expressway, Santa Clara, California 95051 (US).

(72) Inventors: **HUANG, Yanxiang**; 2788 San Tomas Expressway, Santa Clara, California 95051 (US). **DELFELD, James**; 2788 San Tomas Expressway, Santa Clara, California 95051 (US).

(74) Agent: **ADAMS, Scott S.** et al.; c/o IP Docketing Dept., Davis Wright Tremaine LLP, 920 Fifth Avenue, Suite 3300, Seattle, Washington 98104-1610 (US).

(81) Designated States (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, IT, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW.

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

(54) Title: SCHEDULING METHOD FOR LDPC DECODING

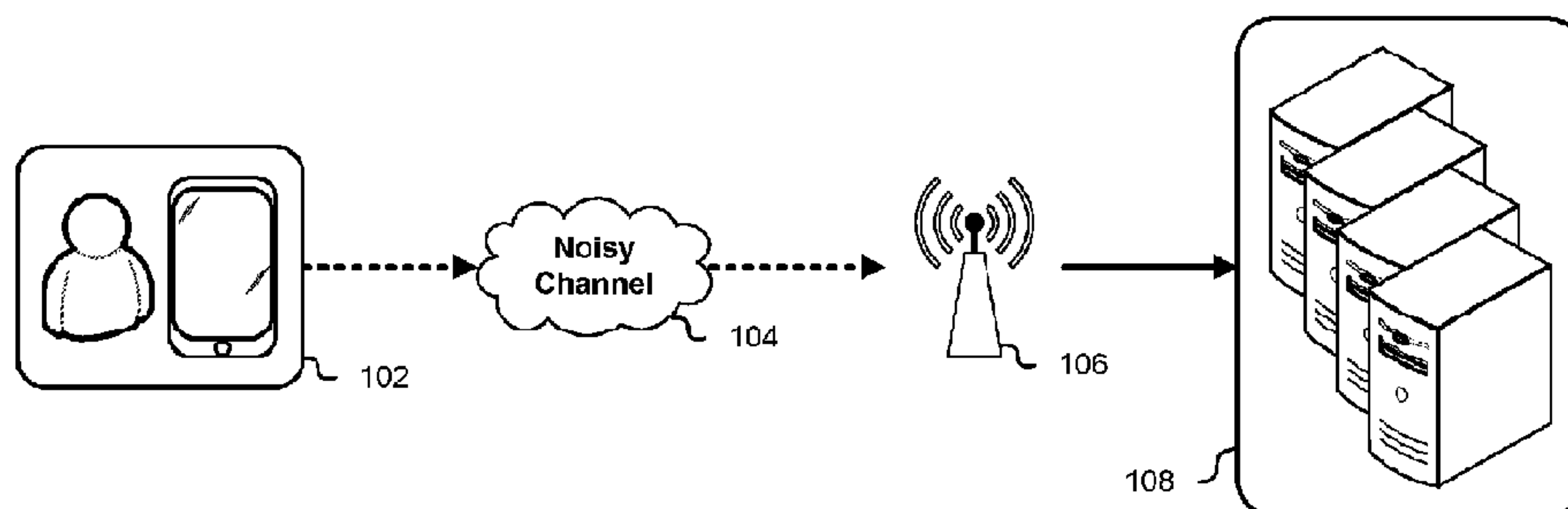


FIG. 1A

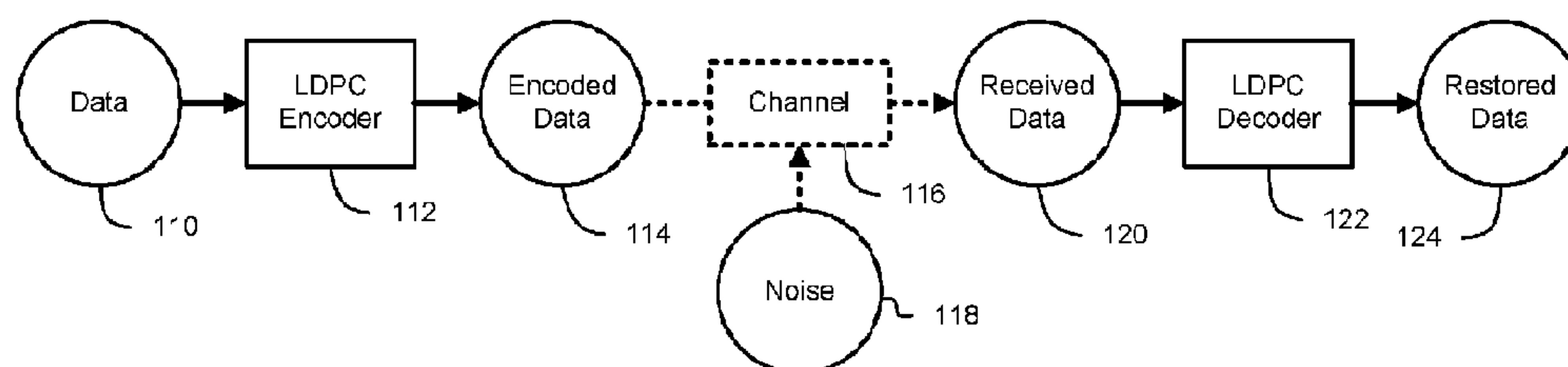


FIG. 1B

(57) Abstract: Apparatuses, systems, and techniques to correct errors in information received from a plurality of fifth-generation new radio antennas. In at least one embodiment, codeword information is decoded by a plurality of variable (or vector) nodes and check nodes, where codewords and nodes are divided among processors to be performed in parallel.

TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

- *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*
- *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii))*

Published:

- *with international search report (Art. 21(3))*

5

SCHEDULING METHOD FOR LDPC DECODING

CROSS REFERENCE TO RELATED APPLICATION

[0001] This application claims priority to U.S. Patent Application No. 16/685,641, filed November 15, 2019, entitled “SCHEDULING METHOD FOR LDPC DECODING,” the entire contents of which is incorporated herein by reference in its entirety and for all
10 purposes.

TECHINICAL FIELD

[0002] At least one embodiment pertains to processing resources used to perform and facilitate wireless and wired communications. For example, at least one embodiment pertains to processors or computing systems used to decode low density parity check codes according
15 to various novel techniques described herein.

BACKGROUND

[0003] As more and more users migrate to using cellular devices as their primary means of communication, mobile data traffic is growing at an exponential rate. To handle this rapid growth, new communication standards have been developed that introduce technology layers
20 that facilitate greater wireless network access and throughput of information. Because of mass throughput and data capacity requirements in new wireless standards, hardware capabilities of network infrastructure must be able to handle challenging levels of information processing. An increase in users and devices in a communication network also introduces an increasing amount of disruptive noise in communication channels. To address accurate
25 communication of information in an increasingly noisy channel, various encoding schemes, such as low density parity check (LDPC) codes, have been integrated as a component of new standards such as Fifth Generation New Radio (5G-NR). Decoding of information that has been encoded as an LDPC code is a complex and time consuming element of 5G-NR processing. Throughput and latency requirements in 5G-NR are very strict, and decoding
30 large LDPC codewords may not meet those strict throughput and latency requirements. Two scheduling approaches commonly implemented in LDPC decoding may not always meet

- 5 these strict throughput and latency requirements. Consequently, a new approach is required for large LDPC codewords.

BRIEF DESCRIPTION OF THE DRAWINGS

- [0004] FIG. 1A illustrates an example of wireless communication over a noisy channel, according to at least one embodiment;
- 10 [0005] FIG. 1B illustrates an example architecture for encoded data in a noisy channel, according to at least one embodiment;
- [0006] FIG. 2 illustrates an example encoding and decoding scheme for communications in a noisy channel, according to at least one embodiment;
- [0007] FIG. 3 illustrates an example of low density parity check (LDPC) decoding,
15 according to at least one embodiment;
- [0008] FIG. 4 illustrates an example general architecture for LDPC decoding, according to at least one embodiment;
- [0009] FIG. 5 illustrates a flooding decoding schedule for LDPC decoding, according to at least one embodiment;
- 20 [0010] FIG. 6 illustrates a layered decoding schedule for LDPC decoding, according to at least one embodiment;
- [0011] FIG. 7A illustrates parallelizable operations in LDPC decoding, according to at least one embodiment;
- [0012] FIG. 7B illustrates non-parallelizable operations in LDPC decoding, according to
25 at least one embodiment;
- [0013] FIG. 8 illustrates a parallel-layered schedule for LDPC decoding, according to at least one embodiment;
- [0014] FIG. 9 illustrates a process for performing parallel-layered LDPC decoding, according to at least one embodiment;

5 [0015] FIG. 10 illustrates an example data center system, according to at least one embodiment;

[0016] FIG. 11A illustrates an example of an autonomous vehicle, according to at least one embodiment;

10 [0017] FIG. 11B illustrates an example of camera locations and fields of view for the autonomous vehicle of FIG. 11A, according to at least one embodiment;

[0018] FIG. 11C is a block diagram illustrating an example system architecture for the autonomous vehicle of FIG. 11A, according to at least one embodiment;

15 [0019] FIG. 11D is a diagram illustrating a system for communication between cloud-based server(s) and the autonomous vehicle of FIG. 11A, according to at least one embodiment;

[0020] FIG. 12 is a block diagram illustrating a computer system, according to at least one embodiment;

[0021] FIG. 13 is a block diagram illustrating computer system, according to at least one embodiment;

20 [0022] FIG. 14 illustrates a computer system, according to at least one embodiment;

[0023] FIG. 15 illustrates a computer system, according at least one embodiment;

[0024] FIG. 16A illustrates a computer system, according to at least one embodiment;

[0025] FIG. 16B illustrates a computer system, according to at least one embodiment;

[0026] FIG. 16C illustrates a computer system, according to at least one embodiment;

25 [0027] FIG. 16D illustrates a computer system, according to at least one embodiment;

[0028] FIGS. 16E and 16F illustrate a shared programming model, according to at least one embodiment;

[0029] FIG. 17 illustrates exemplary integrated circuits and associated graphics processors, according to at least one embodiment;

- 5 [0030] FIGS. 18A and 18B illustrate exemplary integrated circuits and associated graphics processors, according to at least one embodiment;
- [0031] FIGS. 19A and 19B illustrate additional exemplary graphics processor logic according to at least one embodiment;
- [0032] FIG. 20 illustrates a computer system, according to at least one embodiment;
- 10 [0033] FIG. 21A illustrates a parallel processor, according to at least one embodiment;
- [0034] FIG. 21B illustrates a partition unit, according to at least one embodiment;
- [0035] FIG. 21C illustrates a processing cluster, according to at least one embodiment;
- [0036] FIG. 21D illustrates a graphics multiprocessor, according to at least one embodiment;
- 15 [0037] FIG. 22 illustrates a multi-graphics processing unit (GPU) system, according to at least one embodiment;
- [0038] FIG. 23 illustrates a graphics processor, according to at least one embodiment;
- [0039] FIG. 24 is a block diagram illustrating a processor micro-architecture for a processor, according to at least one embodiment;
- 20 [0040] FIG. 25 illustrates at least portions of a graphics processor, according to one or more embodiments;
- [0041] FIG. 26 illustrates at least portions of a graphics processor, according to one or more embodiments;
- [0042] FIG. 27 illustrates at least portions of a graphics processor, according to one or
25 more embodiments;
- [0043] FIG. 28 is a block diagram of a graphics processing engine of a graphics processor in accordance with at least one embodiment;
- [0044] FIG. 29 is a block diagram of at least portions of a graphics processor core, according to at least one embodiment;

5 [0045] FIGS. 30A and 30B illustrate thread execution logic including an array of processing elements of a graphics processor core according to at least one embodiment;

[0046] FIG. 31 illustrates a parallel processing unit (“PPU”), according to at least one embodiment;

[0047] FIG. 32 illustrates a general processing cluster (“GPC”), according to at least one
10 embodiment;

[0048] FIG. 33 illustrates a memory partition unit of a parallel processing unit (“PPU”), according to at least one embodiment;

[0049] FIG. 34 illustrates a streaming multi-processor, according to at least one embodiment;

15 [0050] FIG. 35 illustrates a network for communicating data within a 5G wireless communications network, according to at least one embodiment;

[0051] FIG. 36 illustrates a network architecture for a 5G LTE wireless network, according to at least one embodiment;

[0052] FIG. 37 is a diagram illustrating some basic functionality of a mobile
20 telecommunications network/system operating in accordance with LTE and 5G principles, according to at least one embodiment;

[0053] FIG. 38 illustrates a radio access network which may be part of a 5G network architecture, according to at least one embodiment;

[0054] FIG. 39 provides an example illustration of a 5G mobile communications system
25 in which a plurality of different types of devices is used, according to at least one embodiment;

[0055] FIG. 40 illustrates an example high level system, according to at least one embodiment;

[0056] FIG. 41 illustrates an architecture of a system of a network, according to at least
30 one embodiment;

5 [0057] FIG. 42 illustrates example components of a device, according to at least one embodiment;

[0058] FIG. 43 illustrates example interfaces of baseband circuitry, according to at least one embodiment;

10 [0059] FIG. 44 illustrates an example of an uplink channel, according to at least one embodiment;

[0060] FIG. 45 illustrates an architecture of a system of a network, according to at least one embodiment;

[0061] FIG. 46 illustrates a control plane protocol stack, according to at least one embodiment;

15 [0062] FIG. 47 illustrates a user plane protocol stack, according to at least one embodiment;

[0063] FIG. 48 illustrates components of a core network, according to at least one embodiment; and

20 [0064] FIG. 49 illustrates components of a system to support network function virtualization (NFV), according to at least one embodiment.

DETAILED DESCRIPTION

[0065] FIG. 1A illustrates an example of wireless communication over a noisy channel. In at least one embodiment, a wireless device 102 is connected to at least one antenna 106 on a wireless network containing at least one baseband unit (BBU) 108. In at least one embodiment, a user device 102 may be wireless devices such as cell phones, tablets, smart watches, appliances, or any other connected device that utilizes wireless communication, as described herein.

30 [0066] In at least one embodiment, a BBU 108 may contain one or more computing devices to perform signal processing and codeword decoding to estimate user data from one or more user devices 102. In at least one embodiment, a BBU 108 may be responsible for interpreting baseband frequencies in communication systems, including computer

5 networks, internet, phone networks and radio broadcasting systems. In at least one embodiment, a BBU 108 may switch communications in network data traffic between multiple locations. In at least one embodiment, a BBU 108 may perform decoding for data encoded as low density parity check (LDPC) codewords. In at least one embodiment, a BBU 108 may utilize one or more parallel processing units (PPUs) as described below to
10 performing decoding for data encoded as LDPC codewords. In at least one embodiment, a BBU 108 may switch communications in network data traffic between multiple locations. In at least one embodiment, a BBU 108 may connect to a base station containing an antenna 106, which may also be referred to as a remote radio unit (RRU) or a remote radio head (RRH). In at least one embodiment, an antenna 106 may be used to facilitate wireless
15 communication to one or more wireless devices 102.

[0067] In at least one embodiment, an antenna 106 may be a wire antenna, such as a short dipole antenna, dipole antenna, loop antenna, or monopole antenna. In at least one embodiment, an antenna 106 may be a log periodic antenna, such as a bow tie antenna, log periodic antenna, or log periodic dipole array. In at least one embodiment, an antenna 106
20 may be an aperture antenna, such as a slot antenna or a horn antenna. In at least one embodiment, an antenna 106 may be a microstrip antenna, such as a rectangular microstrip patch antenna or a quarter-wave patch antenna. In at least one embodiment, an antenna 106 may be a reflector antenna, such as a flat-plate reflector antenna, corner reflector antenna, or parabolic reflector antenna. In at least one embodiment, an antenna 106 may be a lens
25 antenna. In at least one embodiment, an antenna 106 may be a travelling wave antenna, such as a long wire antenna, yagi-uda antenna, helical wire antenna, or spiral antenna. In at least one embodiment, an antenna 106 may be an array antenna, such as a two-element array antenna, a linear array antenna, or a phased array antenna. In at least one embodiment, an antenna 106 may be any type of antenna that facilitates long- or short-range wireless
30 communications.

[0068] In at least one embodiment, a wireless device 102 sends data across a noisy channel 104 to an antenna 106, as described herein. In at least one embodiment, a noisy channel 104 may add interference to data values sent between a wireless device 102 and one or more antennas 106. In at least one embodiment, a wireless device 102 will encode data
35 using methods described herein such that it can be decoded at a BBU 108 or other computing device in order to retrieve data lost in a noisy channel 104.

5 [0069] FIG. 1B illustrates an example architecture for encoded data in a noisy channel. In at least one embodiment, data 110 may include any sequence of data encoded in a machine-readable format, as described herein, to be transmitted across an information channel 116 such as a wireless communication network, wired communication network, or other communication medium described herein. In at least one embodiment, data may be
 10 encoded using a low density parity check (LDPC) encoder 112. In at least one embodiment, data 110 may be represented as input bits. In at least one embodiment, data 110 may be fed into an LDPC encoder 112, described herein, in order to generate encoded data 114. In at least one embodiment, encoded data 114 may include input data 110 and parity bits in order to make up code symbols, and blocks of encoded data 114 may be referred to as codewords.
 15 In at least one embodiment, each digit or bit contained in a codeword in encoded data 114 corresponds to a variable in an equation within an LDPC scheme, as described herein.

[0070] In at least one embodiment, encoded data 114 is transmitted across a communication channel 116, which may be wireless or wired. In at least one embodiment, a communication channel 116 may introduce noise 118. In at least one embodiment, noise 118
 20 may include interfering signals and other data or information that may introduce errors into encoded data 114 transmitted across a channel 116. In at least one embodiment, original data 110 may be recovered from received data 120 after transmission through a channel 116 containing noise 118 if data 110 has been encoded 114 using an encoder 112, such as those described herein.

25 [0071] In at least one embodiment, received data 120 may contain errors introduced by noise 118 in a communication channel 116. In at least one embodiment, an LDPC decoder is used to recover original data 110 from received data 120 using techniques described below. In at least one embodiment, restored data 124 may represent original data 110 after received data 120 has been decoded using an LDPC decoder 122.

30 [0072] FIG. 2 illustrates an example encoding and decoding scheme for communications in a noisy channel. In at least one embodiment, an encoder 204 and decoder 228 implement a coding scheme where redundant information is added to input data 202 before transmission over a noisy channel 208 in order to increase robustness of a data communication against interference. In at least one embodiment, a decoder 228 may be able to recover an original
 35 input data value 202 based on a configuration of received data 210, 212, 214, 216, 218, 220, 222, 224.

5 [0073] In at least one embodiment, an error correcting code such as a low density parity check code (LDPC) is accomplished by adding redundant information to transmitted information 206. In at least one embodiment, input data or information 202 may or may not appear in encoded data 206.

[0074] An example error correcting code, in an embodiment, is a (3,1) repetition code,
10 where a data bit 202 is transmitted three times 206. In at least one embodiment, an input data bit 202, for example, contains a value of 0. In at least one embodiment, an input data bit 202, in a (3,1) repetition code, is encoded by an encoder 204 such that each input data bit 202 is repeated three times in an encoded codeword 206. In at least one embodiment, an encoded codeword 206 is transmitted across a noisy channel 208. In at least one embodiment, a noisy
15 channel may introduce uncertainty to a codeword 206, and it may then contain several different values 210, 212, 214, 216, 218, 220, 222, 224, which can be used by a decoder 228 to obtain an original data bit 226. In at least one embodiment, decoding errors may lead to an incorrect recovered data bit 230.

[0075] In at least one embodiment, a decoder 228 may implement a simple majority rule
20 to recover an original data bit 202 from received data 210, 212, 214, 216, 218, 220, 222, 224. In at least one embodiment, if a received codeword, in a (3,1) repetition code, contains two 0 values 210, 212, 214, 216, then its value may be decoded as 0 226, which is a correct encoding. In at least one embodiment, each non-zero value in received data 210, 212, 214, 216, 218, 220, 222, 224 after transmission through a noisy channel 208 is an error introduced
25 through noise in said channel 208. In at least one embodiment, if a received codeword, in a (3,1) repetition code, contains two 1 values 218, 220, 222, 224, then its original value will be decoded incorrectly as 1 230. In at least one embodiment, a (3,1) repetition encoding scheme, as illustrated in FIG. 2, demonstrates a simple error correcting code that allows for limited detection and correction of errors in data 206 transmitted across a noisy channel 208.

30 [0076] FIG. 3 illustrates an example of low density parity check (LDPC) decoding, in general. In at least one embodiment, belief propagation may be used to determine recovered data bits 322 in a codeword given a set of incomplete data bits 302 such as those received from a noisy channel, as described herein. In at least one embodiment, belief propagation is facilitated by use of variable nodes 306 and check nodes 308. In at least one embodiment,
35 variable nodes 306 may store beliefs whether a certain bit in a codeword 302, 310, 322 is 0 or 1. In at least one embodiment, check nodes 308 update belief values contained in variable

5 nodes 306 and may calculate probabilities or utilize other techniques in order to determine belief values for connected variable nodes 306. In at least one embodiment, connections between variable nodes 306 and check nodes 308 are determined by a parity check matrix.

[0077] In at least one embodiment, FIG. 3 illustrates implementation of an LDPC coding scheme with parity check matrix H defined as follows:

$$10 \quad \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 \end{pmatrix}$$

[0078] In at least one embodiment, LDPC codes each require a parity check matrix H , such as that shown above. In at least one embodiment, a matrix H for an LDPC code may be randomly generated. In at least one embodiment, an (n,k) LDPC code such as that decoded in FIG. 3 will have n variable nodes 306 and k check nodes 308. In at least one embodiment, 15 variable nodes 306 will be updated several times, in successive iterations, by check nodes 308 until decoding success or a certain number of iterations have taken place.

[0079] In at least one embodiment, an input codeword 302 that has been transmitted across a noisy channel contains two missing bit values. In at least one embodiment, variable nodes 306 contain data values for each bit in an input codeword 302. In at least one 20 embodiment, check nodes 308 are connected to each variable in a set of variable nodes 306 as described in an associated parity check matrix given above.

[0080] In at least one embodiment, a variable node may be determined if its corresponding check node is connected to no more than one unknown variable node. In at least one embodiment, a first bit and a fourth bit of a received codeword 302 are unknown. 25 In at least one embodiment, step A 304 represents an initial decoding state, including all connections between variable nodes 306 and check nodes 308. At step B 312, a first bit cannot be determined, as its associated check node or nodes are connected to at least one other unknown variable node. In at least one embodiment, decoding continues with the next unknown bit in the fourth position of a partially decoded codeword 310. In at least one 30 embodiment, a variable node containing a fourth bit 316 of an input codeword 310 is connected to a check node 320. In at least one embodiment, a check node 320 with only one unknown connection 316 may determine a value for an unknown variable node 316 based on values of known connected variable nodes 314, 318.

- 5 [0081] In at least one embodiment, each bit in a codeword 302, 310, 322 represents a variable in an equation, and each check node 308 represents an equation that satisfies an equality to zero. In at least one embodiment, each matrix in an LDPC code contains coefficients specifying whether a variable is included in an equation corresponding to each check node 308 where each variable sums to zero in a binary or other number field such as a
- 10 Galois Field like $GF(2^k)$. In at least one embodiment, a matrix H , as described above, corresponds to the following equations:

$$a_1 \oplus a_2 \oplus a_3 \oplus a_4 = 0$$

$$a_3 \oplus a_4 \oplus a_6 = 0$$

$$a_1 \oplus a_4 \oplus a_5 = 0$$

- 15 [0082] In at least one embodiment, an active check node 320 at step B 312 corresponds to an inequality defined as $a_3 \oplus a_4 \oplus a_6 = 0$, which is row 2 in a parity check matrix H :

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 \end{pmatrix}$$

- [0083] In at least one embodiment, values associated with variable nodes a_3 314 and a_6 318 are known and equal to 1, so determining a_4 316 requires solving an inequality $1 \oplus$
- 20 $a_4 \oplus 1 = 0$, and a_4 316 must be 0. In at least one embodiment, step B 312 resolves a fourth bit of an input codeword 310 represented by a variable node 316 using at least one check node 320.

- [0084] In at least one embodiment, step C 324 resolves a first bit of an input codeword 322 once enough information has been determined in previous steps in order to finish
- 25 decoding. In at least one embodiment, enough information has been determined when a check node 334 has no more than one unresolved variable node 326 connected. In at least one embodiment, step C 324 will resolve a first bit of an input codeword 322 according to an equation corresponding to a check node 334 as follows:

$$a_1 \oplus a_2 \oplus a_3 \oplus a_4 = 0$$

- 30 [0085] In at least one embodiment, because previous steps or iterations 304, 312 have determined that a_2 328 is 0, a_3 330 is 1, and a_4 332 is 0, check node 334 will resolve a_2 to be 1, and a codeword 322 is decoded.

5 [0086] FIG. 4 illustrates an example general architecture for low density parity check (LDPC) decoding. In at least one embodiment, an architecture such as that shown in FIG. 4 implements a randomly generated sparse parity check matrix such as:

$$\begin{pmatrix} 0 & 1 & \cdots & 1 & 0 \\ 1 & 0 & \cdots & 1 & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & 0 & \cdots & 1 & 1 \\ 1 & 1 & \cdots & 0 & 1 \end{pmatrix}$$

[0087] In at least one embodiment, this parity check matrix will be used to encode data
 10 402 prior to transmission. In at least one embodiment, a general architecture for low density parity check decoding will receive data 402 that is decoded using both variable, or vector, nodes 406 as well as check nodes 404. In at least one embodiment, variable (or vector) nodes 406 store bitwise values, or other information such as a probability that a vector node (or bit) is a given value. In at least one embodiment, check nodes 404 resolve an equation
 15 corresponding to a row in a parity check matrix, as described above. For example, in an embodiment, check node 404 equations according to an above parity check matrix for this figure are as follows:

$$v_2 \oplus \dots \oplus v_{n-1} = 0$$

$$v_1 \oplus \dots \oplus v_{n-1} = 0$$

20 $\dots$

$$\dots v_{n-1} \oplus v_n = 0$$

$$v_1 \oplus v_2 \oplus \dots \oplus v_n = 0$$

[0088] In at least one embodiment, check nodes 404 may either directly resolve an equation or compute a belief that a variable node contains a given value. In at least one
 25 embodiment, this belief can be expressed as $p(v_n | d_n)$, where v_n is a value of a variable node 406 and d_n is an n^{th} received data 402 value. In at least one embodiment, $p(v_n | d_n)$ corresponds to a probability of a variable node 406 value v_n given a received data 402 value d_n .

[0089] In at least one embodiment, a vector node 406 in a general LDPC decoding
 30 architecture, such as FIG. 4, will first pass its value along its vertices to all connected check nodes 404, as defined by rows of a parity check matrix as shown above. In at least one

5 embodiment, this step may be called “vector-to-check” or “V2C.” In at least one embodiment, a check node 404 computes a value or belief that a variable (or vector) node 406 contains a certain value. In at least one embodiment, values or beliefs calculated by each check node 404 are then passed along vertices between each check node 404 and a vector node 406 for each bitwise position in a codeword. In at least one embodiment, each bit of a
 10 codeword is stored in corresponding vector nodes 406. In at least one embodiment, this step may be called “check-to-vector” or “C2V.”

[0090] In at least one embodiment, variable 406 and check 404 nodes process input data 402 according to a decoding schedule, described herein. In at least one embodiment, a decoding schedule determines an order in which check nodes 404 process and resolve
 15 unknown values contained in variable (or vector) nodes 406. In at least one embodiment, after vector node 406 values converge or a predetermined number of iterations have based, decoded data 408 is complete.

[0091] FIG. 5 illustrates a flooding decoding schedule for low density parity check (LDPC) decoding of data 502 transmitted across a noisy channel, as described herein. In at
 20 least one embodiment, data is received 502 as a codeword of bits, where each bit may have been distorted or corrupted by noise in a transmission channel from a user. In at least one embodiment, each codeword, or sequence of bits, is encoded prior to transmission on a noisy channel using an LDPC scheme, as described herein.

[0092] In at least one embodiment, data is encoded using a matrix H , described above,
 25 which may be randomly generated, but must be known by both an encoder and decoder used in an LDPC scheme. In at least one embodiment, elements of a low density parity check matrix H correspond to vertices between variable (or vector) nodes 506, 512 and check nodes 508, 514 in an LDPC decoding scheme, further described herein. In at least one embodiment, a “1” value at location $[i,j]$ in an $n \times m$ parity check matrix H indicates that an i^{th} check node
 30 508, 514 is connected to a j^{th} variable (or vector) node 506, 512.

[0093] In at least one embodiment, a sparse parity check matrix H corresponding to relationships between check nodes 508, 514 and variable (or vector) nodes 506, 512 as illustrated in FIG. 5 may be defined as:

5

$$\begin{pmatrix} 0 & 1 & \dots & 1 & 0 \\ 1 & 0 & & 1 & 0 \\ \vdots & & \ddots & \vdots & \\ 0 & 0 & \dots & 1 & 1 \\ 1 & 1 & & 0 & 1 \end{pmatrix}$$

[0094] In at least one embodiment, such a sparse parity check matrix H demonstrates the following relationships between check nodes 508 and variable (or vector) nodes 506, 512, where each check node 508, 514 corresponds to one of each inequality equation defined as:

10

$$v_2 \oplus \dots \oplus v_{n-1} = 0$$

$$v_1 \oplus \dots \oplus v_{n-1} = 0$$

...

$$\dots v_{n-1} \oplus v_n = 0$$

$$v_1 \oplus v_2 \oplus \dots \oplus v_n = 0$$

15

[0095] In at least one embodiment, each check node 508, 514 will resolve any unknown variable (or vector) node 506, 512 values along its connected vertices using belief propagation. In at least one embodiment, each check node 508, 514 will determine a probability that a single variable (or vector) node 506, 512 is a certain value given a knowledge of other known, connected variable (or vector) nodes 506, 512, where connections are determined based on a low density parity check matrix H , as described above. In at least one embodiment, performing multiple decoding iterations will lead to a better approximation of a variable (or vector) node 506, 512 value probability.

20

25

[0096] In at least one embodiment, LDPC decoding consists of two steps 504, 510, which may be performed over multiple iterations until decoded results converge on a value or a maximum number of iterations has been reached. In at least one embodiment, a “variable-to-check” or “V2C” step 504 copies (or passes) current values from variable (or vector) nodes 506, 512 to check nodes 508, 514. In at least one embodiment, in an initial iteration, each bit passed from variable (or vector) nodes 506, 512 to check nodes 508, 514 will be a received bit value, which may be known or unknown, and may be incorrect.

30

[0097] In at least one embodiment, a “check-to-variable” or “C2V” step 510 computes, at each check node 508, 514, the probability or likelihood that each connected variable node 506, 512, as described above, contains a certain value. In at least one embodiment,

5 relationships defined in a low density parity check matrix H will allow for values to converge according said relationships over successive iterations. In at least one embodiment, updated values are copied (or passed) from check nodes 508, 514 to variable (or vector) nodes 506, 512 to complete a “C2V” step 510.

[0098] In at least one embodiment, a flooding schedule for LDPC decoding consists of
 10 performing a “V2C” 504 step and “C2V” 510 step for each bit in received data 502 at once. In at least one embodiment, check nodes 508, 514 are “flooded” with data from variable (or vector) nodes 506, 512, and updated results are “flooded” back to each variable (or vector) node 506, 512. In at least one embodiment, at each iteration a “V2C” step 504 and “C2V” step 510 are performed, wherein each step computes updated values or probabilities for each
 15 variable (or vector) node 506, 512. In at least one embodiment, a flooding schedule for LDPC decoding generally requires a larger number of iterations to converge onto a successful value 516. In at least one embodiment, each iteration has latency and energy requirements to compute and copy data between check nodes 506, 512 and variable (or vector) nodes 508, 514. In at least one embodiment, requiring a higher number of iterations will require longer
 20 latency and greater power requirements, which may not be available given limited computing restrictions of 5G-NR.

[0099] FIG. 6 illustrates a layered decoding schedule for low density parity check (LDPC) decoding of data 602 transmitted across a noisy channel, as described herein. In at least one embodiment, a layered decoding schedule may implement an encoding scheme
 25 defined by a parity check matrix H , as follows:

$$\begin{pmatrix} 0 & 1 & \dots & 1 & 0 \\ 1 & 0 & & 1 & 0 \\ \vdots & & \ddots & \vdots & \\ 0 & 0 & \dots & 1 & 1 \\ 1 & 1 & & 0 & 1 \end{pmatrix}$$

[00100] In at least one embodiment, such a parity check matrix H demonstrates the following relationships between check nodes and variable (or vector) nodes, where each check node corresponds to one of each inequality equation defined as:

30
$$v_2 \oplus \dots \oplus v_{n-1} = 0$$

$$v_1 \oplus \dots \oplus v_{n-1} = 0$$

5

...

$$\dots v_{n-1} \oplus v_n = 0$$

$$v_1 \oplus v_2 \oplus \dots \oplus v_n = 0$$

[00101] In at least one embodiment, a layered decoding schedule updates only a single check node operation 604, 608, 612 at each subiteration 618, 620, 622. In at least one
 10 embodiment, after each single check node is updated 604, 608, 612 during a single subiteration 618, 620, 622, each connected variable (or vector) node, described above, is updated 606, 610, 614.

[00102] In at least one embodiment, each check node is sequentially performed and each variable node is updated, with each subiteration 618, 620, 622 performing single a “V2C”
 15 step 604, 608, 612 and a single “C2V” step 618, 620, 622. In at least one embodiment, an initial subiteration 618 copies, at a “V2C” step 604, values from connected variable (or vector) nodes to a first check node. In at least one embodiment, a first check node calculates updated values and copies said updated values back to connected variable (or vector) nodes during a “C2V” step 606.

[00103] In at least one embodiment, a second subiteration 620 copies, at a “V2C” step
 20 608, values from connected variable (or vector) nodes to a second check node. In at least one embodiment, a second check node calculates updated values and copies said updated values back to connected variable (or vector) nodes during a “C2V” step 610. In at least one embodiment, subiterations continue for each check node until a final check node has been
 25 reached, and a final subiteration 622 is performed.

[00104] In at least one embodiment, a final subiteration 622 copies, at a final “V2C” step
 612, values from connected variable (or vector) nodes to a last check node associated with an LDPC scheme. In at least one embodiment, a last check node calculates updated values and copies said updated values back to connected variable (or vector) nodes during a final “C2V”
 30 step 614, and variable (or vector) nodes then contain decoded data 616. In at least one embodiment, variable (or vector) nodes are updated during each subiteration or step 618, 620, 622. In at least one embodiment, because each subiteration or step XX may be dependent on a value calculated during a previous step, parallelization may not be possible, as discussed below.

5 [00105] FIG. 7A illustrates parallelizable operations in low density parity check (LDPC) decoding. In at least one embodiment, adjacent check nodes 706 that contain no shared variable (or vector) node 708 dependencies may be implemented in parallel. In at least one embodiment, each variable (or vector) node may be updated independently during a “V2C” 702 and “C2V” 704 step in a single iteration of a decoding schedule. In at least one
10 embodiment, if no variable (or vector) nodes are shared, then a layered decoding schedule will require fewer iterations.

[00106] FIG. 7B illustrates non-parallelizable operations in low density parity check (LDPC) decoding. In at least one embodiment, variable (or vector) node 716 values are copied along connected vertices to check nodes 714 during a “V2C” step 710. In at least one
15 embodiment, check node 714 updates are passed along vertices back to connected variable (or vector) nodes during a “C2V” step 712. In at least one embodiment, check nodes 714 that depend on and update a single, shared variable node 716 cannot be parallelized. In at least one embodiment, successive “V2C” steps 710 between iterations may not be able to use an updated “C2V” 712 due to dependencies, requiring more iterations to complete.

20 [00107] FIG. 8 illustrates a parallel-layered schedule for low density parity check (LDPC) decoding of data 802 transmitted across a noisy channel, such as those described herein. In at least one embodiment, a parallel-layered schedule for LDPC decoding separates variable (or vector) nodes that may be parallelized 804, 806, as described above, and performs check node operations 810, 814 for each corresponding variable (or vector) node on a parallel
25 processing unit (PPU) 822, 824. In at least one embodiment, a layering schedule, as described above, is used on each PPU 822, 824 to compute partial variable (or vector) node values for each check node implemented on each PPU.

[00108] In at least one embodiment, each PPU 822, 824 is responsible for processing a portion of check nodes associated with parallelizable variable (or vector) nodes 804, 806 for a
30 given LDPC coding scheme. In at least one embodiment, each set of subiterations or steps 816, 818, 820 in a decoding iteration begins by copying 804, 806 a global variable (or vector) node belief 802 into local variable (or vector) node copies. In at least one embodiment, each PPU 822, 824 computes updated values for each check node, where one check node is processed per step or subiteration 816, 818, 820. During each iteration, in an embodiment,
35 both a “V2C” step 808, 812 and “C2V” step 810, 814 are performed for a check node being processed.

5 [00109] In at least one embodiment, as each check node is processed and associated local variable (or vector) nodes are updated on each processor 822, 824. After all steps or subiterations 816, 818, 820 are complete and each check node has computed 810, 814 updated beliefs for each connected variable (or vector) node, a PPU 822, 824 pushes accumulated deltas or values 826, 828 for each local variable (or vector) node to a global
10 variable (or vector) node belief 802.

[00110] In at least one embodiment, a PPU 822, 824 may update a global variable (or vector) node belief 802 if any of its local variable (or vector) node values changes during each subiteration or step 816, 818, 820 of an iteration. In at least one embodiment, LDPC decoding using a parallel-layering schedule is complete when a maximum number of
15 iterations has been reached. In at least one embodiment, an LDPC decoder implementing a parallel-layering schedule may complete early if global variable (or vector) node beliefs or values 802 satisfy a parity check matrix H associated with said LDPC decoder, as described herein. In at least one embodiment, portions of a global variable (or vector) node belief 802 may be stored locally, whereby storage requirements for each PPU 822, 824 are reduced.

20 [00111] In at least one embodiment, each PPU 822, 824 must only maintain check node information or values for a subset of check nodes associated with parallelized variable nodes. In at least one embodiment, retention of a subset of check node information satisfies smaller memory and cache requirements. In at least one embodiment, multiple codewords (or multiple input data values) 802 may be processed simultaneously as long as parallelization
25 requirements, as described above, are met. In at least one embodiment, maintaining a local copy of variable (or vector) node beliefs may enable local layering and prevent modification from other PPUs 822, 824.

[00112] FIG. 9 illustrates a process for performing parallel-layered LDPC decoding. In at least one embodiment, prior to starting 902, variable (or vector) nodes and associated or
30 connected check nodes have been segmented into multiple processors or parallel processing units (PPUs) according to data dependencies, as described above. In at least one embodiment, parallel-layered LDPC decoding begins by copying 904 relevant bits or variable (or vector) nodes of a global vector node belief into each PPU responsible for updating said relevant bits.

5 [00113] In at least one embodiment, local check nodes are processed according to a layering schedule on each PPU 906, as described above, where each check node is processed individually and variable (or vector) nodes are updated during each step or subiteration. In at least one embodiment, after each check node is processed on each PPU, local vector node beliefs are updated with values from a check node at each step 908. In at least one
 10 embodiment, 908 may improve performance by performing in accordance with a layering schedule. In at least one embodiment, a determination 910 is made regarding if all local check nodes have been processed on a PPU. In at least one embodiment, if all local check nodes have not been processed, local PPU processing continues 906. In at least one embodiment, if all local check nodes have been processed, a PPU updates a global variable
 15 (or vector) node belief with values from its local vector node belief 912.

[00114] In at least one embodiment, if a global variable (or vector) node belief satisfies a parity check matrix 914 specific to an LDPC scheme used, as described above, then a parallel-layering decoding process may terminate 918. In at least one embodiment, if a global variable (or vector) node belief does not satisfy a parity check matrix specific to an
 20 LDPC scheme used, as described above, then a parallel-layering decoding process determines if a maximum number of iterations have been reached 916. In at least one embodiment, if a maximum number of iterations have not been reached, then an up-to-date global variable (or vector) node belief is copied to local vector node beliefs for each PPU 904 and processing continues. In at least one embodiment, if a maximum number of iterations have been
 25 reached, parallel-layered decoding terminates 918.

DATA CENTER

[00115] FIG. 10 illustrates an example data center 1000, in which at least one embodiment may be used. In at least one embodiment, data center 1000 includes a data center infrastructure layer 1010, a framework layer 1020, a software layer 1030 and an application
 30 layer 1040.

[00116] In at least one embodiment, as shown in FIG. 10, data center infrastructure layer 1010 may include a resource orchestrator 1012, grouped computing resources 1014, and node computing resources (“node C.R.s”) 1016(1)-1016(N), where “N” represents any whole, positive integer. In at least one embodiment, node C.R.s 1016(1)-1016(N) may include, but
 35 are not limited to, any number of central processing units (“CPUs”) or other processors

5 (including accelerators, field programmable gate arrays (FPGAs), graphics processors, etc.), memory devices (e.g., dynamic read-only memory), storage devices (e.g., solid state or disk drives), network input/output ("NW I/O") devices, network switches, virtual machines ("VMs"), power modules, and cooling modules, etc. In at least one embodiment, one or more node C.R.s from among node C.R.s 1016(1)-1016(N) may be a server having one or more of
10 above-mentioned computing resources.

[00117] In at least one embodiment, grouped computing resources 1014 may include separate groupings of node C.R.s housed within one or more racks (not shown), or many racks housed in data centers at various geographical locations (also not shown). Separate groupings of node C.R.s within grouped computing resources 1014 may include grouped
15 compute, network, memory or storage resources that may be configured or allocated to support one or more workloads. In at least one embodiment, several node C.R.s including CPUs or processors may grouped within one or more racks to provide compute resources to support one or more workloads. In at least one embodiment, one or more racks may also include any number of power modules, cooling modules, and network switches, in any
20 combination.

[00118] In at least one embodiment, resource orchestrator 1012 may configure or otherwise control one or more node C.R.s 1016(1)-1016(N) and/or grouped computing resources 1014. In at least one embodiment, resource orchestrator 1012 may include a software design infrastructure ("SDI") management entity for data center 1000. In at least
25 one embodiment, resource orchestrator may include hardware, software or some combination thereof.

[00119] In at least one embodiment, as shown in FIG. 10, framework layer 1020 includes a job scheduler 1032, a configuration manager 1034, a resource manager 1036 and a distributed file system 1038. In at least one embodiment, framework layer 1020 may include a
30 framework to support software 1032 of software layer 1030 and/or one or more application(s) 1042 of application layer 1040. In at least one embodiment, software 1032 or application(s) 1042 may respectively include web-based service software or applications, such as those provided by Amazon Web Services, Google Cloud and Microsoft Azure. In at least one embodiment, framework layer 1020 may be, but is not limited to, a type of free and open-
35 source software web application framework such as Apache Spark™ (hereinafter "Spark") that may utilize distributed file system 1038 for large-scale data processing (e.g., "big data").

5 In at least one embodiment, job scheduler 1032 may include a Spark driver to facilitate scheduling of workloads supported by various layers of data center 1000. In at least one embodiment, configuration manager 1034 may be capable of configuring different layers such as software layer 1030 and framework layer 1020 including Spark and distributed file system 1038 for supporting large-scale data processing. In at least one embodiment, resource
10 manager 1036 may be capable of managing clustered or grouped computing resources mapped to or allocated for support of distributed file system 1038 and job scheduler 1032. In at least one embodiment, clustered or grouped computing resources may include grouped computing resource 1014 at data center infrastructure layer 1010. In at least one embodiment, resource manager 1036 may coordinate with resource orchestrator 1012 to
15 manage these mapped or allocated computing resources.

[00120] In at least one embodiment, software 1032 included in software layer 1030 may include software used by at least portions of node C.R.s 1016(1)-1016(N), grouped computing resources 1014, and/or distributed file system 1038 of framework layer 1020. One or more types of software may include, but are not limited to, Internet web page search
20 software, e-mail virus scan software, database software, and streaming video content software.

[00121] In at least one embodiment, application(s) 1042 included in application layer 1040 may include one or more types of applications used by at least portions of node C.R.s 1016(1)-1016(N), grouped computing resources 1014, and/or distributed file system 1038 of
25 framework layer 1020. One or more types of applications may include, but are not limited to, any number of a genomics application, a cognitive compute, and a machine learning application, including training or inferencing software, machine learning framework software (e.g., PyTorch, TensorFlow, Caffe, etc.) or other machine learning applications used in conjunction with one or more embodiments.

30 [00122] In at least one embodiment, any of configuration manager 1034, resource manager 1036, and resource orchestrator 1012 may implement any number and type of self-modifying actions based on any amount and type of data acquired in any technically feasible fashion. In at least one embodiment, self-modifying actions may relieve a data center operator of data center 1000 from making possibly bad configuration decisions and possibly
35 avoiding underutilized and/or poor performing portions of a data center.

- 5 [00123] In at least one embodiment, data center 1000 may include tools, services, software or other resources to train one or more machine learning models or predict or infer information using one or more machine learning models according to one or more embodiments described herein. For example, in at least one embodiment, a machine learning model may be trained by calculating weight parameters according to a neural network
10 architecture using software and computing resources described above with respect to data center 1000. In at least one embodiment, trained machine learning models corresponding to one or more neural networks may be used to infer or predict information using resources described above with respect to data center 1000 by using weight parameters calculated through one or more training techniques described herein.
- 15 [00124] In at least one embodiment, data center may use CPUs, application-specific integrated circuits (ASICs), GPUs, FPGAs, or other hardware to perform training and/or inferencing using above-described resources. Moreover, one or more software and/or hardware resources described above may be configured as a service to allow users to train or performing inferencing of information, such as image recognition, speech recognition, or
20 other artificial intelligence services.

Autonomous Vehicle

- [00125] FIG. 11A illustrates an example of an autonomous vehicle 1100, according to at least one embodiment. In at least one embodiment, autonomous vehicle 1100 (alternatively referred to herein as “vehicle 1100”) may be, without limitation, a passenger vehicle, such as
25 a car, a truck, a bus, and/or another type of vehicle that accommodates one or more passengers. In at least one embodiment, vehicle 1100 may be a semi-tractor-trailer truck used for hauling cargo. In at least one embodiment, vehicle 1100 may be an airplane, robotic vehicle, or other kind of vehicle.
- [00126] Autonomous vehicles may be described in terms of automation levels, defined by
30 National Highway Traffic Safety Administration (“NHTSA”), a division of US Department of Transportation, and Society of Automotive Engineers (“SAE”) “Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles” (e.g., Standard No. J3016-201806, published on June 15, 2018, Standard No. J3016-201609, published on September 30, 2016, and previous and future versions of this standard). In one
35 or more embodiments, vehicle 1100 may be capable of functionality in accordance with one or more of level 1 – level 5 of autonomous driving levels. For example, in at least one

5 embodiment, vehicle 1100 may be capable of conditional automation (Level 3), high automation (Level 4), and/or full automation (Level 5), depending on embodiment.

[00127] In at least one embodiment, vehicle 1100 may include, without limitation, components such as a chassis, a vehicle body, wheels (e.g., 2, 4, 6, 8, 18, etc.), tires, axles, and other components of a vehicle. In at least one embodiment, vehicle 1100 may include,
10 without limitation, a propulsion system 1150, such as an internal combustion engine, hybrid electric power plant, an all-electric engine, and/or another propulsion system type. In at least one embodiment, propulsion system 1150 may be connected to a drive train of vehicle 1100, which may include, without limitation, a transmission, to enable propulsion of vehicle 1100. In at least one embodiment, propulsion system 1150 may be controlled in response to
15 receiving signals from a throttle/accelerator(s) 1152.

[00128] In at least one embodiment, a steering system 1154, which may include, without limitation, a steering wheel, is used to steer a vehicle 1100 (e.g., along a desired path or route) when a propulsion system 1150 is operating (e.g., when vehicle is in motion). In at least one embodiment, a steering system 1154 may receive signals from steering actuator(s)
20 1156. Steering wheel may be optional for full automation (Level 5) functionality. In at least one embodiment, a brake sensor system 1146 may be used to operate vehicle brakes in response to receiving signals from brake actuator(s) 1148 and/or brake sensors.

[00129] In at least one embodiment, controller(s) 1136, which may include, without limitation, one or more system on chips (“SoCs”) (not shown in FIG. 11A) and/or graphics
25 processing unit(s) (“GPU(s)”), provide signals (e.g., representative of commands) to one or more components and/or systems of vehicle 1100. For instance, in at least one embodiment, controller(s) 1136 may send signals to operate vehicle brakes via brake actuators 1148, to operate steering system 1154 via steering actuator(s) 1156, to operate propulsion system 1150 via throttle/accelerator(s) 1152. Controller 1136 may include one or more onboard (e.g.,
30 integrated) computing devices (e.g., supercomputers) that process sensor signals, and output operation commands (e.g., signals representing commands) to enable autonomous driving and/or to assist a human driver in driving vehicle 1100. In at least one embodiment, controller(s) 1136 may include a first controller 1136 for autonomous driving functions, a second controller 1136 for functional safety functions, a third controller 1136 for artificial
35 intelligence functionality (e.g., computer vision), a fourth controller 1136 for infotainment functionality, a fifth controller 1136 for redundancy in emergency conditions, and/or other

5 controllers. In at least one embodiment, a single controller 1136 may handle two or more of above functionalities, two or more controllers 1136 may handle a single functionality, and/or any combination thereof.

[00130] In at least one embodiment, controller(s) 1136 provide signals for controlling one or more components and/or systems of vehicle 1100 in response to sensor data received from
10 one or more sensors (e.g., sensor inputs). In at least one embodiment, sensor data may be received from, for example and without limitation, global navigation satellite systems (“GNSS”) sensor(s) 1158 (e.g., Global Positioning System sensor(s)), RADAR sensor(s) 1160, ultrasonic sensor(s) 1162, LIDAR sensor(s) 1164, inertial measurement unit (“IMU”) sensor(s) 1166 (e.g., accelerometer(s), gyroscope(s), magnetic compass(es), magnetometer(s),
15 etc.), microphone(s) 1196, stereo camera(s) 1168, wide-view camera(s) 1170 (e.g., fisheye cameras), infrared camera(s) 1172, surround camera(s) 1174 (e.g., 360 degree cameras), long-range cameras (not shown in Figure 11A), mid-range camera(s) (not shown in Figure 11A), speed sensor(s) 1144 (e.g., for measuring speed of vehicle 1100), vibration sensor(s) 1142, steering sensor(s) 1140, brake sensor(s) (e.g., as part of brake sensor system 1146),
20 and/or other sensor types.

[00131] In at least one embodiment, one or more of controller(s) 1136 may receive inputs (e.g., represented by input data) from an instrument cluster 1132 of vehicle 1100 and provide outputs (e.g., represented by output data, display data, etc.) via a human-machine interface (“HMI”) display 1134, an audible annunciator, a loudspeaker, and/or via other components of
25 vehicle 1100. In at least one embodiment, outputs may include information such as vehicle velocity, speed, time, map data (e.g., a High Definition map (not shown in FIG. 11A), location data (e.g., vehicle’s 1100 location, such as on a map), direction, location of other vehicles (e.g., an occupancy grid), information about objects and status of objects as perceived by controller(s) 1136, etc. For example, in at least one embodiment, HMI display
30 1134 may display information about presence of one or more objects (e.g., a street sign, caution sign, traffic light changing, etc.), and/or information about driving maneuvers vehicle has made, is making, or will make (e.g., changing lanes now, taking exit 34B in two miles, etc.).

[00132] In at least one embodiment, vehicle 1100 further includes a network interface
35 1124, which may use wireless antenna(s) 1126 and/or modem(s) to communicate over one or more networks. For example, in at least one embodiment, network interface 1124 may be

5 capable of communication over Long-Term Evolution (“LTE”), Wideband Code Division Multiple Access (“WCDMA”), Universal Mobile Telecommunications System (“UMTS”), Global System for Mobile communication (“GSM”), IMT-CDMA Multi-Carrier (“CDMA2000”), etc. In at least one embodiment, wireless antenna(s) 1126 may also enable communication between objects in environment (e.g., vehicles, mobile devices, etc.), using
10 local area network(s), such as Bluetooth, Bluetooth Low Energy (“LE”), Z-Wave, ZigBee, etc., and/or low power wide-area network(s) (“LPWANs”), such as LoRaWAN, SigFox, etc.

[00133] FIG. 11B illustrates an example of camera locations and fields of view for autonomous vehicle 1100 of FIG. 11A, according to at least one embodiment. In at least one embodiment, cameras and respective fields of view are one example embodiment and are not
15 intended to be limiting. For instance, in at least one embodiment, additional and/or alternative cameras may be included and/or cameras may be located at different locations on vehicle 1100.

[00134] In at least one embodiment, camera types for cameras may include, but are not limited to, digital cameras that may be adapted for use with components and/or systems of
20 vehicle 1100. Camera may operate at automotive safety integrity level (“ASIL”) B and/or at another ASIL. In at least one embodiment, camera types may be capable of any image capture rate, such as 60 frames per second (fps), 1220 fps, 240 fps, etc., depending on embodiment. In at least one embodiment, cameras may be capable of using rolling shutters, global shutters, another type of shutter, or a combination thereof. In at least one embodiment,
25 color filter array may include a red clear clear clear (“RCCC”) color filter array, a red clear clear blue (“RCCB”) color filter array, a red blue green clear (“RBGC”) color filter array, a Foveon X3 color filter array, a Bayer sensors (“RGGB”) color filter array, a monochrome sensor color filter array, and/or another type of color filter array. In at least one embodiment, clear pixel cameras, such as cameras with an RCCC, an RCCB, and/or an RBGC color filter
30 array, may be used in an effort to increase light sensitivity.

[00135] In at least one embodiment, one or more of camera(s) may be used to perform advanced driver assistance systems (“ADAS”) functions (e.g., as part of a redundant or fail-safe design). For example, in at least one embodiment, a Multi-Function Mono Camera may be installed to provide functions including lane departure warning, traffic sign assist and
35 intelligent headlamp control. In at least one embodiment, one or more of camera(s) (e.g., all of cameras) may record and provide image data (e.g., video) simultaneously.

5 [00136] In at least one embodiment, one or more of cameras may be mounted in a mounting assembly, such as a custom designed (three-dimensional (“3D”) printed) assembly, in order to cut out stray light and reflections from within car (e.g., reflections from dashboard reflected in windshield mirrors) which may interfere with camera’s image data capture abilities. With reference to wing-mirror mounting assemblies, in at least one embodiment,
10 wing-mirror assemblies may be custom 3D printed so that camera mounting plate matches shape of wing-mirror. In at least one embodiment, camera(s) may be integrated into wing-mirror. In at least one embodiment, for side-view cameras, camera(s) may also be integrated within four pillars at each corner of car.

[00137] In at least one embodiment, cameras with a field of view that include portions of
15 environment in front of vehicle 1100 (e.g., front-facing cameras) may be used for surround view, to help identify forward facing paths and obstacles, as well as aid in, with help of one or more of controllers 1136 and/or control SoCs, providing information critical to generating an occupancy grid and/or determining preferred vehicle paths. In at least one embodiment, front-facing cameras may be used to perform many of same ADAS functions as LIDAR,
20 including, without limitation, emergency braking, pedestrian detection, and collision avoidance. In at least one embodiment, front-facing cameras may also be used for ADAS functions and systems including, without limitation, Lane Departure Warnings (“LDW”), Autonomous Cruise Control (“ACC”), and/or other functions such as traffic sign recognition.

[00138] In at least one embodiment, a variety of cameras may be used in a front-facing
25 configuration, including, for example, a monocular camera platform that includes a CMOS (“complementary metal oxide semiconductor”) color imager. In at least one embodiment, wide-view camera 1170 may be used to perceive objects coming into view from periphery (e.g., pedestrians, crossing traffic or bicycles). Although only one wide-view camera 1170 is illustrated in FIG. 11B, in other embodiments, there may be any number (including zero) of
30 wide-view camera(s) 1170 on vehicle 1100. In at least one embodiment, any number of long-range camera(s) 1198 (e.g., a long-view stereo camera pair) may be used for depth-based object detection, especially for objects for which a neural network has not yet been trained. In at least one embodiment, long-range camera(s) 1198 may also be used for object detection and classification, as well as basic object tracking.

35 [00139] In at least one embodiment, any number of stereo camera(s) 1168 may also be included in a front-facing configuration. In at least one embodiment, one or more of stereo

5 camera(s) 1168 may include an integrated control unit comprising a scalable processing unit, which may provide a programmable logic (“FPGA”) and a multi-core micro-processor with an integrated Controller Area Network (“CAN”) or Ethernet interface on a single chip. In at least one embodiment, such a unit may be used to generate a 3D map of environment of vehicle 1100, including a distance estimate for all points in image. In at least one
10 embodiment, one or more of stereo camera(s) 1168 may include, without limitation, compact stereo vision sensor(s) that may include, without limitation, two camera lenses (one each on left and right) and an image processing chip that may measure distance from vehicle 1100 to target object and use generated information (e.g., metadata) to activate autonomous emergency braking and lane departure warning functions. In at least one embodiment, other
15 types of stereo camera(s) 1168 may be used in addition to, or alternatively from, those described herein.

[00140] In at least one embodiment, cameras with a field of view that include portions of environment to side of vehicle 1100 (e.g., side-view cameras) may be used for surround view, providing information used to create and update occupancy grid, as well as to generate side
20 impact collision warnings. For example, in at least one embodiment, surround camera(s) 1174 (e.g., four surround cameras 1174 as illustrated in FIG. 11B) could be positioned on vehicle 1100. Surround camera(s) 1174 may include, without limitation, any number and combination of wide-view camera(s) 1170, fisheye camera(s), 360 degree camera(s), and/or like. For instance, in at least one embodiment, four fisheye cameras may be positioned on
25 front, rear, and sides of vehicle 1100. In at least one embodiment, vehicle 1100 may use three surround camera(s) 1174 (e.g., left, right, and rear), and may leverage one or more other camera(s) (e.g., a forward-facing camera) as a fourth surround-view camera.

[00141] In at least one embodiment, cameras with a field of view that include portions of environment to rear of vehicle 1100 (e.g., rear-view cameras) may be used for park
30 assistance, surround view, rear collision warnings, and creating and updating occupancy grid. In at least one embodiment, a wide variety of cameras may be used including, but not limited to, cameras that are also suitable as a front-facing camera(s) (e.g., long-range cameras 1198 and/or mid-range camera(s) 1176, stereo camera(s) 1168), infrared camera(s) 1172, etc.), as described herein.

35 [00142] FIG. 11C is a block diagram illustrating an example system architecture for autonomous vehicle 1100 of FIG. 11A, according to at least one embodiment. In at least one

5 embodiment, each of components, features, and systems of vehicle 1100 in FIG. 11C are illustrated as being connected via a bus 1102. In at least one embodiment, bus 1102 may include, without limitation, a CAN data interface (alternatively referred to herein as a “CAN bus”). In at least one embodiment, a CAN may be a network inside vehicle 1100 used to aid in control of various features and functionality of vehicle 1100, such as actuation of brakes,
10 acceleration, braking, steering, windshield wipers, etc. In at least one embodiment, bus 1102 may be configured to have dozens or even hundreds of nodes, each with its own unique identifier (e.g., a CAN ID). In at least one embodiment, bus 1102 may be read to find steering wheel angle, ground speed, engine revolutions per minute (“RPMs”), button positions, and/or other vehicle status indicators. In at least one embodiment, bus 1102 may
15 be a CAN bus that is ASIL B compliant.

[00143] In at least one embodiment, in addition to, or alternatively from CAN, FlexRay and/or Ethernet may be used. In at least one embodiment, there may be any number of busses 1102, which may include, without limitation, zero or more CAN busses, zero or more FlexRay busses, zero or more Ethernet busses, and/or zero or more other types of busses
20 using a different protocol. In at least one embodiment, two or more busses 1102 may be used to perform different functions, and/or may be used for redundancy. For example, a first bus 1102 may be used for collision avoidance functionality and a second bus 1102 may be used for actuation control. In at least one embodiment, each bus 1102 may communicate with any of components of vehicle 1100, and two or more busses 1102 may communicate with same
25 components. In at least one embodiment, each of any number of system(s) on chip(s) (“SoC(s)”) 1104, each of controller(s) 1136, and/or each computer within vehicle may have access to same input data (e.g., inputs from sensors of vehicle 1100), and may be connected to a common bus, such CAN bus.

[00144] In at least one embodiment, vehicle 1100 may include one or more controller(s) 1136, such as those described herein with respect to FIG. 11A. Controller 1136 may be used
30 for a variety of functions. In at least one embodiment, controller(s) 1136 may be coupled to any of various other components and systems of vehicle 1100, and may be used for control of vehicle 1100, artificial intelligence of vehicle 1100, infotainment for vehicle 1100, and/or like.

35 [00145] In at least one embodiment, vehicle 1100 may include any number of SoCs 1104. Each of SoCs 1104 may include, without limitation, central processing units (“CPU(s)”) 1104.

5 1106, graphics processing units (“GPU(s)”) 1108, processor(s) 1110, cache(s) 1112, accelerator(s) 1114, data store(s) 1116, and/or other components and features not illustrated. In at least one embodiment, SoC(s) 1104 may be used to control vehicle 1100 in a variety of platforms and systems. For example, in at least one embodiment, SoC(s) 1104 may be combined in a system (e.g., system of vehicle 1100) with a High Definition (“HD”) map 1122
10 which may obtain map refreshes and/or updates via network interface 1124 from one or more servers (not shown in Figure 11C).

[00146] In at least one embodiment, CPU(s) 1106 may include a CPU cluster or CPU complex (alternatively referred to herein as a “CCPLEX”). In at least one embodiment, CPU(s) 1106 may include multiple cores and/or level two (“L2”) caches. For instance, in at
15 least one embodiment, CPU(s) 1106 may include eight cores in a coherent multi-processor configuration. In at least one embodiment, CPU(s) 1106 may include four dual-core clusters where each cluster has a dedicated L2 cache (e.g., a 2 MB L2 cache). In at least one embodiment, CPU(s) 1106 (e.g., CPLEX) may be configured to support simultaneous cluster operation enabling any combination of clusters of CPU(s) 1106 to be active at any
20 given time.

[00147] In at least one embodiment, one or more of CPU(s) 1106 may implement power management capabilities that include, without limitation, one or more of following features: individual hardware blocks may be clock-gated automatically when idle to save dynamic power; each core clock may be gated when core is not actively executing instructions due to
25 execution of Wait for Interrupt (“WFI”)/Wait for Event (“WFE”) instructions; each core may be independently power-gated; each core cluster may be independently clock-gated when all cores are clock-gated or power-gated; and/or each core cluster may be independently power-gated when all cores are power-gated. In at least one embodiment, CPU(s) 1106 may further implement an enhanced algorithm for managing power states, where allowed power states
30 and expected wakeup times are specified, and hardware/microcode determines best power state to enter for core, cluster, and CPLEX. In at least one embodiment, processing cores may support simplified power state entry sequences in software with work offloaded to microcode.

[00148] In at least one embodiment, GPU(s) 1108 may include an integrated GPU
35 (alternatively referred to herein as an “iGPU”). In at least one embodiment, GPU(s) 1108 may be programmable and may be efficient for parallel workloads. In at least one

5 embodiment, GPU(s) 1108, in at least one embodiment, may use an enhanced tensor instruction set. In on embodiment, GPU(s) 1108 may include one or more streaming microprocessors, where each streaming microprocessor may include a level one (“L1”) cache (e.g., an L1 cache with at least 96KB storage capacity), and two or more of streaming microprocessors may share an L2 cache (e.g., an L2 cache with a 512 KB storage capacity).

10 In at least one embodiment, GPU(s) 1108 may include at least eight streaming microprocessors. In at least one embodiment, GPU(s) 1108 may use compute application programming interface(s) (API(s)). In at least one embodiment, GPU(s) 1108 may use one or more parallel computing platforms and/or programming models (e.g., NVIDIA’s CUDA).

[00149] In at least one embodiment, one or more of GPU(s) 1108 may be power-optimized
15 for best performance in automotive and embedded use cases. For example, in on embodiment, GPU(s) 1108 could be fabricated on a Fin field-effect transistor (“FinFET”). In at least one embodiment, each streaming microprocessor may incorporate a number of mixed-precision processing cores partitioned into multiple blocks. For example, and without limitation, 64 PF32 cores and 32 PF64 cores could be partitioned into four processing blocks.
20 In at least one embodiment, each processing block could be allocated 16 FP32 cores, 8 FP64 cores, 16 INT32 cores, two mixed-precision NVIDIA TENSOR COREs for deep learning matrix arithmetic, a level zero (“L0”) instruction cache, a warp scheduler, a dispatch unit, and/or a 64 KB register file. In at least one embodiment, streaming microprocessors may include independent parallel integer and floating-point data paths to provide for efficient
25 execution of workloads with a mix of computation and addressing calculations. In at least one embodiment, streaming microprocessors may include independent thread scheduling capability to enable finer-grain synchronization and cooperation between parallel threads. In at least one embodiment, streaming microprocessors may include a combined L1 data cache and shared memory unit in order to improve performance while simplifying programming.

30 [00150] In at least one embodiment, one or more of GPU(s) 1108 may include a high bandwidth memory (“HBM”) and/or a 16 GB HBM2 memory subsystem to provide, in some examples, about 900 GB/second peak memory bandwidth. In at least one embodiment, in addition to, or alternatively from, HBM memory, a synchronous graphics random-access memory (“SGRAM”) may be used, such as a graphics double data rate type five synchronous
35 random-access memory (“GDDR5”).

- 5 [00151] In at least one embodiment, GPU(s) 1108 may include unified memory technology. In at least one embodiment, address translation services (“ATS”) support may be used to allow GPU(s) 1108 to access CPU(s) 1106 page tables directly. In at least one embodiment, when GPU(s) 1108 memory management unit (“MMU”) experiences a miss, an address translation request may be transmitted to CPU(s) 1106. In
10 response, CPU(s) 1106 may look in its page tables for virtual-to-physical mapping for address and transmits translation back to GPU(s) 1108, in at least one embodiment. In at least one embodiment, unified memory technology may allow a single unified virtual address space for memory of both CPU(s) 1106 and GPU(s) 1108, thereby simplifying GPU(s) 1108 programming and porting of applications to GPU(s) 1108.
- 15 [00152] In at least one embodiment, GPU(s) 1108 may include any number of access counters that may keep track of frequency of access of GPU(s) 1108 to memory of other processors. In at least one embodiment, access counter(s) may help ensure that memory pages are moved to physical memory of processor that is accessing pages most frequently, thereby improving efficiency for memory ranges shared between processors.
- 20 [00153] In at least one embodiment, one or more of SoC(s) 1104 may include any number of cache(s) 1112, including those described herein. For example, in at least one embodiment, cache(s) 1112 could include a level three (“L3”) cache that is available to both CPU(s) 1106 and GPU(s) 1108 (e.g., that is connected both CPU(s) 1106 and GPU(s) 1108). In at least one embodiment, cache(s) 1112 may include a write-back cache that may keep track of states
25 of lines, such as by using a cache coherence protocol (e.g., MEI, MESI, MSI, etc.). In at least one embodiment, L3 cache may include 4 MB or more, depending on embodiment, although smaller cache sizes may be used.
- [00154] In at least one embodiment, one or more of SoC(s) 1104 may include one or more accelerator(s) 1114 (e.g., hardware accelerators, software accelerators, or a combination
30 thereof). In at least one embodiment, SoC(s) 1104 may include a hardware acceleration cluster that may include optimized hardware accelerators and/or large on-chip memory. In at least one embodiment, large on-chip memory (e.g., 4MB of SRAM), may enable hardware acceleration cluster to accelerate neural networks and other calculations. In at least one embodiment, hardware acceleration cluster may be used to complement GPU(s) 1108 and to
35 off-load some of tasks of GPU(s) 1108 (e.g., to free up more cycles of GPU(s) 1108 for performing other tasks). In at least one embodiment, accelerator(s) 1114 could be used for

5 targeted workloads (e.g., perception, convolutional neural networks ("CNNs"), recurrent neural networks ("RNNs"), etc.) that are stable enough to be amenable to acceleration. In at least one embodiment, a CNN may include a region-based or regional convolutional neural networks ("RCNNs") and Fast RCNNs (e.g., as used for object detection) or other type of CNN.

10 [00155] In at least one embodiment, accelerator(s) 1114 (e.g., hardware acceleration cluster) may include a deep learning accelerator(s) ("DLA"). DLA(s) may include, without limitation, one or more Tensor processing units ("TPUs") that may be configured to provide an additional ten trillion operations per second for deep learning applications and inferencing. In at least one embodiment, TPUs may be accelerators configured to, and optimized for,
15 performing image processing functions (e.g., for CNNs, RCNNs, etc.). DLA(s) may further be optimized for a specific set of neural network types and floating point operations, as well as inferencing. In at least one embodiment, design of DLA(s) may provide more performance per millimeter than a typical general-purpose GPU, and typically vastly exceeds performance of a CPU. In at least one embodiment, TPU(s) may perform several functions,
20 including a single-instance convolution function, supporting, for example, INT8, INT16, and FP16 data types for both features and weights, as well as post-processor functions. In at least one embodiment, DLA(s) may quickly and efficiently execute neural networks, especially CNNs, on processed or unprocessed data for any of a variety of functions, including, for example and without limitation: a CNN for object identification and detection using data
25 from camera sensors; a CNN for distance estimation using data from camera sensors; a CNN for emergency vehicle detection and identification and detection using data from microphones 1196; a CNN for facial recognition and vehicle owner identification using data from camera sensors; and/or a CNN for security and/or safety related events.

[00156] In at least one embodiment, DLA(s) may perform any function of GPU(s) 1108,
30 and by using an inference accelerator, for example, a designer may target either DLA(s) or GPU(s) 1108 for any function. For example, in at least one embodiment, designer may focus processing of CNNs and floating point operations on DLA(s) and leave other functions to GPU(s) 1108 and/or other accelerator(s) 1114.

[00157] In at least one embodiment, accelerator(s) 1114 (e.g., hardware acceleration
35 cluster) may include a programmable vision accelerator(s) ("PVA"), which may alternatively be referred to herein as a computer vision accelerator. In at least one embodiment, PVA(s)

5 may be designed and configured to accelerate computer vision algorithms for advanced driver assistance system ("ADAS") 1138, autonomous driving, augmented reality ("AR") applications, and/or virtual reality ("VR") applications. PVA(s) may provide a balance between performance and flexibility. For example, in at least one embodiment, each PVA(s) may include, for example and without limitation, any number of reduced instruction set
10 computer ("RISC") cores, direct memory access ("DMA"), and/or any number of vector processors.

[00158] In at least one embodiment, RISC cores may interact with image sensors (e.g., image sensors of any of cameras described herein), image signal processor(s), and/or like. In at least one embodiment, each of RISC cores may include any amount of memory. In at least
15 one embodiment, RISC cores may use any of a number of protocols, depending on embodiment. In at least one embodiment, RISC cores may execute a real-time operating system ("RTOS"). In at least one embodiment, RISC cores may be implemented using one or more integrated circuit devices, application specific integrated circuits ("ASICs"), and/or memory devices. For example, in at least one embodiment, RISC cores could include an
20 instruction cache and/or a tightly coupled RAM.

[00159] In at least one embodiment, DMA may enable components of PVA(s) to access system memory independently of CPU(s) 1106. In at least one embodiment, DMA may support any number of features used to provide optimization to PVA including, but not limited to, supporting multi-dimensional addressing and/or circular addressing. In at least
25 one embodiment, DMA may support up to six or more dimensions of addressing, which may include, without limitation, block width, block height, block depth, horizontal block stepping, vertical block stepping, and/or depth stepping.

[00160] In at least one embodiment, vector processors may be programmable processors that may be designed to efficiently and flexibly execute programming for computer vision
30 algorithms and provide signal processing capabilities. In at least one embodiment, PVA may include a PVA core and two vector processing subsystem partitions. In at least one embodiment, PVA core may include a processor subsystem, DMA engine(s) (e.g., two DMA engines), and/or other peripherals. In at least one embodiment, vector processing subsystem may operate as primary processing engine of PVA, and may include a vector processing unit
35 ("VPU"), an instruction cache, and/or vector memory (e.g., "VMEM"). In at least one embodiment, VPU core may include a digital signal processor such as, for example, a single

5 instruction, multiple data ("SIMD"), very long instruction word ("VLIW") digital signal processor. In at least one embodiment, a combination of SIMD and VLIW may enhance throughput and speed.

[00161] In at least one embodiment, each of vector processors may include an instruction cache and may be coupled to dedicated memory. As a result, in at least one embodiment,
10 each of vector processors may be configured to execute independently of other vector processors. In at least one embodiment, vector processors that are included in a particular PVA may be configured to employ data parallelism. For instance, in at least one embodiment, plurality of vector processors included in a single PVA may execute same computer vision algorithm, but on different regions of an image. In at least one embodiment,
15 vector processors included in a particular PVA may simultaneously execute different computer vision algorithms, on same image, or even execute different algorithms on sequential images or portions of an image. In at least one embodiment, among other things, any number of PVAs may be included in hardware acceleration cluster and any number of vector processors may be included in each of PVAs. In at least one embodiment, PVA(s)
20 may include additional error correcting code ("ECC") memory, to enhance overall system safety.

[00162] In at least one embodiment, accelerator(s) 1114 (e.g., hardware acceleration cluster) may include a computer vision network on-chip and static random-access memory ("SRAM"), for providing a high-bandwidth, low latency SRAM for accelerator(s) 1114. In at
25 least one embodiment, on-chip memory may include at least 4MB SRAM, consisting of, for example and without limitation, eight field-configurable memory blocks, that may be accessible by both PVA and DLA. In at least one embodiment, each pair of memory blocks may include an advanced peripheral bus ("APB") interface, configuration circuitry, a controller, and a multiplexer. In at least one embodiment, any type of memory may be used.
30 In at least one embodiment, PVA and DLA may access memory via a backbone that provides PVA and DLA with high-speed access to memory. In at least one embodiment, backbone may include a computer vision network on-chip that interconnects PVA and DLA to memory (e.g., using APB).

[00163] In at least one embodiment, computer vision network on-chip may include an
35 interface that determines, before transmission of any control signal/address/data, that both PVA and DLA provide ready and valid signals. In at least one embodiment, an interface may

5 provide for separate phases and separate channels for transmitting control signals/addresses/data, as well as burst-type communications for continuous data transfer. In at least one embodiment, an interface may comply with International Organization for Standardization (“ISO”) 26262 or International Electrotechnical Commission (“IEC”) 61508 standards, although other standards and protocols may be used.

10 [00164] In at least one embodiment, one or more of SoC(s) 1104 may include a real-time ray-tracing hardware accelerator. In at least one embodiment, real-time ray-tracing hardware accelerator may be used to quickly and efficiently determine positions and extents of objects (e.g., within a world model), to generate real-time visualization simulations, for RADAR
15 SONAR systems, for general wave propagation simulation, for comparison to LIDAR data for purposes of localization and/or other functions, and/or for other uses.

[00165] In at least one embodiment, accelerator(s) 1114 (e.g., hardware accelerator cluster) have a wide array of uses for autonomous driving. In at least one embodiment, PVA may be a programmable vision accelerator that may be used for key processing stages in
20 ADAS and autonomous vehicles. In at least one embodiment, PVA’s capabilities are a good match for algorithmic domains needing predictable processing, at low power and low latency. In other words, PVA performs well on semi-dense or dense regular computation, even on small data sets, which need predictable run-times with low latency and low power. In at least one embodiment, autonomous vehicles, such as vehicle 1100, PVAs are designed to run
25 classic computer vision algorithms, as they are efficient at object detection and operating on integer math.

[00166] For example, according to at least one embodiment of technology, PVA is used to perform computer stereo vision. In at least one embodiment, semi-global matching-based algorithm may be used in some examples, although this is not intended to be limiting. In at
30 least one embodiment, applications for Level 3-5 autonomous driving use motion estimation/stereo matching on-the-fly (e.g., structure from motion, pedestrian recognition, lane detection, etc.). In at least one embodiment, PVA may perform computer stereo vision function on inputs from two monocular cameras.

[00167] In at least one embodiment, PVA may be used to perform dense optical flow. For
35 example, in at least one embodiment, PVA could process raw RADAR data (e.g., using a 4D

5 Fast Fourier Transform) to provide processed RADAR data. In at least one embodiment, PVA is used for time of flight depth processing, by processing raw time of flight data to provide processed time of flight data, for example.

[00168] In at least one embodiment, DLA may be used to run any type of network to enhance control and driving safety, including for example and without limitation, a neural
 10 network that outputs a measure of confidence for each object detection. In at least one embodiment, confidence may be represented or interpreted as a probability, or as providing a relative “weight” of each detection compared to other detections. In at least one embodiment, confidence enables a system to make further decisions regarding which detections should be considered as true positive detections rather than false positive detections. For example, in at
 15 least one embodiment, a system may set a threshold value for confidence and consider only detections exceeding threshold value as true positive detections. In an embodiment in which an automatic emergency braking (“AEB”) system is used, false positive detections would cause vehicle to automatically perform emergency braking, which is obviously undesirable. In at least one embodiment, highly confident detections may be considered as triggers for
 20 AEB. In at least one embodiment, DLA may run a neural network for regressing confidence value. In at least one embodiment, neural network may take as its input at least some subset of parameters, such as bounding box dimensions, ground plane estimate obtained (e.g. from another subsystem), output from IMU sensor(s) 1166 that correlates with vehicle 1100 orientation, distance, 3D location estimates of object obtained from neural network and/or
 25 other sensors (e.g., LIDAR sensor(s) 1164 or RADAR sensor(s) 1160), among others.

[00169] In at least one embodiment, one or more of SoC(s) 1104 may include data store(s) 1116 (e.g., memory). In at least one embodiment, data store(s) 1116 may be on-chip memory of SoC(s) 1104, which may store neural networks to be executed on GPU(s) 1108 and/or DLA. In at least one embodiment, data store(s) 1116 may be large enough in capacity
 30 to store multiple instances of neural networks for redundancy and safety. In at least one embodiment, data store(s) 1112 may comprise L2 or L3 cache(s). .

[00170] In at least one embodiment, one or more of SoC(s) 1104 may include any number of processor(s) 1110 (e.g., embedded processors). Processor 1110 may include a boot and power management processor that may be a dedicated processor and subsystem to handle
 35 boot power and management functions and related security enforcement. In at least one embodiment, boot and power management processor may be a part of SoC(s) 1104 boot

5 sequence and may provide runtime power management services. In at least one embodiment, boot power and management processor may provide clock and voltage programming, assistance in system low power state transitions, management of SoC(s) 1104 thermals and temperature sensors, and/or management of SoC(s) 1104 power states. In at least one embodiment, each temperature sensor may be implemented as a ring-oscillator whose output
10 frequency is proportional to temperature, and SoC(s) 1104 may use ring-oscillators to detect temperatures of CPU(s) 1106, GPU(s) 1108, and/or accelerator(s) 1114. In at least one embodiment, if temperatures are determined to exceed a threshold, then boot and power management processor may enter a temperature fault routine and put SoC(s) 1104 into a lower power state and/or put vehicle 1100 into a chauffeur to safe stop mode (e.g., bring
15 vehicle 1100 to a safe stop).

[00171] In at least one embodiment, processor(s) 1110 may further include a set of embedded processors that may serve as an audio processing engine. In at least one embodiment, audio processing engine may be an audio subsystem that enables full hardware support for multi-channel audio over multiple interfaces, and a broad and flexible range of
20 audio I/O interfaces. In at least one embodiment, audio processing engine is a dedicated processor core with a digital signal processor with dedicated RAM.

[00172] In at least one embodiment, processor(s) 1110 may further include an always on processor engine that may provide necessary hardware features to support low power sensor management and wake use cases. In at least one embodiment, always on processor engine
25 may include, without limitation, a processor core, a tightly coupled RAM, supporting peripherals (e.g., timers and interrupt controllers), various I/O controller peripherals, and routing logic.

[00173] In at least one embodiment, processor(s) 1110 may further include a safety cluster engine that includes, without limitation, a dedicated processor subsystem to handle safety
30 management for automotive applications. In at least one embodiment, safety cluster engine may include, without limitation, two or more processor cores, a tightly coupled RAM, support peripherals (e.g., timers, an interrupt controller, etc.), and/or routing logic. In a safety mode, two or more cores may operate, in at least one embodiment, in a lockstep mode and function as a single core with comparison logic to detect any differences between their
35 operations. In at least one embodiment, processor(s) 1110 may further include a real-time camera engine that may include, without limitation, a dedicated processor subsystem for

5 handling real-time camera management. In at least one embodiment, processor(s) 1110 may further include a high-dynamic range signal processor that may include, without limitation, an image signal processor that is a hardware engine that is part of camera processing pipeline.

[00174] In at least one embodiment, processor(s) 1110 may include a video image compositor that may be a processing block (e.g., implemented on a microprocessor) that
10 implements video post-processing functions needed by a video playback application to produce final image for player window. In at least one embodiment, video image compositor may perform lens distortion correction on wide-view camera(s) 1170, surround camera(s) 1174, and/or on in-cabin monitoring camera sensor(s). In at least one embodiment, in-cabin monitoring camera sensor(s) are preferably monitored by a neural network running on
15 another instance of SoC 1104, configured to identify in cabin events and respond accordingly. In at least one embodiment, an in-cabin system may perform, without limitation, lip reading to activate cellular service and place a phone call, dictate emails, change vehicle's destination, activate or change vehicle's infotainment system and settings, or provide voice-activated web surfing. In at least one embodiment, certain functions are
20 available to driver when vehicle is operating in an autonomous mode and are disabled otherwise.

[00175] In at least one embodiment, video image compositor may include enhanced temporal noise reduction for both spatial and temporal noise reduction. For example, in at least one embodiment, where motion occurs in a video, noise reduction weights spatial
25 information appropriately, decreasing weight of information provided by adjacent frames. In at least one embodiment, where an image or portion of an image does not include motion, temporal noise reduction performed by video image compositor may use information from previous image to reduce noise in current image.

[00176] In at least one embodiment, video image compositor may also be configured to
30 perform stereo rectification on input stereo lens frames. In at least one embodiment, video image compositor may further be used for user interface composition when operating system desktop is in use, and GPU(s) 1108 are not required to continuously render new surfaces. In at least one embodiment, when GPU(s) 1108 are powered on and active doing 3D rendering, video image compositor may be used to offload GPU(s) 1108 to improve performance and
35 responsiveness.

5 [00177] In at least one embodiment, one or more of SoC(s) 1104 may further include a mobile industry processor interface (“MIPI”) camera serial interface for receiving video and input from cameras, a high-speed interface, and/or a video input block that may be used for camera and related pixel input functions. In at least one embodiment, one or more of SoC(s) 1104 may further include an input/output controller(s) that may be controlled by software and
10 may be used for receiving I/O signals that are uncommitted to a specific role.

[00178] In at least one embodiment, one or more of SoC(s) 1104 may further include a broad range of peripheral interfaces to enable communication with peripherals, audio encoders/decoders (“codecs”), power management, and/or other devices. SoC(s) 1104 may be used to process data from cameras (e.g., connected over Gigabit Multimedia Serial Link and Ethernet), sensors (e.g., LIDAR sensor(s) 1164, RADAR sensor(s) 1160, etc. that may be
15 connected over Ethernet), data from bus 1102 (e.g., speed of vehicle 1100, steering wheel position, etc.), data from GNSS sensor(s) 1158 (e.g., connected over Ethernet or CAN bus), etc. In at least one embodiment, one or more of SoC(s) 1104 may further include dedicated high-performance mass storage controllers that may include their own DMA engines, and
20 that may be used to free CPU(s) 1106 from routine data management tasks.

[00179] In at least one embodiment, SoC(s) 1104 may be an end-to-end platform with a flexible architecture that spans automation levels 3-5, thereby providing a comprehensive functional safety architecture that leverages and makes efficient use of computer vision and ADAS techniques for diversity and redundancy, provides a platform for a flexible, reliable
25 driving software stack, along with deep learning tools. In at least one embodiment, SoC(s) 1104 may be faster, more reliable, and even more energy-efficient and space-efficient than conventional systems. For example, in at least one embodiment, accelerator(s) 1114, when combined with CPU(s) 1106, GPU(s) 1108, and data store(s) 1116, may provide for a fast, efficient platform for level 3-5 autonomous vehicles.

30 [00180] In at least one embodiment, computer vision algorithms may be executed on CPUs, which may be configured using high-level programming language, such as C programming language, to execute a wide variety of processing algorithms across a wide variety of visual data. However, in at least one embodiment, CPUs are oftentimes unable to meet performance requirements of many computer vision applications, such as those related
35 to execution time and power consumption, for example. In at least one embodiment, many

5 CPUs are unable to execute complex object detection algorithms in real-time, which is used in in-vehicle ADAS applications and in practical Level 3-5 autonomous vehicles.

[00181] Embodiments described herein allow for multiple neural networks to be performed simultaneously and/or sequentially, and for results to be combined together to enable Level 3-5 autonomous driving functionality. For example, in at least one
10 embodiment, a CNN executing on DLA or discrete GPU (e.g., GPU(s) 1120) may include text and word recognition, allowing supercomputer to read and understand traffic signs, including signs for which neural network has not been specifically trained. In at least one embodiment, DLA may further include a neural network that is able to identify, interpret, and provide semantic understanding of sign, and to pass that semantic understanding to path
15 planning modules running on CPU Complex.

[00182] In at least one embodiment, multiple neural networks may be run simultaneously, as for Level 3, 4, or 5 driving. For example, in at least one embodiment, a warning sign consisting of “Caution: flashing lights indicate icy conditions,” along with an electric light, may be independently or collectively interpreted by several neural networks. In at least one
20 embodiment, sign itself may be identified as a traffic sign by a first deployed neural network (e.g., a neural network that has been trained), text “flashing lights indicate icy conditions” may be interpreted by a second deployed neural network, which informs vehicle’s path planning software (preferably executing on CPU Complex) that when flashing lights are detected, icy conditions exist. In at least one embodiment, flashing light may be identified by
25 operating a third deployed neural network over multiple frames, informing vehicle’s path-planning software of presence (or absence) of flashing lights. In at least one embodiment, all three neural networks may run simultaneously, such as within DLA and/or on GPU(s) 1108.

[00183] In at least one embodiment, a CNN for facial recognition and vehicle owner identification may use data from camera sensors to identify presence of an authorized driver
30 and/or owner of vehicle 1100. In at least one embodiment, an always on sensor processing engine may be used to unlock vehicle when owner approaches driver door and turn on lights, and, in security mode, to disable vehicle when owner leaves vehicle. In this way, SoC(s) 1104 provide for security against theft and/or carjacking.

[00184] In at least one embodiment, a CNN for emergency vehicle detection and
35 identification may use data from microphones 1196 to detect and identify emergency vehicle

5 sirens. In at least one embodiment, SoC(s) 1104 use CNN for classifying environmental and urban sounds, as well as classifying visual data. In at least one embodiment, CNN running on DLA is trained to identify relative closing speed of emergency vehicle (e.g., by using Doppler effect). In at least one embodiment, CNN may also be trained to identify emergency vehicles specific to local area in which vehicle is operating, as identified by GNSS sensor(s)
10 1158. In at least one embodiment, when operating in Europe, CNN will seek to detect European sirens, and when in United States CNN will seek to identify only North American sirens. In at least one embodiment, once an emergency vehicle is detected, a control program may be used to execute an emergency vehicle safety routine, slowing vehicle, pulling over to side of road, parking vehicle, and/or idling vehicle, with assistance of ultrasonic sensor(s)
15 1162, until emergency vehicle(s) passes.

[00185] In at least one embodiment, vehicle 1100 may include CPU(s) 1118 (e.g., discrete CPU(s), or dCPU(s)), that may be coupled to SoC(s) 1104 via a high-speed interconnect (e.g., PCIe). In at least one embodiment, CPU(s) 1118 may include an X86 processor, for example. CPU(s) 1118 may be used to perform any of a variety of functions, including
20 arbitrating potentially inconsistent results between ADAS sensors and SoC(s) 1104, and/or monitoring status and health of controller(s) 1136 and/or an infotainment system on a chip (“infotainment SoC”) 1130, for example.

[00186] In at least one embodiment, vehicle 1100 may include GPU(s) 1120 (e.g., discrete GPU(s), or dGPU(s)), that may be coupled to SoC(s) 1104 via a high-speed interconnect
25 (e.g., NVIDIA’s NVLINK). In at least one embodiment, GPU(s) 1120 may provide additional artificial intelligence functionality, such as by executing redundant and/or different neural networks, and may be used to train and/or update neural networks based at least in part on input (e.g., sensor data) from sensors of vehicle 1100.

[00187] In at least one embodiment, vehicle 1100 may further include network interface
30 1124 which may include, without limitation, wireless antenna(s) 1126 (e.g., one or more wireless antennas 1126 for different communication protocols, such as a cellular antenna, a Bluetooth antenna, etc.). In at least one embodiment, network interface 1124 may be used to enable wireless connectivity over Internet with cloud (e.g., with server(s) and/or other network devices), with other vehicles, and/or with computing devices (e.g., client devices of
35 passengers). In at least one embodiment, to communicate with other vehicles, a direct link may be established between vehicle 110 and other vehicle and/or an indirect link may be

5 established (e.g., across networks and over Internet). In at least one embodiment, direct links may be provided using a vehicle-to-vehicle communication link. Vehicle-to-vehicle communication link may provide vehicle 1100 information about vehicles in proximity to vehicle 1100 (e.g., vehicles in front of, on side of, and/or behind vehicle 1100). In at least one embodiment, aforementioned functionality may be part of a cooperative adaptive cruise control functionality of vehicle 1100.

[00188] In at least one embodiment, network interface 1124 may include an SoC that provides modulation and demodulation functionality and enables controller(s) 1136 to communicate over wireless networks. In at least one embodiment, network interface 1124 may include a radio frequency front-end for up-conversion from baseband to radio frequency, and down conversion from radio frequency to baseband. In at least one embodiment, frequency conversions may be performed in any technically feasible fashion. For example, frequency conversions could be performed through well-known processes, and/or using super-heterodyne processes. In at least one embodiment, radio frequency front end functionality may be provided by a separate chip. In at least one embodiment, network interface may include wireless functionality for communicating over LTE, WCDMA, UMTS, GSM, CDMA2000, Bluetooth, Bluetooth LE, Wi-Fi, Z-Wave, ZigBee, LoRaWAN, and/or other wireless protocols.

[00189] In at least one embodiment, vehicle 1100 may further include data store(s) 1128 which may include, without limitation, off-chip (e.g., off SoC(s) 1104) storage. In at least one embodiment, data store(s) 1128 may include, without limitation, one or more storage elements including RAM, SRAM, dynamic random-access memory (“DRAM”), video random-access memory (“VRAM”), Flash, hard disks, and/or other components and/or devices that may store at least one bit of data.

[00190] In at least one embodiment, vehicle 1100 may further include GNSS sensor(s) 1158 (e.g., GPS and/or assisted GPS sensors), to assist in mapping, perception, occupancy grid generation, and/or path planning functions. In at least one embodiment, any number of GNSS sensor(s) 1158 may be used, including, for example and without limitation, a GPS using a USB connector with an Ethernet to Serial (e.g., RS-232) bridge.

[00191] In at least one embodiment, vehicle 1100 may further include RADAR sensor(s) 1160. RADAR sensor(s) 1160 may be used by vehicle 1100 for long-range vehicle detection,

5 even in darkness and/or severe weather conditions. In at least one embodiment, RADAR functional safety levels may be ASIL B. RADAR sensor(s) 1160 may use CAN and/or bus 1102 (e.g., to transmit data generated by RADAR sensor(s) 1160) for control and to access object tracking data, with access to Ethernet to access raw data in some examples. In at least one embodiment, wide variety of RADAR sensor types may be used. For example, and
10 without limitation, RADAR sensor(s) 1160 may be suitable for front, rear, and side RADAR use. In at least one embodiment, one or more of RADAR sensors(s) 1160 are Pulse Doppler RADAR sensor(s).

[00192] In at least one embodiment, RADAR sensor(s) 1160 may include different configurations, such as long-range with narrow field of view, short-range with wide field of
15 view, short-range side coverage, etc. In at least one embodiment, long-range RADAR may be used for adaptive cruise control functionality. In at least one embodiment, long-range RADAR systems may provide a broad field of view realized by two or more independent scans, such as within a 250m range. In at least one embodiment, RADAR sensor(s) 1160 may help in distinguishing between static and moving objects, and may be used by ADAS
20 system 1138 for emergency brake assist and forward collision warning. Sensors 1160(s) included in a long-range RADAR system may include, without limitation, monostatic multimodal RADAR with multiple (e.g., six or more) fixed RADAR antennae and a high-speed CAN and FlexRay interface. In at least one embodiment, with six antennae, central four antennae may create a focused beam pattern, designed to record vehicle's 1100
25 surroundings at higher speeds with minimal interference from traffic in adjacent lanes. In at least one embodiment, other two antennae may expand field of view, making it possible to quickly detect vehicles entering or leaving vehicle's 1100 lane.

[00193] In at least one embodiment, mid-range RADAR systems may include, as an example, a range of up to 160m (front) or 80m (rear), and a field of view of up to 42 degrees
30 (front) or 150 degrees (rear). In at least one embodiment, short-range RADAR systems may include, without limitation, any number of RADAR sensor(s) 1160 designed to be installed at both ends of rear bumper. When installed at both ends of rear bumper, in at least one embodiment, a RADAR sensor system may create two beams that constantly monitor blind spot in rear and next to vehicle. In at least one embodiment, short-range RADAR systems
35 may be used in ADAS system 1138 for blind spot detection and/or lane change assist.

5 [00194] In at least one embodiment, vehicle 1100 may further include ultrasonic sensor(s) 1162. Ultrasonic sensor(s) 1162, which may be positioned at front, back, and/or sides of vehicle 1100, may be used for park assist and/or to create and update an occupancy grid. In at least one embodiment, a wide variety of ultrasonic sensor(s) 1162 may be used, and different ultrasonic sensor(s) 1162 may be used for different ranges of detection (e.g., 2.5m,
10 4m). In at least one embodiment, ultrasonic sensor(s) 1162 may operate at functional safety levels of ASIL B.

[00195] In at least one embodiment, vehicle 1100 may include LIDAR sensor(s) 1164. LIDAR sensor(s) 1164 may be used for object and pedestrian detection, emergency braking, collision avoidance, and/or other functions. In at least one embodiment, LIDAR sensor(s)
15 1164 may be functional safety level ASIL B. In at least one embodiment, vehicle 1100 may include multiple LIDAR sensors 1164 (e.g., two, four, six, etc.) that may use Ethernet (e.g., to provide data to a Gigabit Ethernet switch).

[00196] In at least one embodiment, LIDAR sensor(s) 1164 may be capable of providing a list of objects and their distances for a 360-degree field of view. In at least one embodiment,
20 commercially available LIDAR sensor(s) 1164 may have an advertised range of approximately 100m, with an accuracy of 2cm-3cm, and with support for a 100 Mbps Ethernet connection, for example. In at least one embodiment, one or more non-protruding LIDAR sensors 1164 may be used. In such an embodiment, LIDAR sensor(s) 1164 may be implemented as a small device that may be embedded into front, rear, sides, and/or corners of
25 vehicle 1100. In at least one embodiment, LIDAR sensor(s) 1164, in such an embodiment, may provide up to a 120-degree horizontal and 35-degree vertical field-of-view, with a 200m range even for low-reflectivity objects. In at least one embodiment, front-mounted LIDAR sensor(s) 1164 may be configured for a horizontal field of view between 45 degrees and 135 degrees.

30 [00197] In at least one embodiment, LIDAR technologies, such as 3D flash LIDAR, may also be used. 3D Flash LIDAR uses a flash of a laser as a transmission source, to illuminate surroundings of vehicle 1100 up to approximately 200m. In at least one embodiment, a flash LIDAR unit includes, without limitation, a receptor, which records laser pulse transit time and reflected light on each pixel, which in turn corresponds to range from vehicle 1100 to
35 objects. In at least one embodiment, flash LIDAR may allow for highly accurate and distortion-free images of surroundings to be generated with every laser flash. In at least one

5 embodiment, four flash LIDAR sensors may be deployed, one at each side of vehicle 1100. In at least one embodiment, 3D flash LIDAR systems include, without limitation, a solid-state 3D staring array LIDAR camera with no moving parts other than a fan (e.g., a non-scanning LIDAR device). In at least one embodiment, flash LIDAR device may use a 5 nanosecond class I (eye-safe) laser pulse per frame and may capture reflected laser light in
10 form of 3D range point clouds and co-registered intensity data.

[00198] In at least one embodiment, vehicle may further include IMU sensor(s) 1166. In at least one embodiment, IMU sensor(s) 1166 may be located at a center of rear axle of vehicle 1100, in at least one embodiment. In at least one embodiment, IMU sensor(s) 1166 may include, for example and without limitation, accelerometer(s), magnetometer(s),
15 gyroscope(s), magnetic compass(es), and/or other sensor types. In at least one embodiment, such as in six-axis applications, IMU sensor(s) 1166 may include, without limitation, accelerometers and gyroscopes. , In at least one embodiment, such as in nine-axis applications, IMU sensor(s) 1166 may include, without limitation, accelerometers, gyroscopes, and magnetometers.

20 [00199] In at least one embodiment, IMU sensor(s) 1166 may be implemented as a miniature, high performance GPS-Aided Inertial Navigation System ("GPS/INS") that combines micro-electro-mechanical systems ("MEMS") inertial sensors, a high-sensitivity GPS receiver, and advanced Kalman filtering algorithms to provide estimates of position, velocity, and attitude. In at least one embodiment, IMU sensor(s) 1166 may enable vehicle
25 1100 to estimate heading without requiring input from a magnetic sensor by directly observing and correlating changes in velocity from GPS to IMU sensor(s) 1166. In at least one embodiment, IMU sensor(s) 1166 and GNSS sensor(s) 1158 may be combined in a single integrated unit.

[00200] In at least one embodiment, vehicle 1100 may include microphone(s) 1196 placed
30 in and/or around vehicle 1100. In at least one embodiment, microphone(s) 1196 may be used for emergency vehicle detection and identification, among other things.

[00201] In at least one embodiment, vehicle 1100 may further include any number of camera types, including stereo camera(s) 1168, wide-view camera(s) 1170, infrared camera(s) 1172, surround camera(s) 1174, long-range camera(s) 1198, mid-range camera(s)
35 1176, and/or other camera types. In at least one embodiment, cameras may be used to

5 capture image data around an entire periphery of vehicle 1100. In at least one embodiment, types of cameras used depends vehicle 1100. In at least one embodiment, any combination of camera types may be used to provide necessary coverage around vehicle 1100. In at least one embodiment, number of cameras may differ depending on embodiment. For example, in at least one embodiment, vehicle 1100 could include six cameras, seven cameras, ten cameras,
10 twelve cameras, or another number of cameras. Cameras may support, as an example and without limitation, Gigabit Multimedia Serial Link (“GMSL”) and/or Gigabit Ethernet. In at least one embodiment, each of camera(s) is described with more detail previously herein with respect to FIG. 11A and FIG. 11B.

[00202] In at least one embodiment, vehicle 1100 may further include vibration sensor(s)
15 1142. Vibration sensor(s) 1142 may measure vibrations of components of vehicle 1100, such as axle(s). For example, in at least one embodiment, changes in vibrations may indicate a change in road surfaces. In at least one embodiment, when two or more vibration sensors 1142 are used, differences between vibrations may be used to determine friction or slippage of road surface (e.g., when difference in vibration is between a power-driven axle and a freely
20 rotating axle).

[00203] In at least one embodiment, vehicle 1100 may include ADAS system 1138. ADAS system 1138 may include, without limitation, an SoC, in some examples. In at least one embodiment, ADAS system 1138 may include, without limitation, any number and combination of an autonomous/adaptive/automatic cruise control (“ACC”) system, a
25 cooperative adaptive cruise control (“CACC”) system, a forward crash warning (“FCW”) system, an automatic emergency braking (“AEB”) system, a lane departure warning (“LDW”) system, a lane keep assist (“LKA”) system, a blind spot warning (“BSW”) system, a rear cross-traffic warning (“RCTW”) system, a collision warning (“CW”) system, a lane centering (“LC”) system, and/or other systems, features, and/or functionality.

30 [00204] In at least one embodiment, ACC system may use RADAR sensor(s) 1160, LIDAR sensor(s) 1164, and/or any number of camera(s). In at least one embodiment, ACC system may include a longitudinal ACC system and/or a lateral ACC system. In at least one embodiment, longitudinal ACC system monitors and controls distance to vehicle immediately ahead of vehicle 1100 and automatically adjust speed of vehicle 1100 to maintain a safe
35 distance from vehicles ahead. In at least one embodiment, lateral ACC system performs

5 distance keeping, and advises vehicle 1100 to change lanes when necessary. In at least one embodiment, lateral ACC is related to other ADAS applications such as LC and CW.

[00205] In at least one embodiment, CACC system uses information from other vehicles that may be received via network interface 1124 and/or wireless antenna(s) 1126 from other vehicles via a wireless link, or indirectly, over a network connection (e.g., over Internet). In
10 at least one embodiment, direct links may be provided by a vehicle-to-vehicle (“V2V”) communication link, while indirect links may be provided by an infrastructure-to-vehicle (“I2V”) communication link. In general, V2V communication concept provides information about immediately preceding vehicles (e.g., vehicles immediately ahead of and in same lane as vehicle 1100), while I2V communication concept provides information about traffic
15 further ahead. In at least one embodiment, CACC system may include either or both I2V and V2V information sources. In at least one embodiment, given information of vehicles ahead of vehicle 1100, CACC system may be more reliable and it has potential to improve traffic flow smoothness and reduce congestion on road.

[00206] In at least one embodiment, FCW system is designed to alert driver to a hazard, so
20 that driver may take corrective action. In at least one embodiment, FCW system uses a front-facing camera and/or RADAR sensor(s) 1160, coupled to a dedicated processor, DSP, FPGA, and/or ASIC, that is electrically coupled to driver feedback, such as a display, speaker, and/or vibrating component. In at least one embodiment, FCW system may provide a warning, such as in form of a sound, visual warning, vibration and/or a quick brake pulse.

[00207] In at least one embodiment, AEB system detects an impending forward collision with another vehicle or other object, and may automatically apply brakes if driver does not take corrective action within a specified time or distance parameter. In at least one
25 embodiment, AEB system may use front-facing camera(s) and/or RADAR sensor(s) 1160, coupled to a dedicated processor, DSP, FPGA, and/or ASIC. In at least one embodiment, when AEB system detects a hazard, AEB system typically first alerts driver to take corrective
30 action to avoid collision and, if driver does not take corrective action, AEB system may automatically apply brakes in an effort to prevent, or at least mitigate, impact of predicted collision. In at least one embodiment, AEB system, may include techniques such as dynamic brake support and/or crash imminent braking.

5 [00208] In at least one embodiment, LDW system provides visual, audible, and/or tactile warnings, such as steering wheel or seat vibrations, to alert driver when vehicle 1100 crosses lane markings. In at least one embodiment, LDW system does not activate when driver indicates an intentional lane departure, by activating a turn signal. In at least one embodiment, LDW system may use front-side facing cameras, coupled to a dedicated
10 processor, DSP, FPGA, and/or ASIC, that is electrically coupled to driver feedback, such as a display, speaker, and/or vibrating component. In at least one embodiment, LKA system is a variation of LDW system. LKA system provides steering input or braking to correct vehicle 1100 if vehicle 1100 starts to exit lane.

[00209] In at least one embodiment, BSW system detects and warns driver of vehicles in
15 an automobile's blind spot. In at least one embodiment, BSW system may provide a visual, audible, and/or tactile alert to indicate that merging or changing lanes is unsafe. In at least one embodiment, BSW system may provide an additional warning when driver uses a turn signal. In at least one embodiment, BSW system may use rear-side facing camera(s) and/or RADAR sensor(s) 1160, coupled to a dedicated processor, DSP, FPGA, and/or ASIC, that is
20 electrically coupled to driver feedback, such as a display, speaker, and/or vibrating component.

[00210] In at least one embodiment, RCTW system may provide visual, audible, and/or tactile notification when an object is detected outside rear-camera range when vehicle 1100 is backing up. In at least one embodiment, RCTW system includes AEB system to ensure that
25 vehicle brakes are applied to avoid a crash. In at least one embodiment, RCTW system may use one or more rear-facing RADAR sensor(s) 1160, coupled to a dedicated processor, DSP, FPGA, and/or ASIC, that is electrically coupled to driver feedback, such as a display, speaker, and/or vibrating component.

[00211] In at least one embodiment, conventional ADAS systems may be prone to false
30 positive results which may be annoying and distracting to a driver, but typically are not catastrophic, because conventional ADAS systems alert driver and allow driver to decide whether a safety condition truly exists and act accordingly. In at least one embodiment, vehicle 1100 itself decides, in case of conflicting results, whether to heed result from a primary computer or a secondary computer (e.g., first controller 1136 or second controller
35 1136). For example, in at least one embodiment, ADAS system 1138 may be a backup and/or secondary computer for providing perception information to a backup computer

5 rationality module. In at least one embodiment, backup computer rationality monitor may run a redundant diverse software on hardware components to detect faults in perception and dynamic driving tasks. In at least one embodiment, outputs from ADAS system 1138 may be provided to a supervisory MCU. In at least one embodiment, if outputs from primary computer and secondary computer conflict, supervisory MCU determines how to reconcile
10 conflict to ensure safe operation.

[00212] In at least one embodiment, primary computer may be configured to provide supervisory MCU with a confidence score, indicating primary computer's confidence in chosen result. In at least one embodiment, if confidence score exceeds a threshold, supervisory MCU may follow primary computer's direction, regardless of whether secondary
15 computer provides a conflicting or inconsistent result. In at least one embodiment, where confidence score does not meet threshold, and where primary and secondary computer indicate different results (e.g., a conflict), supervisory MCU may arbitrate between computers to determine appropriate outcome.

[00213] In at least one embodiment, supervisory MCU may be configured to run a neural
20 network(s) that is trained and configured to determine, based at least in part on outputs from primary computer and secondary computer, conditions under which secondary computer provides false alarms. In at least one embodiment, neural network(s) in supervisory MCU may learn when secondary computer's output may be trusted, and when it cannot. For example, in at least one embodiment, when secondary computer is a RADAR-based FCW
25 system, a neural network(s) in supervisory MCU may learn when FCW system is identifying metallic objects that are not, in fact, hazards, such as a drainage grate or manhole cover that triggers an alarm. In at least one embodiment, when secondary computer is a camera-based LDW system, a neural network in supervisory MCU may learn to override LDW when bicyclists or pedestrians are present and a lane departure is, in fact, safest maneuver. In at
30 least one embodiment, supervisory MCU may include at least one of a DLA or GPU suitable for running neural network(s) with associated memory. In at least one embodiment, supervisory MCU may comprise and/or be included as a component of SoC(s) 1104.

[00214] In at least one embodiment, ADAS system 1138 may include a secondary computer that performs ADAS functionality using traditional rules of computer vision. In at
35 least one embodiment, secondary computer may use classic computer vision rules (if-then), and presence of a neural network(s) in supervisory MCU may improve reliability, safety and

5 performance. For example, in at least one embodiment, diverse implementation and intentional non-identity makes overall system more fault-tolerant, especially to faults caused by software (or software-hardware interface) functionality. For example, in at least one embodiment, if there is a software bug or error in software running on primary computer, and non-identical software code running on secondary computer provides same overall result,
10 then supervisory MCU may have greater confidence that overall result is correct, and bug in software or hardware on primary computer is not causing material error.

[00215] In at least one embodiment, output of ADAS system 1138 may be fed into primary computer's perception block and/or primary computer's dynamic driving task block. For example, in at least one embodiment, if ADAS system 1138 indicates a forward crash
15 warning due to an object immediately ahead, perception block may use this information when identifying objects. In at least one embodiment, secondary computer may have its own neural network which is trained and thus reduces risk of false positives, as described herein.

[00216] In at least one embodiment, vehicle 1100 may further include infotainment SoC 1130 (e.g., an in-vehicle infotainment system (IVI)). Although illustrated and described as an
20 SoC, infotainment system 1130, in at least one embodiment, may not be an SoC, and may include, without limitation, two or more discrete components. In at least one embodiment, infotainment SoC 1130 may include, without limitation, a combination of hardware and software that may be used to provide audio (e.g., music, a personal digital assistant, navigational instructions, news, radio, etc.), video (e.g., TV, movies, streaming, etc.), phone
25 (e.g., hands-free calling), network connectivity (e.g., LTE, WiFi, etc.), and/or information services (e.g., navigation systems, rear-parking assistance, a radio data system, vehicle related information such as fuel level, total distance covered, brake fuel level, oil level, door open/close, air filter information, etc.) to vehicle 1100. For example, infotainment SoC 1130 could include radios, disk players, navigation systems, video players, USB and Bluetooth
30 connectivity, carputers, in-car entertainment, WiFi, steering wheel audio controls, hands free voice control, a heads-up display ("HUD"), HMI display 1134, a telematics device, a control panel (e.g., for controlling and/or interacting with various components, features, and/or systems), and/or other components. In at least one embodiment, infotainment SoC 1130 may further be used to provide information (e.g., visual and/or audible) to user(s) of vehicle, such
35 as information from ADAS system 1138, autonomous driving information such as planned

5 vehicle maneuvers, trajectories, surrounding environment information (e.g., intersection information, vehicle information, road information, etc.), and/or other information.

[00217] In at least one embodiment, infotainment SoC 1130 may include any amount and type of GPU functionality. In at least one embodiment, infotainment SoC 1130 may communicate over bus 1102 (e.g., CAN bus, Ethernet, etc.) with other devices, systems,
10 and/or components of vehicle 1100. In at least one embodiment, infotainment SoC 1130 may be coupled to a supervisory MCU such that GPU of infotainment system may perform some self-driving functions in event that primary controller(s) 1136 (e.g., primary and/or backup computers of vehicle 1100) fail. In at least one embodiment, infotainment SoC 1130 may put vehicle 1100 into a chauffeur to safe stop mode, as described herein.

15 [00218] In at least one embodiment, vehicle 1100 may further include instrument cluster 1132 (e.g., a digital dash, an electronic instrument cluster, a digital instrument panel, etc.). Instrument cluster 1132 may include, without limitation, a controller and/or supercomputer (e.g., a discrete controller or supercomputer). In at least one embodiment, instrument cluster 1132 may include, without limitation, any number and combination of a set of
20 instrumentation such as a speedometer, fuel level, oil pressure, tachometer, odometer, turn indicators, gearshift position indicator, seat belt warning light(s), parking-brake warning light(s), engine-malfunction light(s), supplemental restraint system (e.g., airbag) information, lighting controls, safety system controls, navigation information, etc. In some examples, information may be displayed and/or shared among infotainment SoC 1130 and instrument
25 cluster 1132. In at least one embodiment, instrument cluster 1132 may be included as part of infotainment SoC 1130, or vice versa.

[00219] FIG. 11D is a diagram of a system 1176 for communication between cloud-based server(s) and autonomous vehicle 1100 of FIG. 11A, according to at least one embodiment. In at least one embodiment, system 1176 may include, without limitation, server(s) 1178,
30 network(s) 1190, and any number and type of vehicles, including vehicle 1100. server(s) 1178 may include, without limitation, a plurality of GPUs 1184(A)-1184(H) (collectively referred to herein as GPUs 1184), PCIe switches 1182(A)-1182(H) (collectively referred to herein as PCIe switches 1182), and/or CPUs 1180(A)-1180(B) (collectively referred to herein as CPUs 1180).
35 GPUs 1184, CPUs 1180, and PCIe switches 1182 may be interconnected with high-speed interconnects such as, for example and without limitation, NVLink interfaces 1188 developed

5 by NVIDIA and/or PCIe connections 1186. In at least one embodiment, GPUs 1184 are connected via an NVLink and/or NVSwitch SoC and GPUs 1184 and PCIe switches 1182 are connected via PCIe interconnects. In at least one embodiment, although eight GPUs 1184, two CPUs 1180, and four PCIe switches 1182 are illustrated, this is not intended to be limiting. In at least one embodiment, each of server(s) 1178 may include, without limitation,
10 any number of GPUs 1184, CPUs 1180, and/or PCIe switches 1182, in any combination. For example, in at least one embodiment, server(s) 1178 could each include eight, sixteen, thirty-two, and/or more GPUs 1184.

[00220] In at least one embodiment, server(s) 1178 may receive, over network(s) 1190 and from vehicles, image data representative of images showing unexpected or changed road
15 conditions, such as recently commenced road-work. In at least one embodiment, server(s) 1178 may transmit, over network(s) 1190 and to vehicles, neural networks 1192, updated neural networks 1192, and/or map information 1194, including, without limitation, information regarding traffic and road conditions. In at least one embodiment, updates to map information 1194 may include, without limitation, updates for HD map 1122, such as
20 information regarding construction sites, potholes, detours, flooding, and/or other obstructions. In at least one embodiment, neural networks 1192, updated neural networks 1192, and/or map information 1194 may have resulted from new training and/or experiences represented in data received from any number of vehicles in environment, and/or based at least in part on training performed at a data center (e.g., using server(s) 1178 and/or other
25 servers).

[00221] In at least one embodiment, server(s) 1178 may be used to train machine learning models (e.g., neural networks) based at least in part on training data. Training data may be generated by vehicles, and/or may be generated in a simulation (e.g., using a game engine). In at least one embodiment, any amount of training data is tagged (e.g., where associated
30 neural network benefits from supervised learning) and/or undergoes other pre-processing. In at least one embodiment, any amount of training data is not tagged and/or pre-processed (e.g., where associated neural network does not require supervised learning). In at least one embodiment, once machine learning models are trained, machine learning models may be used by vehicles (e.g., transmitted to vehicles over network(s) 1190, and/or machine learning
35 models may be used by server(s) 1178 to remotely monitor vehicles.

5 [00222] In at least one embodiment, server(s) 1178 may receive data from vehicles and apply data to up-to-date real-time neural networks for real-time intelligent inferencing. In at least one embodiment, server(s) 1178 may include deep-learning supercomputers and/or dedicated AI computers powered by GPU(s) 1184, such as a DGX and DGX Station machines developed by NVIDIA. However, in at least one embodiment, server(s) 1178 may
10 include deep learning infrastructure that use CPU-powered data centers.

[00223] In at least one embodiment, deep-learning infrastructure of server(s) 1178 may be capable of fast, real-time inferencing, and may use that capability to evaluate and verify health of processors, software, and/or associated hardware in vehicle 1100. For example, in at least one embodiment, deep-learning infrastructure may receive periodic updates from
15 vehicle 1100, such as a sequence of images and/or objects that vehicle 1100 has located in that sequence of images (e.g., via computer vision and/or other machine learning object classification techniques). In at least one embodiment, deep-learning infrastructure may run its own neural network to identify objects and compare them with objects identified by vehicle 1100 and, if results do not match and deep-learning infrastructure concludes that AI
20 in vehicle 1100 is malfunctioning, then server(s) 1178 may transmit a signal to vehicle 1100 instructing a fail-safe computer of vehicle 1100 to assume control, notify passengers, and complete a safe parking maneuver.

[00224] In at least one embodiment, server(s) 1178 may include GPU(s) 1184 and one or more programmable inference accelerators (e.g., NVIDIA's TensorRT 3). In at least one
25 embodiment, combination of GPU-powered servers and inference acceleration may make real-time responsiveness possible. In at least one embodiment, such as where performance is less critical, servers powered by CPUs, FPGAs, and other processors may be used for inferencing. In at least one embodiment, hardware structure(s) 1015 are used to perform one or more embodiments. Details regarding hardware structure(x) 1015 are provided herein in
30 conjunction with FIGs. 10A and/or 10B.

COMPUTER SYSTEMS

[00225] FIG. 12 is a block diagram illustrating an exemplary computer system, which may be a system with interconnected devices and components, a system-on-a-chip (SOC) or some combination thereof 1200 formed with a processor that may include execution units to
35 execute an instruction, according to at least one embodiment. In at least one embodiment, computer system 1200 may include, without limitation, a component, such as a processor

5 1202 to employ execution units including logic to perform algorithms for process data, in accordance with present disclosure, such as in embodiment described herein. In at least one embodiment, computer system 1200 may include processors, such as PENTIUM® Processor family, Xeon™, Itanium®, XScale™ and/or StrongARM™, Intel® Core™, or Intel® Nervana™ microprocessors available from Intel Corporation of Santa Clara, California,
10 although other systems (including PCs having other microprocessors, engineering workstations, set-top boxes and like) may also be used. In at least one embodiment, computer system 1200 may execute a version of WINDOWS' operating system available from Microsoft Corporation of Redmond, Wash., although other operating systems (UNIX and Linux for example), embedded software, and/or graphical user interfaces, may also be
15 used.

[00226] Embodiments may be used in other devices such as handheld devices and embedded applications. Some examples of handheld devices include cellular phones, Internet Protocol devices, digital cameras, personal digital assistants ("PDAs"), and handheld PCs. In at least one embodiment, embedded applications may include a microcontroller, a
20 digital signal processor ("DSP"), system on a chip, network computers ("NetPCs"), set-top boxes, network hubs, wide area network ("WAN") switches, or any other system that may perform one or more instructions in accordance with at least one embodiment.

[00227] In at least one embodiment, computer system 1200 may include, without limitation, processor 1202 that may include, without limitation, one or more execution units
25 1208 to perform machine learning model training and/or inferencing according to techniques described herein. In at least one embodiment, system 1200 is a single processor desktop or server system, but in another embodiment system 1200 may be a multiprocessor system. In at least one embodiment, processor 1202 may include, without limitation, a complex instruction set computer ("CISC") microprocessor, a reduced instruction set computing
30 ("RISC") microprocessor, a very long instruction word ("VLIW") microprocessor, a processor implementing a combination of instruction sets, or any other processor device, such as a digital signal processor, for example. In at least one embodiment, processor 1202 may be coupled to a processor bus 1210 that may transmit data signals between processor 1202 and other components in computer system 1200.

35 [00228] In at least one embodiment, processor 1202 may include, without limitation, a Level 1 ("L1") internal cache memory ("cache") 1204. In at least one embodiment, processor

5 1202 may have a single internal cache or multiple levels of internal cache. In at least one embodiment, cache memory may reside external to processor 1202. Other embodiments may also include a combination of both internal and external caches depending on particular implementation and needs. In at least one embodiment, register file 1206 may store different types of data in various registers including, without limitation, integer registers, floating point
10 registers, status registers, and instruction pointer register.

[00229] In at least one embodiment, execution unit 1208, including, without limitation, logic to perform integer and floating point operations, also resides in processor 1202. Processor 1202 may also include a microcode (“ucode”) read only memory (“ROM”) that stores microcode for certain macro instructions. In at least one embodiment, execution unit
15 1208 may include logic to handle a packed instruction set 1209. In at least one embodiment, by including packed instruction set 1209 in instruction set of a general-purpose processor 1202, along with associated circuitry to execute instructions, operations used by many multimedia applications may be performed using packed data in a general-purpose processor 1202. In one or more embodiments, many multimedia applications may be accelerated and
20 executed more efficiently by using full width of a processor's data bus for performing operations on packed data, which may eliminate need to transfer smaller units of data across processor's data bus to perform one or more operations one data element at a time.

[00230] In at least one embodiment, execution unit 1208 may also be used in microcontrollers, embedded processors, graphics devices, DSPs, and other types of logic
25 circuits. In at least one embodiment, computer system 1200 may include, without limitation, a memory 1220. In at least one embodiment, memory 1220 may be implemented as a Dynamic Random Access Memory (“DRAM”) device, a Static Random Access Memory (“SRAM”) device, flash memory device, or other memory device. Memory 1220 may store instruction(s) 1219 and/or data 1221 represented by data signals that may be executed by
30 processor 1202.

[00231] In at least one embodiment, system logic chip may be coupled to processor bus 1210 and memory 1220. In at least one embodiment, system logic chip may include, without limitation, a memory controller hub (“MCH”) 1216, and processor 1202 may communicate with MCH 1216 via processor bus 1210. In at least one embodiment, MCH 1216 may
35 provide a high bandwidth memory path 1218 to memory 1220 for instruction and data storage and for storage of graphics commands, data and textures. In at least one embodiment, MCH

5 1216 may direct data signals between processor 1202, memory 1220, and other components in computer system 1200 and to bridge data signals between processor bus 1210, memory 1220, and a system I/O 1222. In at least one embodiment, system logic chip may provide a graphics port for coupling to a graphics controller. In at least one embodiment, MCH 1216 may be coupled to memory 1220 through a high bandwidth memory path 1218 and
10 graphics/video card 1212 may be coupled to MCH 1216 through an Accelerated Graphics Port (“AGP”) interconnect 1214.

[00232] In at least one embodiment, computer system 1200 may use system I/O 1222 that is a proprietary hub interface bus to couple MCH 1216 to I/O controller hub (“ICH”) 1230. In at least one embodiment, ICH 1230 may provide direct connections to some I/O devices
15 via a local I/O bus. In at least one embodiment, local I/O bus may include, without limitation, a high-speed I/O bus for connecting peripherals to memory 1220, chipset, and processor 1202. Examples may include, without limitation, an audio controller 1229, a firmware hub (“flash BIOS”) 1228, a wireless transceiver 1226, a data storage 1224, a legacy I/O controller 1223 containing user input and keyboard interfaces, a serial expansion port
20 1227, such as Universal Serial Bus (“USB”), and a network controller 1234. Data storage 1224 may comprise a hard disk drive, a floppy disk drive, a CD-ROM device, a flash memory device, or other mass storage device.

[00233] In at least one embodiment, FIG. 12 illustrates a system, which includes interconnected hardware devices or “chips”, whereas in other embodiments, FIG. 12 may
25 illustrate an exemplary System on a Chip (“SoC”). In at least one embodiment, devices illustrated in FIG. 12 may be interconnected with proprietary interconnects, standardized interconnects (e.g., PCIe) or some combination thereof. In at least one embodiment, one or more components of system 1200 are interconnected using compute express link (CXL) interconnects.

30 [00234] FIG. 13 is a block diagram illustrating an electronic device 1300 for utilizing a processor 1310, according to at least one embodiment. In at least one embodiment, electronic device 1300 may be, for example and without limitation, a notebook, a tower server, a rack server, a blade server, a laptop, a desktop, a tablet, a mobile device, a phone, an embedded computer, or any other suitable electronic device.

5 [00235] In at least one embodiment, system 1300 may include, without limitation, processor 1310 communicatively coupled to any suitable number or kind of components, peripherals, modules, or devices. In at least one embodiment, processor 1310 coupled using a bus or interface, such as a I²C bus, a System Management Bus (“SMBus”), a Low Pin Count (LPC) bus, a Serial Peripheral Interface (“SPI”), a High Definition Audio (“HDA”) bus, a
 10 Serial Advance Technology Attachment (“SATA”) bus, a Universal Serial Bus (“USB”) (versions 1, 2, 3), or a Universal Asynchronous Receiver/Transmitter (“UART”) bus. In at least one embodiment, FIG. 13 illustrates a system, which includes interconnected hardware devices or “chips”, whereas in other embodiments, FIG. 13 may illustrate an exemplary System on a Chip (“SoC”). In at least one embodiment, devices illustrated in FIG. 13 may be
 15 interconnected with proprietary interconnects, standardized interconnects (e.g., PCIe) or some combination thereof. In at least one embodiment, one or more components of FIG. 13 are interconnected using compute express link (CXL) interconnects.

[00236] In at least one embodiment, FIG 13 may include a display 1324, a touch screen 1325, a touch pad 1330, a Near Field Communications unit (“NFC”) 1345, a sensor hub
 20 1340, a thermal sensor 1346, an Express Chipset (“EC”) 1335, a Trusted Platform Module (“TPM”) 1338, BIOS/firmware/flash memory (“BIOS, FW Flash”) 1322, a DSP 1360, a drive “SSD or HDD”) 1320 such as a Solid State Disk (“SSD”) or a Hard Disk Drive (“HDD”), a wireless local area network unit (“WLAN”) 1350, a Bluetooth unit 1352, a
 Wireless Wide Area Network unit (“WWAN”) 1356, a Global Positioning System (GPS)
 25 1355, a camera (“USB 3.0 camera”) 1354 such as a USB 3.0 camera, or a Low Power Double Data Rate (“LPDDR”) memory unit (“LPDDR3”) 1315 implemented in, for example, LPDDR3 standard. These components may each be implemented in any suitable manner.

[00237] In at least one embodiment, other components may be communicatively coupled to processor 1310 through components discussed above. In at least one embodiment, an
 30 accelerometer 1341, Ambient Light Sensor (“ALS”) 1342, compass 1343, and a gyroscope 1344 may be communicatively coupled to sensor hub 1340. In at least one embodiment, thermal sensor 1339, a fan 1337, a keyboard 1346, and a touch pad 1330 may be communicatively coupled to EC 1335. In at least one embodiment, speaker 1363, a
 headphones 1364, and a microphone (“mic”) 1365 may be communicatively coupled to an
 35 audio unit (“audio codec and class d amp”) 1364, which may in turn be communicatively coupled to DSP 1360. In at least one embodiment, audio unit 1364 may include, for example

5 and without limitation, an audio coder/decoder (“codec”) and a class D amplifier. In at least one embodiment, SIM card (“SIM”) 1357 may be communicatively coupled to WWAN unit 1356. In at least one embodiment, components such as WLAN unit 1350 and Bluetooth unit 1352, as well as WWAN unit 1356 may be implemented in a Next Generation Form Factor (“NGFF”).

10 [00238] FIG. 14 illustrates a computer system 1400, according to at least one embodiment. In at least one embodiment, computer system 1400 is configured to implement various processes and methods described throughout this disclosure.

[00239] In at least one embodiment, computer system 1400 comprises, without limitation, at least one central processing unit (“CPU”) 1402 that is connected to a communication bus
15 1410 implemented using any suitable protocol, such as PCI (“Peripheral Component Interconnect”), peripheral component interconnect express (“PCI-Express”), AGP (“Accelerated Graphics Port”), HyperTransport, or any other bus or point-to-point communication protocol(s). In at least one embodiment, computer system 1400 includes, without limitation, a main memory 1404 and control logic (e.g., implemented as hardware,
20 software, or a combination thereof) and data are stored in main memory 1404 which may take form of random access memory (“RAM”). In at least one embodiment, a network interface subsystem (“network interface”) 1422 provides an interface to other computing devices and networks for receiving data from and transmitting data to other systems from computer system 1400.

25 [00240] In at least one embodiment, computer system 1400, in at least one embodiment, includes, without limitation, input devices 1408, parallel processing system 1412, and display devices 1406 which can be implemented using a conventional cathode ray tube (“CRT”), liquid crystal display (“LCD”), light emitting diode (“LED”), plasma display, or other suitable display technologies. In at least one embodiment, user input is received from input
30 devices 1408 such as keyboard, mouse, touchpad, microphone, and more. In at least one embodiment, each of foregoing modules can be situated on a single semiconductor platform to form a processing system.

[00241] FIG. 15 illustrates a computer system 1500, according to at least one embodiment. In at least one embodiment, computer system 1500 includes, without limitation, a computer
35 1510 and a USB stick 1520. In at least one embodiment, computer 1510 may include,

5 without limitation, any number and type of processor(s) (not shown) and a memory (not shown). In at least one embodiment, computer 1510 includes, without limitation, a server, a cloud instance, a laptop, and a desktop computer.

[00242] In at least one embodiment, USB stick 1520 includes, without limitation, a processing unit 1530, a USB interface 1540, and USB interface logic 1550. In at least one
10 embodiment, processing unit 1530 may be any instruction execution system, apparatus, or device capable of executing instructions. In at least one embodiment, processing unit 1530 may include, without limitation, any number and type of processing cores (not shown). In at least one embodiment, processing core 1530 comprises an application specific integrated circuit (“ASIC”) that is optimized to perform any amount and type of operations associated
15 with machine learning. For instance, in at least one embodiment, processing core 1530 is a tensor processing unit (“TPC”) that is optimized to perform machine learning inference operations. In at least one embodiment, processing core 1530 is a vision processing unit (“VPU”) that is optimized to perform machine vision and machine learning inference operations.

20 [00243] In at least one embodiment, USB interface 1540 may be any type of USB connector or USB socket. For instance, in at least one embodiment, USB interface 1540 is a USB 3.0 Type-C socket for data and power. In at least one embodiment, USB interface 1540 is a USB 3.0 Type-A connector. In at least one embodiment, USB interface logic 1550 may include any amount and type of logic that enables processing unit 1530 to interface with or
25 devices (e.g., computer 1510) via USB connector 1540.

[00244] FIG. 16A illustrates an exemplary architecture in which a plurality of GPUs 1610-1613 is communicatively coupled to a plurality of multi-core processors 1605-1606 over high-speed links 1640-1643 (e.g., buses, point-to-point interconnects, etc.). In one
embodiment, high-speed links 1640-1643 support a communication throughput of 4GB/s,
30 30GB/s, 80GB/s or higher. Various interconnect protocols may be used including, but not limited to, PCIe 4.0 or 5.0 and NVLink 2.0.

[00245] In addition, and in one embodiment, two or more of GPUs 1610-1613 are interconnected over high-speed links 1629-1630, which may be implemented using same or different protocols/links than those used for high-speed links 1640-1643. Similarly, two or
35 more of multi-core processors 1605-1606 may be connected over high speed link 1628 which

5 may be symmetric multi-processor (SMP) buses operating at 20GB/s, 30GB/s, 120GB/s or higher. Alternatively, all communication between various system components shown in FIG. 16A may be accomplished using same protocols/links (e.g., over a common interconnection fabric).

[00246] In one embodiment, each multi-core processor 1605-1606 is communicatively
10 coupled to a processor memory 1601-1602, via memory interconnects 1626-1627,
respectively, and each GPU 1610-1613 is communicatively coupled to GPU memory 1620-
1623 over GPU memory interconnects 1650-1653, respectively. Memory interconnects 1626-
1627 and 1650-1653 may utilize same or different memory access technologies. By way of
example, and not limitation, processor memories 1601-1602 and GPU memories 1620-1623
15 may be volatile memories such as dynamic random access memories (DRAMs) (including
stacked DRAMs), Graphics DDR SDRAM (GDDR) (e.g., GDDR5, GDDR6), or High
Bandwidth Memory (HBM) and/or may be non-volatile memories such as 3D XPoint or
Nano-Ram. In one embodiment, some portion of processor memories 1601-1602 may be
volatile memory and another portion may be non-volatile memory (e.g., using a two-level
20 memory (2LM) hierarchy).

[00247] As described herein, although various processors 1605-1606 and GPUs 1610-1613
may be physically coupled to a particular memory 1601-1602, 1620-1623, respectively, a
unified memory architecture may be implemented in which a same virtual system address
space (also referred to as “effective address” space) is distributed among various physical
25 memories. For example, processor memories 1601-1602 may each comprise 64GB of system
memory address space and GPU memories 1620-1623 may each comprise 32GB of system
memory address space (resulting in a total of 256GB addressable memory in this example).

[00248] FIG. 16B illustrates additional details for an interconnection between a multi-core
processor 1607 and a graphics acceleration module 1646 in accordance with one exemplary
30 embodiment. Graphics acceleration module 1646 may include one or more GPU chips
integrated on a line card which is coupled to processor 1607 via high-speed link 1640.
Alternatively, graphics acceleration module 1646 may be integrated on a same package or
chip as processor 1607.

[00249] In at least one embodiment, illustrated processor 1607 includes a plurality of
35 cores 1660A-1660D, each with a translation lookaside buffer 1661A-1661D and one or more

5 caches 1662A-1662D. In at least one embodiment, cores 1660A-1660D may include various other components for executing instructions and processing data which are not illustrated. Caches 1662A-1662D may comprise level 1 (L1) and level 2 (L2) caches. In addition, one or more shared caches 1656 may be included in caches 1662A-1662D and shared by sets of cores 1660A-1660D. For example, one embodiment of processor 1607 includes 24 cores,
10 each with its own L1 cache, twelve shared L2 caches, and twelve shared L3 caches. In this embodiment, one or more L2 and L3 caches are shared by two adjacent cores. Processor 1607 and graphics acceleration module 1646 connect with system memory 1614, which may include processor memories 1601-1602 of FIG. 16A.

[00250] Coherency is maintained for data and instructions stored in various
15 caches 1662A-1662D, 1656 and system memory 1614 via inter-core communication over a coherence bus 1664. For example, each cache may have cache coherency logic/circuitry associated therewith to communicate to over coherence bus 1664 in response to detected reads or writes to particular cache lines. In one implementation, a cache snooping protocol is implemented over coherence bus 1664 to snoop cache accesses.

20 [00251] In one embodiment, a proxy circuit 1625 communicatively couples graphics acceleration module 1646 to coherence bus 1664, allowing graphics acceleration module 1646 to participate in a cache coherence protocol as a peer of cores 1660A-1660D. In particular, an interface 1635 provides connectivity to proxy circuit 1625 over high-speed link 1640 (e.g., a PCIe bus, NVLink, etc.) and an interface 1637 connects graphics acceleration
25 module 1646 to link 1640.

[00252] In one implementation, an accelerator integration circuit 1636 provides cache management, memory access, context management, and interrupt management services on behalf of a plurality of graphics processing engines 1631, 1632, N of graphics acceleration module 1646. Graphics processing engines 1631, 1632, N may each comprise a separate
30 graphics processing unit (GPU). Alternatively, graphics processing engines 1631, 1632, N may comprise different types of graphics processing engines within a GPU such as graphics execution units, media processing engines (e.g., video encoders/decoders), samplers, and block image transfer (BLIT) engines. In at least one embodiment, graphics acceleration module 1646 may be a GPU with a plurality of graphics processing engines 1631-1632, N or
35 graphics processing engines 1631-1632, N may be individual GPUs integrated on a common package, line card, or chip.

5 [00253] In one embodiment, accelerator integration circuit 1636 includes a memory management unit (MMU) 1639 for performing various memory management functions such as virtual-to-physical memory translations (also referred to as effective-to-real memory translations) and memory access protocols for accessing system memory 1614. MMU 1639 may also include a translation lookaside buffer (TLB) (not shown) for caching
10 virtual/effective to physical/real address translations. In one implementation, a cache 1638 stores commands and data for efficient access by graphics processing engines 1631-1632, N. In one embodiment, data stored in cache 1638 and graphics memories 1633-1634, M is kept coherent with core caches 1662A-1662D, 1656 and system memory 1614. As mentioned, this may be accomplished via proxy circuit 1625 on behalf of cache 1638 and memories
15 1633-1634, M (e.g., sending updates to cache 1638 related to modifications/accesses of cache lines on processor caches 1662A-1662D, 1656 and receiving updates from cache 1638).

[00254] A set of registers 1645 store context data for threads executed by graphics processing engines 1631-1632, N and a context management circuit 1648 manages thread contexts. For example, context management circuit 1648 may perform save and restore
20 operations to save and restore contexts of various threads during contexts switches (e.g., where a first thread is saved and a second thread is stored so that a second thread can be execute by a graphics processing engine). For example, on a context switch, context management circuit 1648 may store current register values to a designated region in memory (e.g., identified by a context pointer). It may then restore register values when returning to a
25 context. In one embodiment, an interrupt management circuit 1647 receives and processes interrupts received from system devices.

[00255] In one implementation, virtual/effective addresses from a graphics processing engine 1631 are translated to real/physical addresses in system memory 1614 by MMU 1639. One embodiment of accelerator integration circuit 1636 supports multiple (e.g., 4, 8, 16)
30 graphics accelerator modules 1646 and/or other accelerator devices. Graphics accelerator module 1646 may be dedicated to a single application executed on processor 1607 or may be shared between multiple applications. In one embodiment, a virtualized graphics execution environment is presented in which resources of graphics processing engines 1631-1632, N are shared with multiple applications or virtual machines (VMs). In at least one embodiment,
35 resources may be subdivided into “slices” which are allocated to different VMs and/or

5 applications based on processing requirements and priorities associated with VMs and/or applications.

[00256] In at least one embodiment, accelerator integration circuit 1636 performs as a bridge to a system for graphics acceleration module 1646 and provides address translation and system memory cache services. In addition, accelerator integration circuit 1636 may
10 provide virtualization facilities for a host processor to manage virtualization of graphics processing engines 1631-1632, interrupts, and memory management.

[00257] Because hardware resources of graphics processing engines 1631-1632, N are mapped explicitly to a real address space seen by host processor 1607, any host processor can address these resources directly using an effective address value. One function of accelerator
15 integration circuit 1636, in one embodiment, is physical separation of graphics processing engines 1631-1632, N so that they appear to a system as independent units.

[00258] In at least one embodiment, one or more graphics memories 1633-1634, M are coupled to each of graphics processing engines 1631-1632, N, respectively. Graphics memories 1633-1634, M store instructions and data being processed by each of graphics
20 processing engines 1631-1632, N. Graphics memories 1633-1634, M may be volatile memories such as DRAMs (including stacked DRAMs), GDDR memory (e.g., GDDR5, GDDR6), or HBM, and/or may be non-volatile memories such as 3D XPoint or Nano-Ram.

[00259] In one embodiment, to reduce data traffic over link 1640, biasing techniques are used to ensure that data stored in graphics memories 1633-1634, M is data which will be used
25 most frequently by graphics processing engines 1631-1632, N and preferably not used by cores 1660A-1660D (at least not frequently). Similarly, a biasing mechanism attempts to keep data needed by cores (and preferably not graphics processing engines 1631-1632, N) within caches 1662A-1662D, 1656 of cores and system memory 1614.

[00260] FIG. 16C illustrates another exemplary embodiment in which accelerator
30 integration circuit 1636 is integrated within processor 1607. In this embodiment, graphics processing engines 1631-1632, N communicate directly over high-speed link 1640 to accelerator integration circuit 1636 via interface 1637 and interface 1635 (which, again, may be utilize any form of bus or interface protocol). Accelerator integration circuit 1636 may perform same operations as those described with respect to FIG. 16B, but potentially at a
35 higher throughput given its close proximity to coherence bus 1664 and caches 1662A-1662D,

5 1656. One embodiment supports different programming models including a dedicated-process programming model (no graphics acceleration module virtualization) and shared programming models (with virtualization), which may include programming models which are controlled by accelerator integration circuit 1636 and programming models which are controlled by graphics acceleration module 1646.

10 [00261] In at least one embodiment, graphics processing engines 1631-1632, N are dedicated to a single application or process under a single operating system. In at least one embodiment, a single application can funnel other application requests to graphics processing engines 1631-1632, N, providing virtualization within a VM/partition.

[00262] In at least one embodiment, graphics processing engines 1631-1632, N, may be
15 shared by multiple VM/application partitions. In at least one embodiment, shared models may use a system hypervisor to virtualize graphics processing engines 1631-1632, N to allow access by each operating system. For single-partition systems without a hypervisor, graphics processing engines 1631-1632, N are owned by an operating system. In at least one embodiment, an operating system can virtualize graphics processing engines 1631-1632, N to
20 provide access to each process or application.

[00263] In at least one embodiment, graphics acceleration module 1646 or an individual graphics processing engine 1631-1632, N selects a process element using a process handle. In one embodiment, process elements are stored in system memory 1614 and are addressable using an effective address to real address translation techniques described herein. In at least
25 one embodiment, a process handle may be an implementation-specific value provided to a host process when registering its context with graphics processing engine 1631-1632, N (that is, calling system software to add a process element to a process element linked list). In at least one embodiment, a lower 16-bits of a process handle may be an offset of the process element within a process element linked list.

30 [00264] FIG. 16D illustrates an exemplary accelerator integration slice 1690. As used herein, a “slice” comprises a specified portion of processing resources of accelerator integration circuit 1636. Application effective address space 1682 within system memory 1614 stores process elements 1683. In one embodiment, process elements 1683 are stored in response to GPU invocations 1681 from applications 1680 executed on processor 1607. A
35 process element 1683 contains process state for corresponding application 1680. A work

5 descriptor (WD) 1684 contained in process element 1683 can be a single job requested by an application or may contain a pointer to a queue of jobs. In at least one embodiment, WD 1684 is a pointer to a job request queue in an application's address space 1682.

[00265] Graphics acceleration module 1646 and/or individual graphics processing engines 1631-1632, N can be shared by all or a subset of processes in a system. In at least one
10 embodiment, an infrastructure for setting up process state and sending a WD 1684 to a graphics acceleration module 1646 to start a job in a virtualized environment may be included.

[00266] In at least one embodiment, a dedicated-process programming model is implementation-specific. In this model, a single process owns graphics acceleration module
15 1646 or an individual graphics processing engine 1631. Because graphics acceleration module 1646 is owned by a single process, a hypervisor initializes accelerator integration circuit 1636 for an owning partition and an operating system initializes accelerator integration circuit 1636 for an owning process when graphics acceleration module 1646 is assigned.

[00267] In operation, a WD fetch unit 1691 in accelerator integration slice 1690 fetches
20 next WD 1684 which includes an indication of work to be done by one or more graphics processing engines of graphics acceleration module 1646. Data from WD 1684 may be stored in registers 1645 and used by MMU 1639, interrupt management circuit 1647 and/or context management circuit 1648 as illustrated. For example, one embodiment of MMU 1639 includes segment/page walk circuitry for accessing segment/page tables 1686 within OS
25 virtual address space 1685. Interrupt management circuit 1647 may process interrupt events 1692 received from graphics acceleration module 1646. When performing graphics operations, an effective address 1693 generated by a graphics processing engine 1631-1632, N is translated to a real address by MMU 1639.

[00268] In one embodiment, a same set of registers 1645 are duplicated for each graphics
30 processing engine 1631-1632, N and/or graphics acceleration module 1646 and may be initialized by a hypervisor or operating system. Each of these duplicated registers may be included in an accelerator integration slice 1690. Exemplary registers that may be initialized by a hypervisor are shown in Table 1.

5

Table 1 –Hypervisor Initialized Registers

1	Slice Control Register
2	Real Address (RA) Scheduled Processes Area Pointer
3	Authority Mask Override Register
4	Interrupt Vector Table Entry Offset
5	Interrupt Vector Table Entry Limit
6	State Register
7	Logical Partition ID
8	Real address (RA) Hypervisor Accelerator Utilization Record Pointer
9	Storage Description Register

[00269] Exemplary registers that may be initialized by an operating system are shown in Table 2.

Table 2 –Operating System Initialized Registers

1	Process and Thread Identification
2	Effective Address (EA) Context Save/Restore Pointer
3	Virtual Address (VA) Accelerator Utilization Record Pointer
4	Virtual Address (VA) Storage Segment Table Pointer
5	Authority Mask
6	Work descriptor

10

[00270] In one embodiment, each WD 1684 is specific to a particular graphics acceleration module 1646 and/or graphics processing engines 1631-1632, N. It contains all information

5 required by a graphics processing engine 1631-1632, N to do work or it can be a pointer to a memory location where an application has set up a command queue of work to be completed.

[00271] FIG. 16E illustrates additional details for one exemplary embodiment of a shared model. This embodiment includes a hypervisor real address space 1698 in which a process element list 1699 is stored. Hypervisor real address space 1698 is accessible via a hypervisor
10 1696, which virtualizes graphics acceleration module engines for operating system 1695.

[00272] In at least one embodiment, shared programming models allow for all or a subset of processes from all or a subset of partitions in a system to use a graphics acceleration module 1646. There are two programming models where graphics acceleration module 1646 is shared by multiple processes and partitions: time-sliced shared and graphics directed
15 shared.

[00273] In this model, system hypervisor 1696 owns graphics acceleration module 1646 and makes its function available to all operating systems 1695. For a graphics acceleration module 1646 to support virtualization by system hypervisor 1696, graphics acceleration module 1646 may adhere to the following: 1) An application's job request must be
20 autonomous (that is, state does not need to be maintained between jobs), or graphics acceleration module 1646 must provide a context save and restore mechanism. 2) An application's job request is guaranteed by graphics acceleration module 1646 to complete in a specified amount of time, including any translation faults, or graphics acceleration module 1646 provides an ability to preempt processing of a job. 3) Graphics acceleration module
25 1646 must be guaranteed fairness between processes when operating in a directed shared programming model.

[00274] In at least one embodiment, application 1680 is required to make an operating system 1695 system call with a graphics acceleration module 1646 type, a work descriptor (WD), an authority mask register (AMR) value, and a context save/restore area pointer
30 (CSRP). In at least one embodiment, graphics acceleration module 1646 type describes a targeted acceleration function for a system call. In at least one embodiment, graphics acceleration module 1646 type may be a system-specific value. In at least one embodiment, WD is formatted specifically for graphics acceleration module 1646 and can be in a form of a graphics acceleration module 1646 command, an effective address pointer to a user-defined
35 structure, an effective address pointer to a queue of commands, or any other data structure to

5 describe work to be done by graphics acceleration module 1646. In one embodiment, an AMR value is an AMR state to use for a current process. In at least one embodiment, a value passed to an operating system is similar to an application setting an AMR. If accelerator integration circuit 1636 and graphics acceleration module 1646 implementations do not support a User Authority Mask Override Register (UAMOR), an operating system may apply
 10 a current UAMOR value to an AMR value before passing an AMR in a hypervisor call. Hypervisor 1696 may optionally apply a current Authority Mask Override Register (AMOR) value before placing an AMR into process element 1683. In at least one embodiment, CSRP is one of registers 1645 containing an effective address of an area in an application's address space 1682 for graphics acceleration module 1646 to save and restore context state. This
 15 pointer is optional if no state is required to be saved between jobs or when a job is preempted. In at least one embodiment, context save/restore area may be pinned system memory.

[00275] Upon receiving a system call, operating system 1695 may verify that application 1680 has registered and been given authority to use graphics acceleration module 1646. Operating system 1695 then calls hypervisor 1696 with information shown in Table 3.

20 **Table 3 –OS to Hypervisor Call Parameters**

1	A work descriptor (WD)
2	An Authority Mask Register (AMR) value (potentially masked)
3	An effective address (EA) Context Save/Restore Area Pointer (CSRP)
4	A process ID (PID) and optional thread ID (TID)
5	A virtual address (VA) accelerator utilization record pointer (AURP)
6	Virtual address of storage segment table pointer (SSTP)
7	A logical interrupt service number (LISN)

[00276] Upon receiving a hypervisor call, hypervisor 1696 verifies that operating system 1695 has registered and been given authority to use graphics acceleration module 1646. Hypervisor 1696 then puts process element 1683 into a process element linked list for a

5 corresponding graphics acceleration module 1646 type. A process element may include information shown in Table 4.

Table 4 –Process Element Information

1	A work descriptor (WD)
2	An Authority Mask Register (AMR) value (potentially masked).
3	An effective address (EA) Context Save/Restore Area Pointer (CSRP)
4	A process ID (PID) and optional thread ID (TID)
5	A virtual address (VA) accelerator utilization record pointer (AURP)
6	Virtual address of storage segment table pointer (SSTP)
7	A logical interrupt service number (LISN)
8	Interrupt vector table, derived from hypervisor call parameters
9	A state register (SR) value
10	A logical partition ID (LPID)
11	A real address (RA) hypervisor accelerator utilization record pointer
12	Storage Descriptor Register (SDR)

[00277] In at least one embodiment, hypervisor initializes a plurality of accelerator
10 integration slice 1690 registers 1645.

[00278] As illustrated in FIG. 16F, in at least one embodiment, a unified memory is used, addressable via a common virtual memory address space used to access physical processor memories 1601-1602 and GPU memories 1620-1623. In this implementation, operations
15 executed on GPUs 1610-1613 utilize a same virtual/effective memory address space to access processor memories 1601-1602 and vice versa, thereby simplifying programmability. In one embodiment, a first portion of a virtual/effective address space is allocated to processor memory 1601, a second portion to second processor memory 1602, a third portion to GPU

5 memory 1620, and so on. In at least one embodiment, an entire virtual/effective memory space (sometimes referred to as an effective address space) is thereby distributed across each of processor memories 1601-1602 and GPU memories 1620-1623, allowing any processor or GPU to access any physical memory with a virtual address mapped to that memory.

[00279] In one embodiment, bias/coherence management circuitry 1694A-1694E within
10 one or more of MMUs 1639A-1639E ensures cache coherence between caches of one or more host processors (e.g., 1605) and GPUs 1610-1613 and implements biasing techniques indicating physical memories in which certain types of data should be stored. While multiple instances of bias/coherence management circuitry 1694A-1694E are illustrated in FIG. 16F, bias/coherence circuitry may be implemented within an MMU of one or more host processors
15 1605 and/or within accelerator integration circuit 1636.

[00280] One embodiment allows GPU-attached memory 1620-1623 to be mapped as part of system memory, and accessed using shared virtual memory (SVM) technology, but without suffering performance drawbacks associated with full system cache coherence. In at least one embodiment, an ability for GPU-attached memory 1620-1623 to be accessed as
20 system memory without onerous cache coherence overhead provides a beneficial operating environment for GPU offload. This arrangement allows host processor 1605 software to setup operands and access computation results, without overhead of traditional I/O DMA data copies. Such traditional copies involve driver calls, interrupts and memory mapped I/O (MMIO) accesses that are all inefficient relative to simple memory accesses. In at least one
25 embodiment, an ability to access GPU attached memory 1620-1623 without cache coherence overheads can be critical to execution time of an offloaded computation. In cases with substantial streaming write memory traffic, for example, cache coherence overhead can significantly reduce an effective write bandwidth seen by a GPU 1610-1613. In at least one embodiment, efficiency of operand setup, efficiency of results access, and efficiency of GPU
30 computation may play a role in determining effectiveness of a GPU offload.

[00281] In at least one embodiment, selection of GPU bias and host processor bias is driven by a bias tracker data structure. A bias table may be used, for example, which may be a page-granular structure (i.e., controlled at a granularity of a memory page) that includes 1 or 2 bits per GPU-attached memory page. In at least one embodiment, a bias table may be
35 implemented in a stolen memory range of one or more GPU-attached memories 1620-1623,

5 with or without a bias cache in GPU 1610-1613 (e.g., to cache frequently/recently used entries of a bias table). Alternatively, an entire bias table may be maintained within a GPU.

[00282] In at least one embodiment, a bias table entry associated with each access to GPU-attached memory 1620-1623 is accessed prior to actual access to a GPU memory, causing the following operations. First, local requests from GPU 1610-1613 that find their page in GPU
10 bias are forwarded directly to a corresponding GPU memory 1620-1623. Local requests from a GPU that find their page in host bias are forwarded to processor 1605 (e.g., over a high-speed link as discussed above). In one embodiment, requests from processor 1605 that find a requested page in host processor bias complete a request like a normal memory read. Alternatively, requests directed to a GPU-biased page may be forwarded to GPU 1610-1613.
15 In at least one embodiment, a GPU may then transition a page to a host processor bias if it is not currently using a page. In at least one embodiment, bias state of a page can be changed either by a software-based mechanism, a hardware-assisted software-based mechanism, or, for a limited set of cases, a purely hardware-based mechanism.

[00283] One mechanism for changing bias state employs an API call (e.g. OpenCL),
20 which, in turn, calls a GPU's device driver which, in turn, sends a message (or enqueues a command descriptor) to a GPU directing it to change a bias state and, for some transitions, perform a cache flushing operation in a host. In at least one embodiment, cache flushing operation is used for a transition from host processor 1605 bias to GPU bias, but is not for an opposite transition.

25 [00284] In one embodiment, cache coherency is maintained by temporarily rendering GPU-biased pages uncacheable by host processor 1605. To access these pages, processor 1605 may request access from GPU 1610, which may or may not grant access right away. Thus, to reduce communication between processor 1605 and GPU 1610 it is beneficial to ensure that GPU-biased pages are those which are required by a GPU but not host processor
30 1605 and vice versa.

[00285] Hardware structure(s) 1015 are used to perform one or more embodiments. Details regarding the hardware structure(x) 1015 are provided herein in conjunction with FIGs. 10A and/or 10B.

[00286] FIG. 17 illustrates exemplary integrated circuits and associated graphics
35 processors that may be fabricated using one or more IP cores, according to various

5 embodiments described herein. In addition to what is illustrated, other logic and circuits may be included in at least one embodiment, including additional graphics processors/cores, peripheral interface controllers, or general-purpose processor cores.

[00287] FIG. 17 is a block diagram illustrating an exemplary system on a chip integrated circuit 1700 that may be fabricated using one or more IP cores, according to at least one
10 embodiment. In at least one embodiment, integrated circuit 1700 includes one or more application processor(s) 1705 (e.g., CPUs), at least one graphics processor 1710, and may additionally include an image processor 1715 and/or a video processor 1720, any of which may be a modular IP core. In at least one embodiment, integrated circuit 1700 includes peripheral or bus logic including a USB controller 1725, UART controller 1730, an
15 SPI/SDIO controller 1735, and an I.sup.2S/I.sup.2C controller 1740. In at least one embodiment, integrated circuit 1700 can include a display device 1745 coupled to one or more of a high-definition multimedia interface (HDMI) controller 1750 and a mobile industry processor interface (MIPI) display interface 1755. In at least one embodiment, storage may be provided by a flash memory subsystem 1760 including flash memory and a flash memory
20 controller. In at least one embodiment, memory interface may be provided via a memory controller 1765 for access to SDRAM or SRAM memory devices. In at least one embodiment, some integrated circuits additionally include an embedded security engine 1770.

[00288] FIGS. 18A-18B illustrate exemplary integrated circuits and associated graphics
25 processors that may be fabricated using one or more IP cores, according to various embodiments described herein. In addition to what is illustrated, other logic and circuits may be included in at least one embodiment, including additional graphics processors/cores, peripheral interface controllers, or general-purpose processor cores.

[00289] FIGS. 18A-18B are block diagrams illustrating exemplary graphics processors for
30 use within an SoC, according to embodiments described herein. FIG. 18A illustrates an exemplary graphics processor 1810 of a system on a chip integrated circuit that may be fabricated using one or more IP cores, according to at least one embodiment. FIG. 18B illustrates an additional exemplary graphics processor 1840 of a system on a chip integrated circuit that may be fabricated using one or more IP cores, according to at least one
35 embodiment. In at least one embodiment, graphics processor 1810 of FIG. 18A is a low power graphics processor core. In at least one embodiment, graphics processor 1840 of FIG.

5 18B is a higher performance graphics processor core. In at least one embodiment, each of graphics processors 1810, 1840 can be variants of graphics processor 1710 of FIG. 17.

[00290] In at least one embodiment, graphics processor 1810 includes a vertex processor 1805 and one or more fragment processor(s) 1815A-1815N (e.g., 1815A, 1815B, 1815C, 1815D, through 1815N-1, and 1815N). In at least one embodiment, graphics processor 1810
 10 can execute different shader programs via separate logic, such that vertex processor 1805 is optimized to execute operations for vertex shader programs, while one or more fragment processor(s) 1815A-1815N execute fragment (e.g., pixel) shading operations for fragment or pixel shader programs. In at least one embodiment, vertex processor 1805 performs a vertex processing stage of a 3D graphics pipeline and generates primitives and vertex data. In at
 15 least one embodiment, fragment processor(s) 1815A-1815N use primitive and vertex data generated by vertex processor 1805 to produce a framebuffer that is displayed on a display device. In at least one embodiment, fragment processor(s) 1815A-1815N are optimized to execute fragment shader programs as provided for in an OpenGL API, which may be used to perform similar operations as a pixel shader program as provided for in a Direct 3D API.

[00291] In at least one embodiment, graphics processor 1810 additionally includes one or more memory management units (MMUs) 1820A-1820B, cache(s) 1825A-1825B, and circuit interconnect(s) 1830A-1830B. In at least one embodiment, one or more MMU(s) 1820A-1820B provide for virtual to physical address mapping for graphics processor 1810, including for vertex processor 1805 and/or fragment processor(s) 1815A-1815N, which may reference
 25 vertex or image/texture data stored in memory, in addition to vertex or image/texture data stored in one or more cache(s) 1825A-1825B. In at least one embodiment, one or more MMU(s) 1820A-1820B may be synchronized with other MMUs within system, including one or more MMUs associated with one or more application processor(s) 1705, image processors 1715, and/or video processors 1720 of FIG. 17, such that each processor 1705-
 30 1720 can participate in a shared or unified virtual memory system. In at least one embodiment, one or more circuit interconnect(s) 1830A-1830B enable graphics processor 1810 to interface with other IP cores within SoC, either via an internal bus of SoC or via a direct connection.

[00292] In at least one embodiment, graphics processor 1840 includes one or more
 35 MMU(s) 1820A-1820B, caches 1825A-1825B, and circuit interconnects 1830A-1830B of graphics processor 1810 of FIG. 18A. In at least one embodiment, graphics processor 1840

5 includes one or more shader core(s) 1855A-1855N (e.g., 1855A, 1855B, 1855C, 1855D, 1855E, 1855F, through 1855N-1, and 1855N), which provides for a unified shader core architecture in which a single core or type or core can execute all types of programmable shader code, including shader program code to implement vertex shaders, fragment shaders, and/or compute shaders. In at least one embodiment, a number of shader cores can vary. In
 10 at least one embodiment, graphics processor 1840 includes an inter-core task manager 1845, which acts as a thread dispatcher to dispatch execution threads to one or more shader cores 1855A-1855N and a tiling unit 1858 to accelerate tiling operations for tile-based rendering, in which rendering operations for a scene are subdivided in image space, for example to exploit local spatial coherence within a scene or to optimize use of internal caches.

15 [00293] FIGS. 19A-19B illustrate additional exemplary graphics processor logic according to embodiments described herein. FIG. 19A illustrates a graphics core 1900 that may be included within graphics processor 1710 of FIG. 17, in at least one embodiment, and may be a unified shader core 1855A-1855N as in FIG. 18B in at least one embodiment. FIG. 19B illustrates a highly-parallel general-purpose graphics processing unit 1930 suitable for
 20 deployment on a multi-chip module in at least one embodiment.

[00294] In at least one embodiment, graphics core 1900 includes a shared instruction cache 1902, a texture unit 1918, and a cache/shared memory 1920 that are common to execution resources within graphics core 1900. In at least one embodiment, graphics core 1900 can include multiple slices 1901A-1901N or partition for each core, and a graphics
 25 processor can include multiple instances of graphics core 1900. Slices 1901A-1901N can include support logic including a local instruction cache 1904A-1904N, a thread scheduler 1906A-1906N, a thread dispatcher 1908A-1908N, and a set of registers 1910A-1910N. In at least one embodiment, slices 1901A-1901N can include a set of additional function units (AFUs 1912A-1912N), floating-point units (FPU 1914A-1914N), integer arithmetic logic
 30 units (ALUs 1916-1916N), address computational units (ACU 1913A-1913N), double-precision floating-point units (DPFPU 1915A-1915N), and matrix processing units (MPU 1917A-1917N).

[00295] In at least one embodiment, FPUs 1914A-1914N can perform single-precision (32-bit) and half-precision (16-bit) floating point operations, while DPFPU 1915A-1915N
 35 perform double precision (64-bit) floating point operations. In at least one embodiment, ALUs 1916A-1916N can perform variable precision integer operations at 8-bit, 16-bit, and

5 32-bit precision, and can be configured for mixed precision operations. In at least one embodiment, MPUs 1917A-1917N can also be configured for mixed precision matrix operations, including half-precision floating point and 8-bit integer operations. In at least one embodiment, MPUs 1917-1917N can perform a variety of matrix operations to accelerate machine learning application frameworks, including enabling support for accelerated general
10 matrix to matrix multiplication (GEMM). In at least one embodiment, AFUs 1912A-1912N can perform additional logic operations not supported by floating-point or integer units, including trigonometric operations (e.g., Sine, Cosine, etc.).

[00296] FIG. 19B illustrates a general-purpose processing unit (GPGPU) 1930 that can be configured to enable highly-parallel compute operations to be performed by an array of
15 graphics processing units, in at least one embodiment. In at least one embodiment, GPGPU 1930 can be linked directly to other instances of GPGPU 1930 to create a multi-GPU cluster to improve training speed for deep neural networks. In at least one embodiment, GPGPU 1930 includes a host interface 1932 to enable a connection with a host processor. In at least one embodiment, host interface 1932 is a PCI Express interface. In at least one embodiment,
20 host interface 1932 can be a vendor specific communications interface or communications fabric. In at least one embodiment, GPGPU 1930 receives commands from a host processor and uses a global scheduler 1934 to distribute execution threads associated with those commands to a set of compute clusters 1936A-1936H. In at least one embodiment, compute clusters 1936A-1936H share a cache memory 1938. In at least one embodiment, cache
25 memory 1938 can serve as a higher-level cache for cache memories within compute clusters 1936A-1936H.

[00297] In at least one embodiment, GPGPU 1930 includes memory 1944A-1944B coupled with compute clusters 1936A-1936H via a set of memory controllers 1942A-1942B. In at least one embodiment, memory 1944A-1944B can include various types of memory
30 devices including dynamic random access memory (DRAM) or graphics random access memory, such as synchronous graphics random access memory (SGRAM), including graphics double data rate (GDDR) memory.

[00298] In at least one embodiment, compute clusters 1936A-1936H each include a set of graphics cores, such as graphics core 1900 of FIG. 19A, which can include multiple types of
35 integer and floating point logic units that can perform computational operations at a range of precisions including suited for machine learning computations. For example, in at least one

5 embodiment, at least a subset of floating point units in each of compute clusters 1936A-1936H can be configured to perform 16-bit or 32-bit floating point operations, while a different subset of floating point units can be configured to perform 64-bit floating point operations.

[00299] In at least one embodiment, multiple instances of GPGPU 1930 can be configured
10 to operate as a compute cluster. In at least one embodiment, communication used by compute clusters 1936A-1936H for synchronization and data exchange varies across embodiments. In at least one embodiment, multiple instances of GPGPU 1930 communicate over host interface 1932. In at least one embodiment, GPGPU 1930 includes an I/O hub 1939 that couples GPGPU 1930 with a GPU link 1940 that enables a direct connection to other
15 instances of GPGPU 1930. In at least one embodiment, GPU link 1940 is coupled to a dedicated GPU-to-GPU bridge that enables communication and synchronization between multiple instances of GPGPU 1930. In at least one embodiment GPU link, 1940 couples with a high speed interconnect to transmit and receive data to other GPGPUs or parallel processors. In at least one embodiment, multiple instances of GPGPU 1930 are located in
20 separate data processing systems and communicate via a network device that is accessible via host interface 1932. In at least one embodiment GPU link, 1940 can be configured to enable a connection to a host processor in addition to or as an alternative to host interface 1932.

[00300] In at least one embodiment, GPGPU 1930 can be configured to train neural networks. In at least one embodiment, GPGPU 1930 can be used within a inferencing
25 platform. In at least one embodiment, in which GPGPU 1930 is used for inferencing, GPGPU may include fewer compute clusters 1936A-1936H relative to when GPGPU is used for training a neural network. In at least one embodiment, memory technology associated with memory 1944A-1944B may differ between inferencing and training configurations, with higher bandwidth memory technologies devoted to training configurations. In at least one
30 embodiment, inferencing configuration of GPGPU 1930 can support inferencing specific instructions. For example, in at least one embodiment, an inferencing configuration can provide support for one or more 8-bit integer dot product instructions, which may be used during inferencing operations for deployed neural networks.

[00301] FIG. 20 is a block diagram illustrating a computing system 2000 according to at
35 least one embodiment. In at least one embodiment, computing system 2000 includes a processing subsystem 2001 having one or more processor(s) 2002 and a system memory

5 2004 communicating via an interconnection path that may include a memory hub 2005. In at least one embodiment, memory hub 2005 may be a separate component within a chipset component or may be integrated within one or more processor(s) 2002. In at least one embodiment, memory hub 2005 couples with an I/O subsystem 2011 via a communication link 2006. In at least one embodiment, I/O subsystem 2011 includes an I/O hub 2007 that
10 can enable computing system 2000 to receive input from one or more input device(s) 2008. In at least one embodiment, I/O hub 2007 can enable a display controller, which may be included in one or more processor(s) 2002, to provide outputs to one or more display device(s) 2010A. In at least one embodiment, one or more display device(s) 2010A coupled with I/O hub 2007 can include a local, internal, or embedded display device.

15 [00302] In at least one embodiment, processing subsystem 2001 includes one or more parallel processor(s) 2012 coupled to memory hub 2005 via a bus or other communication link 2013. In at least one embodiment, communication link 2013 may be one of any number of standards based communication link technologies or protocols, such as, but not limited to PCI Express, or may be a vendor specific communications interface or communications
20 fabric. In at least one embodiment, one or more parallel processor(s) 2012 form a computationally focused parallel or vector processing system that can include a large number of processing cores and/or processing clusters, such as a many integrated core (MIC) processor. In at least one embodiment, one or more parallel processor(s) 2012 form a graphics processing subsystem that can output pixels to one of one or more display device(s)
25 2010A coupled via I/O Hub 2007. In at least one embodiment, one or more parallel processor(s) 2012 can also include a display controller and display interface (not shown) to enable a direct connection to one or more display device(s) 2010B.

[00303] In at least one embodiment, a system storage unit 2014 can connect to I/O hub 2007 to provide a storage mechanism for computing system 2000. In at least one
30 embodiment, an I/O switch 2016 can be used to provide an interface mechanism to enable connections between I/O hub 2007 and other components, such as a network adapter 2018 and/or wireless network adapter 2019 that may be integrated into platform, and various other devices that can be added via one or more add-in device(s) 2020. In at least one embodiment, network adapter 2018 can be an Ethernet adapter or another wired network adapter. In at least
35 one embodiment, wireless network adapter 2019 can include one or more of a Wi-Fi,

5 Bluetooth, near field communication (NFC), or other network device that includes one or more wireless radios.

[00304] In at least one embodiment, computing system 2000 can include other components not explicitly shown, including USB or other port connections, optical storage drives, video capture devices, and like, may also be connected to I/O hub 2007. In at least one
10 embodiment, communication paths interconnecting various components in FIG. 20 may be implemented using any suitable protocols, such as PCI (Peripheral Component Interconnect) based protocols (e.g., PCI-Express), or other bus or point-to-point communication interfaces and/or protocol(s), such as NV-Link high-speed interconnect, or interconnect protocols.

[00305] In at least one embodiment, one or more parallel processor(s) 2012 incorporate
15 circuitry optimized for graphics and video processing, including, for example, video output circuitry, and constitutes a graphics processing unit (GPU). In at least one embodiment, one or more parallel processor(s) 2012 incorporate circuitry optimized for general purpose processing. In at least one embodiment, components of computing system 2000 may be integrated with one or more other system elements on a single integrated circuit. For
20 example, in at least one embodiment, one or more parallel processor(s) 2012, memory hub 2005, processor(s) 2002, and I/O hub 2007 can be integrated into a system on chip (SoC) integrated circuit. In at least one embodiment, components of computing system 2000 can be integrated into a single package to form a system in package (SIP) configuration. In at least one embodiment, at least a portion of components of computing system 2000 can be
25 integrated into a multi-chip module (MCM), which can be interconnected with other multi-chip modules into a modular computing system.

PROCESSORS

[00306] FIG. 21A illustrates a parallel processor 2100 according to at least one
embodiment. In at least one embodiment, various components of parallel processor 2100
30 may be implemented using one or more integrated circuit devices, such as programmable processors, application specific integrated circuits (ASICs), or field programmable gate arrays (FPGA). In at least one embodiment, illustrated parallel processor 2100 is a variant of one or more parallel processor(s) 2012 shown in FIG. 20 according to an exemplary embodiment.

5 [00307] In at least one embodiment, parallel processor 2100 includes a parallel processing unit 2102. In at least one embodiment, parallel processing unit 2102 includes an I/O unit 2104 that enables communication with other devices, including other instances of parallel processing unit 2102. In at least one embodiment, I/O unit 2104 may be directly connected to other devices. In at least one embodiment, I/O unit 2104 connects with other devices via use
10 of a hub or switch interface, such as memory hub 2005. In at least one embodiment, connections between memory hub 2005 and I/O unit 2104 form a communication link 2013. In at least one embodiment, I/O unit 2104 connects with a host interface 2106 and a memory crossbar 2116, where host interface 2106 receives commands directed to performing processing operations and memory crossbar 2116 receives commands directed to performing
15 memory operations.

[00308] In at least one embodiment, when host interface 2106 receives a command buffer via I/O unit 2104, host interface 2106 can direct work operations to perform those commands to a front end 2108. In at least one embodiment, front end 2108 couples with a scheduler 2110, which is configured to distribute commands or other work items to a processing cluster
20 array 2112. In at least one embodiment, scheduler 2110 ensures that processing cluster array 2112 is properly configured and in a valid state before tasks are distributed to processing cluster array 2112 of processing cluster array 2112. In at least one embodiment, scheduler 2110 is implemented via firmware logic executing on a microcontroller. In at least one embodiment, microcontroller implemented scheduler 2110 is configurable to perform
25 complex scheduling and work distribution operations at coarse and fine granularity, enabling rapid preemption and context switching of threads executing on processing array 2112. In at least one embodiment, host software can provide workloads for scheduling on processing array 2112 via one of multiple graphics processing doorbells. In at least one embodiment, workloads can then be automatically distributed across processing array 2112 by scheduler
30 2110 logic within a microcontroller including scheduler 2110.

[00309] In at least one embodiment, processing cluster array 2112 can include up to “N” processing clusters (e.g., cluster 2114A, cluster 2114B, through cluster 2114N). In at least one embodiment, each cluster 2114A-2114N of processing cluster array 2112 can execute a large number of concurrent threads. In at least one embodiment, scheduler 2110 can allocate
35 work to clusters 2114A-2114N of processing cluster array 2112 using various scheduling and/or work distribution algorithms, which may vary depending on workload arising for each

5 type of program or computation. In at least one embodiment, scheduling can be handled dynamically by scheduler 2110, or can be assisted in part by compiler logic during compilation of program logic configured for execution by processing cluster array 2112. In at least one embodiment, different clusters 2114A-2114N of processing cluster array 2112 can be allocated for processing different types of programs or for performing different types
10 of computations.

[00310] In at least one embodiment, processing cluster array 2112 can be configured to perform various types of parallel processing operations. In at least one embodiment, processing cluster array 2112 is configured to perform general-purpose parallel compute operations. For example, in at least one embodiment, processing cluster array 2112 can
15 include logic to execute processing tasks including filtering of video and/or audio data, performing modeling operations, including physics operations, and performing data transformations.

[00311] In at least one embodiment, processing cluster array 2112 is configured to perform parallel graphics processing operations. In at least one embodiment, processing cluster
20 array 2112 can include additional logic to support execution of such graphics processing operations, including, but not limited to texture sampling logic to perform texture operations, as well as tessellation logic and other vertex processing logic. In at least one embodiment, processing cluster array 2112 can be configured to execute graphics processing related shader programs such as, but not limited to vertex shaders, tessellation shaders, geometry shaders,
25 and pixel shaders. In at least one embodiment, parallel processing unit 2102 can transfer data from system memory via I/O unit 2104 for processing. In at least one embodiment, during processing, transferred data can be stored to on-chip memory (e.g., parallel processor memory 2122) during processing, then written back to system memory.

[00312] In at least one embodiment, when parallel processing unit 2102 is used to perform
30 graphics processing, scheduler 2110 can be configured to divide a processing workload into approximately equal sized tasks, to better enable distribution of graphics processing operations to multiple clusters 2114A-2114N of processing cluster array 2112. In at least one embodiment, portions of processing cluster array 2112 can be configured to perform different types of processing. For example, in at least one embodiment, a first portion may be
35 configured to perform vertex shading and topology generation, a second portion may be configured to perform tessellation and geometry shading, and a third portion may be

5 configured to perform pixel shading or other screen space operations, to produce a rendered image for display. In at least one embodiment, intermediate data produced by one or more of clusters 2114A-2114N may be stored in buffers to allow intermediate data to be transmitted between clusters 2114A-2114N for further processing.

[00313] In at least one embodiment, processing cluster array 2112 can receive processing
10 tasks to be executed via scheduler 2110, which receives commands defining processing tasks from front end 2108. In at least one embodiment, processing tasks can include indices of data to be processed, e.g., surface (patch) data, primitive data, vertex data, and/or pixel data, as well as state parameters and commands defining how data is to be processed (e.g., what program is to be executed). In at least one embodiment, scheduler 2110 may be configured to
15 fetch indices corresponding to tasks or may receive indices from front end 2108. In at least one embodiment, front end 2108 can be configured to ensure processing cluster array 2112 is configured to a valid state before a workload specified by incoming command buffers (e.g., batch-buffers, push buffers, etc.) is initiated.

[00314] In at least one embodiment, each of one or more instances of parallel processing
20 unit 2102 can couple with parallel processor memory 2122. In at least one embodiment, parallel processor memory 2122 can be accessed via memory crossbar 2116, which can receive memory requests from processing cluster array 2112 as well as I/O unit 2104. In at least one embodiment, memory crossbar 2116 can access parallel processor memory 2122 via a memory interface 2118. In at least one embodiment, memory interface 2118 can include
25 multiple partition units (e.g., partition unit 2120A, partition unit 2120B, through partition unit 2120N) that can each couple to a portion (e.g., memory unit) of parallel processor memory 2122. In at least one embodiment, a number of partition units 2120A-2120N is configured to be equal to a number of memory units, such that a first partition unit 2120A has a corresponding first memory unit 2124A, a second partition unit 2120B has a corresponding
30 memory unit 2124B, and an Nth partition unit 2120N has a corresponding Nth memory unit 2124N. In at least one embodiment, a number of partition units 2120A-2120N may not be equal to a number of memory devices.

[00315] In at least one embodiment, memory units 2124A-2124N can include various
types of memory devices, including dynamic random access memory (DRAM) or graphics
35 random access memory, such as synchronous graphics random access memory (SGRAM), including graphics double data rate (GDDR) memory. In at least one embodiment, memory

5 units 2124A-2124N may also include 3D stacked memory, including but not limited to high bandwidth memory (HBM). In at least one embodiment, render targets, such as frame buffers or texture maps may be stored across memory units 2124A-2124N, allowing partition units 2120A-2120N to write portions of each render target in parallel to efficiently use available bandwidth of parallel processor memory 2122. In at least one embodiment, a local instance of
10 parallel processor memory 2122 may be excluded in favor of a unified memory design that utilizes system memory in conjunction with local cache memory.

[00316] In at least one embodiment, any one of clusters 2114A-2114N of processing cluster array 2112 can process data that will be written to any of memory units 2124A-2124N within parallel processor memory 2122. In at least one embodiment, memory crossbar 2116
15 can be configured to transfer an output of each cluster 2114A-2114N to any partition unit 2120A-2120N or to another cluster 2114A-2114N, which can perform additional processing operations on an output. In at least one embodiment, each cluster 2114A-2114N can communicate with memory interface 2118 through memory crossbar 2116 to read from or write to various external memory devices. In at least one embodiment, memory crossbar
20 2116 has a connection to memory interface 2118 to communicate with I/O unit 2104, as well as a connection to a local instance of parallel processor memory 2122, enabling processing units within different processing clusters 2114A-2114N to communicate with system memory or other memory that is not local to parallel processing unit 2102. In at least one embodiment, memory crossbar 2116 can use virtual channels to separate traffic streams
25 between clusters 2114A-2114N and partition units 2120A-2120N.

[00317] In at least one embodiment, multiple instances of parallel processing unit 2102 can be provided on a single add-in card, or multiple add-in cards can be interconnected. In at least one embodiment, different instances of parallel processing unit 2102 can be configured to inter-operate even if different instances have different numbers of processing cores,
30 different amounts of local parallel processor memory, and/or other configuration differences. For example, in at least one embodiment, some instances of parallel processing unit 2102 can include higher precision floating point units relative to other instances. In at least one embodiment, systems incorporating one or more instances of parallel processing unit 2102 or parallel processor 2100 can be implemented in a variety of configurations and form factors,
35 including but not limited to desktop, laptop, or handheld personal computers, servers, workstations, game consoles, and/or embedded systems.

5 [00318] FIG. 21B is a block diagram of a partition unit 2120 according to at least one embodiment. In at least one embodiment, partition unit 2120 is an instance of one of partition units 2120A-2120N of FIG. 21A. In at least one embodiment, partition unit 2120 includes an L2 cache 2121, a frame buffer interface 2125, and a ROP 2126 (raster operations unit). L2 cache 2121 is a read/write cache that is configured to perform load and store operations
10 received from memory crossbar 2116 and ROP 2126. In at least one embodiment, read misses and urgent write-back requests are output by L2 cache 2121 to frame buffer interface 2125 for processing. In at least one embodiment, updates can also be sent to a frame buffer via frame buffer interface 2125 for processing. In at least one embodiment, frame buffer interface 2125 interfaces with one of memory units in parallel processor memory, such as memory
15 units 2124A-2124N of FIG. 21 (e.g., within parallel processor memory 2122).

[00319] In at least one embodiment, ROP 2126 is a processing unit that performs raster operations such as stencil, z test, blending, and like. In at least one embodiment, ROP 2126 then outputs processed graphics data that is stored in graphics memory. In at least one embodiment, ROP 2126 includes compression logic to compress depth or color data that is
20 written to memory and decompress depth or color data that is read from memory. In at least one embodiment, compression logic can be lossless compression logic that makes use of one or more of multiple compression algorithms. Type of compression that is performed by ROP 2126 can vary based on statistical characteristics of data to be compressed. For example, in at least one embodiment, delta color compression is performed on depth and color data on a
25 per-tile basis.

[00320] In In at least one embodiment, ROP 2126 is included within each processing cluster (e.g., cluster 2114A-2114N of FIG. 21) instead of within partition unit 2120. In at least one embodiment, read and write requests for pixel data are transmitted over memory crossbar 2116 instead of pixel fragment data. In at least one embodiment, processed graphics
30 data may be displayed on a display device, such as one of one or more display device(s) 2010 of FIG. 20, routed for further processing by processor(s) 2002, or routed for further processing by one of processing entities within parallel processor 2100 of FIG. 21A.

[00321] FIG. 21C is a block diagram of a processing cluster 2114 within a parallel processing unit according to at least one embodiment. In at least one embodiment, a
35 processing cluster is an instance of one of processing clusters 2114A-2114N of FIG. 21. In at least one embodiment, processing cluster 2114 can be configured to execute many threads in

5 parallel, where term “thread” refers to an instance of a particular program executing on a particular set of input data. In at least one embodiment, single-instruction, multiple-data (SIMD) instruction issue techniques are used to support parallel execution of a large number of threads without providing multiple independent instruction units. In at least one embodiment, single-instruction, multiple-thread (SIMT) techniques are used to support
10 parallel execution of a large number of generally synchronized threads, using a common instruction unit configured to issue instructions to a set of processing engines within each one of processing clusters.

[00322] In at least one embodiment, operation of processing cluster 2114 can be controlled via a pipeline manager 2132 that distributes processing tasks to SIMT parallel processors. In
15 at least one embodiment, pipeline manager 2132 receives instructions from scheduler 2110 of FIG. 21 and manages execution of those instructions via a graphics multiprocessor 2134 and/or a texture unit 2136. In at least one embodiment, graphics multiprocessor 2134 is an exemplary instance of a SIMT parallel processor. However, in at least one embodiment, various types of SIMT parallel processors of differing architectures may be included within
20 processing cluster 2114. In at least one embodiment, one or more instances of graphics multiprocessor 2134 can be included within a processing cluster 2114. In at least one embodiment, graphics multiprocessor 2134 can process data and a data crossbar 2140 can be used to distribute processed data to one of multiple possible destinations, including other shader units. In at least one embodiment, pipeline manager 2132 can facilitate distribution of
25 processed data by specifying destinations for processed data to be distributed via data crossbar 2140.

[00323] In at least one embodiment, each graphics multiprocessor 2134 within processing cluster 2114 can include an identical set of functional execution logic (e.g., arithmetic logic units, load-store units, etc.). In at least one embodiment, functional execution logic can be
30 configured in a pipelined manner in which new instructions can be issued before previous instructions are complete. In at least one embodiment, functional execution logic supports a variety of operations including integer and floating point arithmetic, comparison operations, Boolean operations, bit-shifting, and computation of various algebraic functions. In at least one embodiment, same functional-unit hardware can be leveraged to perform different
35 operations and any combination of functional units may be present.

5 [00324] In at least one embodiment, instructions transmitted to processing cluster 2114 constitute a thread. In at least one embodiment, a set of threads executing across a set of parallel processing engines is a thread group. In at least one embodiment, thread group executes a program on different input data. In at least one embodiment, each thread within a thread group can be assigned to a different processing engine within a graphics
10 multiprocessor 2134. In at least one embodiment, a thread group may include fewer threads than a number of processing engines within graphics multiprocessor 2134. In at least one embodiment, when a thread group includes fewer threads than a number of processing engines, one or more of processing engines may be idle during cycles in which that thread group is being processed. In at least one embodiment, a thread group may also include more
15 threads than a number of processing engines within graphics multiprocessor 2134. In at least one embodiment, when a thread group includes more threads than number of processing engines within graphics multiprocessor 2134, processing can be performed over consecutive clock cycles. In at least one embodiment, multiple thread groups can be executed concurrently on a graphics multiprocessor 2134.

20 [00325] In at least one embodiment, graphics multiprocessor 2134 includes an internal cache memory to perform load and store operations. In at least one embodiment, graphics multiprocessor 2134 can forego an internal cache and use a cache memory (e.g., L1 cache 2148) within processing cluster 2114. In at least one embodiment, each graphics multiprocessor 2134 also has access to L2 caches within partition units (e.g., partition units
25 2120A-2120N of FIG. 21) that are shared among all processing clusters 2114 and may be used to transfer data between threads. In at least one embodiment, graphics multiprocessor 2134 may also access off-chip global memory, which can include one or more of local parallel processor memory and/or system memory. In at least one embodiment, any memory external to parallel processing unit 2102 may be used as global memory. In at least one
30 embodiment, processing cluster 2114 includes multiple instances of graphics multiprocessor 2134 can share common instructions and data, which may be stored in L1 cache 2148.

[00326] In at least one embodiment, each processing cluster 2114 may include an MMU 2145 (memory management unit) that is configured to map virtual addresses into physical addresses. In at least one embodiment, one or more instances of MMU 2145 may reside
35 within memory interface 2118 of FIG. 21. In at least one embodiment, MMU 2145 includes a set of page table entries (PTEs) used to map a virtual address to a physical address of a tile

5 (talk more about tiling) and optionally a cache line index. In at least one embodiment, MMU 2145 may include address translation lookaside buffers (TLB) or caches that may reside within graphics multiprocessor 2134 or L1 cache or processing cluster 2114. In at least one embodiment, physical address is processed to distribute surface data access locality to allow efficient request interleaving among partition units. In at least one embodiment, cache line
10 index may be used to determine whether a request for a cache line is a hit or miss.

[00327] In at least one embodiment, a processing cluster 2114 may be configured such that each graphics multiprocessor 2134 is coupled to a texture unit 2136 for performing texture mapping operations, e.g., determining texture sample positions, reading texture data, and filtering texture data. In at least one embodiment, texture data is read from an internal
15 texture L1 cache (not shown) or from an L1 cache within graphics multiprocessor 2134 and is fetched from an L2 cache, local parallel processor memory, or system memory, as needed. In at least one embodiment, each graphics multiprocessor 2134 outputs processed tasks to data crossbar 2140 to provide processed task to another processing cluster 2114 for further processing or to store processed task in an L2 cache, local parallel processor memory, or
20 system memory via memory crossbar 2116. In at least one embodiment, preROP 2142 (pre-raster operations unit) is configured to receive data from graphics multiprocessor 2134, direct data to ROP units, which may be located with partition units as described herein (e.g., partition units 2120A-2120N of FIG. 21). In at least one embodiment, PreROP 2142 unit can perform optimizations for color blending, organize pixel color data, and perform address
25 translations.

[00328] FIG. 21D shows a graphics multiprocessor 2134 according to at least one embodiment. In at least one embodiment, graphics multiprocessor 2134 couples with pipeline manager 2132 of processing cluster 2114. In at least one embodiment, graphics multiprocessor 2134 has an execution pipeline including but not limited to an instruction
30 cache 2152, an instruction unit 2154, an address mapping unit 2156, a register file 2158, one or more general purpose graphics processing unit (GPGPU) cores 2162, and one or more load/store units 2166. GPGPU cores 2162 and load/store units 2166 are coupled with cache memory 2172 and shared memory 2170 via a memory and cache interconnect 2168.

[00329] In at least one embodiment, instruction cache 2152 receives a stream of
35 instructions to execute from pipeline manager 2132. In at least one embodiment, instructions are cached in instruction cache 2152 and dispatched for execution by instruction unit 2154.

5 In at least one embodiment, instruction unit 2154 can dispatch instructions as thread groups (e.g., warps), with each thread of thread group assigned to a different execution unit within GPGPU core 2162. In at least one embodiment, an instruction can access any of a local, shared, or global address space by specifying an address within a unified address space. In at
10 unified address space into a distinct memory address that can be accessed by load/store units 2166.

[00330] In at least one embodiment, register file 2158 provides a set of registers for functional units of graphics multiprocessor 2134. In at least one embodiment, register file 2158 provides temporary storage for operands connected to data paths of functional units
15 (e.g., GPGPU cores 2162, load/store units 2166) of graphics multiprocessor 2134. In at least one embodiment, register file 2158 is divided between each of functional units such that each functional unit is allocated a dedicated portion of register file 2158. In at least one embodiment, register file 2158 is divided between different warps being executed by graphics multiprocessor 2134.

20 [00331] In at least one embodiment, GPGPU cores 2162 can each include floating point units (FPUs) and/or integer arithmetic logic units (ALUs) that are used to execute instructions of graphics multiprocessor 2134. GPGPU cores 2162 can be similar in architecture or can differ in architecture. In at least one embodiment, a first portion of GPGPU cores 2162 include a single precision FPU and an integer ALU while a second portion of GPGPU cores
25 include a double precision FPU. In at least one embodiment, FPUs can implement IEEE 754-2008 standard for floating point arithmetic or enable variable precision floating point arithmetic. In at least one embodiment, graphics multiprocessor 2134 can additionally include one or more fixed function or special function units to perform specific functions such as copy rectangle or pixel blending operations. In at least one embodiment one or more of
30 GPGPU cores can also include fixed or special function logic.

[00332] In at least one embodiment, GPGPU cores 2162 include SIMD logic capable of performing a single instruction on multiple sets of data. In at least one embodiment GPGPU cores 2162 can physically execute SIMD4, SIMD8, and SIMD16 instructions and logically execute SIMD1, SIMD2, and SIMD32 instructions. In at least one embodiment, SIMD
35 instructions for GPGPU cores can be generated at compile time by a shader compiler or automatically generated when executing programs written and compiled for single program

5 multiple data (SPMD) or SIMT architectures. In at least one embodiment, multiple threads of a program configured for an SIMT execution model can be executed via a single SIMD instruction. For example, in at least one embodiment, eight SIMT threads that perform same or similar operations can be executed in parallel via a single SIMD8 logic unit.

[00333] In at least one embodiment, memory and cache interconnect 2168 is an
10 interconnect network that connects each functional unit of graphics multiprocessor 2134 to register file 2158 and to shared memory 2170. In at least one embodiment, memory and cache interconnect 2168 is a crossbar interconnect that allows load/store unit 2166 to implement load and store operations between shared memory 2170 and register file 2158. In at least one embodiment, register file 2158 can operate at a same frequency as GPGPU cores
15 2162, thus data transfer between GPGPU cores 2162 and register file 2158 is very low latency. In at least one embodiment, shared memory 2170 can be used to enable communication between threads that execute on functional units within graphics multiprocessor 2134. In at least one embodiment, cache memory 2172 can be used as a data cache for example, to cache texture data communicated between functional units and texture
20 unit 2136. In at least one embodiment, shared memory 2170 can also be used as a program managed cache. In at least one embodiment, threads executing on GPGPU cores 2162 can programmatically store data within shared memory in addition to automatically cached data that is stored within cache memory 2172.

[00334] In at least one embodiment, a parallel processor or GPGPU as described herein is
25 communicatively coupled to host/processor cores to accelerate graphics operations, machine-learning operations, pattern analysis operations, and various general purpose GPU (GPGPU) functions. In at least one embodiment, GPU may be communicatively coupled to host processor/cores over a bus or other interconnect (e.g., a high speed interconnect such as PCIe or NVLink). In at least one embodiment, GPU may be integrated on same package or chip as
30 cores and communicatively coupled to cores over an internal processor bus/interconnect (i.e., internal to package or chip). In at least one embodiment, regardless of manner in which GPU is connected, processor cores may allocate work to GPU in form of sequences of commands/instructions contained in a work descriptor. In at least one embodiment, GPU then uses dedicated circuitry/logic for efficiently processing these commands/instructions.

35 [00335] FIG. 22 illustrates a multi-GPU computing system 2200, according to at least one embodiment. In at least one embodiment, multi-GPU computing system 2200 can include a

5 processor 2202 coupled to multiple general purpose graphics processing units (GPGPUs) 2206A-D via a host interface switch 2204. In at least one embodiment, host interface switch 2204 is a PCI express switch device that couples processor 2202 to a PCI express bus over which processor 2202 can communicate with GPGPUs 2206A-D. GPGPUs 2206A-D can interconnect via a set of high-speed point to point GPU to GPU links 2216. In at least one
10 embodiment, GPU to GPU links 2216 connect to each of GPGPUs 2206A-D via a dedicated GPU link. In at least one embodiment, P2P GPU links 2216 enable direct communication between each of GPGPUs 2206A-D without requiring communication over host interface bus 2204 to which processor 2202 is connected. In at least one embodiment, with GPU-to-GPU traffic directed to P2P GPU links 2216, host interface bus 2204 remains available for system
15 memory access or to communicate with other instances of multi-GPU computing system 2200, for example, via one or more network devices. While in at least one embodiment GPGPUs 2206A-D connect to processor 2202 via host interface switch 2204, in at least one embodiment processor 2202 includes direct support for P2P GPU links 2216 and can connect directly to GPGPUs 2206A-D.

20 [00336] FIG. 23 is a block diagram of a graphics processor 2300, according to at least one embodiment. In at least one embodiment, graphics processor 2300 includes a ring interconnect 2302, a pipeline front-end 2304, a media engine 2337, and graphics cores 2380A-2380N. In at least one embodiment, ring interconnect 2302 couples graphics processor 2300 to other processing units, including other graphics processors or one or more
25 general-purpose processor cores. In at least one embodiment, graphics processor 2300 is one of many processors integrated within a multi-core processing system.

[00337] In at least one embodiment, graphics processor 2300 receives batches of commands via ring interconnect 2302. In at least one embodiment, incoming commands are interpreted by a command streamer 2303 in pipeline front-end 2304. In at least one
30 embodiment, graphics processor 2300 includes scalable execution logic to perform 3D geometry processing and media processing via graphics core(s) 2380A-2380N. In at least one embodiment, for 3D geometry processing commands, command streamer 2303 supplies commands to geometry pipeline 2336. In at least one embodiment, for at least some media processing commands, command streamer 2303 supplies commands to a video front end
35 2334, which couples with a media engine 2337. In at least one embodiment, media engine 2337 includes a Video Quality Engine (VQE) 2330 for video and image post-processing and

5 a multi-format encode/decode (MFX) 2333 engine to provide hardware-accelerated media data encode and decode. In at least one embodiment, geometry pipeline 2336 and media engine 2337 each generate execution threads for thread execution resources provided by at least one graphics core 2380A.

[00338] In at least one embodiment, graphics processor 2300 includes scalable thread
10 execution resources featuring modular cores 2380A-2380N (sometimes referred to as core slices), each having multiple sub-cores 2350A-2350N, 2360A-2360N (sometimes referred to as core sub-slices). In at least one embodiment, graphics processor 2300 can have any number of graphics cores 2380A through 2380N. In at least one embodiment, graphics processor 2300 includes a graphics core 2380A having at least a first sub-core 2350A and a
15 second sub-core 2360A. In at least one embodiment, graphics processor 2300 is a low power processor with a single sub-core (e.g., 2350A). In at least one embodiment, graphics processor 2300 includes multiple graphics cores 2380A-2380N, each including a set of first sub-cores 2350A-2350N and a set of second sub-cores 2360A-2360N. In at least one embodiment, each sub-core in first sub-cores 2350A-2350N includes at least a first set of
20 execution units 2352A-2352N and media/texture samplers 2354A-2354N. In at least one embodiment, each sub-core in second sub-cores 2360A-2360N includes at least a second set of execution units 2362A-2362N and samplers 2364A-2364N. In at least one embodiment, each sub-core 2350A-2350N, 2360A-2360N shares a set of shared resources 2370A-2370N. In at least one embodiment, shared resources include shared cache memory and pixel
25 operation logic.

[00339] FIG. 24 is a block diagram illustrating micro-architecture for a processor 2400 that may include logic circuits to perform instructions, according to at least one embodiment. In at least one embodiment, processor 2400 may perform instructions, including x86 instructions, ARM instructions, specialized instructions for application-specific integrated
30 circuits (ASICs), etc. In at least one embodiment, processor 2410 may include registers to store packed data, such as 64-bit wide MMX™ registers in microprocessors enabled with MMX technology from Intel Corporation of Santa Clara, Calif. In at least one embodiment, MMX registers, available in both integer and floating point forms, may operate with packed data elements that accompany single instruction, multiple data (“SIMD”) and streaming
35 SIMD extensions (“SSE”) instructions. In at least one embodiment, 128-bit wide XMM registers relating to SSE2, SSE3, SSE4, AVX, or beyond (referred to generically as “SSEx”)

5 technology may hold such packed data operands. In at least one embodiment, processors 2410 may perform instructions to accelerate machine learning or deep learning algorithms, training, or inferencing.

[00340] In at least one embodiment, processor 2400 includes an in-order front end (“front end”) 2401 to fetch instructions to be executed and prepare instructions to be used later in
10 processor pipeline. In at least one embodiment, front end 2401 may include several units. In at least one embodiment, an instruction prefetcher 2426 fetches instructions from memory and feeds instructions to an instruction decoder 2428 which in turn decodes or interprets instructions. For example, in at least one embodiment, instruction decoder 2428 decodes a received instruction into one or more operations called “micro-instructions” or “micro-
15 operations” (also called “micro ops” or “uops”) that machine may execute. In at least one embodiment, instruction decoder 2428 parses instruction into an opcode and corresponding data and control fields that may be used by micro-architecture to perform operations in accordance with at least one embodiment. In at least one embodiment, a trace cache 2430 may assemble decoded uops into program ordered sequences or traces in a uop queue 2434
20 for execution. In at least one embodiment, when trace cache 2430 encounters a complex instruction, a microcode ROM 2432 provides uops needed to complete operation.

[00341] In at least one embodiment, some instructions may be converted into a single micro-op, whereas others need several micro-ops to complete full operation. In at least one embodiment, if more than four micro-ops are needed to complete an instruction, instruction
25 decoder 2428 may access microcode ROM 2432 to perform instruction. In at least one embodiment, an instruction may be decoded into a small number of micro-ops for processing at instruction decoder 2428. In at least one embodiment, an instruction may be stored within microcode ROM 2432 should a number of micro-ops be needed to accomplish operation. In at least one embodiment, trace cache 2430 refers to an entry point programmable logic array
30 (“PLA”) to determine a correct micro-instruction pointer for reading microcode sequences to complete one or more instructions from microcode ROM 2432 in accordance with at least one embodiment. In at least one embodiment, after microcode ROM 2432 finishes sequencing micro-ops for an instruction, front end 2401 of machine may resume fetching micro-ops from trace cache 2430.

35 [00342] In at least one embodiment, out-of-order execution engine (“out of order engine”) 2403 may prepare instructions for execution. In at least one embodiment, out-of-

5 order execution logic has a number of buffers to smooth out and re-order flow of instructions to optimize performance as they go down pipeline and get scheduled for execution. out-of-order execution engine 2403 includes, without limitation, an allocator/register renamer 2440, a memory uop queue 2442, an integer/floating point uop queue 2444, a memory scheduler 2446, a fast scheduler 2402, a slow/general floating point scheduler (“slow/general FP
10 scheduler”) 2404, and a simple floating point scheduler (“simple FP scheduler”) 2406. In at least one embodiment, fast scheduler 2402, slow/general floating point scheduler 2404, and simple floating point scheduler 2406 are also collectively referred to herein as “uop schedulers 2402, 2404, 2406.” Allocator/register renamer 2440 allocates machine buffers and resources that each uop needs in order to execute. In at least one embodiment,
15 allocator/register renamer 2440 renames logic registers onto entries in a register file. In at least one embodiment, allocator/register renamer 2440 also allocates an entry for each uop in one of two uop queues, memory uop queue 2442 for memory operations and integer/floating point uop queue 2444 for non-memory operations, in front of memory scheduler 2446 and uop schedulers 2402, 2404, 2406. In at least one embodiment, uop schedulers 2402, 2404,
20 2406, determine when a uop is ready to execute based on readiness of their dependent input register operand sources and availability of execution resources uops need to complete their operation. In at least one embodiment, fast scheduler 2402 of at least one embodiment may schedule on each half of main clock cycle while slow/general floating point scheduler 2404 and simple floating point scheduler 2406 may schedule once per main processor clock cycle.
25 In at least one embodiment, uop schedulers 2402, 2404, 2406 arbitrate for dispatch ports to schedule uops for execution.

[00343] In at least one embodiment, execution block b11 includes, without limitation, an integer register file/bypass network 2408, a floating point register file/bypass network (“FP register file/bypass network”) 2410, address generation units (“AGUs”) 2412 and 2414, fast
30 Arithmetic Logic Units (ALUs) (“fast ALUs”) 2416 and 2418, a slow Arithmetic Logic Unit (“slow ALU”) 2420, a floating point ALU (“FP”) 2422, and a floating point move unit (“FP move”) 2424. In at least one embodiment, integer register file/bypass network 2408 and floating point register file/bypass network 2410 are also referred to herein as “register files 2408, 2410.” In at least one embodiment, AGUs 2412 and 2414, fast ALUs 2416 and
35 2418, slow ALU 2420, floating point ALU 2422, and floating point move unit 2424 are also referred to herein as “execution units 2412, 2414, 2416, 2418, 2420, 2422, and 2424.” In at least one embodiment, execution block b11 may include, without limitation, any number

5 (including zero) and type of register files, bypass networks, address generation units, and execution units, in any combination.

[00344] In at least one embodiment, register files 2408, 2410 may be arranged between uop schedulers 2402, 2404, 2406, and execution units 2412, 2414, 2416, 2418, 2420, 2422, and 2424. In at least one embodiment, integer register file/bypass network 2408 performs
 10 integer operations. In at least one embodiment, floating point register file/bypass network 2410 performs floating point operations. In at least one embodiment, each of register files 2408, 2410 may include, without limitation, a bypass network that may bypass or forward just completed results that have not yet been written into register file to new dependent uops. In at least one embodiment, register files 2408, 2410 may communicate data with each other.
 15 In at least one embodiment, integer register file/bypass network 2408 may include, without limitation, two separate register files, one register file for low-order thirty-two bits of data and a second register file for high order thirty-two bits of data. In at least one embodiment, floating point register file/bypass network 2410 may include, without limitation, 128-bit wide entries because floating point instructions typically have operands from 64 to 128 bits in
 20 width.

[00345] In at least one embodiment, execution units 2412, 2414, 2416, 2418, 2420, 2422, 2424 may execute instructions. In at least one embodiment, register files 2408, 2410 store integer and floating point data operand values that micro-instructions need to execute. In at least one embodiment, processor 2400 may include, without limitation, any number and
 25 combination of execution units 2412, 2414, 2416, 2418, 2420, 2422, 2424. In at least one embodiment, floating point ALU 2422 and floating point move unit 2424, may execute floating point, MMX, SIMD, AVX and SSE, or other operations, including specialized machine learning instructions. In at least one embodiment, floating point ALU 2422 may include, without limitation, a 64-bit by 64-bit floating point divider to execute divide, square
 30 root, and remainder micro ops. In at least one embodiment, instructions involving a floating point value may be handled with floating point hardware. In at least one embodiment, ALU operations may be passed to fast ALUs 2416, 2418. In at least one embodiment, fast ALUS 2416, 2418 may execute fast operations with an effective latency of half a clock cycle. In at least one embodiment, most complex integer operations go to slow ALU 2420 as slow ALU
 35 2420 may include, without limitation, integer execution hardware for long-latency type of operations, such as a multiplier, shifts, flag logic, and branch processing. In at least one

5 embodiment, memory load/store operations may be executed by AGUS 2412, 2414. In at least one embodiment, fast ALU 2416, fast ALU 2418, and slow ALU 2420 may perform integer operations on 64-bit data operands. In at least one embodiment, fast ALU 2416, fast ALU 2418, and slow ALU 2420 may be implemented to support a variety of data bit sizes including sixteen, thirty-two, 128, 256, etc. In at least one embodiment, floating point
10 ALU 2422 and floating point move unit 2424 may be implemented to support a range of operands having bits of various widths. In at least one embodiment, floating point ALU 2422 and floating point move unit 2424 may operate on 128-bit wide packed data operands in conjunction with SIMD and multimedia instructions.

[00346] In at least one embodiment, uop schedulers 2402, 2404, 2406, dispatch dependent
15 operations before parent load has finished executing. In at least one embodiment, as uops may be speculatively scheduled and executed in processor 2400, processor 2400 may also include logic to handle memory misses. In at least one embodiment, if a data load misses in data cache, there may be dependent operations in flight in pipeline that have left scheduler with temporarily incorrect data. In at least one embodiment, a replay mechanism tracks and
20 re-executes instructions that use incorrect data. In at least one embodiment, dependent operations might need to be replayed and independent ones may be allowed to complete. In at least one embodiment, schedulers and replay mechanism of at least one embodiment of a processor may also be designed to catch instruction sequences for text string comparison operations.

25 [00347] In at least one embodiment, term “registers” may refer to on-board processor storage locations that may be used as part of instructions to identify operands. In at least one embodiment, registers may be those that may be usable from outside of processor (from a programmer's perspective). In at least one embodiment, registers might not be limited to a particular type of circuit. Rather, in at least one embodiment, a register may store data,
30 provide data, and perform functions described herein. In at least one embodiment, registers described herein may be implemented by circuitry within a processor using any number of different techniques, such as dedicated physical registers, dynamically allocated physical registers using register renaming, combinations of dedicated and dynamically allocated physical registers, etc. In at least one embodiment, integer registers store 32-bit integer data.
35 A register file of at least one embodiment also contains eight multimedia SIMD registers for packed data.

5 [00348] FIG. 25 is a block diagram of a processing system, according to at least one embodiment. In at least one embodiment, system 2500 includes one or more processors 2502 and one or more graphics processors 2508, and may be a single processor desktop system, a multiprocessor workstation system, or a server system having a large number of processors 2502 or processor cores 2507. In at least one embodiment, system 2500 is a processing
10 platform incorporated within a system-on-a-chip (SoC) integrated circuit for use in mobile, handheld, or embedded devices.

[00349] In at least one embodiment, system 2500 can include, or be incorporated within a server-based gaming platform, a game console, including a game and media console, a mobile gaming console, a handheld game console, or an online game console. In at least one
15 embodiment, system 2500 is a mobile phone, smart phone, tablet computing device or mobile Internet device. In at least one embodiment, processing system 2500 can also include, couple with, or be integrated within a wearable device, such as a smart watch wearable device, smart eyewear device, augmented reality device, or virtual reality device. In at least one embodiment, processing system 2500 is a television or set top box device having one or more
20 processors 2502 and a graphical interface generated by one or more graphics processors 2508.

[00350] In at least one embodiment, one or more processors 2502 each include one or more processor cores 2507 to process instructions which, when executed, perform operations for system and user software. In at least one embodiment, each of one or more processor
25 cores 2507 is configured to process a specific instruction set 2509. In at least one embodiment, instruction set 2509 may facilitate Complex Instruction Set Computing (CISC), Reduced Instruction Set Computing (RISC), or computing via a Very Long Instruction Word (VLIW). In at least one embodiment, processor cores 2507 may each process a different instruction set 2509, which may include instructions to facilitate emulation of other
30 instruction sets. In at least one embodiment, processor core 2507 may also include other processing devices, such as a Digital Signal Processor (DSP).

[00351] In at least one embodiment, processor 2502 includes cache memory 2504. In at least one embodiment, processor 2502 can have a single internal cache or multiple levels of internal cache. In at least one embodiment, cache memory is shared among various
35 components of processor 2502. In at least one embodiment, processor 2502 also uses an external cache (e.g., a Level-3 (L3) cache or Last Level Cache (LLC)) (not shown), which

5 may be shared among processor cores 2507 using known cache coherency techniques. In at least one embodiment, register file 2506 is additionally included in processor 2502, which may include different types of registers for storing different types of data (e.g., integer registers, floating point registers, status registers, and an instruction pointer register). In at least one embodiment, register file 2506 may include general-purpose registers or other
10 registers.

[00352] In at least one embodiment, one or more processor(s) 2502 are coupled with one or more interface bus(es) 2510 to transmit communication signals such as address, data, or control signals between processor 2502 and other components in system 2500. In at least one embodiment interface bus 2510, in one embodiment, can be a processor bus, such as a
15 version of a Direct Media Interface (DMI) bus. In at least one embodiment, interface 2510 is not limited to a DMI bus, and may include one or more Peripheral Component Interconnect buses (e.g., PCI, PCI Express), memory busses, or other types of interface busses. In at least one embodiment processor(s) 2502 include an integrated memory controller 2516 and a platform controller hub 2530. In at least one embodiment, memory controller 2516 facilitates
20 communication between a memory device and other components of system 2500, while platform controller hub (PCH) 2530 provides connections to I/O devices via a local I/O bus.

[00353] In at least one embodiment, memory device 2520 can be a dynamic random access memory (DRAM) device, a static random access memory (SRAM) device, flash memory device, phase-change memory device, or some other memory device having suitable
25 performance to serve as process memory. In at least one embodiment memory device 2520 can operate as system memory for system 2500, to store data 2522 and instructions 2521 for use when one or more processors 2502 executes an application or process. In at least one embodiment, memory controller 2516 also couples with an optional external graphics processor 2512, which may communicate with one or more graphics processors 2508 in
30 processors 2502 to perform graphics and media operations. In at least one embodiment, a display device 2511 can connect to processor(s) 2502. In at least one embodiment display device 2511 can include one or more of an internal display device, as in a mobile electronic device or a laptop device or an external display device attached via a display interface (e.g., DisplayPort, etc.). In at least one embodiment, display device 2511 can include a head
35 mounted display (HMD) such as a stereoscopic display device for use in virtual reality (VR) applications or augmented reality (AR) applications.

5 [00354] In at least one embodiment, platform controller hub 2530 enables peripherals to connect to memory device 2520 and processor 2502 via a high-speed I/O bus. In at least one embodiment, I/O peripherals include, but are not limited to, an audio controller 2546, a network controller 2534, a firmware interface 2528, a wireless transceiver 2526, touch sensors 2525, a data storage device 2524 (e.g., hard disk drive, flash memory, etc.). In at least one embodiment, data storage device 2524 can connect via a storage interface (e.g., SATA) or via a peripheral bus, such as a Peripheral Component Interconnect bus (e.g., PCI, PCI Express). In at least one embodiment, touch sensors 2525 can include touch screen sensors, pressure sensors, or fingerprint sensors. In at least one embodiment, wireless transceiver 2526 can be a Wi-Fi transceiver, a Bluetooth transceiver, or a mobile network transceiver such as a 15 3G, 4G, or Long Term Evolution (LTE) transceiver. In at least one embodiment, firmware interface 2528 enables communication with system firmware, and can be, for example, a unified extensible firmware interface (UEFI). In at least one embodiment, network controller 2534 can enable a network connection to a wired network. In at least one embodiment, a high-performance network controller (not shown) couples with interface bus 2510. In at least one embodiment, audio controller 2546 is a multi-channel high definition audio controller. In at least one embodiment, system 2500 includes an optional legacy I/O controller 2540 for coupling legacy (e.g., Personal System 2 (PS/2)) devices to system. In at least one embodiment, platform controller hub 2530 can also connect to one or more Universal Serial Bus (USB) controllers 2542 connect input devices, such as keyboard and mouse 2543 25 combinations, a camera 2544, or other USB input devices.

[00355] In at least one embodiment, an instance of memory controller 2516 and platform controller hub 2530 may be integrated into a discreet external graphics processor, such as external graphics processor 2512. In at least one embodiment, platform controller hub 2530 and/or memory controller 2516 may be external to one or more processor(s) 2502. For 30 example, in at least one embodiment, system 2500 can include an external memory controller 2516 and platform controller hub 2530, which may be configured as a memory controller hub and peripheral controller hub within a system chipset that is in communication with processor(s) 2502.

[00356] FIG. 26 is a block diagram of a processor 2600 having one or more processor 35 cores 2602A-2602N, an integrated memory controller 2614, and an integrated graphics processor 2608, according to at least one embodiment. In at least one embodiment, processor

5 2600 can include additional cores up to and including additional core 2602N represented by dashed lined boxes. In at least one embodiment, each of processor cores 2602A-2602N includes one or more internal cache units 2604A-2604N. In at least one embodiment, each processor core also has access to one or more shared cached units 2606.

[00357] In at least one embodiment, internal cache units 2604A-2604N and shared cache
10 units 2606 represent a cache memory hierarchy within processor 2600. In at least one embodiment, cache memory units 2604A-2604N may include at least one level of instruction and data cache within each processor core and one or more levels of shared mid-level cache, such as a Level 2 (L2), Level 3 (L3), Level 4 (L4), or other levels of cache, where a highest level of cache before external memory is classified as an LLC. In at least one embodiment,
15 cache coherency logic maintains coherency between various cache units 2606 and 2604A-2604N.

[00358] In at least one embodiment, processor 2600 may also include a set of one or more bus controller units 2616 and a system agent core 2610. In at least one embodiment, one or more bus controller units 2616 manage a set of peripheral buses, such as one or more PCI or
20 PCI express busses. In at least one embodiment, system agent core 2610 provides management functionality for various processor components. In at least one embodiment, system agent core 2610 includes one or more integrated memory controllers 2614 to manage access to various external memory devices (not shown).

[00359] In at least one embodiment, one or more of processor cores 2602A-2602N include
25 support for simultaneous multi-threading. In at least one embodiment, system agent core 2610 includes components for coordinating and operating cores 2602A-2602N during multi-threaded processing. In at least one embodiment, system agent core 2610 may additionally include a power control unit (PCU), which includes logic and components to regulate one or more power states of processor cores 2602A-2602N and graphics processor 2608.

30 [00360] In at least one embodiment, processor 2600 additionally includes graphics processor 2608 to execute graphics processing operations. In at least one embodiment, graphics processor 2608 couples with shared cache units 2606, and system agent core 2610, including one or more integrated memory controllers 2614. In at least one embodiment, system agent core 2610 also includes a display controller 2611 to drive graphics processor
35 output to one or more coupled displays. In at least one embodiment, display controller 2611

5 may also be a separate module coupled with graphics processor 2608 via at least one interconnect, or may be integrated within graphics processor 2608.

[00361] In at least one embodiment, a ring based interconnect unit 2612 is used to couple internal components of processor 2600. In at least one embodiment, an alternative interconnect unit may be used, such as a point-to-point interconnect, a switched interconnect,
10 or other techniques. In at least one embodiment, graphics processor 2608 couples with ring interconnect 2612 via an I/O link 2613.

[00362] In at least one embodiment, I/O link 2613 represents at least one of multiple varieties of I/O interconnects, including an on package I/O interconnect which facilitates communication between various processor components and a high-performance embedded
15 memory module 2618, such as an eDRAM module. In at least one embodiment, each of processor cores 2602A-2602N and graphics processor 2608 use embedded memory modules 2618 as a shared Last Level Cache.

[00363] In at least one embodiment, processor cores 2602A-2602N are homogenous cores
20 executing a common instruction set architecture. In at least one embodiment, processor cores 2602A-2602N are heterogeneous in terms of instruction set architecture (ISA), where one or more of processor cores 2602A-2602N execute a common instruction set, while one or more other cores of processor cores 2602A-26-02N executes a subset of a common instruction set or a different instruction set. In at least one embodiment, processor cores
25 2602A-2602N are heterogeneous in terms of microarchitecture, where one or more cores having a relatively higher power consumption couple with one or more power cores having a lower power consumption. In at least one embodiment, processor 2600 can be implemented on one or more chips or as an SoC integrated circuit.

[00364] FIG. 27 is a block diagram of a graphics processor 2700, which may be a discrete
30 graphics processing unit, or may be a graphics processor integrated with a plurality of processing cores. In at least one embodiment, graphics processor 2700 communicates via a memory mapped I/O interface to registers on graphics processor 2700 and with commands placed into memory. In at least one embodiment, graphics processor 2700 includes a memory interface 2714 to access memory. In at least one embodiment, memory interface 2714 is an

5 interface to local memory, one or more internal caches, one or more shared external caches, and/or to system memory.

[00365] In at least one embodiment, graphics processor 2700 also includes a display controller 2702 to drive display output data to a display device 2720. In at least one embodiment, display controller 2702 includes hardware for one or more overlay planes for display device 2720 and composition of multiple layers of video or user interface elements. In at least one embodiment, display device 2720 can be an internal or external display device. In at least one embodiment, display device 2720 is a head mounted display device, such as a virtual reality (VR) display device or an augmented reality (AR) display device. In at least one embodiment, graphics processor 2700 includes a video codec engine 2706 to encode, decode, or transcode media to, from, or between one or more media encoding formats, including, but not limited to Moving Picture Experts Group (MPEG) formats such as MPEG-2, Advanced Video Coding (AVC) formats such as H.264/MPEG-4 AVC, as well as the Society of Motion Picture & Television Engineers (SMPTE) 421M/VC-1, and Joint Photographic Experts Group (JPEG) formats such as JPEG, and Motion JPEG (MJPEG) formats.

[00366] In at least one embodiment, graphics processor 2700 includes a block image transfer (BLIT) engine 2704 to perform two-dimensional (2D) rasterizer operations including, for example, bit-boundary block transfers. However, in at least one embodiment, 2D graphics operations are performed using one or more components of graphics processing engine (GPE) 2710. In at least one embodiment, GPE 2710 is a compute engine for performing graphics operations, including three-dimensional (3D) graphics operations and media operations.

[00367] In at least one embodiment, GPE 2710 includes a 3D pipeline 2712 for performing 3D operations, such as rendering three-dimensional images and scenes using processing functions that act upon 3D primitive shapes (e.g., rectangle, triangle, etc.). 3D pipeline 2712 includes programmable and fixed function elements that perform various tasks and/or spawn execution threads to a 3D/Media sub-system 2715. While 3D pipeline 2712 can be used to perform media operations, in at least one embodiment, GPE 2710 also includes a media pipeline 2716 that is used to perform media operations, such as video post-processing and image enhancement.

5 [00368] In at least one embodiment, media pipeline 2716 includes fixed function or programmable logic units to perform one or more specialized media operations, such as video decode acceleration, video de-interlacing, and video encode acceleration in place of, or on behalf of video codec engine 2706. In at least one embodiment, media pipeline 2716 additionally includes a thread spawning unit to spawn threads for execution on 3D/Media
10 sub-system 2715. In at least one embodiment, spawned threads perform computations for media operations on one or more graphics execution units included in 3D/Media sub-system 2715.

[00369] In at least one embodiment, 3D/Media subsystem 2715 includes logic for executing threads spawned by 3D pipeline 2712 and media pipeline 2716. In at least one
15 embodiment, 3D pipeline 2712 and media pipeline 2716 send thread execution requests to 3D/Media subsystem 2715, which includes thread dispatch logic for arbitrating and dispatching various requests to available thread execution resources. In at least one embodiment, execution resources include an array of graphics execution units to process 3D and media threads. In at least one embodiment, 3D/Media subsystem 2715 includes one or
20 more internal caches for thread instructions and data. In at least one embodiment, subsystem 2715 also includes shared memory, including registers and addressable memory, to share data between threads and to store output data.

[00370] FIG. 28 is a block diagram of a graphics processing engine 2810 of a graphics processor in accordance with at least one embodiment. In at least one embodiment, graphics
25 processing engine (GPE) 2810 is a version of GPE 2710 shown in FIG. 27. In at least one embodiment, media pipeline 2816 is optional and may not be explicitly included within GPE 2810. In at least one embodiment, a separate media and/or image processor is coupled to GPE 2810.

[00371] In at least one embodiment, GPE 2810 is coupled to or includes a command
30 streamer 2803, which provides a command stream to 3D pipeline 2812 and/or media pipelines 2816. In at least one embodiment, command streamer 2803 is coupled to memory, which can be system memory, or one or more of internal cache memory and shared cache memory. In at least one embodiment, command streamer 2803 receives commands from memory and sends commands to 3D pipeline 2812 and/or media pipeline 2816. In at least one
35 embodiment, commands are instructions, primitives, or micro-operations fetched from a ring buffer, which stores commands for 3D pipeline 2812 and media pipeline 2816. In at least one

5 embodiment, a ring buffer can additionally include batch command buffers storing batches of multiple commands. In at least one embodiment, commands for 3D pipeline 2812 can also include references to data stored in memory, such as but not limited to vertex and geometry data for 3D pipeline 2812 and/or image data and memory objects for media pipeline 2816. In at least one embodiment, 3D pipeline 2812 and media pipeline 2816 process commands and
10 data by performing operations or by dispatching one or more execution threads to a graphics core array 2814. In at least one embodiment, graphics core array 2814 includes one or more blocks of graphics cores (e.g., graphics core(s) 2815A, graphics core(s) 2815B), each block including one or more graphics cores. In at least one embodiment, each graphics core
15 includes a set of graphics execution resources that includes general-purpose and graphics specific execution logic to perform graphics and compute operations, as well as fixed function texture processing and/or machine learning and artificial intelligence acceleration logic.

[00372] In at least one embodiment, 3D pipeline 2812 includes fixed function and programmable logic to process one or more shader programs, such as vertex shaders,
20 geometry shaders, pixel shaders, fragment shaders, compute shaders, or other shader programs, by processing instructions and dispatching execution threads to graphics core array 2814. In at least one embodiment, graphics core array 2814 provides a unified block of execution resources for use in processing shader programs. In at least one embodiment, multi-purpose execution logic (e.g., execution units) within graphics core(s) 2815A-2815B of
25 graphic core array 2814 includes support for various 3D API shader languages and can execute multiple simultaneous execution threads associated with multiple shaders.

[00373] In at least one embodiment, graphics core array 2814 also includes execution logic to perform media functions, such as video and/or image processing. In at least one embodiment, execution units additionally include general-purpose logic that is programmable
30 to perform parallel general-purpose computational operations, in addition to graphics processing operations.

[00374] In at least one embodiment, output data generated by threads executing on graphics core array 2814 can output data to memory in a unified return buffer (URB) 2818. URB 2818 can store data for multiple threads. In at least one embodiment, URB 2818 may be
35 used to send data between different threads executing on graphics core array 2814. In at least

5 one embodiment, URB 2818 may additionally be used for synchronization between threads on graphics core array 2814 and fixed function logic within shared function logic 2820.

[00375] In at least one embodiment, graphics core array 2814 is scalable, such that graphics core array 2814 includes a variable number of graphics cores, each having a variable number of execution units based on a target power and performance level of GPE 2810. In at
10 least one embodiment, execution resources are dynamically scalable, such that execution resources may be enabled or disabled as needed.

[00376] In at least one embodiment, graphics core array 2814 is coupled to shared function logic 2820 that includes multiple resources that are shared between graphics cores in graphics core array 2814. In at least one embodiment, shared functions performed by shared function
15 logic 2820 are embodied in hardware logic units that provide specialized supplemental functionality to graphics core array 2814. In at least one embodiment, shared function logic 2820 includes but is not limited to sampler 2821, math 2822, and inter-thread communication (ITC) 2823 logic. In at least one embodiment, one or more cache(s) 2825 are included in or couple to shared function logic 2820.

20 [00377] In at least one embodiment, a shared function is used if demand for a specialized function is insufficient for inclusion within graphics core array 2814. In at least one embodiment, a single instantiation of a specialized function is used in shared function logic 2820 and shared among other execution resources within graphics core array 2814. In at least one embodiment, specific shared functions within shared function logic 2820 that are used
25 extensively by graphics core array 2814 may be included within shared function logic 2816 within graphics core array 2814. In at least one embodiment, shared function logic 2816 within graphics core array 2814 can include some or all logic within shared function logic 2820. In at least one embodiment, all logic elements within shared function logic 2820 may be duplicated within shared function logic 2816 of graphics core array 2814. In at least one
30 embodiment, shared function logic 2820 is excluded in favor of shared function logic 2816 within graphics core array 2814.

[00378] FIG. 29 is a block diagram of hardware logic of a graphics processor core 2900, according to at least one embodiment described herein. In at least one embodiment, graphics processor core 2900 is included within a graphics core array. In at least one embodiment,
35 graphics processor core 2900, sometimes referred to as a core slice, can be one or multiple

5 graphics cores within a modular graphics processor. In at least one embodiment, graphics processor core 2900 is exemplary of one graphics core slice, and a graphics processor as described herein may include multiple graphics core slices based on target power and performance envelopes. In at least one embodiment, each graphics core 2900 can include a fixed function block 2930 coupled with multiple sub-cores 2901A-2901F, also referred to as
10 sub-slices, that include modular blocks of general-purpose and fixed function logic.

[00379] In at least one embodiment, fixed function block 2930 includes a geometry/fixed function pipeline 2936 that can be shared by all sub-cores in graphics processor 2900, for example, in lower performance and/or lower power graphics processor implementations. In at least one embodiment, geometry/fixed function pipeline 2936 includes a 3D fixed function
15 pipeline, a video front-end unit, a thread spawner and thread dispatcher, and a unified return buffer manager, which manages unified return buffers.

[00380] In at least one embodiment fixed function block 2930 also includes a graphics SoC interface 2937, a graphics microcontroller 2938, and a media pipeline 2939. Graphics SoC interface 2937 provides an interface between graphics core 2900 and other processor
20 cores within a system on a chip integrated circuit. In at least one embodiment, graphics microcontroller 2938 is a programmable sub-processor that is configurable to manage various functions of graphics processor 2900, including thread dispatch, scheduling, and pre-emption. In at least one embodiment, media pipeline 2939 includes logic to facilitate decoding, encoding, pre-processing, and/or post-processing of multimedia data, including image and
25 video data. In at least one embodiment, media pipeline 2939 implement media operations via requests to compute or sampling logic within sub-cores 2901-2901F.

[00381] In at least one embodiment, SoC interface 2937 enables graphics core 2900 to communicate with general-purpose application processor cores (e.g., CPUs) and/or other components within an SoC, including memory hierarchy elements such as a shared last level
30 cache memory, system RAM, and/or embedded on-chip or on-package DRAM. In at least one embodiment, SoC interface 2937 can also enable communication with fixed function devices within an SoC, such as camera imaging pipelines, and enables use of and/or implements global memory atomics that may be shared between graphics core 2900 and CPUs within an SoC. In at least one embodiment, SoC interface 2937 can also implement
35 power management controls for graphics core 2900 and enable an interface between a clock domain of graphic core 2900 and other clock domains within an SoC. In at least one

5 embodiment, SoC interface 2937 enables receipt of command buffers from a command streamer and global thread dispatcher that are configured to provide commands and instructions to each of one or more graphics cores within a graphics processor. In at least one embodiment, commands and instructions can be dispatched to media pipeline 2939, when media operations are to be performed, or a geometry and fixed function pipeline (e.g.,
10 geometry and fixed function pipeline 2936, geometry and fixed function pipeline 2914) when graphics processing operations are to be performed.

[00382] In at least one embodiment, graphics microcontroller 2938 can be configured to perform various scheduling and management tasks for graphics core 2900. In at least one embodiment, graphics microcontroller 2938 can perform graphics and/or compute
15 workload scheduling on various graphics parallel engines within execution unit (EU) arrays 2902A-2902F, 2904A-2904F within sub-cores 2901A-2901F. In at least one embodiment, host software executing on a CPU core of an SoC including graphics core 2900 can submit workloads one of multiple graphic processor doorbells, which invokes a scheduling operation on an appropriate graphics engine. In at least one embodiment,
20 scheduling operations include determining which workload to run next, submitting a workload to a command streamer, pre-empting existing workloads running on an engine, monitoring progress of a workload, and notifying host software when a workload is complete. In at least one embodiment, graphics microcontroller 2938 can also facilitate low-power or idle states for graphics core 2900, providing graphics core 2900 with an ability to save and
25 restore registers within graphics core 2900 across low-power state transitions independently from an operating system and/or graphics driver software on a system.

[00383] In at least one embodiment, graphics core 2900 may have greater than or fewer than illustrated sub-cores 2901A-2901F, up to N modular sub-cores. For each set of N sub-cores, in at least one embodiment, graphics core 2900 can also include shared function logic
30 2910, shared and/or cache memory 2912, a geometry/fixed function pipeline 2914, as well as additional fixed function logic 2916 to accelerate various graphics and compute processing operations. In at least one embodiment, shared function logic 2910 can include logic units (e.g., sampler, math, and/or inter-thread communication logic) that can be shared by each N sub-cores within graphics core 2900. Shared and/or cache memory 2912 can be a last-level
35 cache for N sub-cores 2901A-2901F within graphics core 2900 and can also serve as shared memory that is accessible by multiple sub-cores. In at least one embodiment, geometry/fixed

5 function pipeline 2914 can be included instead of geometry/fixed function pipeline 2936 within fixed function block 2930 and can include same or similar logic units.

[00384] In at least one embodiment, graphics core 2900 includes additional fixed function logic 2916 that can include various fixed function acceleration logic for use by graphics core 2900. In at least one embodiment, additional fixed function logic 2916 includes an
 10 additional geometry pipeline for use in position only shading. In position-only shading, at least two geometry pipelines exist, whereas in a full geometry pipeline within geometry/fixed function pipeline 2916, 2936, and a cull pipeline, which is an additional geometry pipeline which may be included within additional fixed function logic 2916. In at least one
 15 embodiment, cull pipeline is a trimmed down version of a full geometry pipeline. In at least one embodiment, a full pipeline and a cull pipeline can execute different instances of an application, each instance having a separate context. In at least one embodiment, position only shading can hide long cull runs of discarded triangles, enabling shading to be completed earlier in some instances. For example, in at least one embodiment, cull pipeline logic within additional fixed function logic 2916 can execute position shaders in parallel with a main
 20 application and generally generates critical results faster than a full pipeline, as cull pipeline fetches and shades position attribute of vertices, without performing rasterization and rendering of pixels to a frame buffer. In at least one embodiment, cull pipeline can use generated critical results to compute visibility information for all triangles without regard to whether those triangles are culled. In at least one embodiment, full pipeline (which in this
 25 instance may be referred to as a replay pipeline) can consume visibility information to skip culled triangles to shade only visible triangles that are finally passed to a rasterization phase.

[00385] In at least one embodiment, additional fixed function logic 2916 can also include machine-learning acceleration logic, such as fixed function matrix multiplication logic, for implementations including optimizations for machine learning training or inferencing.

30 [00386] In at least one embodiment, within each graphics sub-core 2901A-2901F includes a set of execution resources that may be used to perform graphics, media, and compute operations in response to requests by graphics pipeline, media pipeline, or shader programs. In at least one embodiment, graphics sub-cores 2901A-2901F include multiple EU arrays 2902A-2902F, 2904A-2904F, thread dispatch and inter-thread communication (TD/IC) logic
 35 2903A-2903F, a 3D (e.g., texture) sampler 2905A-2905F, a media sampler 2906A-2906F, a shader processor 2907A-2907F, and shared local memory (SLM) 2908A-2908F. EU

5 arrays 2902A-2902F, 2904A-2904F each include multiple execution units, which are general-purpose graphics processing units capable of performing floating-point and integer/fixed-point logic operations in service of a graphics, media, or compute operation, including graphics, media, or compute shader programs. In at least one embodiment, TD/IC logic 2903A-2903F performs local thread dispatch and thread control operations for execution

10 units within a sub-core and facilitate communication between threads executing on execution units of a sub-core. In at least one embodiment, 3D sampler 2905A-2905F can read texture or other 3D graphics related data into memory. In at least one embodiment, 3D sampler can read texture data differently based on a configured sample state and texture format associated with a given texture. In at least one embodiment, media sampler 2906A-2906F can perform

15 similar read operations based on a type and format associated with media data. In at least one embodiment, each graphics sub-core 2901A-2901F can alternately include a unified 3D and media sampler. In at least one embodiment, threads executing on execution units within each of sub-cores 2901A-2901F can make use of shared local memory 2908A-2908F within each sub-core, to enable

20 threads executing within a thread group to execute using a common pool of on-chip memory.

[00387] FIGS. 30A-30B illustrate thread execution logic 3000 including an array of processing elements of a graphics processor core according to at least one embodiment. FIG. 30A illustrates at least one embodiment, in which thread execution logic 3000 is used. FIG. 30B illustrates exemplary internal details of an execution unit, according to at least one

25 embodiment.

[00388] As illustrated in FIG. 30A, in at least one embodiment, thread execution logic 3000 includes a shader processor 3002, a thread dispatcher 3004, instruction cache 3006, a scalable execution unit array including a plurality of execution units 3008A-3008N, a sampler 3010, a data cache 3012, and a data port 3014. In at least one embodiment a scalable

30 execution unit array can dynamically scale by enabling or disabling one or more execution units (e.g., any of execution unit 3008A, 3008B, 3008C, 3008D, through 3008N-1 and 3008N) based on computational requirements of a workload, for example. In at least one embodiment, scalable execution units are interconnected via an interconnect fabric that links to each of execution unit. In at least one embodiment, thread execution logic 3000 includes

35 one or more connections to memory, such as system memory or cache memory, through one or more of instruction cache 3006, data port 3014, sampler 3010, and execution units 3008A-

- 5 3008N. In at least one embodiment, each execution unit (e.g., 3008A) is a stand-alone programmable general-purpose computational unit that is capable of executing multiple simultaneous hardware threads while processing multiple data elements in parallel for each thread. In at least one embodiment, array of execution units 3008A-3008N is scalable to include any number individual execution units.
- 10 [00389] In at least one embodiment, execution units 3008A-3008N are primarily used to execute shader programs. In at least one embodiment, shader processor 3002 can process various shader programs and dispatch execution threads associated with shader programs via a thread dispatcher 3004. In at least one embodiment, thread dispatcher 3004 includes logic to
- 15 requested threads on one or more execution units in execution units 3008A-3008N. For example, in at least one embodiment, a geometry pipeline can dispatch vertex, tessellation, or geometry shaders to thread execution logic for processing. In at least one embodiment, thread dispatcher 3004 can also process runtime thread spawning requests from executing shader programs.
- 20 [00390] In at least one embodiment, execution units 3008A-3008N support an instruction set that includes native support for many standard 3D graphics shader instructions, such that shader programs from graphics libraries (e.g., Direct 3D and OpenGL) are executed with a minimal translation. In at least one embodiment, execution units support vertex and
- 25 geometry processing (e.g., vertex programs, geometry programs, vertex shaders), pixel processing (e.g., pixel shaders, fragment shaders) and general-purpose processing (e.g., compute and media shaders). In at least one embodiment, each of execution units 3008A-3008N, which include one or more arithmetic logic units (ALUs), is capable of multi-issue single instruction multiple data (SIMD) execution and multi-threaded operation enables an efficient execution environment despite higher latency memory accesses. In at least one
- 30 embodiment, each hardware thread within each execution unit has a dedicated high-bandwidth register file and associated independent thread-state. In at least one embodiment, execution is multi-issue per clock to pipelines capable of integer, single and double precision floating point operations, SIMD branch capability, logical operations, transcendental operations, and other miscellaneous operations. In at least one embodiment, while waiting
- 35 for data from memory or one of shared functions, dependency logic within execution units 3008A-3008N causes a waiting thread to sleep until requested data has been returned. In at

5 least one embodiment, while awaiting thread is sleeping, hardware resources may be devoted to processing other threads. For example, in at least one embodiment, during a delay associated with a vertex shader operation, an execution unit can perform operations for a pixel shader, fragment shader, or another type of shader program, including a different vertex shader.

10 [00391] In at least one embodiment, each execution unit in execution units 3008A-3008N operates on arrays of data elements. In at least one embodiment, a number of data elements is "execution size," or number of channels for an instruction. In at least one embodiment, an execution channel is a logical unit of execution for data element access, masking, and flow control within instructions. In at least one embodiment, a number of channels may be
15 independent of a number of physical Arithmetic Logic Units (ALUs) or Floating Point Units (FPUs) for a particular graphics processor. In at least one embodiment, execution units 3008A-3008N support integer and floating-point data types.

[00392] In at least one embodiment, an execution unit instruction set includes SIMD instructions. In at least one embodiment, various data elements can be stored as a packed
20 data type in a register and execution unit will process various elements based on data size of elements. For example, in at least one embodiment, when operating on a 256-bit wide vector, 256 bits of a vector are stored in a register and an execution unit operates on a vector as four separate 64-bit packed data elements (Quad-Word (QW) size data elements), eight separate 32-bit packed data elements (Double Word (DW) size data elements), sixteen
25 separate 16-bit packed data elements (Word (W) size data elements), or thirty-two separate 8-bit data elements (byte (B) size data elements). However, in at least one embodiment, different vector widths and register sizes are possible.

[00393] In at least one embodiment, one or more execution units can be combined into a fused execution unit 3009A-3009N having thread control logic (3007A-3007N) that is
30 common to fused EUs. In at least one embodiment, multiple EUs can be fused into an EU group. In at least one embodiment, each EU in fused EU group can be configured to execute a separate SIMD hardware thread. Number of EUs in a fused EU group can vary according to various embodiments. In at least one embodiment, various SIMD widths can be performed per-EU, including but not limited to SIMD8, SIMD16, and SIMD32. In at least one
35 embodiment, each fused graphics execution unit 3009A-3009N includes at least two execution units. For example, in at least one embodiment, fused execution unit 3009A

5 includes a first EU 3008A, second EU 3008B, and thread control logic 3007A that is common to first EU 3008A and second EU 3008B. In at least one embodiment, thread control logic 3007A controls threads executed on fused graphics execution unit 3009A, allowing each EU within fused execution units 3009A-3009N to execute using a common instruction pointer register.

10 [00394] In at least one embodiment, one or more internal instruction caches (e.g., 3006) are included in thread execution logic 3000 to cache thread instructions for execution units. In at least one embodiment, one or more data caches (e.g., 3012) are included to cache thread data during thread execution. In at least one embodiment, a sampler 3010 is included to provide texture sampling for 3D operations and media sampling for media operations. In at
15 least one embodiment, sampler 3010 includes specialized texture or media sampling functionality to process texture or media data during sampling process before providing sampled data to an execution unit.

[00395] During execution, in at least one embodiment, graphics and media pipelines send thread initiation requests to thread execution logic 3000 via thread spawning and dispatch
20 logic. In at least one embodiment, once a group of geometric objects has been processed and rasterized into pixel data, pixel processor logic (e.g., pixel shader logic, fragment shader logic, etc.) within shader processor 3002 is invoked to further compute output information and cause results to be written to output surfaces (e.g., color buffers, depth buffers, stencil buffers, etc.). In at least one embodiment, a pixel shader or fragment shader calculates values
25 of various vertex attributes that are to be interpolated across a rasterized object. In at least one embodiment, pixel processor logic within shader processor 3002 then executes an application programming interface (API)-supplied pixel or fragment shader program. In at least one embodiment, to execute a shader program, shader processor 3002 dispatches threads to an execution unit (e.g., 3008A) via thread dispatcher 3004. In at least one embodiment, shader
30 processor 3002 uses texture sampling logic in sampler 3010 to access texture data in texture maps stored in memory. In at least one embodiment, arithmetic operations on texture data and input geometry data compute pixel color data for each geometric fragment, or discards one or more pixels from further processing.

[00396] In at least one embodiment, data port 3014 provides a memory access mechanism
35 for thread execution logic 3000 to output processed data to memory for further processing on a graphics processor output pipeline. In at least one embodiment, data port 3014 includes or

5 couples to one or more cache memories (e.g., data cache 3012) to cache data for memory access via a data port.

[00397] As illustrated in FIG. 30B, in at least one embodiment, a graphics execution unit 3008 can include an instruction fetch unit 3037, a general register file array (GRF) 3024, an architectural register file array (ARF) 3026, a thread arbiter 3022, a send unit 3030, a branch
10 unit 3032, a set of SIMD floating point units (FPUs) 3034, and In at least one embodiment a set of dedicated integer SIMD ALUs 3035. In at least one embodiment, GRF 3024 and ARF 3026 includes a set of general register files and architecture register files associated with each simultaneous hardware thread that may be active in graphics execution unit 3008. In at least
15 one embodiment, per thread architectural state is maintained in ARF 3026, while data used during thread execution is stored in GRF 3024. In at least one embodiment, execution state of each thread, including instruction pointers for each thread, can be held in thread-specific registers in ARF 3026.

[00398] In at least one embodiment, graphics execution unit 3008 has an architecture that is a combination of Simultaneous Multi-Threading (SMT) and fine-grained Interleaved
20 Multi-Threading (IMT). In at least one embodiment, architecture has a modular configuration that can be fine-tuned at design time based on a target number of simultaneous threads and number of registers per execution unit, where execution unit resources are divided across logic used to execute multiple simultaneous threads.

[00399] In at least one embodiment, graphics execution unit 3008 can co-issue multiple
25 instructions, which may each be different instructions. In at least one embodiment, thread arbiter 3022 of graphics execution unit thread 3008 can dispatch instructions to one of send unit 3030, branch unit 3042, or SIMD FPU(s) 3034 for execution. In at least one embodiment, each execution thread can access 128 general-purpose registers within GRF 3024, where each register can store 32 bytes, accessible as a SIMD 8-element vector of 32-bit
30 data elements. In at least one embodiment, each execution unit thread has access to 4 Kbytes within GRF 3024, although embodiments are not so limited, and greater or fewer register resources may be provided in other embodiments. In at least one embodiment, up to seven threads can execute simultaneously, although a number of threads per execution unit can also vary according to embodiments. In at least one embodiment, in which seven threads may
35 access 4 Kbytes, GRF 3024 can store a total of 28 Kbytes. In at least one embodiment,

5 flexible addressing modes can permit registers to be addressed together to build effectively wider registers or to represent strided rectangular block data structures.

[00400] In at least one embodiment, memory operations, sampler operations, and other longer-latency system communications are dispatched via "send" instructions that are executed by message passing send unit 3030. In at least one embodiment, branch instructions
10 are dispatched to a dedicated branch unit 3032 to facilitate SIMD divergence and eventual convergence.

[00401] In at least one embodiment graphics execution unit 3008 includes one or more SIMD floating point units (FPU(s)) 3034 to perform floating-point operations. In at least one embodiment, FPU(s) 3034 also support integer computation. In at least one embodiment
15 FPU(s) 3034 can SIMD execute up to M number of 32-bit floating-point (or integer) operations, or SIMD execute up to 2M 16-bit integer or 16-bit floating-point operations. In at least one embodiment, at least one of FPU(s) provides extended math capability to support high-throughput transcendental math functions and double precision 64-bit floating-point. In at least one embodiment, a set of 8-bit integer SIMD ALUs 3035 are also present, and may be
20 specifically optimized to perform operations associated with machine learning computations.

[00402] In at least one embodiment, arrays of multiple instances of graphics execution unit 3008 can be instantiated in a graphics sub-core grouping (e.g., a sub-slice). In at least one embodiment execution unit 3008 can execute instructions across a plurality of execution channels. In at least one embodiment, each thread executed on graphics execution unit 3008
25 is executed on a different channel.

[00403] FIG. 31 illustrates a parallel processing unit ("PPU") 3100, according to at least one embodiment. In at least one embodiment, PPU 3100 is configured with machine-readable code that, if executed by PPU 3100, causes PPU 3100 to perform some or all of processes and techniques described throughout this disclosure. In at least one embodiment,
30 PPU 3100 is a multi-threaded processor that is implemented on one or more integrated circuit devices and that utilizes multithreading as a latency-hiding technique designed to process computer-readable instructions (also referred to as machine-readable instructions or simply instructions) on multiple threads in parallel. In at least one embodiment, a thread refers to a thread of execution and is an instantiation of a set of instructions configured to be executed
35 by PPU 3100. In at least one embodiment, PPU 3100 is a graphics processing unit ("GPU")

5 configured to implement a graphics rendering pipeline for processing three-dimensional (“3D”) graphics data in order to generate two-dimensional (“2D”) image data for display on a display device such as a liquid crystal display (“LCD”) device. In at least one embodiment, PPU 3100 is utilized to perform computations such as linear algebra operations and machine-learning operations. FIG. 31 illustrates an example parallel processor for illustrative purposes
10 only and should be construed as a non-limiting example of processor architectures contemplated within scope of this disclosure and that any suitable processor may be employed to supplement and/or substitute for same.

[00404] In at least one embodiment, one or more PPUs 3100 are configured to accelerate High Performance Computing (“HPC”), data center, and machine learning applications. In at
15 least one embodiment, PPU 3100 is configured to accelerate deep learning systems and applications including following non-limiting examples: autonomous vehicle platforms, deep learning, high-accuracy speech, image, text recognition systems, intelligent video analytics, molecular simulations, drug discovery, disease diagnosis, weather forecasting, big data analytics, astronomy, molecular dynamics simulation, financial modeling, robotics, factory
20 automation, real-time language translation, online search optimizations, and personalized user recommendations, and more.

[00405] In at least one embodiment, PPU 3100 includes, without limitation, an Input/Output (“I/O”) unit 3106, a front-end unit 3110, a scheduler unit 3112, a work
distribution unit 3114, a hub 3116, a crossbar (“Xbar”) 3120, one or more general processing
25 clusters (“GPCs”) 3118, and one or more partition units (“memory partition units”) 3122. In at least one embodiment, PPU 3100 is connected to a host processor or other PPUs 3100 via one or more high-speed GPU interconnects (“GPU interconnects”) 3108. In at least one embodiment, PPU 3100 is connected to a host processor or other peripheral devices via an interconnect 3102. In at least one embodiment, PPU 3100 is connected to a local memory
30 comprising one or more memory devices (“memory”) 3104. In at least one embodiment, memory devices 3104 include, without limitation, one or more dynamic random access memory (“DRAM”) devices. In at least one embodiment, one or more DRAM devices are configured and/or configurable as high-bandwidth memory (“HBM”) subsystems, with multiple DRAM dies stacked within each device.

35 [00406] In at least one embodiment, high-speed GPU interconnect 3108 may refer to a wire-based multi-lane communications link that is used by systems to scale and include one

5 or more PPUs 3100 combined with one or more central processing units (“CPUs”), supports cache coherence between PPUs 3100 and CPUs, and CPU mastering. In at least one embodiment, data and/or commands are transmitted by high-speed GPU interconnect 3108 through hub 3116 to/from other units of PPU 3100 such as one or more copy engines, video encoders, video decoders, power management units, and other components which may not be
10 explicitly illustrated in FIG. 31.

[00407] In at least one embodiment, I/O unit 3106 is configured to transmit and receive communications (e.g., commands, data) from a host processor (not illustrated in FIG. 31) over system bus 3102. In at least one embodiment, I/O unit 3106 communicates with host processor directly via system bus 3102 or through one or more intermediate devices such as a
15 memory bridge. In at least one embodiment, I/O unit 3106 may communicate with one or more other processors, such as one or more of PPUs 3100 via system bus 3102. In at least one embodiment, I/O unit 3106 implements a Peripheral Component Interconnect Express (“PCIe”) interface for communications over a PCIe bus. In at least one embodiment, I/O unit 3106 implements interfaces for communicating with external devices.

20 [00408] In at least one embodiment, I/O unit 3106 decodes packets received via system bus 3102. In at least one embodiment, at least some packets represent commands configured to cause PPU 3100 to perform various operations. In at least one embodiment, I/O unit 3106 transmits decoded commands to various other units of PPU 3100 as specified by commands. In at least one embodiment, commands are transmitted to front-end unit 3110 and/or
25 transmitted to hub 3116 or other units of PPU 3100 such as one or more copy engines, a video encoder, a video decoder, a power management unit, etc. (not explicitly illustrated in FIG. 31). In at least one embodiment, I/O unit 3106 is configured to route communications between and among various logical units of PPU 3100.

[00409] In at least one embodiment, a program executed by host processor encodes a
30 command stream in a buffer that provides workloads to PPU 3100 for processing. In at least one embodiment, a workload comprises instructions and data to be processed by those instructions. In at least one embodiment, buffer is a region in a memory that is accessible (e.g., read/write) by both host processor and PPU 3100 — a host interface unit may be configured to access buffer in a system memory connected to system bus 3102 via memory
35 requests transmitted over system bus 3102 by I/O unit 3106. In at least one embodiment, host processor writes command stream to buffer and then transmits a pointer to start of command

5 stream to PPU 3100 such that front-end unit 3110 receives pointers to one or more command streams and manages one or more command streams, reading commands from command streams and forwarding commands to various units of PPU 3100.

[00410] In at least one embodiment, front-end unit 3110 is coupled to scheduler unit 3112 that configures various GPCs 3118 to process tasks defined by one or more command
10 streams. In at least one embodiment, scheduler unit 3112 is configured to track state information related to various tasks managed by scheduler unit 3112 where state information may indicate which of GPCs 3118 a task is assigned to, whether task is active or inactive, a priority level associated with task, and so forth. In at least one embodiment, scheduler unit 3112 manages execution of a plurality of tasks on one or more of GPCs 3118.

15 [00411] In at least one embodiment, scheduler unit 3112 is coupled to work distribution unit 3114 that is configured to dispatch tasks for execution on GPCs 3118. In at least one embodiment, work distribution unit 3114 tracks a number of scheduled tasks received from scheduler unit 3112 and work distribution unit 3114 manages a pending task pool and an active task pool for each of GPCs 3118. In at least one embodiment, pending task pool
20 comprises a number of slots (e.g., 32 slots) that contain tasks assigned to be processed by a particular GPC 3118; active task pool may comprise a number of slots (e.g., 4 slots) for tasks that are actively being processed by GPCs 3118 such that as one of GPCs 3118 completes execution of a task, that task is evicted from active task pool for GPC 3118 and one of other tasks from pending task pool is selected and scheduled for execution on GPC 3118. In at
25 least one embodiment, if an active task is idle on GPC 3118, such as while waiting for a data dependency to be resolved, then active task is evicted from GPC 3118 and returned to pending task pool while another task in pending task pool is selected and scheduled for execution on GPC 3118.

[00412] In at least one embodiment, work distribution unit 3114 communicates with one
30 or more GPCs 3118 via XBar 3120. In at least one embodiment, XBar 3120 is an interconnect network that couples many of units of PPU 3100 to other units of PPU 3100 and can be configured to couple work distribution unit 3114 to a particular GPC 3118. In at least one embodiment, one or more other units of PPU 3100 may also be connected to XBar 3120 via hub 3116.

5 [00413] In at least one embodiment, tasks are managed by scheduler unit 3112 and dispatched to one of GPCs 3118 by work distribution unit 3114. GPC 3118 is configured to process task and generate results. In at least one embodiment, results may be consumed by other tasks within GPC 3118, routed to a different GPC 3118 via XBar 3120, or stored in memory 3104. In at least one embodiment, results can be written to memory 3104 via
10 partition units 3122, which implement a memory interface for reading and writing data to/from memory 3104. In at least one embodiment, results can be transmitted to another PPU 3104 or CPU via high-speed GPU interconnect 3108. In at least one embodiment, PPU 3100 includes, without limitation, a number U of partition units 3122 that is equal to number of separate and distinct memory devices 3104 coupled to PPU 3100. In at least one
15 embodiment, partition unit 3122 will be described in more detail herein in conjunction with FIG. 33.

[00414] In at least one embodiment, a host processor executes a driver kernel that implements an application programming interface (“API”) that enables one or more applications executing on host processor to schedule operations for execution on PPU 3100.
20 In at least one embodiment, multiple compute applications are simultaneously executed by PPU 3100 and PPU 3100 provides isolation, quality of service (“QoS”), and independent address spaces for multiple compute applications. In at least one embodiment, an application generates instructions (e.g., in form of API calls) that cause driver kernel to generate one or more tasks for execution by PPU 3100 and driver kernel outputs tasks to one or more streams
25 being processed by PPU 3100. In at least one embodiment, each task comprises one or more groups of related threads, which may be referred to as a warp. In at least one embodiment, a warp comprises a plurality of related threads (e.g., 32 threads) that can be executed in parallel. In at least one embodiment, cooperating threads can refer to a plurality of threads including instructions to perform task and that exchange data through shared memory. In at
30 least one embodiment, threads and cooperating threads are described in more detail, in accordance with at least one embodiment, in conjunction with FIG. 33.

[00415] FIG. 32 illustrates a general processing cluster (“GPC”) 3200, according to at least one embodiment. In at least one embodiment, GPC 3200 is GPC 3118 of FIG. 31. In at least one embodiment, each GPC 3200 includes, without limitation, a number of hardware units
35 for processing tasks and each GPC 3200 includes, without limitation, a pipeline manager 3202, a pre-raster operations unit (“PROP”) 3204, a raster engine 3208, a work distribution

5 crossbar (“WDX”) 3216, a memory management unit (“MMU”) 3218, one or more Data Processing Clusters (“DPCs”) 3206, and any suitable combination of parts.

[00416] In at least one embodiment, operation of GPC 3200 is controlled by pipeline manager 3202. In at least one embodiment, pipeline manager 3202 manages configuration of one or more DPCs 3206 for processing tasks allocated to GPC 3200. In at least one
10 embodiment, pipeline manager 3202 configures at least one of one or more DPCs 3206 to implement at least a portion of a graphics rendering pipeline. In at least one embodiment, DPC 3206 is configured to execute a vertex shader program on a programmable streaming multi-processor (“SM”) 3214. In at least one embodiment, pipeline manager 3202 is configured to route packets received from a work distribution unit to appropriate logical units
15 within GPC 3200, in at least one embodiment, and some packets may be routed to fixed function hardware units in PROP 3204 and/or raster engine 3208 while other packets may be routed to DPCs 3206 for processing by a primitive engine 3212 or SM 3214. In at least one embodiment, pipeline manager 3202 configures at least one of DPCs 3206 to implement a neural network model and/or a computing pipeline.

20 [00417] In at least one embodiment, PROP unit 3204 is configured, in at least one embodiment, to route data generated by raster engine 3208 and DPCs 3206 to a Raster Operations (“ROP”) unit in partition unit 3122, described in more detail above in conjunction with FIG. 31. In at least one embodiment, PROP unit 3204 is configured to perform optimizations for color blending, organize pixel data, perform address translations, and more.
25 In at least one embodiment, raster engine 3208 includes, without limitation, a number of fixed function hardware units configured to perform various raster operations, in at least one embodiment, and raster engine 3208 includes, without limitation, a setup engine, a coarse raster engine, a culling engine, a clipping engine, a fine raster engine, a tile coalescing engine, and any suitable combination thereof. In at least one embodiment, setup engine
30 receives transformed vertices and generates plane equations associated with geometric primitive defined by vertices; plane equations are transmitted to coarse raster engine to generate coverage information (e.g., an x, y coverage mask for a tile) for primitive; output of coarse raster engine is transmitted to culling engine where fragments associated with primitive that fail a z-test are culled, and transmitted to a clipping engine where fragments
35 lying outside a viewing frustum are clipped. In at least one embodiment, fragments that survive clipping and culling are passed to fine raster engine to generate attributes for pixel

5 fragments based on plane equations generated by setup engine. In at least one embodiment, output of raster engine 3208 comprises fragments to be processed by any suitable entity such as by a fragment shader implemented within DPC 3206.

[00418] In at least one embodiment, each DPC 3206 included in GPC 3200 comprise, without limitation, an M-Pipe Controller (“MPC”) 3210; primitive engine 3212; one or more
10 SMs 3214; and any suitable combination thereof. In at least one embodiment, MPC 3210 controls operation of DPC 3206, routing packets received from pipeline manager 3202 to appropriate units in DPC 3206. In at least one embodiment, packets associated with a vertex are routed to primitive engine 3212, which is configured to fetch vertex attributes associated with vertex from memory; in contrast, packets associated with a shader program may be
15 transmitted to SM 3214.

[00419] In at least one embodiment, SM 3214 comprises, without limitation, a programmable streaming processor that is configured to process tasks represented by a number of threads. In at least one embodiment, SM 3214 is multi-threaded and configured to execute a plurality of threads (e.g., 32 threads) from a particular group of threads
20 concurrently and implements a Single-Instruction, Multiple-Data (“SIMD”) architecture where each thread in a group of threads (e.g., a warp) is configured to process a different set of data based on same set of instructions. In at least one embodiment, all threads in group of threads execute same instructions. In at least one embodiment, SM 3214 implements a Single-Instruction, Multiple Thread (“SIMT”) architecture wherein each thread in a group of
25 threads is configured to process a different set of data based on same set of instructions, but where individual threads in group of threads are allowed to diverge during execution. In at least one embodiment, a program counter, call stack, and execution state is maintained for each warp, enabling concurrency between warps and serial execution within warps when threads within warp diverge. In another embodiment, a program counter, call stack, and
30 execution state is maintained for each individual thread, enabling equal concurrency between all threads, within and between warps. In at least one embodiment, execution state is maintained for each individual thread and threads executing same instructions may be converged and executed in parallel for better efficiency. At least one embodiment of SM 3214 are described in more detail herein.

35 [00420] In at least one embodiment, MMU 3218 provides an interface between GPC 3200 and memory partition unit (e.g., partition unit 3122 of FIG. 31) and MMU 3218 provides

5 translation of virtual addresses into physical addresses, memory protection, and arbitration of memory requests. In at least one embodiment, MMU 3218 provides one or more translation lookaside buffers (“TLBs”) for performing translation of virtual addresses into physical addresses in memory.

[00421] FIG. 33 illustrates a memory partition unit 3300 of a parallel processing unit (“PPU”), in accordance with at least one embodiment. In at least one embodiment, memory partition unit 3300 includes, without limitation, a Raster Operations (“ROP”) unit 3302; a level two (“L2”) cache 3304; a memory interface 3306; and any suitable combination thereof. Memory interface 3306 is coupled to memory. Memory interface 3306 may implement 32, 64, 128, 1024-bit data buses, or like, for high-speed data transfer. In at least one
10 embodiment, PPU incorporates U memory interfaces 3306, one memory interface 3306 per pair of partition units 3300, where each pair of partition units 3300 is connected to a corresponding memory device. For example, in at least one embodiment, PPU may be connected to up to Y memory devices, such as high bandwidth memory stacks or graphics double-data-rate, version 5, synchronous dynamic random access memory (“GDDR5
15 SDRAM”).
20

[00422] In at least one embodiment, memory interface 3306 implements a high bandwidth memory second generation (“HBM2”) memory interface and Y equals half U. In at least one embodiment, HBM2 memory stacks are located on same physical package as PPU, providing substantial power and area savings compared with conventional GDDR5 SDRAM systems.
25 In at least one embodiment, each HBM2 stack includes, without limitation, four memory dies and Y equals 4, with each HBM2 stack including two 128-bit channels per die for a total of 8 channels and a data bus width of 1024 bits. In at least one embodiment, memory supports Single-Error Correcting Double-Error Detecting (“SECCDED”) Error Correction Code (“ECC”) to protect data. ECC provides higher reliability for compute applications that are
30 sensitive to data corruption.

[00423] In at least one embodiment, PPU implements a multi-level memory hierarchy. In at least one embodiment, memory partition unit 3300 supports a unified memory to provide a single unified virtual address space for central processing unit (“CPU”) and PPU memory, enabling data sharing between virtual memory systems. In at least one embodiment
35 frequency of accesses by a PPU to memory located on other processors is traced to ensure that memory pages are moved to physical memory of PPU that is accessing pages more

5 frequently. In at least one embodiment, high-speed GPU interconnect 3108 supports address translation services allowing PPU to directly access a CPU's page tables and providing full access to CPU memory by PPU.

[00424] In at least one embodiment, copy engines transfer data between multiple PPUs or between PPUs and CPUs. In at least one embodiment, copy engines can generate page faults
10 for addresses that are not mapped into page tables and memory partition unit 3300 then services page faults, mapping addresses into page table, after which copy engine performs transfer. In at least one embodiment, memory is pinned (i.e., non-pageable) for multiple copy engine operations between multiple processors, substantially reducing available memory. In at least one embodiment, with hardware page faulting, addresses can be passed to copy
15 engines without regard as to whether memory pages are resident, and copy process is transparent.

[00425] Data from memory 3104 of FIG. 31 or other system memory is fetched by memory partition unit 3300 and stored in L2 cache 3304, which is located on-chip and is shared between various GPCs, in accordance with at least one embodiment. Each memory
20 partition unit 3300, in at least one embodiment, includes, without limitation, at least a portion of L2 cache associated with a corresponding memory device. In at least one embodiment, lower level caches are implemented in various units within GPCs. In at least one embodiment, each of SMs 3214 may implement a level one ("L1") cache wherein L1 cache is private memory that is dedicated to a particular SM 3214 and data from L2 cache 3304 is
25 fetched and stored in each of L1 caches for processing in functional units of SMs 3214. In at least one embodiment, L2 cache 3304 is coupled to memory interface 3306 and XBar 3120.

[00426] ROP unit 3302 performs graphics raster operations related to pixel color, such as color compression, pixel blending, and more, in at least one embodiment. ROP unit 3302, in at least one embodiment, implements depth testing in conjunction with raster engine 3208,
30 receiving a depth for a sample location associated with a pixel fragment from culling engine of raster engine 3208. In at least one embodiment, depth is tested against a corresponding depth in a depth buffer for a sample location associated with fragment. In at least one embodiment, if fragment passes depth test for sample location, then ROP unit 3302 updates depth buffer and transmits a result of depth test to raster engine 3208. It will be appreciated
35 that number of partition units 3300 may be different than number of GPCs and, therefore, each ROP unit 3302 can, in at least one embodiment, be coupled to each of GPCs. In at least

5 one embodiment, ROP unit 3302 tracks packets received from different GPCs and determines which that a result generated by ROP unit 3302 is routed to through XBar 3120.

[00427] FIG. 34 illustrates a streaming multi-processor (“SM”) 3400, according to at least one embodiment. In at least one embodiment, SM 3400 is SM of FIG. 32. In at least one embodiment, SM 3400 includes, without limitation, an instruction cache 3402; one or more
10 scheduler units 3404; a register file 3408; one or more processing cores (“cores”) 3410; one or more special function units (“SFUs”) 3412; one or more load/store units (“LSUs”) 3414; an interconnect network 3416; a shared memory/level one (“L1”) cache 3418; and any suitable combination thereof. In at least one embodiment, a work distribution unit dispatches tasks for execution on general processing clusters (“GPCs”) of parallel processing units
15 (“PPUs”) and each task is allocated to a particular Data Processing Cluster (“DPC”) within a GPC and, if task is associated with a shader program, task is allocated to one of SMs 3400. In at least one embodiment, scheduler unit 3404 receives tasks from work distribution unit and manages instruction scheduling for one or more thread blocks assigned to SM 3400. In at least one embodiment, scheduler unit 3404 schedules thread blocks for execution as warps
20 of parallel threads, wherein each thread block is allocated at least one warp. In at least one embodiment, each warp executes threads. In at least one embodiment, scheduler unit 3404 manages a plurality of different thread blocks, allocating warps to different thread blocks and then dispatching instructions from plurality of different cooperative groups to various functional units (e.g., processing cores 3410, SFUs 3412, and LSUs 3414) during each clock
25 cycle.

[00428] In at least one embodiment, Cooperative Groups may refer to a programming model for organizing groups of communicating threads that allows developers to express granularity at which threads are communicating, enabling expression of richer, more efficient parallel decompositions. In at least one embodiment, cooperative launch APIs support
30 synchronization amongst thread blocks for execution of parallel algorithms. In at least one embodiment, applications of conventional programming models provide a single, simple construct for synchronizing cooperating threads: a barrier across all threads of a thread block (e.g., syncthreads() function). However, in at least one embodiment, programmers may define groups of threads at smaller than thread block granularities and synchronize within
35 defined groups to enable greater performance, design flexibility, and software reuse in form of collective group-wide function interfaces. In at least one embodiment, Cooperative

5 Groups enables programmers to define groups of threads explicitly at sub-block (i.e., as small as a single thread) and multi-block granularities, and to perform collective operations such as synchronization on threads in a cooperative group. Programming model supports clean composition across software boundaries, so that libraries and utility functions can synchronize safely within their local context without having to make assumptions about
10 convergence. In at least one embodiment, Cooperative Groups primitives enable new patterns of cooperative parallelism, including, without limitation, producer-consumer parallelism, opportunistic parallelism, and global synchronization across an entire grid of thread blocks.

[00429] In at least one embodiment, a dispatch unit 3406 is configured to transmit
15 instructions to one or more of functional units and scheduler unit 3404 includes, without limitation, two dispatch units 3406 that enable two different instructions from same warp to be dispatched during each clock cycle. In at least one embodiment, each scheduler unit 3404 includes a single dispatch unit 3406 or additional dispatch units 3406.

[00430] In at least one embodiment, each SM 3400, in at least one embodiment, includes,
20 without limitation, register file 3408 that provides a set of registers for functional units of SM 3400. In at least one embodiment, register file 3408 is divided between each of functional units such that each functional unit is allocated a dedicated portion of register file 3408. In at least one embodiment, register file 3408 is divided between different warps being executed by SM 3400 and register file 3408 provides temporary storage for operands
25 connected to data paths of functional units. In at least one embodiment, each SM 3400 comprises, without limitation, a plurality of L processing cores 3410. In at least one embodiment, SM 3400 includes, without limitation, a large number (e.g., 128 or more) of distinct processing cores 3410. In at least one embodiment, each processing core 3410, in at least one embodiment, includes, without limitation, a fully-pipelined, single-precision,
30 double-precision, and/or mixed precision processing unit that includes, without limitation, a floating point arithmetic logic unit and an integer arithmetic logic unit. In at least one embodiment, floating point arithmetic logic units implement IEEE 754-2008 standard for floating point arithmetic. In at least one embodiment, processing cores 3410 include, without limitation, 64 single-precision (32-bit) floating point cores, 64 integer cores, 32 double-
35 precision (64-bit) floating point cores, and 8 tensor cores.

5 [00431] Tensor cores are configured to perform matrix operations in accordance with at least one embodiment. In at least one embodiment, one or more tensor cores are included in processing cores 3410. In at least one embodiment, tensor cores are configured to perform deep learning matrix arithmetic, such as convolution operations for neural network training and inferencing. In at least one embodiment, each tensor core operates on a 4x4 matrix and
10 performs a matrix multiply and accumulate operation $D = A \times B + C$, where A, B, C, and D are 4x4 matrices.

[00432] In at least one embodiment, matrix multiply inputs A and B are 16-bit floating point matrices and accumulation matrices C and D are 16-bit floating point or 32-bit floating point matrices. In at least one embodiment, tensor cores operate on 16-bit floating point
15 input data with 32-bit floating point accumulation. In at least one embodiment, 16-bit floating point multiply uses 64 operations and results in a full precision product that is then accumulated using 32-bit floating point addition with other intermediate products for a 4x4x4 matrix multiply. Tensor cores are used to perform much larger two-dimensional or higher dimensional matrix operations, built up from these smaller elements, in at least one
20 embodiment. In at least one embodiment, an API, such as CUDA 9 C++ API, exposes specialized matrix load, matrix multiply and accumulate, and matrix store operations to efficiently use tensor cores from a CUDA-C++ program. In at least one embodiment, at CUDA level, warp-level interface assumes 16x16 size matrices spanning all 32 threads of warp.

25 [00433] In at least one embodiment, each SM 3400 comprises, without limitation, M SFUs 3412 that perform special functions (e.g., attribute evaluation, reciprocal square root, and like). In at least one embodiment, SFUs 3412 include, without limitation, a tree traversal unit configured to traverse a hierarchical tree data structure. In at least one embodiment, SFUs 3412 include, without limitation, a texture unit configured to perform texture map filtering
30 operations. In at least one embodiment, texture units are configured to load texture maps (e.g., a 2D array of texels) from memory and sample texture maps to produce sampled texture values for use in shader programs executed by SM 3400. In at least one embodiment, texture maps are stored in shared memory/L1 cache 3418. In at least one embodiment, texture units implement texture operations such as filtering operations using mip-maps (e.g., texture maps
35 of varying levels of detail), in accordance with at least one embodiment. In at least one embodiment, each SM 3400 includes, without limitation, two texture units.

5 [00434] Each SM 3400 comprises, without limitation, N LSUs 3414 that implement load and store operations between shared memory/L1 cache 3418 and register file 3408, in at least one embodiment. Each SM 3400 includes, without limitation, interconnect network 3416 that connects each of functional units to register file 3408 and LSU 3414 to register file 3408 and shared memory/ L1 cache 3418 in at least one embodiment. In at least one embodiment, 10 interconnect network 3416 is a crossbar that can be configured to connect any of functional units to any of registers in register file 3408 and connect LSUs 3414 to register file 3408 and memory locations in shared memory/L1 cache 3418.

[00435] In at least one embodiment, shared memory/L1 cache 3418 is an array of on-chip memory that allows for data storage and communication between SM 3400 and primitive 15 engine and between threads in SM 3400, in at least one embodiment. In at least one embodiment, shared memory/L1 cache 3418 comprises, without limitation, 128KB of storage capacity and is in path from SM 3400 to partition unit. In at least one embodiment, shared memory/L1 cache 3418, in at least one embodiment, is used to cache reads and writes. In at least one embodiment, one or more of shared memory/L1 cache 3418, L2 cache, and memory 20 are backing stores.

[00436] Combining data cache and shared memory functionality into a single memory block provides improved performance for both types of memory accesses, in at least one embodiment. In at least one embodiment, capacity is used or is usable as a cache by programs that do not use shared memory, such as if shared memory is configured to use half 25 of capacity, texture and load/store operations can use remaining capacity. Integration within shared memory/L1 cache 3418 enables shared memory/L1 cache 3418 to function as a high-throughput conduit for streaming data while simultaneously providing high-bandwidth and low-latency access to frequently reused data, in accordance with at least one embodiment. In at least one embodiment, when configured for general purpose parallel computation, a 30 simpler configuration can be used compared with graphics processing. In at least one embodiment, fixed function graphics processing units are bypassed, creating a much simpler programming model. In general purpose parallel computation configuration, work distribution unit assigns and distributes blocks of threads directly to DPCs, in at least one embodiment. In at least one embodiment, threads in a block execute same program, using a 35 unique thread ID in calculation to ensure each thread generates unique results, using SM 3400 to execute program and perform calculations, shared memory/L1 cache 3418 to

5 communicate between threads, and LSU 3414 to read and write global memory through shared memory/L1 cache 3418 and memory partition unit. In at least one embodiment, when configured for general purpose parallel computation, SM 3400 writes commands that scheduler unit 3404 can use to launch new work on DPCs.

[00437] In at least one embodiment, PPU is included in or coupled to a desktop computer,
10 a laptop computer, a tablet computer, servers, supercomputers, a smart-phone (e.g., a wireless, hand-held device), personal digital assistant (“PDA”), a digital camera, a vehicle, a head mounted display, a hand-held electronic device, and more. In at least one embodiment, PPU is embodied on a single semiconductor substrate. In at least one embodiment, PPU is included in a system-on-a-chip (“SoC”) along with one or more other devices such as
15 additional PPUs, memory, a reduced instruction set computer (“RISC”) CPU, a memory management unit (“MMU”), a digital-to-analog converter (“DAC”), and like.

[00438] In at least one embodiment, PPU may be included on a graphics card that includes one or more memory devices. Graphics card may be configured to interface with a PCIe slot on a motherboard of a desktop computer. In at least one embodiment, PPU may be an
20 integrated graphics processing unit (“iGPU”) included in chipset of motherboard.

[00439] In at least one embodiment, a single semiconductor platform may refer to a sole unitary semiconductor-based integrated circuit or chip. In at least one embodiment, multi-chip modules may be used with increased connectivity which simulate on-chip operation, and make substantial improvements over utilizing a conventional central processing unit (“CPU”) and bus implementation. In at least one embodiment, various modules may also be situated
25 separately or in various combinations of semiconductor platforms per desires of user.

[00440] In at least one embodiment, computer programs in form of machine-readable executable code or computer control logic algorithms are stored in main memory 1404 and/or secondary storage. Computer programs, if executed by one or more processors, enable
30 system 1400 to perform various functions in accordance with at least one embodiment. Memory 1404, storage, and/or any other storage are possible examples of computer-readable media. In at least one embodiment, secondary storage may refer to any suitable storage device or system such as a hard disk drive and/or a removable storage drive, representing a floppy disk drive, a magnetic tape drive, a compact disk drive, digital versatile disk (“DVD”) drive, recording device, universal serial bus (“USB”) flash memory, etc. In at least one
35

5 embodiment, architecture and/or functionality of various previous figures are implemented in context of CPU 1402; parallel processing system 1412; an integrated circuit capable of at least a portion of capabilities of both CPU 1402; parallel processing system 1412; a chipset (e.g., a group of integrated circuits designed to work and sold as a unit for performing related functions, etc.); and any suitable combination of integrated circuit(s).

10 [00441] In at least one embodiment, architecture and/or functionality of various previous figures are implemented in context of a general computer system, a circuit board system, a game console system dedicated for entertainment purposes, an application-specific system, and more. In at least one embodiment, computer system 1400 may take form of a desktop computer, a laptop computer, a tablet computer, servers, supercomputers, a smart-phone (e.g.,
15 a wireless, hand-held device), personal digital assistant (“PDA”), a digital camera, a vehicle, a head mounted display, a hand-held electronic device, a mobile phone device, a television, workstation, game consoles, embedded system, and/or any other type of logic.

[00442] In at least one embodiment, parallel processing system 1412 includes, without limitation, a plurality of parallel processing units (“PPUs”) 1414 and associated memories
20 1416. In at least one embodiment, PPUs 1414 are connected to a host processor or other peripheral devices via an interconnect 1418 and a switch 1420 or multiplexer. In at least one embodiment, parallel processing system 1412 distributes computational tasks across PPUs 1414, which can be parallelizable — for example, as part of distribution of computational tasks across multiple graphics processing unit (“GPU”) thread blocks. In at least one
25 embodiment, memory is shared and accessible (e.g., for read and/or write access) across some or all of PPUs 1414, although such shared memory may incur performance penalties relative to use of local memory and registers resident to a PPU 1414. In at least one embodiment, operation of PPUs 1414 is synchronized through use of a command such as __syncthreads(), wherein all threads in a block (e.g., executed across multiple PPUs 1414) to
30 reach a certain point of execution of code before proceeding.

NETWORKS

[00443] FIG. 35 illustrates a network 3500 for communicating data within a 5G wireless communications network, in accordance with at least one embodiment. In at least one embodiment, network 3500 comprises a base station 3506 having a coverage area 3504, a
35 plurality of mobile devices 3508, and a backhaul network 3502. In at least one embodiment,

5 as shown, base station 3506 establishes uplink and/or downlink connections with mobile devices 3508, which serve to carry data from mobile devices 3508 to base station 3506 and vice-versa. In at least one embodiment, data carried over uplink/downlink connections may include data communicated between mobile devices 3508, as well as data communicated to/from a remote-end (not shown) by way of backhaul network 3502. In at least one
 10 embodiment, term “base station” refers to any component (or collection of components) configured to provide wireless access to a network, such as an enhanced base station (eNB), a macro-cell, a femtocell, a Wi-Fi access point (AP), or other wirelessly enabled devices. In at least one embodiment, base stations may provide wireless access in accordance with one or more wireless communication protocols, e.g., long term evolution (LTE), LTE advanced
 15 (LTE-A), High Speed Packet Access (HSPA), Wi-Fi 802.11a/b/g/n/ac, etc. In at least one embodiment, term “mobile device” refers to any component (or collection of components) capable of establishing a wireless connection with a base station, such as a user equipment (UE), a mobile station (STA), and other wirelessly enabled devices. In some embodiments, network 3500 may comprise various other wireless devices, such as relays, low power nodes,
 20 etc.

[00444] FIG. 36 illustrates a network architecture 3600 for a 5G wireless network, in accordance with at least one embodiment. In at least one embodiment, as shown, network architecture 3600 includes a radio access network (RAN) 3604, an evolved packet core (EPC) 3602, which may be referred to as a core network, and a home network 3616 of a UE
 25 3608 attempting to access RAN 3604. In at least one embodiment, RAN 3604 and EPC 3602 form a serving wireless network. In at least one embodiment, RAN 3604 includes a base station 3606, and EPC 3602 includes a mobility management entity (MME) 3612, a serving gateway (SGW) 3610, and a packet data network (PDN) gateway (PGW) 3614. In at least one embodiment, home network 3616 includes an application server 3618 and a home
 30 subscriber server (HSS) 3620. In at least one embodiment, HSS 3620 may be part of home network 3616, EPC 3602, and/or variations thereof.

[00445] In at least one embodiment, MME 3612 is a termination point in a network for ciphering/integrity protection for NAS signaling and handles security key management. In at least one embodiment, it should be appreciated that term “MME” is used in 4G LTE
 35 networks, and that 5G LTE networks may include a Security Anchor Node (SEAN) or a Security Access Function (SEAF) that performs similar functions. In at least one

5 embodiment, terms “MME,” “SEAN,” and “SEAF” may be used interchangeably. In at least one embodiment, MME 3612 also provides control plane function for mobility between LTE and 2G/3G access networks, as well as an interface to home networks of roaming UEs. In at least one embodiment, SGW 3610 routes and forwards user data packets, while also acting as a mobility anchor for a user plane during handovers. In at least one embodiment, PGW 3614
10 provides connectivity from UEs to external packet data networks by being a point of exit and entry of traffic for UEs. In at least one embodiment, HSS 3620 is a central database that contains user-related and subscription-related information. In at least one embodiment, application server 3618 is a central database that contains user-related information regarding various applications that may utilize and communicate via network architecture 3600.

15 [00446] FIG. 37 is a diagram illustrating some basic functionality of a mobile telecommunications network/system operating in accordance with LTE and 5G principles, in accordance with at least one embodiment. In at least one embodiment, a mobile telecommunications system includes infrastructure equipment comprising base stations 3714 which are connected to a core network 3702, which operates in accordance with a
20 conventional arrangement which will be understood by those acquainted with communications technology. In at least one embodiment, infrastructure equipment 3714 may also be referred to as a base station, network element, enhanced NodeB (eNodeB) or a coordinating entity for example, and provides a wireless access interface to one or more communications devices within a coverage area or cell represented by a broken line 3704,
25 which may be referred to as a radio access network. In at least one embodiment, one or more mobile communications devices 3706 may communicate data via transmission and reception of signals representing data using a wireless access interface. In at least one embodiment, core network 3702 may also provide functionality including authentication, mobility management, charging and so on for communications devices served by a network entity.

30 [00447] In at least one embodiment, mobile communications devices of FIG. 37 may also be referred to as communications terminals, user equipment (UE), terminal devices and so forth, and are configured to communicate with one or more other communications devices served by a same or a different coverage area via a network entity. In at least one embodiment, these communications may be performed by transmitting and receiving signals
35 representing data using a wireless access interface over two-way communications links.

5 [00448] In at least one embodiment, as shown in FIG. 37, one of eNodeBs 3714a is shown in more detail to include a transmitter 3712 for transmitting signals via a wireless access interface to one or more communications devices or UEs 3706, and a receiver 3710 to receive signals from one or more UEs within coverage area 3704. In at least one embodiment, controller 3708 controls transmitter 3712 and receiver 3710 to transmit and receive signals
10 via a wireless access interface. In at least one embodiment, controller 3708 may perform a function of controlling allocation of communications resource elements of a wireless access interface and may in some examples include a scheduler for scheduling transmissions via a wireless access interface for both uplink and downlink.

[00449] In at least one embodiment, an example UE 3706a is shown in more detail to
15 include a transmitter 3720 for transmitting signals on an uplink of a wireless access interface to eNodeB 3714 and a receiver 3718 for receiving signals transmitted by eNodeB 3714 on a downlink via a wireless access interface. In at least one embodiment, transmitter 3720 and receiver 3718 are controlled by a controller 3716.

[00450] FIG. 38 illustrates a radio access network 3800, which may be part of a 5G
20 network architecture, in accordance with at least one embodiment. In at least one embodiment, radio access network 3800 covers a geographic region divided into a number of cellular regions (cells) that can be uniquely identified by a user equipment (UE) based on an identification broadcasted over a geographical area from one access point or base station. In at least one embodiment, macrocells 3840, 3828, and 3816, and a small cell 3830, may
25 include one or more sectors. In at least one embodiment, a sector is a sub-area of a cell and all sectors within one cell are served by a same base station. In at least one embodiment, a single logical identification belonging to that sector can identify a radio link within a sector. In at least one embodiment, multiple sectors within a cell can be formed by groups of antennas with each antenna responsible for communication with UEs in a portion of a cell.

30 [00451] In at least one embodiment, each cell is served by a base station (BS). In at least one embodiment, a base station is a network element in a radio access network responsible for radio transmission and reception in one or more cells to or from a UE. In at least one embodiment, a base station may also be referred to as a base transceiver station (BTS), a radio base station, a radio transceiver, a transceiver function, a basic service set (BSS), an
35 extended service set (ESS), an access point (AP), a Node B (NB), an eNode B (eNB), a gNode B (gNB), or some other suitable terminology. In at least one embodiment, base

5 stations may include a backhaul interface for communication with a backhaul portion of a network. In at least one embodiment, a base station has an integrated antenna or is connected to an antenna or remote radio head (RRH) by feeder cables.

[00452] In at least one embodiment, a backhaul may provide a link between a base station and a core network, and in some examples, a backhaul may provide interconnection between
10 respective base stations. In at least one embodiment, a core network is a part of a wireless communication system that is generally independent of radio access technology used in a radio access network. In at least one embodiment, various types of backhaul interfaces, such as a direct physical connection, a virtual network, or like using any suitable transport network, may be employed. In at least one embodiment, some base stations may be
15 configured as integrated access and backhaul (IAB) nodes, where a wireless spectrum may be used both for access links (i.e., wireless links with UEs), and for backhaul links, which is sometimes referred to as wireless self-backhauling. In at least one embodiment, through wireless self-backhauling, a wireless spectrum utilized for communication between a base station and UE may be leveraged for backhaul communication, enabling fast and easy
20 deployment of highly dense small cell networks, as opposed to requiring each new base station deployment to be outfitted with its own hard-wired backhaul connection.

[00453] In at least one embodiment, high-power base stations 3836 and 3820 are shown in cells 3840 and 3828, and a high-power base station 3810 is shown controlling a remote radio head (RRH) 3812 in cell 3816. In at least one embodiment, cells 3840, 3828, and 3816 may
25 be referred to as large size cells or macrocells. In at least one embodiment, a low-power base station 3834 is shown in small cell 3830 (e.g., a microcell, picocell, femtocell, home base station, home Node B, home eNode B, etc.) which may overlap with one or more macrocells, and may be referred to as a small cell or small size cell. In at least one embodiment, cell sizing can be done according to system design as well as component constraints. In at least
30 one embodiment, a relay node may be deployed to extend size or coverage area of a given cell. In at least one embodiment, radio access network 3800 may include any number of wireless base stations and cells. In at least one embodiment, base stations 3836, 3820, 3810, 3834 provide wireless access points to a core network for any number of mobile apparatuses.

[00454] In at least one embodiment, a quadcopter or drone 3842 may be configured to
35 function as a base station. In at least one embodiment, a cell may not necessarily be

5 stationary, and a geographic area of a cell may move according to a location of a mobile base station such as quadcopter 3842.

[00455] In at least one embodiment, radio access network 3800 supports wireless communications for multiple mobile apparatuses. In at least one embodiment, a mobile apparatus is commonly referred to as user equipment (UE), but may also be referred to as a
10 mobile station (MS), a subscriber station, a mobile unit, a subscriber unit, a wireless unit, a remote unit, a mobile device, a wireless device, a wireless communications device, a remote device, a mobile subscriber station, an access terminal (AT), a mobile terminal, a wireless terminal, a remote terminal, a handset, a terminal, a user agent, a mobile client, a client, or some other suitable terminology. In at least one embodiment, a UE may be an apparatus that
15 provides a user with access to network services.

[00456] In at least one embodiment, a “mobile” apparatus need not necessarily have a capability to move, and may be stationary. In at least one embodiment, mobile apparatus or mobile device broadly refers to a diverse array of devices and technologies. In at least one embodiment, a mobile apparatus may be a mobile, a cellular (cell) phone, a smart phone, a
20 session initiation protocol (SIP) phone, a laptop, a personal computer (PC), a notebook, a netbook, a smartbook, a tablet, a personal digital assistant (PDA), a broad array of embedded systems, e.g., corresponding to an “Internet of things” (IoT), an automotive or other transportation vehicle, a remote sensor or actuator, a robot or robotics device, a satellite radio, a global positioning system (GPS) device, an object tracking device, a drone, a multi-copter, a quad-copter, a remote control device, a consumer and/or wearable device, such as eyewear,
25 a wearable camera, a virtual reality device, a smart watch, a health or fitness tracker, a digital audio player (e.g., MP3 player), a camera, a game console, a digital home or smart home device such as a home audio, video, and/or multimedia device, an appliance, a vending machine, intelligent lighting, a home security system, a smart meter, a security device, a solar panel or solar array, a municipal infrastructure device controlling electric power (e.g., a smart
30 grid), lighting, water, etc., an industrial automation and enterprise device, a logistics controller, agricultural equipment, military defense equipment, vehicles, aircraft, ships, and weaponry, etc. In at least one embodiment, a mobile apparatus may provide for connected medicine or telemedicine support, i.e., health care at a distance. In at least one embodiment,
35 telehealth devices may include telehealth monitoring devices and telehealth administration devices, whose communication may be given preferential treatment or prioritized access over

5 other types of information, e.g., in terms of prioritized access for transport of critical service data, and/or relevant QoS for transport of critical service data.

[00457] In at least one embodiment, cells of radio access network 3800 may include UEs that may be in communication with one or more sectors of each cell. In at least one embodiment, UEs 3814 and 3808 may be in communication with base station 3810 by way of
10 RRH 3812; UEs 3822 and 3826 may be in communication with base station 3820; UE 3832 may be in communication with low-power base station 3834; UEs 3838 and 3818 may be in communication with base station 3836; and UE 3844 may be in communication with mobile base station 3842. In at least one embodiment, each base station 3810, 3820, 3834, 3836, and 3842 may be configured to provide an access point to a core network (not shown) for all UEs
15 in respective cells and transmissions from a base station (e.g., base station 3836) to one or more UEs (e.g., UEs 3838 and 3818) may be referred to as downlink (DL) transmission, while transmissions from a UE (e.g., UE 3838) to a base station may be referred to as uplink (UL) transmissions. In at least one embodiment, downlink may refer to a point-to-multipoint transmission, which may be referred to as broadcast channel multiplexing. In at least one
20 embodiment, uplink may refer to a point-to-point transmission.

[00458] In at least one embodiment, quadcopter 3842, which may be referred to as a mobile network node, may be configured to function as a UE within cell 3840 by communicating with base station 3836. In at least one embodiment, multiple UEs (e.g., UEs 3822 and 3826) may communicate with each other using peer to peer (P2P) or sidelink
25 signals 3824, which may bypass a base station such as base station 3820.

[00459] In at least one embodiment, ability for a UE to communicate while moving, independent of its location, is referred to as mobility. In at least one embodiment, a mobility management entity (MME) sets up, maintains, and releases various physical channels between a UE and a radio access network. In at least one embodiment, DL-based mobility or
30 UL-based mobility may be utilized by a radio access network 3800 to enable mobility and handovers (i.e., transfer of a UE's connection from one radio channel to another). In at least one embodiment, a UE, in a network configured for DL-based mobility, may monitor various parameters of a signal from its serving cell as well as various parameters of neighboring cells, and, depending on a quality of these parameters, a UE may maintain communication with one
35 or more neighboring cells. In at least one embodiment, if signal quality from a neighboring cell exceeds that from a serving cell for a given amount of time, or if a UE moves from one

5 cell to another, a UE may undertake a handoff or handover from a serving cell to a neighboring (target) cell. In at least one embodiment, UE 3818 (illustrated as a vehicle, although any suitable form of UE may be used) may move from a geographic area corresponding to a cell, such as serving cell 3840, to a geographic area corresponding to a neighbor cell, such as neighbor cell 3816. In at least one embodiment, UE 3818 may transmit
10 a reporting message to its serving base station 3836 indicating its condition when signal strength or quality from a neighbor cell 3816 exceeds that of its serving cell 3840 for a given amount of time. In at least one embodiment, UE 3818 may receive a handover command, and may undergo a handover to cell 3816.

[00460] In at least one embodiment, UL reference signals from each UE may be utilized
15 by a network configured for UL-based mobility to select a serving cell for each UE. In at least one embodiment, base stations 3836, 3820, and 3810/3812 may broadcast unified synchronization signals (e.g., unified Primary Synchronization Signals (PSSs), unified Secondary Synchronization Signals (SSSs) and unified Physical Broadcast Channels (PBCH)). In at least one embodiment, UEs 3838, 3818, 3822, 3826, 3814, and 3808 may
20 receive unified synchronization signals, derive a carrier frequency and slot timing from synchronization signals, and in response to deriving timing, transmit an uplink pilot or reference signal. In at least one embodiment, two or more cells (e.g., base stations 3836 and 3810/3812) within radio access network 3800 may concurrently receive an uplink pilot signal transmitted by a UE (e.g., UE 3818). In at least one embodiment, cells may measure a
25 strength of a pilot signal, and a radio access network (e.g., one or more of base stations 3836 and 3810/3812 and/or a central node within a core network) may determine a serving cell for UE 3818. In at least one embodiment, a network may continue to monitor an uplink pilot signal transmitted by UE 3818 as UE 3818 moves through radio access network 3800. In at least one embodiment, a network 3800 may handover UE 3818 from a serving cell to a
30 neighboring cell, with or without informing UE 3818, when a signal strength or quality of a pilot signal measured by a neighboring cell exceeds that of a signal strength or quality measured by a serving cell.

[00461] In at least one embodiment, synchronization signals transmitted by base stations 3836, 3820, and 3810/3812 may be unified, but may not identify a particular cell and
35 rather may identify a zone of multiple cells operating on a same frequency and/or with a same timing. In at least one embodiment, zones in 5G networks or other next generation

5 communication networks enable uplink-based mobility framework and improves efficiency of both a UE and a network, since amounts of mobility messages that need to be exchanged between a UE and a network may be reduced.

[00462] In at least one embodiment, air interface in a radio access network 3800 may utilize unlicensed spectrum, licensed spectrum, or shared spectrum. In at least one
10 embodiment, unlicensed spectrum provides for shared use of a portion of a spectrum without need for a government-granted license, however, while compliance with some technical rules is generally still required to access an unlicensed spectrum, generally, any operator or device may gain access. In at least one embodiment, licensed spectrum provides for exclusive use of a portion of a spectrum, generally by virtue of a mobile network operator purchasing a license
15 from a government regulatory body. In at least one embodiment, shared spectrum may fall between licensed and unlicensed spectrum, wherein technical rules or limitations may be required to access a spectrum, but a spectrum may still be shared by multiple operators and/or multiple RATs. In at least one embodiment, for example, a holder of a license for a portion of licensed spectrum may provide licensed shared access (LSA) to share that spectrum with
20 other parties, e.g., with suitable licensee-determined conditions to gain access.

[00463] FIG. 39 provides an example illustration of a 5G mobile communications system in which a plurality of different types of devices is used, in accordance with at least one embodiment. In at least one embodiment, as shown in FIG. 39, a first base station 3918 may be provided to a large cell or macro cell in which transmission of signals is over several
25 kilometers. In at least one embodiment, however, system may also support transmission via a very small cell such as transmitted by a second infrastructure equipment 3916, which transmits and receives signals over a distance of hundreds of meters thereby forming a so called “Pico” cell. In at least one embodiment, a third type of infrastructure equipment 3912 may transmit and receive signals over a distance of tens of meters and therefore can be used
30 to form a so called “Femto” cell.

[00464] In at least one embodiment, also shown in FIG. 39, different types of communications devices may be used to transmit and receive signals via different types of infrastructure equipment 3912, 3916, 3918 and communication of data may be adapted in accordance with different types of infrastructure equipment using different communications
35 parameters. In at least one embodiment, conventionally, a mobile communications device may be configured to communicate data to and from a mobile communications network via

5 available communication resources of network. In at least one embodiment, a wireless access system is configured to provide highest data rates to devices such as smart phones 3906. In at least one embodiment, “internet of things” may be provided in which low power machine type communications devices transmit and receive data at very low power, low bandwidth and may have a low complexity. In at least one embodiment, an example of such a machine
10 type communication device 3914 may communicate via a Pico cell 3916. In at least one embodiment, a very high data rate and a low mobility may be characteristic of communications with, for example, a television 3904, which may be communicating via a Pico cell. In at least one embodiment, a very high data rate and low latency may be required by a virtual reality headset 3908. In at least one embodiment, a relay device 3910 may be
15 deployed to extend size or coverage area of a given cell or network.

[00465] FIG. 40 illustrates an example high level system 4000, in which at least one embodiment may be used. In at least one embodiment, high level system 4000 includes applications 4002, system software + libraries 4004, framework software 4006 and a datacenter infrastructure + resource orchestrator 4008. In at least one embodiment, high level
20 system 4000 may be implemented as a cloud service, physical service, virtual service, network service, and/or variations thereof.

[00466] In at least one embodiment, as shown in FIG. 40, datacenter infrastructure + resource orchestrator 4008 may include 5G radio resource orchestrator 4010, GPU packet processing & I/O 4012, and node computing resources (“node C.R.s”) 4016(1)-4016(N),
25 where “N” represents any whole, positive integer. In at least one embodiment, node C.R.s 4016(1)-4016(N) may include, but are not limited to, any number of central processing units (“CPUs”) or other processors (including accelerators, field programmable gate arrays (FPGAs), graphics processors (“GPUs”), etc.), memory devices (e.g., dynamic read-only memory), storage devices (e.g., solid state or disk drives), network input/output (“NW I/O”) devices, network switches, virtual machines (“VMs”), power modules, and cooling modules,
30 etc. In at least one embodiment, one or more node C.R.s from among node C.R.s 4016(1)-4016(N) may be a server having one or more of above-mentioned computing resources.

[00467] In at least one embodiment, 5G radio resource orchestrator 4010 may configure or otherwise control one or more node C.R.s 4016(1)-4016(N) and/or other various components
35 and resources a 5G network architecture may comprise. In at least one embodiment, 5G radio resource orchestrator 4010 may include a software design infrastructure (“SDI”) management

5 entity for high level system 4000. In at least one embodiment, 5G radio resource orchestrator 4010 may include hardware, software or some combination thereof. In at least one embodiment, 5G radio resource orchestrator 4010 may be utilized to configure or otherwise control various medium access control sublayers, radio access networks, physical layers or sublayers, and/or variations thereof, which may be part of a 5G network
10 architecture. In at least one embodiment, 5G radio resource orchestrator 4010 may configure or allocate grouped compute, network, memory or storage resources to support one or more workloads, which may be executed as part of a 5G network architecture.

[00468] In at least one embodiment, GPU packet processing & I/O 4012 may configure or otherwise process various inputs and outputs, as well as packets such as data packets, which
15 may be transmitted/received as part of a 5G network architecture, which may be implemented by high level system 4000. In at least one embodiment, a packet may be data formatted to be provided by a network and may be typically divided into control information and payload (i.e., user data). In at least one embodiment, types of packets may include Internet Protocol version 4 (IPv4) packets, Internet Protocol version 6 (IPv6) packets, and Ethernet II frame
20 packets. In at least one embodiment, control data of a data packet may be classified into data integrity fields and semantic fields. In at least one embodiment, network connections that a data packet may be received upon include a local area network, a wide-area network, a virtual private network, Internet, an intranet, an extranet, a public switched telephone network, an infrared network, a wireless network, a satellite network and any combination thereof.

25 [00469] In at least one embodiment, framework software 4006 includes an AI Model Architecture + Training + Use Cases 4022. In at least one embodiment, AI Model Architecture + Training + Use Cases 4022 may include tools, services, software or other resources to train one or more machine learning models or predict or infer information using one or more machine learning models according to one or more embodiments. For example,
30 in at least one embodiment, a machine learning model may be trained by calculating weight parameters according to a neural network architecture using software and computing resources described above with respect to high level system 4000. In at least one embodiment, trained machine learning models corresponding to one or more neural networks may be used to infer or predict information using resources described above with respect to
35 high level system 4000 by using weight parameters calculated through one or more training

5 techniques. In at least one embodiment, framework software 4006 may include a framework to support system software + libraries 4004 and applications 4002.

[00470] In at least one embodiment, system software + libraries 4004 or applications 4002 may respectively include web-based service software or applications, such as those provided by Amazon Web Services, Google Cloud and Microsoft Azure. In at least one embodiment,
10 framework software 4006 may include, but is not limited to, a type of free and open-source software web application framework such as Apache Spark™ (hereinafter “Spark”). In at least one embodiment, system software + libraries 4004 may include software used by at least portions of node C.R.s 4016(1)-4016(N). In at least one embodiment, one or more types of software may include, but are not limited to, Internet web page search software, e-mail virus
15 scan software, database software, and streaming video content software.

[00471] In at least one embodiment, PHY 4018 is a set of system software and libraries configured to provide an interface with a physical layer of a wireless technology, which may be a physical layer such as a 5G New Radio (NR) physical layer. In at least one embodiment, an NR physical layer utilizes a flexible and scalable design and may comprise various
20 components and technologies, such as modulation schemes, waveform structures, frame structures, reference signals, multi-antenna transmission and channel coding.

[00472] In at least one embodiment, a NR physical layer supports quadrature phase shift keying (QPSK), 16 quadrature amplitude modulation (QAM), 64 QAM and 256 QAM modulation formats. In at least one embodiment, different modulation schemes for different
25 user entity (UE) categories may also be included in a NR physical layer. In at least one embodiment, a NR physical layer may utilize cyclic prefix orthogonal frequency division multiplexing (CP-OFDM) with a scalable numerology (subcarrier spacing, cyclic prefix) in both uplink (UL) and downlink (DL) up to at least 52.6GHz. In at least one embodiment, a NR physical layer may support discrete Fourier transform spread orthogonal frequency
30 division multiplexing (DFT-SOFDM) in UL for coverage-limited scenarios, with single stream transmissions (that is, without spatial multiplexing).

[00473] In at least one embodiment, a NR frame supports time division duplex (TDD) and frequency division duplex (FDD) transmissions and operation in both licensed and unlicensed spectrum, which enables very low latency, fast hybrid automatic repeat request (HARQ)
35 acknowledgements, dynamic TDD, coexistence with LTE and transmissions of variable

5 length (for example, short duration for ultra-reliable low-latency communications (URLLC) and long duration for enhanced mobile broadband (eMBB)). In at least one embodiment, NR frame structure follows three key design principles to enhance forward compatibility and reduce interactions between different features.

[00474] In at least one embodiment, a first principle is that transmissions are self-
10 contained, which can refer to a scheme in which data in a slot and in a beam are decodable on its own without dependency on other slots and beams. In at least one embodiment, this implies that reference signals required for demodulation of data are included in a given slot and a given beam. In at least one embodiment, a second principle is that transmissions are well confined in time and frequency, which results in a scheme in which new types of
15 transmissions in parallel with legacy transmissions may be introduced. In at least one embodiment, a third principle is avoiding static and/or strict timing relations across slots and across different transmission directions. In at least one embodiment, usage of a third principle can entail utilizing asynchronous hybrid automatic repeat request (HARQ) instead of predefined retransmission time.

20 [00475] In at least one embodiment, NR frame structure also allows for rapid HARQ acknowledgement, in which decoding is performed during reception of DL data and HARQ acknowledgement is prepared by a UE during a guard period, when switching from DL reception to UL transmission. In at least one embodiment, to obtain low latency, a slot (or a set of slots in case of slot aggregation) is front-loaded with control signals and reference
25 signals at a beginning of a slot (or set of slots).

[00476] In at least one embodiment, NR has an ultra-lean design that minimizes always-on transmissions to enhance network energy efficiency and ensure forward compatibility. In at least one embodiment, reference signals in NR are transmitted only when necessary. In at least one embodiment, four main reference signals are demodulation reference signal
30 (DMRS), phase-tracking reference signal (PTRS), sounding reference signal (SRS) and channel-state information reference signal (CSI-RS).

[00477] In at least one embodiment, DMRS is used to estimate a radio channel for demodulation. In at least one embodiment, DMRS is UE-specific, can be beamformed, confined in a scheduled resource, and transmitted only when necessary, both in DL and UL.
35 In at least one embodiment, to support multiple-layer multiple-input, multiple-output

5 (MIMO) transmission, multiple orthogonal DMRS ports can be scheduled, one for each layer. In at least one embodiment, a basic DMRS pattern is front loaded, as a DMRS design takes into account an early decoding requirement to support low-latency applications. In at least one embodiment, for low-speed scenarios, DMRS uses low density in a time domain. In at least one embodiment, however, for high-speed scenarios, a time density of DMRS is
10 increased to track fast changes in a radio channel.

[00478] In at least one embodiment, PTRS is introduced in NR to enable compensation of oscillator phase noise. In at least one embodiment, typically, phase noise increases as a function of oscillator carrier frequency. In at least one embodiment, PTRS can therefore be utilized at high carrier frequencies (such as mmWave) to mitigate phase noise. In at least one
15 embodiment, PTRS is UE-specific, confined in a scheduled resource and can be beamformed. In at least one embodiment, PTRS is configurable depending on a quality of oscillators, carrier frequency, OFDM sub-carrier spacing, and modulation and coding schemes used for transmission.

[00479] In at least one embodiment, SRS is transmitted in UL to perform channel state
20 information (CSI) measurements mainly for scheduling and link adaptation. In at least one embodiment, for NR, SRS is also utilized for reciprocity-based precoder design for massive MIMO and UL beam management. In at least one embodiment, SRS has a modular and flexible design to support different procedures and UE capabilities. In at least one embodiment, an approach for channel state information reference signal (CSI-RS) is similar.

25 [00480] In at least one embodiment, NR employs different antenna solutions and techniques depending on which part of a spectrum is used for its operation. In at least one embodiment, for lower frequencies, a low to moderate number of active antennas (up to around 32 transmitter chains) is assumed and FDD operation is common. In at least one embodiment, acquisition of CSI requires transmission of CSI-RS in a DL and CSI reporting
30 in an UL. In at least one embodiment, limited bandwidths available in this frequency region require high spectral efficiency enabled by multi-user MIMO (MU-MIMO) and higher order spatial multiplexing, which is achieved via higher resolution CSI reporting compared with LTE.

[00481] In at least one embodiment, for higher frequencies, a larger number of antennas
35 can be employed in a given aperture, which increases a capability for beamforming and multi

5 user (MU)-MIMO. In at least one embodiment, here, spectrum allocations are of TDD type and reciprocity-based operation is assumed. In at least one embodiment, high-resolution CSI in a form of explicit channel estimations is acquired by UL channel sounding. In at least one embodiment, such high-resolution CSI enables sophisticated precoding algorithms to be employed at a base station (BS). In at least one embodiment, for even higher frequencies (in
10 mmWave range) an analog beamforming implementation is typically required currently, which limits transmission to a single beam direction per time unit and radio chain. In at least one embodiment, since an isotropic antenna element is very small in this frequency region owing to a short carrier wavelength, a great number of antenna elements is required to maintain coverage. In at least one embodiment, beamforming needs to be applied at both
15 transmitter and receiver ends to combat increased path loss, even for control channel transmission.

[00482] In at least one embodiment, to support these diverse use cases, NR features a highly flexible but unified CSI framework, in which there is reduced coupling between CSI measurement, CSI reporting and an actual DL transmission in NR compared with LTE. In at
20 least one embodiment, NR also supports more advanced schemes such as multi-point transmission and coordination. In at least one embodiment, control and data transmissions follow a self-contained principle, where all information required to decode a transmission (such as accompanying DMRS) is contained within a transmission itself. In at least one embodiment, as a result, a network can seamlessly change a transmission point or beam as an
25 UE moves in a network.

[00483] In at least one embodiment, MAC 4020 is a set of system software and libraries configured to provide an interface with a medium access control (MAC) layer, which may be part of a 5G network architecture. In at least one embodiment, a MAC layer controls hardware responsible for interaction with a wired, optical or wireless transmission medium.
30 In at least one embodiment, MAC provides flow control and multiplexing for a transmission medium.

[00484] In at least one embodiment, a MAC sublayer provides an abstraction of a physical layer such that complexities of a physical link control are invisible to a logical link control (LLC) and upper layers of a network stack. In at least one embodiment, any LLC sublayer
35 (and higher layers) may be used with any MAC. In at least one embodiment, any MAC can be used with any physical layer, independent of transmission medium. In at least one

5 embodiment, a MAC sublayer, when sending data to another device on a network, encapsulates higher-level frames into frames appropriate for a transmission medium, adds a frame check sequence to identify transmission errors, and then forwards data to a physical layer as soon as appropriate channel access method permits it. In at least one embodiment, MAC is also responsible for compensating for collisions if a jam signal is detected, in which
10 a MAC may initiate retransmission.

[00485] In at least one embodiment, applications 4002 may include one or more types of applications used by at least portions of node C.R.s 4016(1)-4016(N) and/or framework software 4006. In at least one embodiment, one or more types of applications may include, but are not limited to, any number of a genomics application, a cognitive compute, and a
15 machine learning application, including training or inferencing software, machine learning framework software (e.g., PyTorch, TensorFlow, Caffe, etc.) or other machine learning applications used in conjunction with one or more embodiments.

[00486] In at least one embodiment, RAN APIs 4014 may be a set of subroutine definitions, communication protocols, and/or software tools that provide a method of
20 communication with components of a radio access network (RAN) which may be part of a 5G network architecture. In at least one embodiment, a radio access network is part of a network communications system and may implement a radio access technology. In at least one embodiment, radio access network functionality is typically provided by a silicon chip residing in both a core network as well as user equipment. Further information regarding a
25 radio access network can be found in the description of FIG. 38.

[00487] In at least one embodiment, high level system 4000 may use CPUs, application-specific integrated circuits (ASICs), GPUs, FPGAs, or other hardware to perform training, inferencing, and/or other various processes using above-described resources. In at least one embodiment, moreover, one or more software and/or hardware resources described above
30 may be configured as a service to allow users to train or performing inferencing of information, such as image recognition, speech recognition, or other artificial intelligence services, as well as other services such as services that allow users to configure and implement various aspects of a 5G network architecture.

[00488] FIG. 41 illustrates an architecture of a system 4100 of a network, in accordance
35 with at least one embodiment. In at least one embodiment, system 4100 is shown to include a

5 user equipment (UE) 4102 and a UE 4104. In at least one embodiment, UEs 4102 and 4104 are illustrated as smartphones (e.g., handheld touchscreen mobile computing devices connectable to one or more cellular networks) but may also comprise any mobile or non-mobile computing device, such as Personal Data Assistants (PDAs), pagers, laptop computers, desktop computers, wireless handsets, or any computing device including a
10 wireless communications interface.

[00489] In at least one embodiment, any of UEs 4102 and 4104 can comprise an Internet of Things (IoT) UE, which can comprise a network access layer designed for low-power IoT applications utilizing short-lived UE connections. In at least one embodiment, an IoT UE can utilize technologies such as machine-to-machine (M2M) or machine-type communications
15 (MTC) for exchanging data with an MTC server or device via a public land mobile network (PLMN), Proximity-Based Service (ProSe) or device-to-device (D2D) communication, sensor networks, or IoT networks. In at least one embodiment, a M2M or MTC exchange of data may be a machine-initiated exchange of data. In at least one embodiment, an IoT network describes interconnecting IoT UEs, which may include uniquely identifiable
20 embedded computing devices (within Internet infrastructure), with short-lived connections. In at least one embodiment, an IoT UEs may execute background applications (e.g., keep alive messages, status updates, etc.) to facilitate connections of an IoT network.

[00490] In at least one embodiment, UEs 4102 and 4104 may be configured to connect, e.g., communicatively couple, with a radio access network (RAN) 4116. In at least one
25 embodiment, RAN 4116 may be, for example, an Evolved Universal Mobile Telecommunications System (UMTS) Terrestrial Radio Access Network (E-UTRAN), a NextGen RAN (NG RAN), or some other type of RAN. In at least one embodiment, UEs 4102 and 4104 utilize connections 4112 and 4114, respectively, each of which comprises a physical communications interface or layer. In at least one embodiment, connections 4112
30 and 4114 are illustrated as an air interface to enable communicative coupling, and can be consistent with cellular communications protocols, such as a Global System for Mobile Communications (GSM) protocol, a code-division multiple access (CDMA) network protocol, a Push-to-Talk (PTT) protocol, a PTT over Cellular (POC) protocol, a Universal Mobile Telecommunications System (UMTS) protocol, a 3GPP Long Term Evolution (LTE)
35 protocol, a fifth generation (5G) protocol, a New Radio (NR) protocol, and variations thereof.

5 [00491] In at least one embodiment, UEs 4102 and 4104 may further directly exchange communication data via a ProSe interface 4106. In at least one embodiment, ProSe interface 4106 may alternatively be referred to as a sidelink interface comprising one or more logical channels, including but not limited to a Physical Sidelink Control Channel (PSCCH), a Physical Sidelink Shared Channel (PSSCH), a Physical Sidelink Discovery Channel
10 (PSDCH), and a Physical Sidelink Broadcast Channel (PSBCH).

[00492] In at least one embodiment, UE 4104 is shown to be configured to access an access point (AP) 4110 via connection 4108. In at least one embodiment, connection 4108 can comprise a local wireless connection, such as a connection consistent with any IEEE 802.11 protocol, wherein AP 4110 would comprise a wireless fidelity (WiFi®) router. In at
15 least one embodiment, AP 4110 is shown to be connected to an Internet without connecting to a core network of a wireless system.

[00493] In at least one embodiment, RAN 4116 can include one or more access nodes that enable connections 4112 and 4114. In at least one embodiment, these access nodes (ANs) can be referred to as base stations (BSs), NodeBs, evolved NodeBs (eNBs), next Generation
20 NodeBs (gNB), RAN nodes, and so forth, and can comprise ground stations (e.g., terrestrial access points) or satellite stations providing coverage within a geographic area (e.g., a cell). In at least one embodiment, RAN 4116 may include one or more RAN nodes for providing macrocells, e.g., macro RAN node 4118, and one or more RAN nodes for providing femtocells or picocells (e.g., cells having smaller coverage areas, smaller user capacity, or
25 higher bandwidth compared to macrocells), e.g., low power (LP) RAN node 4120.

[00494] In at least one embodiment, any of RAN nodes 4118 and 4120 can terminate an air interface protocol and can be a first point of contact for UEs 4102 and 4104. In at least one embodiment, any of RAN nodes 4118 and 4120 can fulfill various logical functions for RAN 4116 including, but not limited to, radio network controller (RNC) functions such as
30 radio bearer management, uplink and downlink dynamic radio resource management and data packet scheduling, and mobility management.

[00495] In at least one embodiment, UEs 4102 and 4104 can be configured to communicate using Orthogonal Frequency-Division Multiplexing (OFDM) communication signals with each other or with any of RAN nodes 4118 and 4120 over a multi-carrier
35 communication channel in accordance various communication techniques, such as, but not

5 limited to, an Orthogonal Frequency Division Multiple Access (OFDMA) communication technique (e.g., for downlink communications) or a Single Carrier Frequency Division Multiple Access (SC-FDMA) communication technique (e.g., for uplink and ProSe or sidelink communications), and/or variations thereof. In at least one embodiment, OFDM signals can comprise a plurality of orthogonal sub-carriers.

10 [00496] In at least one embodiment, a downlink resource grid can be used for downlink transmissions from any of RAN nodes 4118 and 4120 to UEs 4102 and 4104, while uplink transmissions can utilize similar techniques. In at least one embodiment, a grid can be a time frequency grid, called a resource grid or time-frequency resource grid, which is a physical resource in a downlink in each slot. In at least one embodiment, such a time frequency plane
15 representation is a common practice for OFDM systems, which makes it intuitive for radio resource allocation. In at least one embodiment, each column and each row of a resource grid corresponds to one OFDM symbol and one OFDM subcarrier, respectively. In at least one embodiment, a duration of a resource grid in a time domain corresponds to one slot in a radio frame. In at least one embodiment, a smallest time-frequency unit in a resource grid is
20 denoted as a resource element. In at least one embodiment, each resource grid comprises a number of resource blocks, which describe a mapping of certain physical channels to resource elements. In at least one embodiment, each resource block comprises a collection of resource elements. In at least one embodiment, in a frequency domain, this may represent a smallest quantity of resources that currently can be allocated. In at least one embodiment,
25 there are several different physical downlink channels that are conveyed using such resource blocks.

[00497] In at least one embodiment, a physical downlink shared channel (PDSCH) may carry user data and higher-layer signaling to UEs 4102 and 4104. In at least one embodiment, a physical downlink control channel (PDCCH) may carry information about a
30 transport format and resource allocations related to PDSCH channel, among other things. In at least one embodiment, it may also inform UEs 4102 and 4104 about a transport format, resource allocation, and HARQ (Hybrid Automatic Repeat Request) information related to an uplink shared channel. In at least one embodiment, typically, downlink scheduling (assigning control and shared channel resource blocks to UE 4102 within a cell) may be performed at
35 any of RAN nodes 4118 and 4120 based on channel quality information fed back from any of

5 UEs 4102 and 4104. In at least one embodiment, downlink resource assignment information may be sent on a PDCCH used for (e.g., assigned to) each of UEs 4102 and 4104.

[00498] In at least one embodiment, a PDCCH may use control channel elements (CCEs) to convey control information. In at least one embodiment, before being mapped to resource elements, PDCCH complex valued symbols may first be organized into quadruplets, which
 10 may then be permuted using a sub-block interleaver for rate matching. In at least one embodiment, each PDCCH may be transmitted using one or more of these CCEs, where each CCE may correspond to nine sets of four physical resource elements known as resource element groups (REGs). In at least one embodiment, four Quadrature Phase Shift Keying (QPSK) symbols may be mapped to each REG. In at least one embodiment, PDCCH can be
 15 transmitted using one or more CCEs, depending on a size of a downlink control information (DCI) and a channel condition. In at least one embodiment, there can be four or more different PDCCH formats defined in LTE with different numbers of CCEs (e.g., aggregation level, $L=1, 2, 4$, or 8).

[00499] In at least one embodiment, an enhanced physical downlink control channel
 20 (EPDCCH) that uses PDSCH resources may be utilized for control information transmission. In at least one embodiment, EPDCCH may be transmitted using one or more enhanced control channel elements (ECCEs). In at least one embodiment, each ECCE may correspond to nine sets of four physical resource elements known as an enhanced resource element groups (EREGs). In at least one embodiment, an ECCE may have other numbers of EREGs
 25 in some situations.

[00500] In at least one embodiment, RAN 4116 is shown to be communicatively coupled to a core network (CN) 4138 via an S1 interface 4122. In at least one embodiment, CN 4138 may be an evolved packet core (EPC) network, a NextGen Packet Core (NPC) network, or some other type of CN. In at least one embodiment, S1 interface 4122 is split into two parts:
 30 S1-U interface 4126, which carries traffic data between RAN nodes 4118 and 4120 and serving gateway (S-GW) 4130, and a S1-mobility management entity (MME) interface 4124, which is a signaling interface between RAN nodes 4118 and 4120 and MMEs 4128.

[00501] In at least one embodiment, CN 4138 comprises MMEs 4128, S-GW 4130, Packet Data Network (PDN) Gateway (P-GW) 4134, and a home subscriber server (HSS) 4132. In at
 35 least one embodiment, MMEs 4128 may be similar in function to a control plane of legacy

5 Serving General Packet Radio Service (GPRS) Support Nodes (SGSN). In at least one embodiment, MMEs 4128 may manage mobility aspects in access such as gateway selection and tracking area list management. In at least one embodiment, HSS 4132 may comprise a database for network users, including subscription related information to support a network entities' handling of communication sessions. In at least one embodiment, CN 4138 may
10 comprise one or several HSSs 4132, depending on a number of mobile subscribers, on a capacity of an equipment, on an organization of a network, etc. In at least one embodiment, HSS 4132 can provide support for routing/roaming, authentication, authorization, naming/addressing resolution, location dependencies, etc.

[00502] In at least one embodiment, S-GW 4130 may terminate a S1 interface 4122
15 towards RAN 4116, and routes data packets between RAN 4116 and CN 4138. In at least one embodiment, S-GW 4130 may be a local mobility anchor point for inter-RAN node handovers and also may provide an anchor for inter-3GPP mobility. In at least one embodiment, other responsibilities may include lawful intercept, charging, and some policy enforcement.

20 [00503] In at least one embodiment, P-GW 4134 may terminate a SGi interface toward a PDN. In at least one embodiment, P-GW 4134 may route data packets between an EPC network 4138 and external networks such as a network including application server 4140 (alternatively referred to as application function (AF)) via an Internet Protocol (IP) interface 4142. In at least one embodiment, application server 4140 may be an element
25 offering applications that use IP bearer resources with a core network (e.g., UMTS Packet Services (PS) domain, LTE PS data services, etc.). In at least one embodiment, P-GW 4134 is shown to be communicatively coupled to an application server 4140 via an IP communications interface 4142. In at least one embodiment, application server 4140 can also be configured to support one or more communication services (e.g., Voice-over-Internet
30 Protocol (VoIP) sessions, PTT sessions, group communication sessions, social networking services, etc.) for UEs 4102 and 4104 via CN 4138.

[00504] In at least one embodiment, P-GW 4134 may further be a node for policy enforcement and charging data collection. In at least one embodiment, policy and Charging Enforcement Function (PCRF) 4136 is a policy and charging control element of CN 4138. In
35 at least one embodiment, in a non-roaming scenario, there may be a single PCRF in a Home Public Land Mobile Network (HPLMN) associated with a UE's Internet Protocol

5 Connectivity Access Network (IP-CAN) session. In at least one embodiment, in a roaming scenario with local breakout of traffic, there may be two PCRFs associated with a UE's IP-CAN session: a Home PCRF (H-PCRF) within a HPLMN and a Visited PCRF (V-PCRF) within a Visited Public Land Mobile Network (VPLMN). In at least one embodiment, PCRF 4136 may be communicatively coupled to application server 4140 via P-GW 4134. In at least
10 one embodiment, application server 4140 may signal PCRF 4136 to indicate a new service flow and select an appropriate Quality of Service (QoS) and charging parameters. In at least one embodiment, PCRF 4136 may provision this rule into a Policy and Charging Enforcement Function (PCEF) (not shown) with an appropriate traffic flow template (TFT) and QoS class of identifier (QCI), which commences a QoS and charging as specified by
15 application server 4140.

[00505] FIG. 42 illustrates example components of a device 4200 in accordance with at least one embodiment. In at least one embodiment, device 4200 may include application circuitry 4204, baseband circuitry 4208, Radio Frequency (RF) circuitry 4210, front-end module (FEM) circuitry 4202, one or more antennas 4212, and power management circuitry
20 (PMC) 4206 coupled together at least as shown. In at least one embodiment, components of illustrated device 4200 may be included in a UE or a RAN node. In at least one embodiment, device 4200 may include less elements (e.g., a RAN node may not utilize application circuitry 4204, and instead include a processor/controller to process IP data received from an EPC). In at least one embodiment, device 4200 may include additional elements such as, for
25 example, memory/storage, display, camera, sensor, or input/output (I/O) interface. In at least one embodiment, components described below may be included in more than one device (e.g., said circuitries may be separately included in more than one device for Cloud-RAN (C-RAN) implementations).

[00506] In at least one embodiment, application circuitry 4204 may include one or more
30 application processors. In at least one embodiment, application circuitry 4204 may include circuitry such as, but not limited to, one or more single-core or multi-core processors. In at least one embodiment, processor(s) may include any combination of general purpose processors and dedicated processors (e.g., graphics processors, application processors, etc.). In at least one embodiment, processors may be coupled with or may include memory/storage
35 and may be configured to execute instructions stored in memory/storage to enable various

5 applications or operating systems to run on device 4200. In at least one embodiment, processors of application circuitry 4204 may process IP data packets received from an EPC.

[00507] In at least one embodiment, baseband circuitry 4208 may include circuitry such as, but not limited to, one or more single-core or multi-core processors. In at least one embodiment, baseband circuitry 4208 may include one or more baseband processors or
10 control logic to process baseband signals received from a receive signal path of RF circuitry 4210 and to generate baseband signals for a transmit signal path of RF circuitry 4210. In at least one embodiment, baseband processing circuitry 4208 may interface with application circuitry 4204 for generation and processing of baseband signals and for controlling operations of RF circuitry 4210. In at least one embodiment, baseband circuitry 4208 may
15 include a third generation (3G) baseband processor 4208A, a fourth generation (4G) baseband processor 4208B, a fifth generation (5G) baseband processor 4208C, or other baseband processor(s) 4208D for other existing generations, generations in development or to be developed (e.g., second generation (2G), sixth generation (6G), etc.). In at least one embodiment, baseband circuitry 4208 (e.g., one or more of base-band processors 4208A-D)
20 may handle various radio control functions that enable communication with one or more radio networks via RF circuitry 4210. In at least one embodiment, some or all of a functionality of baseband processors 4208A-D may be included in modules stored in memory 4208G and executed via a Central Processing Unit (CPU) 4208E. In at least one embodiment, radio control functions may include, but are not limited to, signal modulation/demodulation,
25 encoding/decoding, radio frequency shifting, etc. In at least one embodiment, modulation/demodulation circuitry of baseband circuitry 4208 may include Fast-Fourier Transform (FFT), precoding, or constellation mapping/demapping functionality. In at least one embodiment, encoding/decoding circuitry of baseband circuitry 4208 may include convolution, tailbiting convolution, turbo, Viterbi, or Low Density Parity Check (LDPC)
30 encoder/decoder functionality.

[00508] In at least one embodiment, baseband circuitry 4208 may include one or more audio digital signal processor(s) (DSP) 4208F. In at least one embodiment, audio DSP(s) 4208F may include elements for compression/decompression and echo cancellation and may include other suitable processing elements in other embodiments. In at least one
35 embodiment, components of baseband circuitry may be suitably combined in a single chip, a single chipset, or disposed on a same circuit board in some embodiments. In at least one

5 embodiment, some or all of constituent components of baseband circuitry 4208 and application circuitry 4204 may be implemented together such as, for example, on a system on a chip (SOC).

[00509] In at least one embodiment, baseband circuitry 4208 may provide for communication compatible with one or more radio technologies. In at least one embodiment,
10 baseband circuitry 4208 may support communication with an evolved universal terrestrial radio access network (EUTRAN) or other wireless metropolitan area networks (WMAN), a wireless local area network (WLAN), a wireless personal area network (WPAN). In at least one embodiment, baseband circuitry 4208 is configured to support radio communications of more than one wireless protocol and may be referred to as multimode baseband circuitry.

15 [00510] In at least one embodiment, RF circuitry 4210 may enable communication with wireless networks using modulated electromagnetic radiation through a non-solid medium. In at least one embodiment, RF circuitry 4210 may include switches, filters, amplifiers, etc. to facilitate communication with a wireless network. In at least one embodiment, RF circuitry 4210 may include a receive signal path which may include circuitry to down-convert RF
20 signals received from FEM circuitry 4202 and provide baseband signals to baseband circuitry 4208. In at least one embodiment, RF circuitry 4210 may also include a transmit signal path which may include circuitry to up-convert baseband signals provided by baseband circuitry 4208 and provide RF output signals to FEM circuitry 4202 for transmission.

[00511] In at least one embodiment, receive signal path of RF circuitry 4210 may include
25 mixer circuitry 4210a, amplifier circuitry 4210b and filter circuitry 4210c. In at least one embodiment, a transmit signal path of RF circuitry 4210 may include filter circuitry 4210c and mixer circuitry 4210a. In at least one embodiment, RF circuitry 4210 may also include synthesizer circuitry 4210d for synthesizing a frequency for use by mixer circuitry 4210a of a receive signal path and a transmit signal path. In at least one embodiment, mixer circuitry
30 4210a of a receive signal path may be configured to down-convert RF signals received from FEM circuitry 4202 based on a synthesized frequency provided by synthesizer circuitry 4210d. In at least one embodiment, amplifier circuitry 4210b may be configured to amplify down-converted signals and filter circuitry 4210c may be a low-pass filter (LPF) or band-pass filter (BPF) configured to remove unwanted signals from down-converted signals to generate
35 output baseband signals. In at least one embodiment, output baseband signals may be provided to baseband circuitry 4208 for further processing. In at least one embodiment,

5 output baseband signals may be zero-frequency baseband signals, although this is not a requirement. In at least one embodiment, mixer circuitry 4210a of a receive signal path may comprise passive mixers.

[00512] In at least one embodiment, mixer circuitry 4210a of a transmit signal path may be configured to up-convert input baseband signals based on a synthesized frequency provided
10 by synthesizer circuitry 4210d to generate RF output signals for FEM circuitry 4202. In at least one embodiment, baseband signals may be provided by baseband circuitry 4208 and may be filtered by filter circuitry 4210c.

[00513] In at least one embodiment, mixer circuitry 4210a of a receive signal path and mixer circuitry 4210a of a transmit signal path may include two or more mixers and may be
15 arranged for quadrature down conversion and up conversion, respectively. In at least one embodiment, mixer circuitry 4210a of a receive signal path and mixer circuitry 4210a of a transmit signal path may include two or more mixers and may be arranged for image rejection (e.g., Hartley image rejection). In at least one embodiment, mixer circuitry 4210a of a receive signal path and mixer circuitry 4210a may be arranged for direct down conversion
20 and direct up conversion, respectively. In at least one embodiment, mixer circuitry 4210a of a receive signal path and mixer circuitry 4210a of a transmit signal path may be configured for super-heterodyne operation.

[00514] In at least one embodiment, output baseband signals and input baseband signals may be analog baseband signals. In at least one embodiment, output baseband signals and
25 input baseband signals may be digital baseband signals. In at least one embodiment, RF circuitry 4210 may include analog-to-digital converter (ADC) and digital-to-analog converter (DAC) circuitry and baseband circuitry 4208 may include a digital baseband interface to communicate with RF circuitry 4210.

[00515] In at least one embodiment, a separate radio IC circuitry may be provided for
30 processing signals for each spectrum. In at least one embodiment, synthesizer circuitry 4210d may be a fractional-N synthesizer or a fractional N/N+1 synthesizer. In at least one embodiment, synthesizer circuitry 4210d may be a delta-sigma synthesizer, a frequency multiplier, or a synthesizer comprising a phase-locked loop with a frequency divider

[00516] In at least one embodiment, synthesizer circuitry 4210d may be configured to
35 synthesize an output frequency for use by mixer circuitry 4210a of RF circuitry 4210 based

5 on a frequency input and a divider control input. In at least one embodiment, synthesizer circuitry 4210d may be a fractional $N/N+1$ synthesizer.

[00517] In at least one embodiment, frequency input may be provided by a voltage-controlled oscillator (VCO). In at least one embodiment, divider control input may be provided by either baseband circuitry 4208 or applications processor 4204 depending on a
10 desired output frequency. In at least one embodiment, a divider control input (e.g., N) may be determined from a look-up table based on a channel indicated by applications processor 4204.

[00518] In at least one embodiment, synthesizer circuitry 4210d of RF circuitry 4210 may include a divider, a delay-locked loop (DLL), a multiplexer and a phase accumulator. In at least one embodiment, divider may be a dual modulus divider (DMD) and phase accumulator
15 may be a digital phase accumulator (DPA). In at least one embodiment, DMD may be configured to divide an input signal by either N or $N+1$ (e.g., based on a carry out) to provide a fractional division ratio. In at least one embodiment, DLL may include a set of cascaded, tunable, delay elements, a phase detector, a charge pump and a D-type flip-flop. In at least one embodiment, delay elements may be configured to break a VCO period up into N_d equal
20 packets of phase, where N_d is a number of delay elements in a delay line. In at least one embodiment, in this way, DLL provides negative feedback to help ensure that total delay through a delay line is one VCO cycle.

[00519] In at least one embodiment, synthesizer circuitry 4210d may be configured to generate a carrier frequency as an output frequency, while in other embodiments, output
25 frequency may be a multiple of a carrier frequency (e.g., twice a carrier frequency, four times a carrier frequency) and used in conjunction with quadrature generator and divider circuitry to generate multiple signals at a carrier frequency with multiple different phases with respect to each other. In at least one embodiment, output frequency may be a LO frequency (f_{LO}). In at least one embodiment, RF circuitry 4210 may include an IQ/polar converter.

30 [00520] In at least one embodiment, FEM circuitry 4202 may include a receive signal path which may include circuitry configured to operate on RF signals received from one or more antennas 4212, amplify received signals and provide amplified versions of received signals to RF circuitry 4210 for further processing. In at least one embodiment, FEM circuitry 4202 may also include a transmit signal path which may include circuitry configured to amplify
35 signals for transmission provided by RF circuitry 4210 for transmission by one or more of

5 one or more antennas 4212. In at least one embodiment, amplification through a transmit or receive signal paths may be done solely in RF circuitry 4210, solely in FEM 4202, or in both RF circuitry 4210 and FEM 4202.

[00521] In at least one embodiment, FEM circuitry 4202 may include a TX/RX switch to switch between transmit mode and receive mode operation. In at least one embodiment, FEM
10 circuitry may include a receive signal path and a transmit signal path. In at least one embodiment, a receive signal path of FEM circuitry may include an LNA to amplify received RF signals and provide amplified received RF signals as an output (e.g., to RF circuitry 4210). In at least one embodiment, a transmit signal path of FEM circuitry 4202 may include a power amplifier (PA) to amplify input RF signals (e.g., provided by RF circuitry 4210), and
15 one or more filters to generate RF signals for subsequent transmission (e.g., by one or more of one or more antennas 4212).

[00522] In at least one embodiment, PMC 4206 may manage power provided to baseband circuitry 4208. In at least one embodiment, PMC 4206 may control power-source selection, voltage scaling, battery charging, or DC-to-DC conversion. In at least one embodiment,
20 PMC 4206 may often be included when device 4200 is capable of being powered by a battery, for example, when device is included in a UE. In at least one embodiment, PMC 4206 may increase power conversion efficiency while providing desirable implementation size and heat dissipation characteristics.

[00523] In at least one embodiment, PMC 4206 may be additionally or alternatively
25 coupled with, and perform similar power management operations for, other components such as, but not limited to, application circuitry 4204, RF circuitry 4210, or FEM 4202.

[00524] In at least one embodiment, PMC 4206 may control, or otherwise be part of, various power saving mechanisms of device 4200. In at least one embodiment, if device 4200 is in an RRC Connected state, where it is still connected to a RAN node as it expects to
30 receive traffic shortly, then it may enter a state known as Discontinuous Reception Mode (DRX) after a period of inactivity. In at least one embodiment, during this state, device 4200 may power down for brief intervals of time and thus save power.

[00525] In at least one embodiment, if there is no data traffic activity for an extended period of time, then device 4200 may transition off to an RRC Idle state, where it disconnects
35 from a network and does not perform operations such as channel quality feedback, handover,

5 etc. In at least one embodiment, device 4200 goes into a very low power state and it performs paging where again it periodically wakes up to listen to a network and then powers down again. In at least one embodiment, device 4200 may not receive data in this state, in order to receive data, it must transition back to RRC Connected state.

[00526] In at least one embodiment, an additional power saving mode may allow a device
10 to be unavailable to a network for periods longer than a paging interval (ranging from seconds to a few hours). In at least one embodiment, during this time, a device is totally unreachable to a network and may power down completely. In at least one embodiment, any data sent during this time incurs a large delay and it is assumed delay is acceptable.

[00527] In at least one embodiment, processors of application circuitry 4204 and
15 processors of baseband circuitry 4208 may be used to execute elements of one or more instances of a protocol stack. In at least one embodiment, processors of baseband circuitry 4208, alone or in combination, may be used execute Layer 3, Layer 2, or Layer 1 functionality, while processors of application circuitry 4208 may utilize data (e.g., packet data) received from these layers and further execute Layer 4 functionality (e.g., transmission
20 communication protocol (TCP) and user datagram protocol (UDP) layers). In at least one embodiment, layer 3 may comprise a radio resource control (RRC) layer. In at least one embodiment, Layer 2 may comprise a medium access control (MAC) layer, a radio link control (RLC) layer, and a packet data convergence protocol (PDCP) layer. In at least one embodiment, Layer 1 may comprise a physical (PHY) layer of a UE/RAN node.

25 [00528] FIG. 43 illustrates example interfaces of baseband circuitry, in accordance with at least one embodiment. In at least one embodiment, as discussed above, baseband circuitry 4208 of FIG. 42 may comprise processors 4208A-4208E and a memory 4208G utilized by said processors. In at least one embodiment, each of processors 4208A-4208E may include a memory interface, 4302A-4302E, respectively, to send/receive data to/from memory 4208G.

30 [00529] In at least one embodiment, baseband circuitry 4208 may further include one or more interfaces to communicatively couple to other circuitries/devices, such as a memory interface 4304 (e.g., an interface to send/receive data to/from memory external to baseband circuitry 4208), an application circuitry interface 4306 (e.g., an interface to send/receive data to/from application circuitry 4204 of FIG. 42), an RF circuitry interface 4308 (e.g., an
35 interface to send/receive data to/from RF circuitry 4210 of FIG. 42), a wireless hardware

5 connectivity interface 4310 (e.g., an interface to send/receive data to/from Near Field Communication (NFC) components, Bluetooth® components (e.g., Bluetooth® Low Energy), Wi-Fi® components, and other communication components), and a power management interface 4312 (e.g., an interface to send/receive power or control signals to/from PMC 4206).

10 [00530] FIG. 44 illustrates an example of an uplink channel, in accordance with at least one embodiment. In at least one embodiment, FIG. 44 illustrates transmitting and receiving data within a physical uplink shared channel (PUSCH) in 5G NR, which may be part of a physical layer of a mobile device network.

[00531] In at least one embodiment, Physical Uplink Shared Channel (PUSCH) in 5G NR
15 is designated to carry multiplexed control information and user application data. In at least one embodiment, 5G NR provides much more flexibility and reliability comparing to its predecessor, which in some examples may be referred to as 4G LTE, including more elastic pilot arrangements and support for both cyclic prefix (CP)-OFDM and Discrete Fourier Transform spread (DFT-s)-OFDM waveforms. In at least one embodiment, standard
20 introduced filtered OFDM (f-OFDM) technique is utilized to add additional filtering to reduce Out-of-Band emission and improve performance at higher modulation orders. In at least one embodiment, modifications in Forward Error Correction (FEC) were imposed to replace Turbo Codes used in 4G LTE by Quasi-Cyclic Low Density Parity Check (QC-LDPC) codes, which were proven to achieve better transmission rates and provide
25 opportunities for more efficient hardware implementations.

[00532] In at least one embodiment, transmission of 5G NR downlink and uplink data is organized into frames of 10 ms duration, each divided into 10 subframes of 1 ms each. In at least one embodiment, subframes are composed of a variable number of slots, depending on a selected subcarrier spacing which is parameterized in 5G NR. In at least one embodiment, a
30 slot is built from 14 OFDMA symbols, each prepended with a cyclic prefix. In at least one embodiment, a subcarrier that is located within a passband and is designated for transmission is called a Resource Element (RE). In at least one embodiment, a group of 12 neighboring RE in a same symbol form a Physical Resource Block (PRB).

[00533] In at least one embodiment, 5G NR standard defined two types of reference
35 signals associated with transmission within a PUSCH channel. In at least one embodiment,

5 Demodulation Reference Signal (DMRS) is a user specific reference signal with high frequency density. In at least one embodiment, DMRS is transmitted within dedicated orthogonal frequency-division multiple access (OFDMA) symbols only and designated for frequency-selective channel estimation. In at least one embodiment, a number of DMRS symbols within a slot may vary between 1 and 4 depending on configuration, where a denser
10 DMRS symbol spacing in time is designated for fast time-varying channels to obtain more accurate estimates within a coherence time of a channel. In at least one embodiment, in a frequency domain, DMRS PRB are mapped within a whole transmission allocation. In at least one embodiment, spacing between a DMRS resource element (RE) assigned for a same Antenna Port (AP) may be chosen between 2 and 3. In at least one embodiment, in a case of
15 2-2 multiple-input, multiple-output (MIMO), a standard allows for orthogonal assignment of RE between AP. In at least one embodiment, a receiver may perform partial single input, multiple output (SIMO) channel estimation based on a DMRS RE prior to MIMO equalization, neglecting spatial correlation.

[00534] In at least one embodiment, a second type of reference signal is a Phase Tracking
20 Reference Signal (PTRS). In at least one embodiment, PTRS subcarriers are arranged in a comb structure having high density in a time domain. In at least one embodiment, it is used mainly in mmWave frequency bands to track and correct phase noise, which is a considerable source of performance losses. In at least one embodiment, usage of PTRS is optional, as it may lower a total spectral efficiency of a transmission when effects of phase noise are
25 negligible.

[00535] In at least one embodiment, for transmission of data, a transport block may be generated from a MAC layer and given to a physical layer. In at least one embodiment, a transport block may be data that is intended to be transmitted. In at least one embodiment, a transmission in a physical layer starts with grouped resource data, which may be referred to
30 as transport blocks. In at least one embodiment, a transport block is received by a cyclic redundancy check (CRC) 4402. In at least one embodiment, a cyclic redundancy check is appended to each transport block for error detection. In at least one embodiment, a cyclic redundancy check is used for error detection in transport blocks. In at least one embodiment, an entire transport block is used to calculate CRC parity bits and these parity bits are then
35 attached to an end of a transport block. In at least one embodiment, minimum and maximum code block sizes are specified so blocks sizes are compatible with further processes. In at

5 least one embodiment, an input block is segmented when an input block is greater than a maximum code block size.

[00536] In at least one embodiment, a transport block is received and encoded by a low-density parity-check (LDPC) encode 4404. In at least one embodiment, NR employs low-density parity-check (LDPC) codes for a data channel and polar codes for a control channel.

10 In at least one embodiment, LDPC codes are defined by their parity-check matrices, with each column representing a coded bit, and each row representing a parity-check equation. In at least one embodiment, LDPC codes are decoded by exchanging messages between variables and parity checks in an iterative manner. In at least one embodiment, LDPC codes proposed for NR use a quasi-cyclic structure, where a parity-check matrix is defined by a
15 smaller base matrix. In at least one embodiment, each entry of the base matrix represents either a $Z \times Z$ zero matrix or a shifted $Z \times Z$ identity matrix

[00537] In at least one embodiment, an encoded transport block is received by rate match 4406. In at least one embodiment, an encoded block is used to create an output bit stream with a desired code rate. In at least one embodiment, rate match 4406 is utilized to
20 create an output bit stream to be transmitted with a desired code rate. In at least one embodiment, bits are selected and pruned from a buffer to create an output bit stream with a desired code rate. In at least one embodiment, a Hybrid Automatic Repeat Request (HARQ) error correction scheme is incorporated.

[00538] In at least one embodiment, output bits are scrambled, which may aid in privacy,
25 in scramble 4408. In at least one embodiment, codewords are bit-wise multiplied with an orthogonal sequence and a UE-specific scrambling sequence. In at least one embodiment, output of scramble 4408 may be input into modulation/mapping/precoding and other processes 4410. In at least one embodiment, various modulation, mapping, and precoding processes are performed.

30 [00539] In at least one embodiment, bits output from scramble 4408 are modulated with a modulation scheme, resulting in blocks of modulation symbols. In at least one embodiment, scrambled codewords undergo modulation using one of modulation schemes QPSK, 16 QAM, 64 QAM, resulting in a block of modulation symbols. In at least one embodiment, a channel interleaver process may be utilized that implements a first time mapping of
35 modulation symbols onto a transmit waveform while ensuring that HARQ information is

5 present on both slots. In at least one embodiment, modulation symbols are mapped to various layers based on transmit antennas. In at least one embodiment, symbols may be precoded, in which they are divided into sets, and an Inverse Fast Fourier Transform may be performed. In at least one embodiment, transport data and control multiplexing may be performed such that HARQ acknowledge (ACK) information is present in both slots and is mapped to resources
 10 around demodulation reference signals. In at least one embodiment, various precoding processes are performed.

[00540] In at least one embodiment, symbols are mapped to allocated physical resource elements in resource element mapping 4412. In at least one embodiment, allocation sizes may be limited to values whose prime factors are 2, 3 and 5. In at least one embodiment, symbols
 15 are mapped in increasing order beginning with subcarriers. In at least one embodiment, subcarrier mapped modulation symbols data are orthogonal frequency-division multiple access (OFDMA) modulated through IFFT operation in OFDMA modulation 4414. In at least one embodiment, time domain representations of each symbol are concatenated and filtered using transmit FIR filter to attenuate unwanted Out of Band emission to adjacent frequency
 20 bands caused by phase discontinuities and utilization of different numerologies. In at least one embodiment, an output of OFDMA modulation 4414 may be transmitted to be received and processed by another system.

[00541] In at least one embodiment, a transmission may be received by OFDMA demodulation 4416. In at least one embodiment, a transmission may originate from user
 25 mobile devices over a cellular network, although other contexts may be present. In at least one embodiment, a transmission may be demodulated through IFFT processing. In at least one embodiment, once OFDMA demodulation through IFFT processing has been accomplished, an estimation and correction of residual Sample Time Offset (STO) and Carrier Frequency Offset (CFO) may be performed. In at least one embodiment, both CFO
 30 and STO corrections have to be performed in frequency domain, because a received signal can be a superposition of transmissions coming from multiple UEs multiplexed in frequency, each suffering from a specific residual synchronization error. In at least one embodiment, residual CFO is estimated as a phase rotation between pilot subcarriers belonging to different OFDM symbols and corrected by a circular convolution operation in frequency domain.

35 [00542] In at least one embodiment, output of OFDMA demodulation 4416 may be received by resource element demapping 4418. In at least one embodiment, resource element

5 demapping 4418 may determine symbols and demap symbols from allocated physical resource elements. In at least one embodiment, a channel estimation and equalization is performed in channel estimation 4420 in order to compensate for effects of multipath propagation. In at least one embodiment, channel estimation 4420 may be utilized to minimize effects of noise originating from various transmission layers and antennae. In at
10 least one embodiment, channel estimation 4420 may generate equalized symbols from an output of resource element demapping 4418. In at least one embodiment, demodulation/demapping 4422 may receive equalized symbols from channel estimation 4420. In at least one embodiment, equalized symbols are demapped and permuted through a layer demapping operation. In at least one embodiment, a Maximum A Posteriori Probability
15 (MAP) demodulation approach may be utilized to produce values representing beliefs regarding a received bit being 0 or 1, expressed in a form of Log-Likelihood Ratio (LLR).

[00543] In at least one embodiment, soft-demodulated bits are processed using various operations, including descrambling, deinterleaving and rate unmatching with LLR soft-combining using a circular buffer prior to LDPC decoding. In at least one embodiment,
20 descramble 4424 may involve processes that reverse one or more processes of scramble 4408. In at least one embodiment, rate unmatch 4426 may involve processes that reverse one or more processes of rate match 4406. In at least one embodiment, descramble 4424 may receive output from demodulation/demapping 4422, and descramble received bits. In at least one embodiment, rate unmatch 4426 may receive descrambled bits, and utilize LLR soft-
25 combining utilizing a circular buffer prior to LDPC decode 4428.

[00544] In at least one embodiment, decoding of LDPC codes in practical applications is done based on iterative belief propagation algorithms. In at least one embodiment, an LDPC code can be represented in a form of a bipartite graph with parity check matrix H of size $M \times N$ being an adjacency matrix defining connections between graph nodes. In at least one
30 embodiment, M rows of matrix H corresponds to parity check nodes, whereas N columns corresponds to variable nodes, i.e. received codeword bits. In at least one embodiment, a principle of belief propagation algorithms is based on iterative message exchange, in which A Posteriori probabilities between a variable and check nodes are updated, until a valid codeword is obtained. In at least one embodiment, LDPC decode 4428 may output a
35 transport block comprising data.

5 [00545] In at least one embodiment, CRC check 4430 may determine errors and perform one or more actions based on parity bits attached to a received transport block. In at least one embodiment, CRC check 4430 may analyze and process parity bits attached to a received transport block, or otherwise any information associated with a CRC. In at least one
10 further processing.

[00546] It should be noted that, in various embodiments, transmitting and receiving data, which may be a transport block or other variation thereof, may include various processes not depicted in FIG. 44. In at least one embodiment, processes depicted in FIG. 44 are not intended to be exhaustive and further processes such as additional modulation, mapping,
15 multiplexing, precoding, constellation mapping/demapping, MIMO detection, detection, decoding and variations thereof may be utilized in transmitting and receiving data as part of a network.

[00547] FIG. 45 illustrates an architecture of a system 4500 of a network in accordance with some embodiments. In at least one embodiment, system 4500 is shown to include a
20 UE 4502, a 5G access node or RAN node (shown as (R)AN node 4508), a User Plane Function (shown as UPF 4504), a Data Network (DN 4506), which may be, for example, operator services, Internet access or 3rd party services, and a 5G Core Network (5GC) (shown as CN 4510).

[00548] In at least one embodiment, CN 4510 includes an Authentication Server
25 Function (AUSF 4514); a Core Access and Mobility Management Function (AMF 4512); a Session Management Function (SMF 4518); a Network Exposure Function (NEF 4516); a Policy Control Function (PCF 4522); a Network Function (NF) Repository Function (NRF 4520); a Unified Data Management (UDM 4524); and an Application Function (AF 4526). In at least one embodiment, CN 4510 may also include other elements that are
30 not shown, such as a Structured Data Storage network function (SDSF), an Unstructured Data Storage network function (UDSF), and variations thereof.

[00549] In at least one embodiment, UPF 4504 may act as an anchor point for intra-RAT and inter-RAT mobility, an external PDU session point of interconnect to DN 4506, and a branching point to support multi-homed PDU session. In at least one embodiment,
35 UPF 4504 may also perform packet routing and forwarding, packet inspection, enforce

5 user plane part of policy rules, lawfully intercept packets (UP collection); traffic usage reporting, perform QoS handling for user plane (e.g. packet filtering, gating, UL/DL rate enforcement), perform Uplink Traffic verification (e.g., SDF to QoS flow mapping), transport level packet marking in uplink and downlink, and downlink packet buffering and downlink data notification triggering. In at least one embodiment, UPF 4504 may
10 include an uplink classifier to support routing traffic flows to a data network. In at least one embodiment, DN 4506 may represent various network operator services, Internet access, or third party services.

[00550] In at least one embodiment, AUSF 4514 may store data for authentication of UE 4502 and handle authentication related functionality. In at least one embodiment,
15 AUSF 4514 may facilitate a common authentication framework for various access types.

[00551] In at least one embodiment, AMF 4512 may be responsible for registration management (e.g., for registering UE 4502, etc.), connection management, reachability management, mobility management, and lawful interception of AMF-related events, and access authentication and authorization. In at least one embodiment, AMF 4512 may
20 provide transport for SM messages for SMF 4518, and act as a transparent proxy for routing SM messages. In at least one embodiment, AMF 4512 may also provide transport for short message service (SMS) messages between UE 4502 and an SMS function (SMSF) (not shown by FIG. 45). In at least one embodiment, AMF 4512 may act as Security Anchor Function (SEA), which may include interaction with AUSF 4514 and UE 4502 and receipt
25 of an intermediate key that was established as a result of UE 4502 authentication process. In at least one embodiment, where USIM based authentication is used, AMF 4512 may retrieve security material from AUSF 4514. In at least one embodiment, AMF 4512 may also include a Security Context Management (SCM) function, which receives a key from SEA that it uses to derive access-network specific keys. In at least one embodiment, furthermore,
30 AMF 4512 may be a termination point of RAN CP interface (N2 reference point), a termination point of NAS (NI) signaling, and perform NAS ciphering and integrity protection.

[00552] In at least one embodiment, AMF 4512 may also support NAS signaling with a UE 4502 over an N3 interworking-function (IWF) interface. In at least one embodiment,
35 N3IWF may be used to provide access to untrusted entities. In at least one embodiment, N3IWF may be a termination point for N2 and N3 interfaces for control plane and user

5 plane, respectively, and as such, may handle N2 signaling from SMF and AMF for PDU sessions and QoS, encapsulate/de-encapsulate packets for IPsec and N3 tunneling, mark N3 user-plane packets in uplink, and enforce QoS corresponding to N3 packet marking taking into account QoS requirements associated to such marking received over N2. In at least one embodiment, N3IWF may also relay uplink and downlink control-plane NAS
 10 (NI) signaling between UE 4502 and AMF 4512, and relay uplink and downlink user-plane packets between UE 4502 and UPF 4504. In at least one embodiment, N3IWF also provides mechanisms for IPsec tunnel establishment with UE 4502.

[00553] In at least one embodiment, SMF 4518 may be responsible for session management (e.g., session establishment, modify and release, including tunnel maintain
 15 between UPF and AN node); UE IP address allocation & management (including optional Authorization); Selection and control of UP function; Configures traffic steering at UPF to route traffic to proper destination; termination of interfaces towards Policy control functions; control part of policy enforcement and QoS; lawful intercept (for SM events and interface to LI System); termination of SM parts of NAS messages; downlink Data
 20 Notification; initiator of AN specific SM information, sent via AMF over N2 to AN; determine SSC mode of a session. In at least one embodiment, SMF 4518 may include following roaming functionality: handle local enforcement to apply QoS SLAB (VPLMN); charging data collection and charging interface (VPLMN); lawful intercept (in VPLMN for SM events and interface to LI System); support for interaction with external DN for
 25 transport of signaling for PDU session authorization/ authentication by external DN.

[00554] In at least one embodiment, NEF 4516 may provide means for securely exposing services and capabilities provided by 3GPP network functions for third party, internal exposure/re-exposure, Application Functions (e.g., AF 4526), edge computing or fog computing systems, etc. In at least one embodiment, NEF 4516 may authenticate,
 30 authorize, and/or throttle AFs. In at least one embodiment, NEF 4516 may also translate information exchanged with AF 4526 and information exchanged with internal network functions. In at least one embodiment, NEF 4516 may translate between an AF-Service-Identifier and an internal 5GC information. In at least one embodiment, NEF 4516 may also receive information from other network functions (NFs) based on exposed capabilities
 35 of other network functions. In at least one embodiment, this information may be stored at NEF 4516 as structured data, or at a data storage NF using a standardized interfaces. In at

5 least one embodiment, stored information can then be re-exposed by NEF 4516 to other NFs and AFs, and/or used for other purposes such as analytics.

[00555] In at least one embodiment, NRF 4520 may support service discovery functions, receive NF Discovery Requests from NF instances, and provide information of discovered NF instances to NF instances. In at least one embodiment, NRF 4520 also
10 maintains information of available NF instances and their supported services.

[00556] In at least one embodiment, PCF 4522 may provide policy rules to control plane function(s) to enforce them, and may also support unified policy framework to govern network behavior. In at least one embodiment, PCF 4522 may also implement a front end (FE) to access subscription information relevant for policy decisions in a UDR of UDM
15 4524.

[00557] In at least one embodiment, UDM 4524 may handle subscription-related information to support a network entities' handling of communication sessions, and may store subscription data of UE 4502. In at least one embodiment, UDM 4524 may include two parts, an application FE and a User Data Repository (UDR). In at least one
20 embodiment, UDM may include a UDM FE, which is in charge of processing of credentials, location management, subscription management and so on. In at least one embodiment, several different front ends may serve a same user in different transactions. In at least one embodiment, UDM-FE accesses subscription information stored in an UDR and performs authentication credential processing; user identification handling; access
25 authorization; registration/mobility management; and subscription management. In at least one embodiment, UDR may interact with PCF 4522. In at least one embodiment, UDM 4524 may also support SMS management, wherein an SMS-FE implements a similar application logic as discussed previously.

[00558] In at least one embodiment, AF 4526 may provide application influence on
30 traffic routing, access to a Network Capability Exposure (NCE), and interact with a policy framework for policy control. In at least one embodiment, NCE may be a mechanism that allows a 5GC and AF 4526 to provide information to each other via NEF 4516, which may be used for edge computing implementations. In at least one embodiment, network operator and third party services may be hosted close to UE 4502 access point of attachment to
35 achieve an efficient service delivery through a reduced end-to-end latency and load on a

5 transport network. In at least one embodiment, for edge computing implementations, 5GC may select a UPF 4504 close to UE 4502 and execute traffic steering from UPF 4504 to DN 4506 via N6 interface. In at least one embodiment, this may be based on UE subscription data, UE location, and information provided by AF 4526. In at least one embodiment, AF 4526 may influence UPF (re)selection and traffic routing. In at least one embodiment,
 10 based on operator deployment, when AF 4526 is considered to be a trusted entity, a network operator may permit AF 4526 to interact directly with relevant NFs.

[00559] In at least one embodiment, CN 4510 may include an SMSF, which may be responsible for SMS subscription checking and verification, and relaying SM messages to/from UE 4502 to/from other entities, such as an SMS-GMSC/IWMSC/SMS-router. In
 15 at least one embodiment, SMS may also interact with AMF 4512 and UDM 4524 for notification procedure that UE 4502 is available for SMS transfer (e.g., set a UE not reachable flag, and notifying UDM 4524 when UE 4502 is available for SMS).

[00560] In at least one embodiment, system 4500 may include following service-based interfaces: Namf: Service-based interface exhibited by AMF; Nsmf: Service-based
 20 interface exhibited by SMF; Nnef: Service-based interface exhibited by NEF; Npcf: Service-based interface exhibited by PCF; Nudm: Service-based interface exhibited by UDM; Naf: Service-based interface exhibited by AF; Nnrf: Service-based interface exhibited by NRF; and Nausf: Service-based interface exhibited by AUSF.

[00561] In at least one embodiment, system 4500 may include following reference points:
 25 N1: Reference point between UE and AMF; N2: Reference point between (R)AN and AMF; N3: Reference point between (R)AN and UPF; N4: Reference point between SMF and UPF; and N6: Reference point between UPF and a Data Network. In at least one embodiment, there may be many more reference points and/or service-based interfaces between a NF services in NFs, however, these interfaces and reference points have been omitted for
 30 clarity. In at least one embodiment, an NS reference point may be between a PCF and AF; an N7 reference point may be between PCF and SMF; an N11 reference point between AMF and SMF; etc. In at least one embodiment, CN 4510 may include an Nx interface, which is an inter-CN interface between MME and AMF 4512 in order to enable interworking between CN 4510 and CN 7245.

5 [00562] In at least one embodiment, system 4500 may include multiple RAN nodes (such as (R)AN node 4508) wherein an Xn interface is defined between two or more (R)AN node 4508 (e.g., gNBs) that connecting to 5GC 410, between a (R)AN node 4508 (e.g., gNB) connecting to CN 4510 and an eNB (e.g., a macro RAN node), and/or between two eNBs connecting to CN 4510.

10 [00563] In at least one embodiment, Xn interface may include an Xn user plane (Xn-U) interface and an Xn control plane (Xn-C) interface. In at least one embodiment, Xn-U may provide non-guaranteed delivery of user plane PDUs and support/provide data forwarding and flow control functionality. In at least one embodiment, Xn-C may provide management and error handling functionality, functionality to manage a Xn-C interface;
 15 mobility support for UE 4502 in a connected mode (e.g., CM-CONNECTED) including functionality to manage UE mobility for connected mode between one or more (R)AN node 4508. In at least one embodiment, mobility support may include context transfer from an old (source) serving (R)AN node 4508 to new (target) serving (R)AN node 4508; and control of user plane tunnels between old (source) serving (R)AN node 4508 to new
 20 (target) serving (R)AN node 4508.

[00564] In at least one embodiment, a protocol stack of a Xn-U may include a transport network layer built on Internet Protocol (IP) transport layer, and a GTP-U layer on top of a UDP and/or IP layer(s) to carry user plane PDUs. In at least one embodiment, Xn-C
 25 protocol stack may include an application layer signaling protocol (referred to as Xn Application Protocol (Xn-AP)) and a transport network layer that is built on an SCTP layer. In at least one embodiment, SCTP layer may be on top of an IP layer. In at least one embodiment, SCTP layer provides a guaranteed delivery of application layer messages. In at least one embodiment, in a transport IP layer point-to-point transmission is used to deliver signaling PDUs. In at least one embodiment, Xn-U protocol stack and/or a Xn-C
 30 protocol stack may be same or similar to an user plane and/or control plane protocol stack(s) shown and described herein.

[00565] FIG. 46 is an illustration of a control plane protocol stack in accordance with some embodiments. In at least one embodiment, a control plane 4600 is shown as a communications protocol stack between UE 4102 (or alternatively, UE 4104), RAN 4116,
 35 and MME(s) 4128.

5 [00566] In at least one embodiment, PHY layer 4602 may transmit or receive information used by MAC layer 4604 over one or more air interfaces. In at least one embodiment, PHY layer 4602 may further perform link adaptation or adaptive modulation and coding (AMC), power control, cell search (e.g., for initial
10 such as an RRC layer 4610. In at least one embodiment, PHY layer 4602 may still further perform error detection on transport channels, forward error correction (FEC) coding/decoding of transport channels, modulation/demodulation of physical channels, interleaving, rate matching, mapping onto physical channels, and Multiple Input Multiple Output (MIMO) antenna processing.

15 [00567] In at least one embodiment, MAC layer 4604 may perform mapping between logical channels and transport channels, multiplexing of MAC service data units (SDUs) from one or more logical channels onto transport blocks (TB) to be delivered to PHY via transport channels, de-multiplexing MAC SDUs to one or more logical channels from transport blocks (TB) delivered from PHY via transport channels, multiplexing MAC
20 SDUs onto TBs, scheduling information reporting, error correction through hybrid automatic repeat request (HARD), and logical channel prioritization.

[00568] In at least one embodiment, RLC layer 4606 may operate in a plurality of modes of operation, including: Transparent Mode (TM), Unacknowledged Mode (UM), and Acknowledged Mode (AM). In at least one embodiment, RLC layer 4606 may execute
25 transfer of upper layer protocol data units (PDUs), error correction through automatic repeat request (ARQ) for AM data transfers, and concatenation, segmentation and reassembly of RLC SDUs for UM and AM data transfers. In at least one embodiment, RLC layer 4606 may also execute re-segmentation of RLC data PDUs for AM data transfers, reorder RLC data PDUs for UM and AM data transfers, detect duplicate data for UM and
30 AM data transfers, discard RLC SDUs for UM and AM data transfers, detect protocol errors for AM data transfers, and perform RLC re-establishment.

[00569] In at least one embodiment, PDCP layer 4608 may execute header compression and decompression of IP data, maintain PDCP Sequence Numbers (SNs), perform in-sequence delivery of upper layer PDUs at re-establishment of lower layers, eliminate
35 duplicates of lower layer SDUs at re-establishment of lower layers for radio bearers mapped on RLC AM, cipher and decipher control plane data, perform integrity protection

5 and integrity verification of control plane data, control timer-based discard of data, and perform security operations (e.g., ciphering, deciphering, integrity protection, integrity verification, etc.).

[00570] In at least one embodiment, main services and functions of a RRC layer 4610 may include broadcast of system information (e.g., included in Master Information Blocks
10 (MIBs) or System Information Blocks (SIBs) related to a non-access stratum (NAS)), broadcast of system information related to an access stratum (AS), paging, establishment, maintenance and release of an RRC connection between an UE and E-UTRAN (e.g., RRC connection paging, RRC connection establishment, RRC connection modification, and RRC connection release), establishment, configuration, maintenance and release of point-
15 to-point radio bearers, security functions including key management, inter radio access technology (RAT) mobility, and measurement configuration for UE measurement reporting. In at least one embodiment, said MIBs and SIBs may comprise one or more information elements (IEs), which may each comprise individual data fields or data structures.

20 [00571] In at least one embodiment, UE 4102 and RAN 4116 may utilize a Uu interface (e.g., an LTE-Uu interface) to exchange control plane data via a protocol stack comprising PHY layer 4602, MAC layer 4604, RLC layer 4606, PDCP layer 4608, and RRC layer 4610.

[00572] In at least one embodiment, non-access stratum (NAS) protocols (NAS
25 protocols 4612) form a highest stratum of a control plane between UE 4102 and MME(s) 4128. In at least one embodiment, NAS protocols 4612 support mobility of UE 4102 and session management procedures to establish and maintain IP connectivity between UE 4102 and P-GW 4134.

[00573] In at least one embodiment, Si Application Protocol (S1-AP) layer (Si-AP
30 layer 4622) may support functions of a Si interface and comprise Elementary Procedures (EPs). In at least one embodiment, an EP is a unit of interaction between RAN 4116 and CN 4128. In at least one embodiment, S1 -AP layer services may comprise two groups: UE-associated services and non UE-associated services. In at least one embodiment, these services perform functions including, but not limited to: E-UTRAN Radio Access Bearer

5 (E-RAB) management, UE capability indication, mobility, NAS signaling transport, RAN Information Management (RIM), and configuration transfer.

[00574] In at least one embodiment, Stream Control Transmission Protocol (SCTP) layer (alternatively referred to as a stream control transmission protocol/internet protocol (SCTP/IP) layer) (SCTP layer 4620) may ensure reliable delivery of signaling messages
10 between RAN 4116 and MME(s) 4128 based, in part, on an IP protocol, supported by an IP layer 4618. In at least one embodiment, L2 layer 4616 and an L1 layer 4614 may refer to communication links (e.g., wired or wireless) used by a RAN node and MME to exchange information.

[00575] In at least one embodiment, RAN 4116 and MME(s) 4128 may utilize an S1 -
15 MME interface to exchange control plane data via a protocol stack comprising a L1 layer 4614, L2 layer 4616, IP layer 4618, SCTP layer 4620, and Si -AP layer 4622.

[00576] FIG. 47 is an illustration of a user plane protocol stack in accordance with at least one embodiment. In at least one embodiment, a user plane 4700 is shown as a communications protocol stack between a UE 4102, RAN 4116, S-GW 4130, and P-GW
20 4134. In at least one embodiment, user plane 4700 may utilize a same protocol layers as control plane 4600. In at least one embodiment, for example, UE 4102 and RAN 4116 may utilize a Uu interface (e.g., an LTE-Uu interface) to exchange user plane data via a protocol stack comprising PHY layer 4602, MAC layer 4604, RLC layer 4606, PDCP layer 4608.

[00577] In at least one embodiment, General Packet Radio Service (GPRS) Tunneling
25 Protocol for a user plane (GTP-U) layer (GTP-U layer 4704) may be used for carrying user data within a GPRS core network and between a radio access network and a core network. In at least one embodiment, user data transported can be packets in any of IPv4, IPv6, or PPP formats, for example. In at least one embodiment, UDP and IP security (UDP/IP) layer (UDP/IP layer 4702) may provide checksums for data integrity, port numbers for
30 addressing different functions at a source and destination, and encryption and authentication on selected data flows. In at least one embodiment, RAN 4116 and S-GW 4130 may utilize an S1 -U interface to exchange user plane data via a protocol stack comprising L1 layer 4614, L2 layer 4616, UDP/IP layer 4702, and GTP-U layer 4704. In at least one embodiment, S-GW 4130 and P-GW 4134 may utilize an S5/S8a interface to
35 exchange user plane data via a protocol stack comprising L1 layer 4614, L2 layer 4616,

5 UDP/IP layer 4702, and GTP-U layer 4704. In at least one embodiment, as discussed above with respect to FIG. 46, NAS protocols support a mobility of UE 4102 and session management procedures to establish and maintain IP connectivity between UE 4102 and P-GW 4134.

[00578] FIG. 48 illustrates components 4800 of a core network in accordance with at
10 least one embodiment. In at least one embodiment, components of CN 4138 may be implemented in one physical node or separate physical nodes including components to read and execute instructions from a machine-readable or computer-readable medium (e.g., a non-transitory machine-readable storage medium). In at least one embodiment, Network Functions Virtualization (NFV) is utilized to virtualize any or all of above
15 described network node functions via executable instructions stored in one or more computer readable storage mediums (described in further detail below). In at least one embodiment, a logical instantiation of CN 4138 may be referred to as a network slice 4802 (e.g., network slice 4802 is shown to include HSS 4132, MME(s) 4128, and S-GW 4130). In at least one embodiment, a logical instantiation of a portion of CN 4138 may be referred
20 to as a network sub-slice 4804 (e.g., network sub-slice 4804 is shown to include P-GW 4134 and PCRF 4136).

[00579] In at least one embodiment, NFV architectures and infrastructures may be used to virtualize one or more network functions, alternatively performed by proprietary
hardware, onto physical resources comprising a combination of industry-standard server
25 hardware, storage hardware, or switches. In at least one embodiment, NFV systems can be used to execute virtual or reconfigurable implementations of one or more EPC components/functions.

[00580] FIG. 49 is a block diagram illustrating components, according to at least one
embodiment, of a system 4900 to support network function virtualization (NFV). In at
30 least one embodiment, system 4900 is illustrated as including a virtualized infrastructure manager (shown as VIM 4902), a network function virtualization infrastructure (shown as NFVI 4904), a VNF manager (shown as VNFM 4906), virtualized network functions (shown as VNF 4908), an element manager (shown as EM 4910), an NFV Orchestrator (shown as NFVO 4912), and a network manager (shown as NM 4914).

5 [00581] In at least one embodiment, VIM 4902 manages resources of NFVI 4904. In at least one embodiment, NFVI 4904 can include physical or virtual resources and applications (including hypervisors) used to execute system 4900. In at least one embodiment, VIM 4902 may manage a life cycle of virtual resources with NFVI 4904 (e.g., creation, maintenance, and tear down of virtual machines (VMs) associated with
10 one or more physical resources), track VM instances, track performance, fault and security of VM instances and associated physical resources, and expose VM instances and associated physical resources to other management systems.

[00582] In at least one embodiment, VNFM 4906 may manage VNF 4908. In at least one embodiment, VNF 4908 may be used to execute EPC components/ functions. In at
15 least one embodiment, VNFM 4906 may manage a life cycle of VNF 4908 and track performance, fault and security of virtual aspects of VNF 4908. In at least one embodiment, EM 4910 may track performance, fault and security of functional aspects of VNF 4908. In at least one embodiment, tracking data from VNFM 4906 and EM 4910 may comprise, for example, performance measurement (PM) data used by VIM 4902 or NFVI
20 4904. In at least one embodiment, both VNFM 4906 and EM 4910 can scale up/down a quantity of VNFs of system 4900.

[00583] In at least one embodiment, NFVO 4912 may coordinate, authorize, release and engage resources of NFVI 4904 in order to provide a requested service (e.g., to execute an EPC function, component, or slice). In at least one embodiment, NM 4914
25 may provide a package of end-user functions with responsibility for a management of a network, which may include network elements with VNFs, non-virtualized network functions, or both (management of the VNFs may occur via the EM 4910).

[00584] Other variations are within spirit of present disclosure. Thus, while disclosed techniques are susceptible to various modifications and alternative constructions, certain
30 illustrated embodiments thereof are shown in drawings and have been described above in detail. It should be understood, however, that there is no intention to limit disclosure to specific form or forms disclosed, but on contrary, intention is to cover all modifications, alternative constructions, and equivalents falling within spirit and scope of disclosure, as defined in appended claims.

5 [00585] At least one embodiment of the disclosure can be described in view of the following clauses:

1. A processor comprising:

one or more circuits to cause information received from a plurality of fifth-generation (5G) new radio antennas to be error corrected in parallel by a corresponding plurality of
10 processor pipelines.

2. The processor of clause 1, wherein error correcting the information comprises:

storing, in a memory of the processor, a plurality of variable nodes containing values corresponding to the information;

15 performing a set of instructions to implement one or more check nodes to perform error correction of the information, wherein each of the one or more check nodes corresponds to one or more variable nodes; and

updating the plurality of variable nodes with one or more new values calculated by performing the set of instructions implementing one or more check nodes.

20 3. The processor of clause 1 or 2, wherein each of the plurality of processor pipelines error correct the information according to different portions of a parity check matrix.

4. The processor of any of clauses 1-3, wherein a variable node of a plurality of variable nodes used to error correct the information comprises a master value
25 corresponding to the information and all other variable nodes of the plurality of variable nodes store local values corresponding to the information on the plurality of processor pipelines.

5. The processor of any of clauses 1-4, wherein the one or more circuits implement a check node to calculate an indication that a variable node of the plurality of
30 variable nodes contains a specific value.

6. The processor of any of clauses 1-5, wherein the plurality of processor pipelines are parallel processing units (PPUs).

7. The processor of any of clauses 1-6, wherein the information has been encoded by a low density parity check (LDPC) coding scheme.

35 8. A system comprising:

5 one or more processors to be configured to perform error correction on information received from a plurality of 5G new radio antennas, where the information is error corrected in parallel by a corresponding plurality of processor pipelines.

9. The system of clause 8, further comprising memory to store:
a plurality of variable nodes comprising belief values corresponding to the
10 information;

a set of instructions that, when performed, implement a plurality of check nodes to perform one or more error correction operations, wherein each check node corresponds to one or more variable nodes of the plurality of variable nodes.

10. The system of clause 9, wherein a variable node of the plurality of
15 variable nodes comprises a master value corresponding to the information and all other variable nodes of the plurality of variable nodes store local values corresponding to the information on the plurality of processor pipelines.

11. The system of clause 9 or 10, wherein the one or more error correction operations calculate a belief value indicating a likelihood that a variable node of the plurality
20 of variable nodes indicates a value determined by a parity check matrix.

12. The processor of any of clauses 8-11, wherein each of the plurality of processor pipelines error correct the information according to different portions of a parity check matrix.

13. The system of clause 12, wherein each of the plurality of processor
25 pipelines error correct the information iteratively.

14. The system of any of clauses 8-13, wherein the plurality of processor pipelines are PPUs.

15. The system of any of clauses 8-14, wherein the information received from a plurality of 5G new radio antennas has been encoded by an LDPC coding scheme.

30 16. A machine-readable medium having stored thereon a set of instructions, which if performed by one or more processors, cause the one or more processors to at least:

cause information received from a plurality of 5G new radio antennas to be error corrected in parallel by a corresponding plurality of processor pipelines.

5 17. The machine-readable medium of clause 16, wherein the instructions, if performed by the one or more processors, further cause the one or more processors to at least:

 store belief values corresponding to the information in a plurality of variable nodes;
 perform one or more operations by a plurality of check nodes, where each check node
10 corresponds to one or more variable nodes of the plurality of variable nodes; and
 calculate, for each check node of the plurality of check nodes, a belief value for each of the one or more variable nodes corresponding to the check node.

 18. The machine-readable medium of clause 17, wherein the instructions, if performed by the one or more processors, further cause the one or more processors to store
15 a master belief value in a master variable node of the plurality of variable nodes, and store local belief values in a remainder of variable nodes of the plurality of variable nodes on the plurality of processor pipelines.

 19. The machine-readable medium of clause 17 or 18, wherein the instructions further cause the plurality of processor pipelines to perform each of the one or
20 more operations for each check node of the plurality of check nodes.

 20. The machine-readable medium of any of clauses 16-19, wherein the instructions further cause each of the plurality of processor pipelines error correct the information according to different portions of a parity check matrix.

 21. The machine-readable medium of any of clauses 16-20, wherein the
25 instructions further cause each of the plurality of processor pipelines error correct the information iteratively.

 22. The machine-readable medium of any of clauses 16-21, wherein the plurality of processor pipelines are PPUs.

 23. The machine-readable medium of any of clauses 16-22, wherein the
30 information received from a plurality of 5G new radio antennas has been encoded by an LDPC coding scheme.

 24. A method, comprising:
 error correcting information received from a plurality of fifth-generation (5G) new radio antennas in parallel by a corresponding plurality of processor pipelines.

5 25. The method of clause 24, further comprising:
storing belief values corresponding to the information in a plurality of variable nodes;
performing one or more operations by a plurality of check nodes, where each check
node corresponds to one or more variable nodes of the plurality of variable nodes; and
calculating, for each check node of the plurality of check nodes, a belief value for
10 each of the one or more variable nodes corresponding to the check node.

 26. The method of clause 25, further comprising storing a master belief
value corresponding to the information in a variable node of the plurality of variable nodes,
and storing local belief values in the plurality of variable nodes on the plurality of processor
pipelines.

15 27. The method of clause 25 or 26, further comprising causing the plurality
of processor pipelines to perform each of the one or more operations for each check node of
the plurality of check nodes.

 28. The method of any of clauses 24-27, further comprising error
correcting the information according to different portions of a parity check matrix in each of
20 the plurality of processor pipelines.

 29. The method of any of clauses 24-28, further comprising error
correcting a subset of the information iteratively in each of the plurality of processor
pipelines.

 30. The method of any of clauses 24-29, wherein the plurality of processor
25 pipelines are PPUs, and the PPU each contain one or more processing cores.

 31. The method of any of clauses 24-30, wherein the information received
from a plurality of 5G new radio antennas has been encoded by an LDPC coding scheme.

[00586] Use of terms “a” and “an” and “the” and similar referents in context of describing
disclosed embodiments (especially in context of following claims) are to be construed to
30 cover both singular and plural, unless otherwise indicated herein or clearly contradicted by
context, and not as a definition of a term. Terms “comprising,” “having,” “including,” and
“containing” are to be construed as open-ended terms (meaning “including, but not limited
to,”) unless otherwise noted. term “connected,” when unmodified and referring to physical
connections, is to be construed as partly or wholly contained within, attached to, or joined
35 together, even if there is something intervening. Recitation of ranges of values herein are

5 merely intended to serve as a shorthand method of referring individually to each separate value falling within range, unless otherwise indicated herein and each separate value is incorporated into specification as if it were individually recited herein. Use of term “set” (e.g., “a set of items”) or “subset” unless otherwise noted or contradicted by context, is to be construed as a nonempty collection comprising one or more members. Further, unless
10 otherwise noted or contradicted by context, term “subset” of a corresponding set does not necessarily denote a proper subset of corresponding set, but subset and corresponding set may be equal.

[00587] Conjunctive language, such as phrases of form “at least one of A, B, and C,” or “at least one of A, B and C,” unless specifically stated otherwise or otherwise clearly
15 contradicted by context, is otherwise understood with context as used in general to present that an item, term, etc., may be either A or B or C, or any nonempty subset of set of A and B and C. For instance, in illustrative example of a set having three members, conjunctive phrases “at least one of A, B, and C” and “at least one of A, B and C” refer to any of following sets: {A}, {B}, {C}, {A, B}, {A, C}, {B, C}, {A, B, C}. Thus, such conjunctive
20 language is not generally intended to imply that certain embodiments require at least one of A, at least one of B and at least one of C each to be present. In addition, unless otherwise noted or contradicted by context, term “plurality” indicates a state of being plural (e.g., “a plurality of items” indicates multiple items). Number of items in a plurality is at least two, but can be more when so indicated either explicitly or by context. Further, unless stated
25 otherwise or otherwise clear from context, phrase “based on” means “based at least in part on” and not “based solely on.”

[00588] Operations of processes described herein can be performed in any suitable order unless otherwise indicated herein or otherwise clearly contradicted by context. In at least one embodiment, a process such as those processes described herein (or variations and/or
30 combinations thereof) is performed under control of one or more computer systems configured with executable instructions and is implemented as code (e.g., executable instructions, one or more computer programs or one or more applications) executing collectively on one or more processors, by hardware or combinations thereof. In at least one embodiment, code is stored on a computer-readable storage medium, for example, in form of
35 a computer program comprising a plurality of instructions executable by one or more processors. In at least one embodiment, a computer-readable storage medium is a non-

5 transitory computer-readable storage medium that excludes transitory signals (e.g., a propagating transient electric or electromagnetic transmission) but includes non-transitory data storage circuitry (e.g., buffers, cache, and queues) within transceivers of transitory signals. In at least one embodiment, code (e.g., executable code or source code) is stored on a set of one or more non-transitory computer-readable storage media having stored thereon
10 executable instructions (or other memory to store executable instructions) that, when executed (i.e., as a result of being executed) by one or more processors of a computer system, cause computer system to perform operations described herein. Set of non-transitory computer-readable storage media, in at least one embodiment, comprises multiple non-transitory computer-readable storage media and one or more of individual non-transitory
15 storage media of multiple non-transitory computer-readable storage media lack all of code while multiple non-transitory computer-readable storage media collectively store all of code. In at least one embodiment, executable instructions are executed such that different instructions are executed by different processors — for example, a non-transitory computer-readable storage medium store instructions and a main central processing unit (“CPU”) executes some of instructions while a graphics processing unit (“GPU”) executes other
20 instructions. In at least one embodiment, different components of a computer system have separate processors and different processors execute different subsets of instructions.

[00589] Accordingly, in at least one embodiment, computer systems are configured to implement one or more services that singly or collectively perform operations of processes
25 described herein and such computer systems are configured with applicable hardware and/or software that enable performance of operations. Further, a computer system that implements at least one embodiment of present disclosure is a single device and, in another embodiment, is a distributed computer system comprising multiple devices that operate differently such that distributed computer system performs operations described herein and such that a single
30 device does not perform all operations.

[00590] Use of any and all examples, or exemplary language (e.g., “such as”) provided herein, is intended merely to better illuminate embodiments of disclosure and does not pose a limitation on scope of disclosure unless otherwise claimed. No language in specification should be construed as indicating any non-claimed element as essential to practice of
35 disclosure.

5 [00591] All references, including publications, patent applications, and patents, cited herein are hereby incorporated by reference to same extent as if each reference were individually and specifically indicated to be incorporated by reference and were set forth in its entirety herein.

[00592] In description and claims, terms “coupled” and “connected,” along with their
10 derivatives, may be used. It should be understood that these terms may be not intended as synonyms for each other. Rather, in particular examples, “connected” or “coupled” may be used to indicate that two or more elements are in direct or indirect physical or electrical contact with each other. “Coupled” may also mean that two or more elements are not in direct contact with each other, but yet still co-operate or interact with each other.

15 [00593] Unless specifically stated otherwise, it may be appreciated that throughout specification terms such as “processing,” “computing,” “calculating,” “determining,” or like, refer to action and/or processes of a computer or computing system, or similar electronic computing device, that manipulate and/or transform data represented as physical, such as electronic, quantities within computing system’s registers and/or memories into other data
20 similarly represented as physical quantities within computing system’s memories, registers or other such information storage, transmission or display devices.

[00594] In a similar manner, term “processor” may refer to any device or portion of a device that processes electronic data from registers and/or memory and transform that electronic data into other electronic data that may be stored in registers and/or memory. As
25 non-limiting examples, “processor” may be a CPU or a GPU. A “computing platform” may comprise one or more processors. As used herein, “software” processes may include, for example, software and/or hardware entities that perform work over time, such as tasks, threads, and intelligent agents. Also, each process may refer to multiple processes, for carrying out instructions in sequence or in parallel, continuously or intermittently. Terms
30 “system” and “method” are used herein interchangeably insofar as system may embody one or more methods and methods may be considered a system.

[00595] In present document, references may be made to obtaining, acquiring, receiving, or inputting analog or digital data into a subsystem, computer system, or computer-
implemented machine. Process of obtaining, acquiring, receiving, or inputting analog and
35 digital data can be accomplished in a variety of ways such as by receiving data as a parameter

5 of a function call or a call to an application programming interface. In some implementations, process of obtaining, acquiring, receiving, or inputting analog or digital data can be accomplished by transferring data via a serial or parallel interface. In another implementation, process of obtaining, acquiring, receiving, or inputting analog or digital data can be accomplished by transferring data via a computer network from providing entity to
10 acquiring entity. References may also be made to providing, outputting, transmitting, sending, or presenting analog or digital data. In various examples, process of providing, outputting, transmitting, sending, or presenting analog or digital data can be accomplished by transferring data as an input or output parameter of a function call, a parameter of an application programming interface or interprocess communication mechanism.

15 [00596] Although discussion above sets forth example implementations of described techniques, other architectures may be used to implement described functionality, and are intended to be within scope of this disclosure. Furthermore, although specific distributions of responsibilities are defined above for purposes of discussion, various functions and responsibilities might be distributed and divided in different ways, depending on
20 circumstances.

[00597] Furthermore, although subject matter has been described in language specific to structural features and/or methodological acts, it is to be understood that subject matter claimed in appended claims is not necessarily limited to specific features or acts described. Rather, specific features and acts are disclosed as exemplary forms of implementing the
25 claims.

CLAIMS

WHAT IS CLAIMED IS:

1. A processor comprising:

one or more circuits to cause information received from a plurality of fifth-generation (5G) new radio antennas to be error corrected in parallel by a corresponding plurality of processor pipelines.
2. The processor of claim 1, wherein error correcting the information comprises:

storing, in a memory of the processor, a plurality of variable nodes containing values corresponding to the information;

performing a set of instructions to implement one or more check nodes to perform error correction of the information, wherein each of the one or more check nodes corresponds to one or more variable nodes; and

updating the plurality of variable nodes with one or more new values calculated by performing the set of instructions implementing one or more check nodes.
3. The processor of claim 1, wherein each of the plurality of processor pipelines error correct the information according to different portions of a parity check matrix.
4. The processor of claim 1, wherein a variable node of a plurality of variable nodes used to error correct the information comprises a master value corresponding to the information and all other variable nodes of the plurality of variable nodes store local values corresponding to the information on the plurality of processor pipelines.
5. The processor of claim 1, wherein the one or more circuits implement a check node to calculate an indication that a variable node of the plurality of variable nodes contains a specific value.
6. The processor of claim 1, wherein the plurality of processor pipelines are parallel processing units (PPUs).

7. The processor of claim 1, wherein the information has been encoded by a low density parity check (LDPC) coding scheme.

8. A system comprising:
one or more processors to be configured to perform error correction on information received from a plurality of 5G new radio antennas, where the information is error corrected in parallel by a corresponding plurality of processor pipelines.

9. The system of claim 8, further comprising memory to store:
a plurality of variable nodes comprising belief values corresponding to the information;
a set of instructions that, when performed, implement a plurality of check nodes to perform one or more error correction operations, wherein each check node corresponds to one or more variable nodes of the plurality of variable nodes.

10. The system of claim 9, wherein a variable node of the plurality of variable nodes comprises a master value corresponding to the information and all other variable nodes of the plurality of variable nodes store local values corresponding to the information on the plurality of processor pipelines.

11. The system of claim 9, wherein the one or more error correction operations calculate a belief value indicating a likelihood that a variable node of the plurality of variable nodes indicates a value determined by a parity check matrix.

12. The processor of claim 8, wherein each of the plurality of processor pipelines error correct the information according to different portions of a parity check matrix.

13. The system of claim 12, wherein each of the plurality of processor pipelines error correct the information iteratively.

14. The system of claim 8, wherein the plurality of processor pipelines are PPUs.

15. The system of claim 8, wherein the information received from a plurality of 5G new radio antennas has been encoded by an LDPC coding scheme.

16. A machine-readable medium having stored thereon a set of instructions, which if performed by one or more processors, cause the one or more processors to at least:

cause information received from a plurality of 5G new radio antennas to be error corrected in parallel by a corresponding plurality of processor pipelines.

17. The machine-readable medium of claim 16, wherein the instructions, if performed by the one or more processors, further cause the one or more processors to at least:

store belief values corresponding to the information in a plurality of variable nodes;

perform one or more operations by a plurality of check nodes, where each check node corresponds to one or more variable nodes of the plurality of variable nodes; and

calculate, for each check node of the plurality of check nodes, a belief value for each of the one or more variable nodes corresponding to the check node.

18. The machine-readable medium of claim 17, wherein the instructions, if performed by the one or more processors, further cause the one or more processors to store a master belief value in a master variable node of the plurality of variable nodes, and store local belief values in a remainder of variable nodes of the plurality of variable nodes on the plurality of processor pipelines.

19. The machine-readable medium of claim 17, wherein the instructions further cause the plurality of processor pipelines to perform each of the one or more operations for each check node of the plurality of check nodes.

20. The machine-readable medium of claim 16, wherein the instructions further cause each of the plurality of processor pipelines error correct the information according to different portions of a parity check matrix.

21. The machine-readable medium of claim 16, wherein the instructions further cause each of the plurality of processor pipelines error correct the information iteratively.

22. The machine-readable medium of claim 16, wherein the plurality of processor pipelines are PPUs.

23. The machine-readable medium of claim 16, wherein the information received from a plurality of 5G new radio antennas has been encoded by an LDPC coding scheme.

24. A method, comprising:
error correcting information received from a plurality of fifth-generation (5G) new radio antennas in parallel by a corresponding plurality of processor pipelines.

25. The method of claim 24, further comprising:
storing belief values corresponding to the information in a plurality of variable nodes;
performing one or more operations by a plurality of check nodes, where each check node corresponds to one or more variable nodes of the plurality of variable nodes; and
calculating, for each check node of the plurality of check nodes, a belief value for each of the one or more variable nodes corresponding to the check node.

26. The method of claim 25, further comprising storing a master belief value corresponding to the information in a variable node of the plurality of variable nodes, and storing local belief values in the plurality of variable nodes on the plurality of processor pipelines.

27. The method of claim 25, further comprising causing the plurality of processor pipelines to perform each of the one or more operations for each check node of the plurality of check nodes.

28. The method of claim 24, further comprising error correcting the information according to different portions of a parity check matrix in each of the plurality of processor pipelines.

29. The method of claim 24, further comprising error correcting a subset of the information iteratively in each of the plurality of processor pipelines.

30. The method of claim 24, wherein the plurality of processor pipelines are PPUs, and the PPU each contain one or more processing cores.

31. The method of claim 24, wherein the information received from a plurality of 5G new radio antennas has been encoded by an LDPC coding scheme.

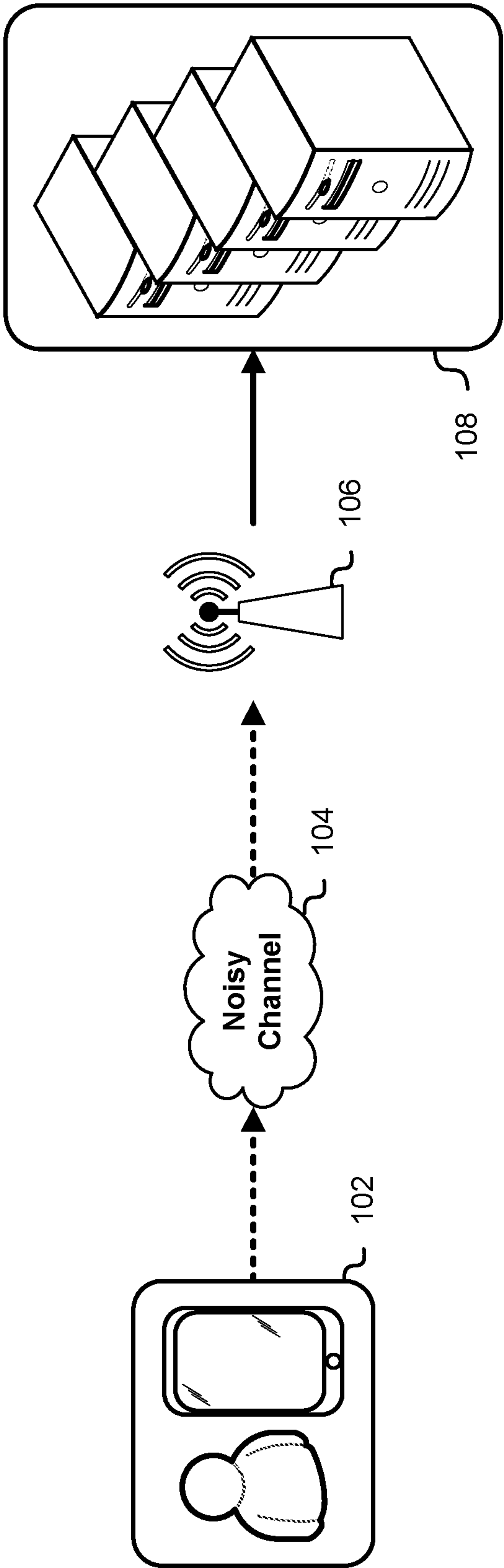


FIG. 1A

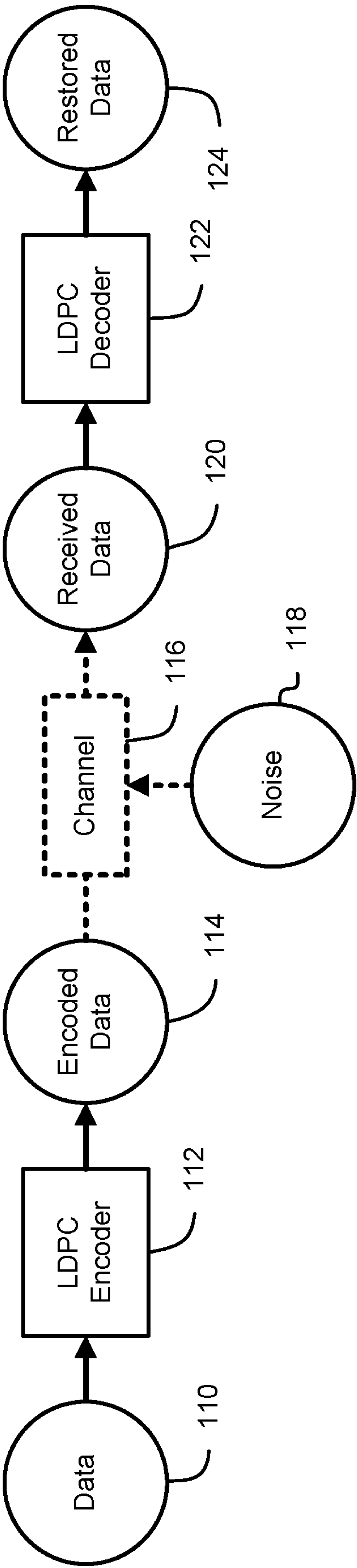


FIG. 1B

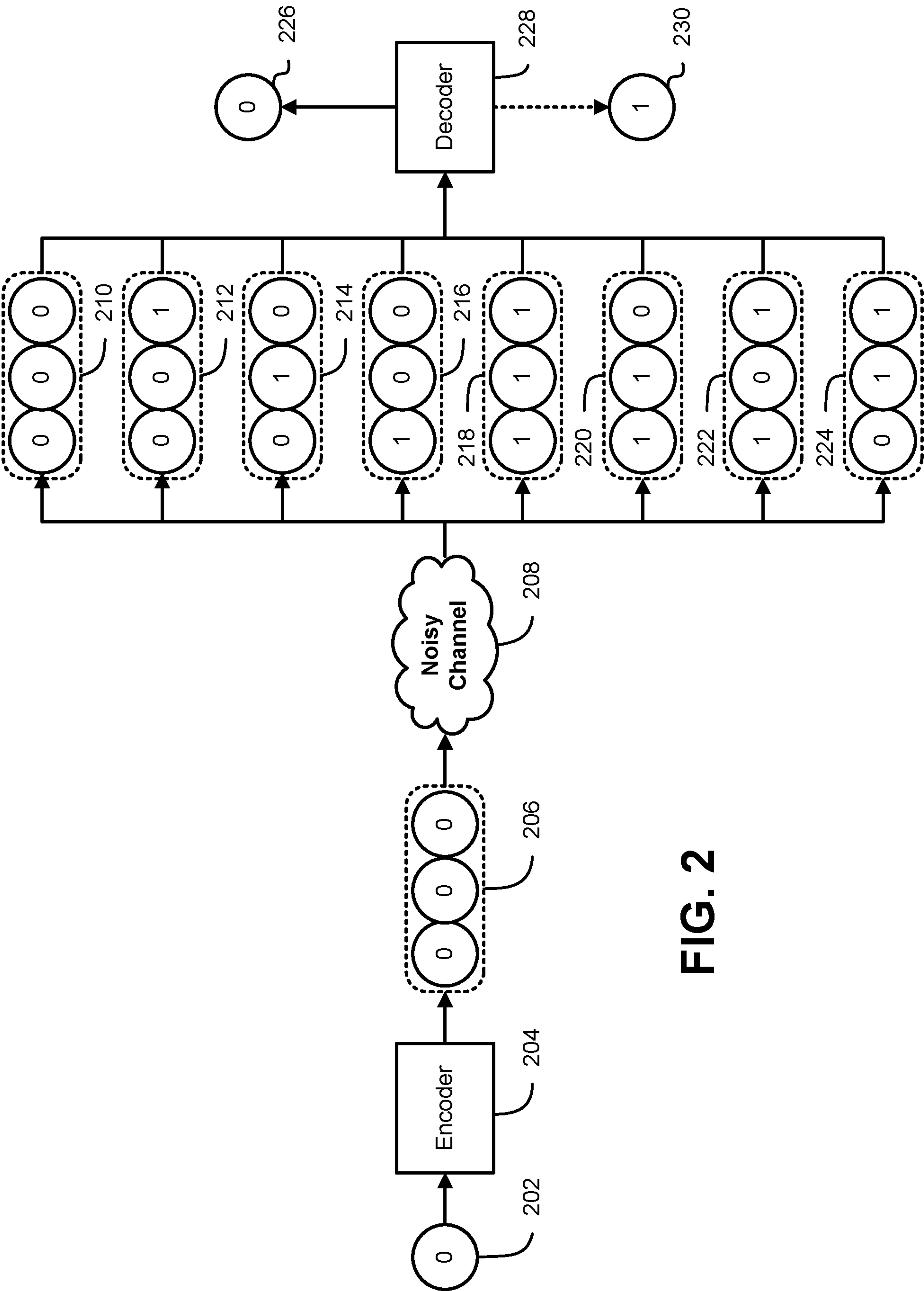


FIG. 2

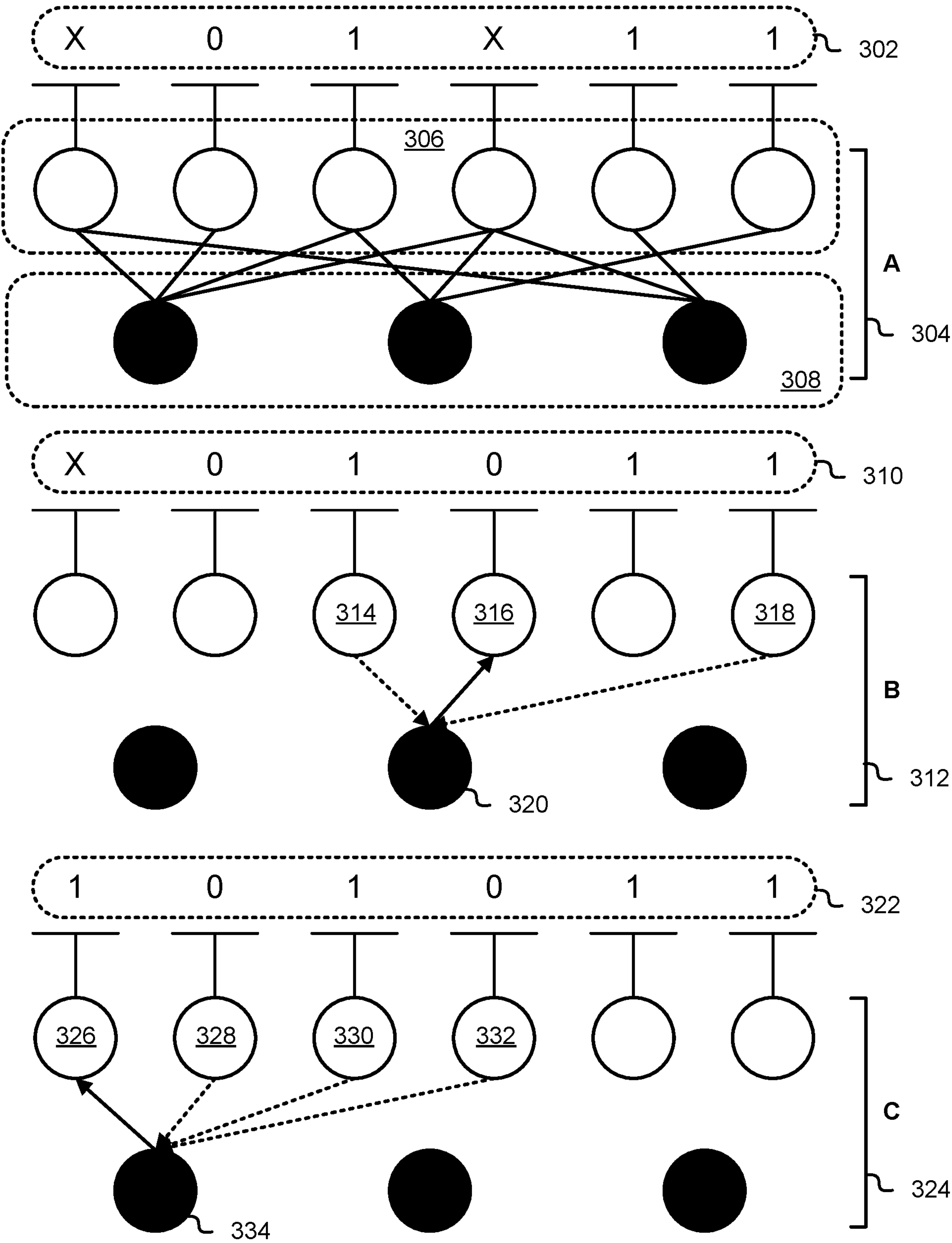


FIG. 3

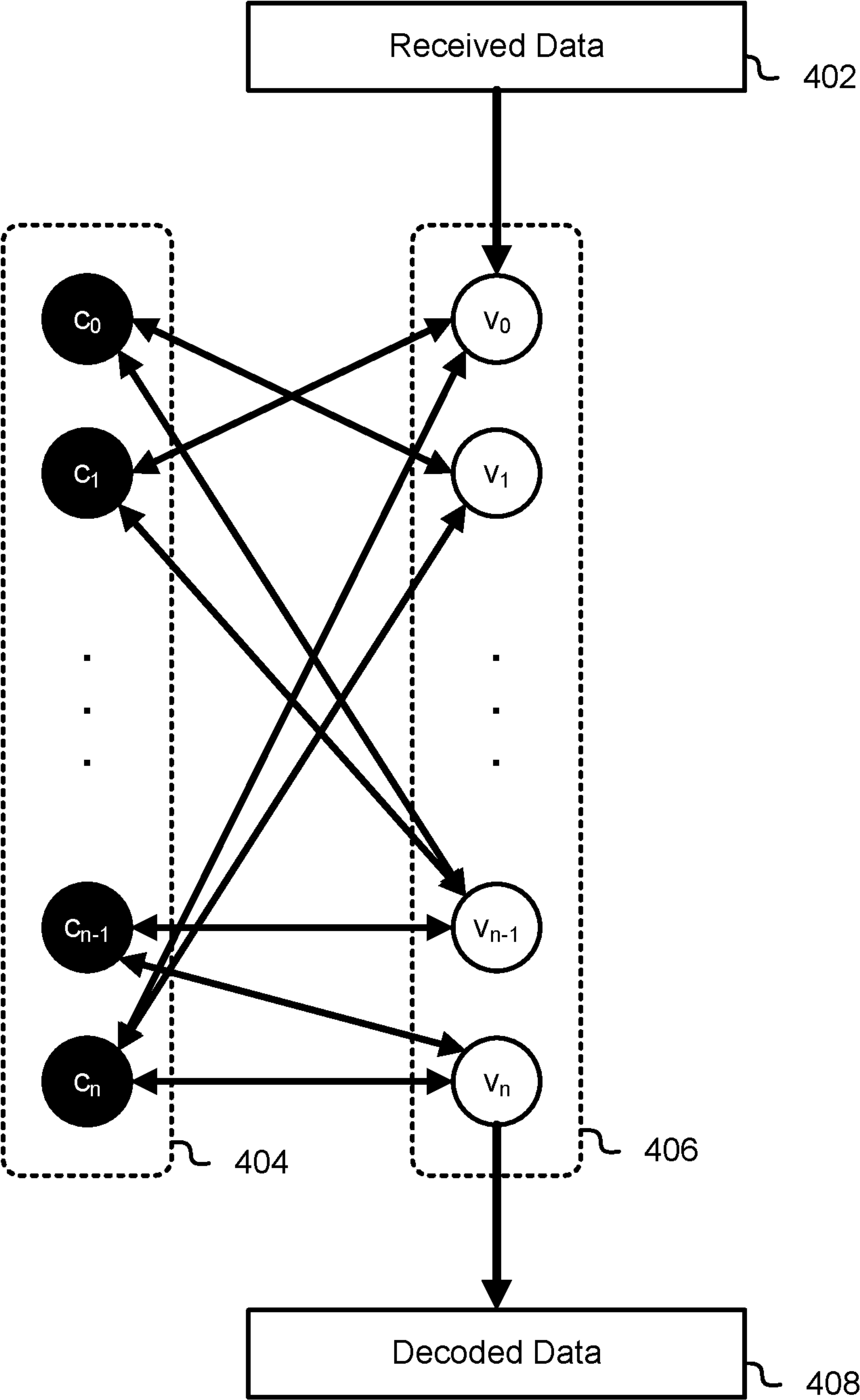


FIG. 4

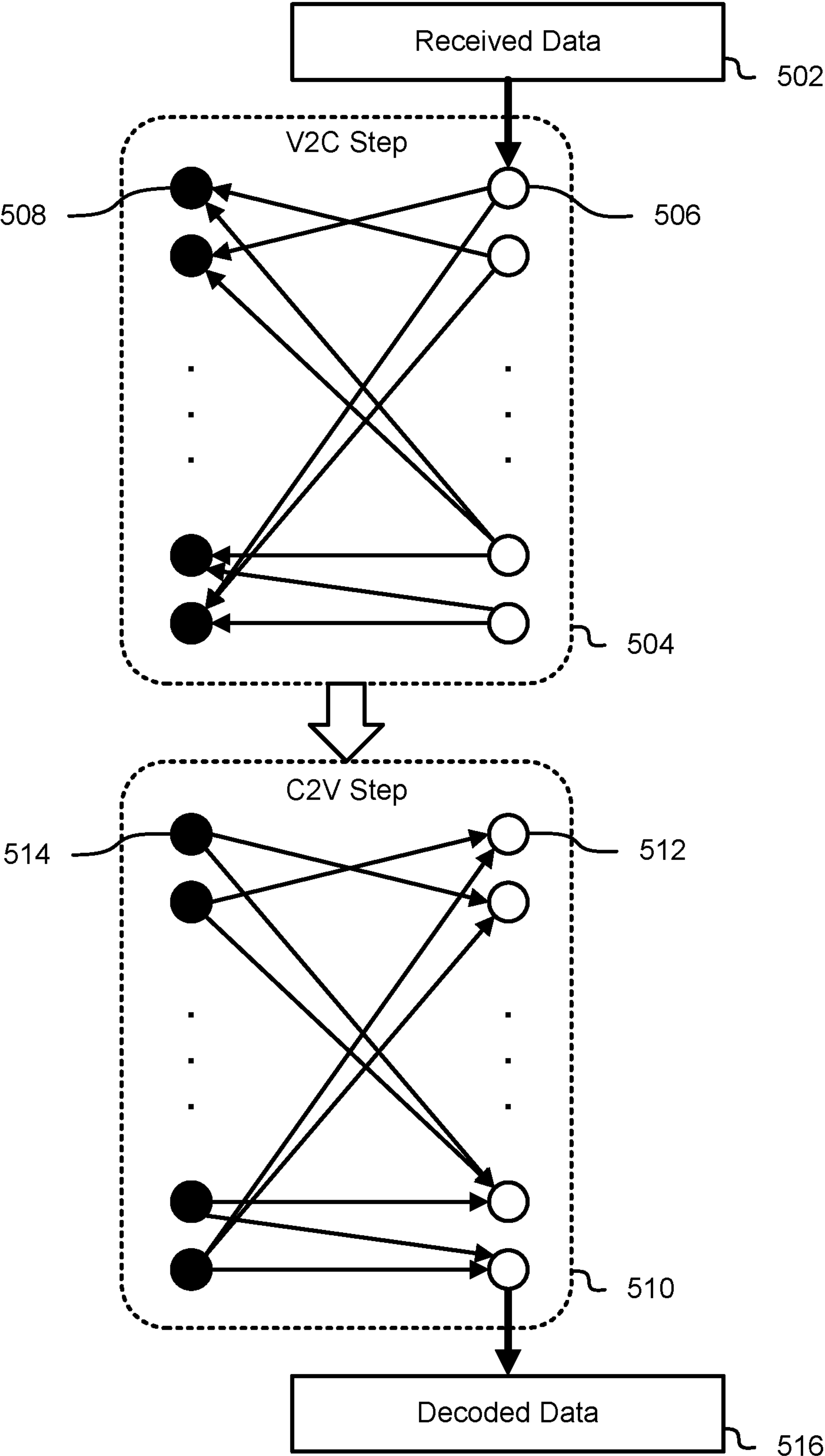
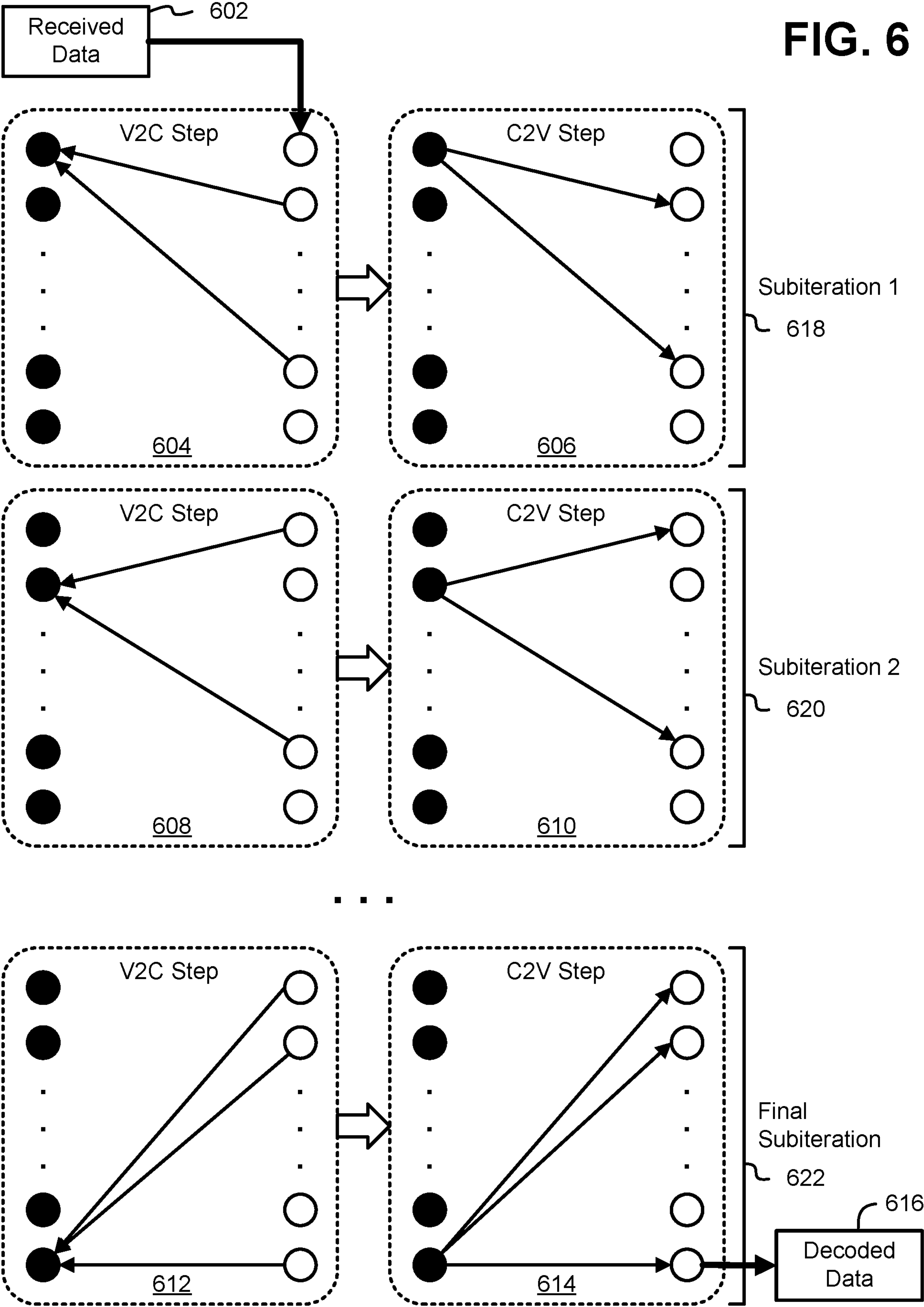


FIG. 5

FIG. 6



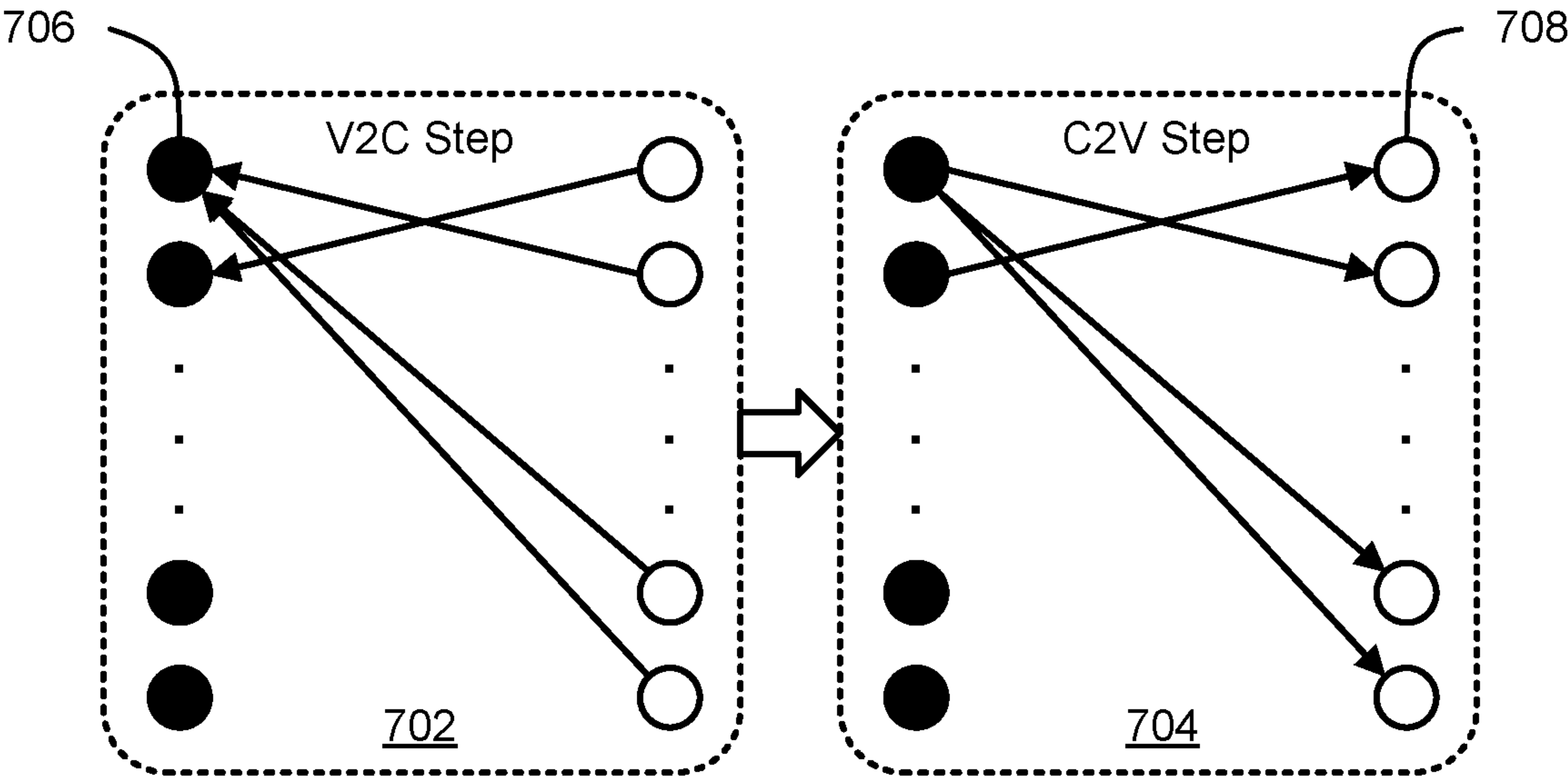


FIG. 7A

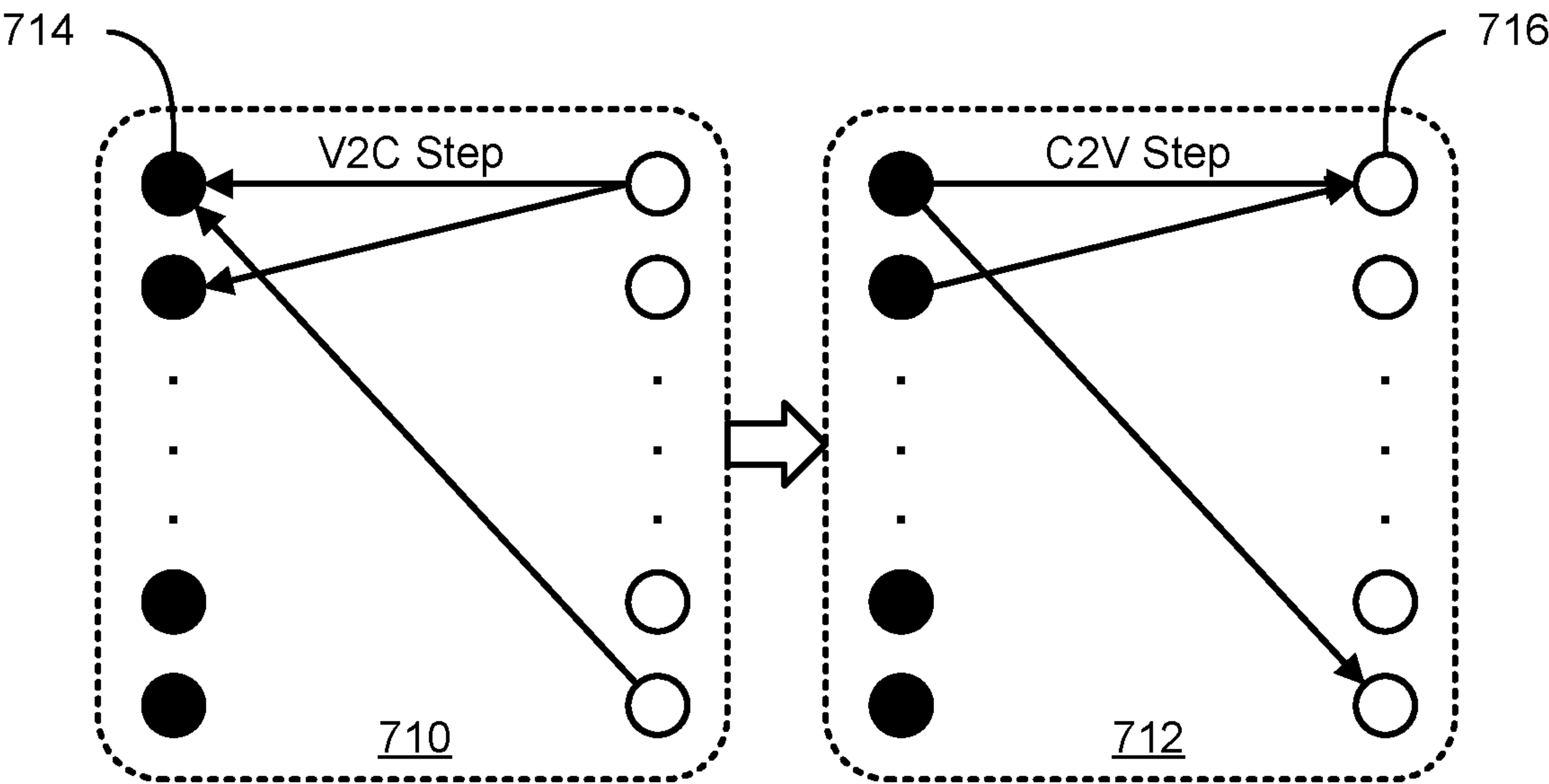


FIG. 7B

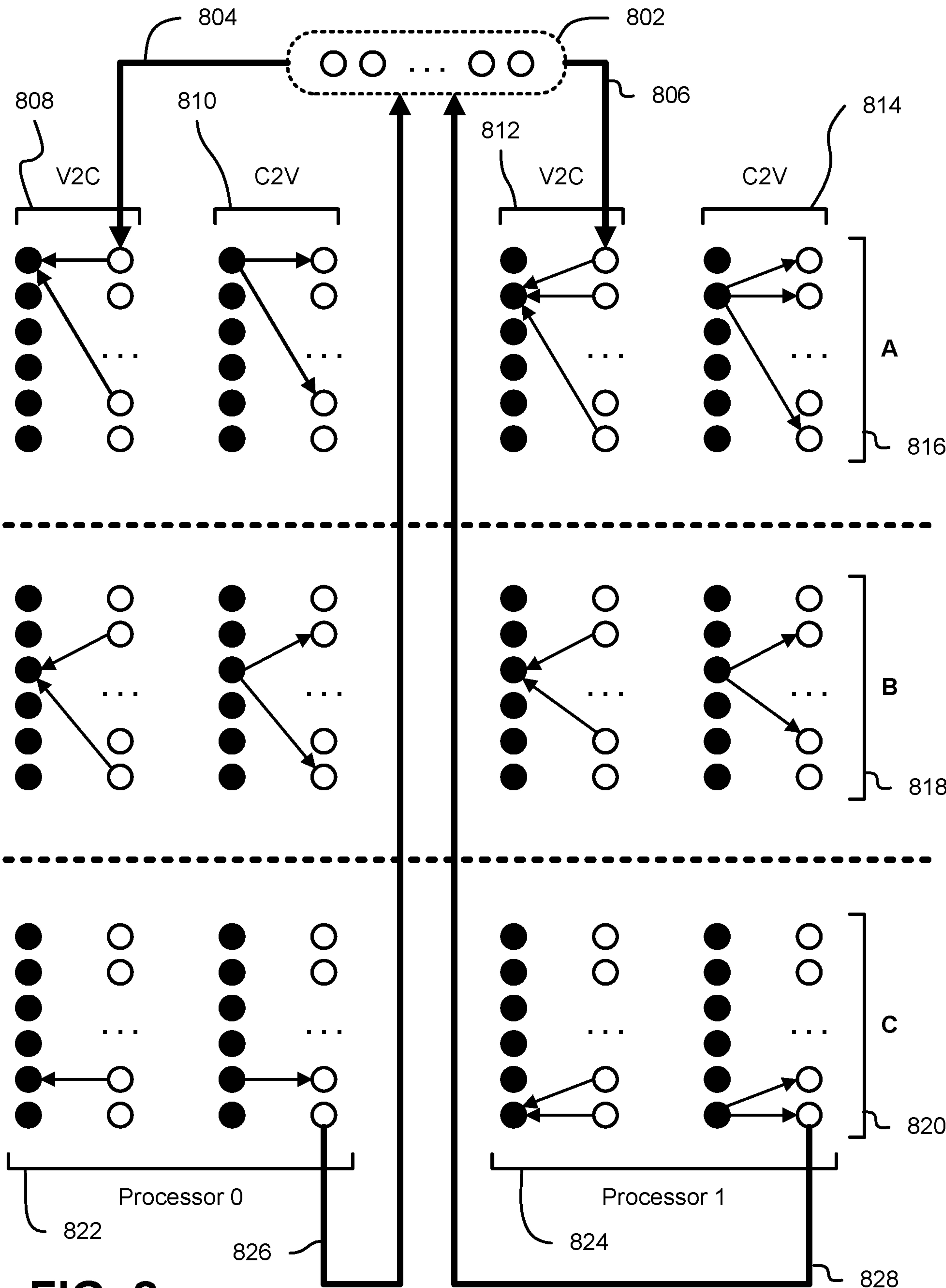


FIG. 8

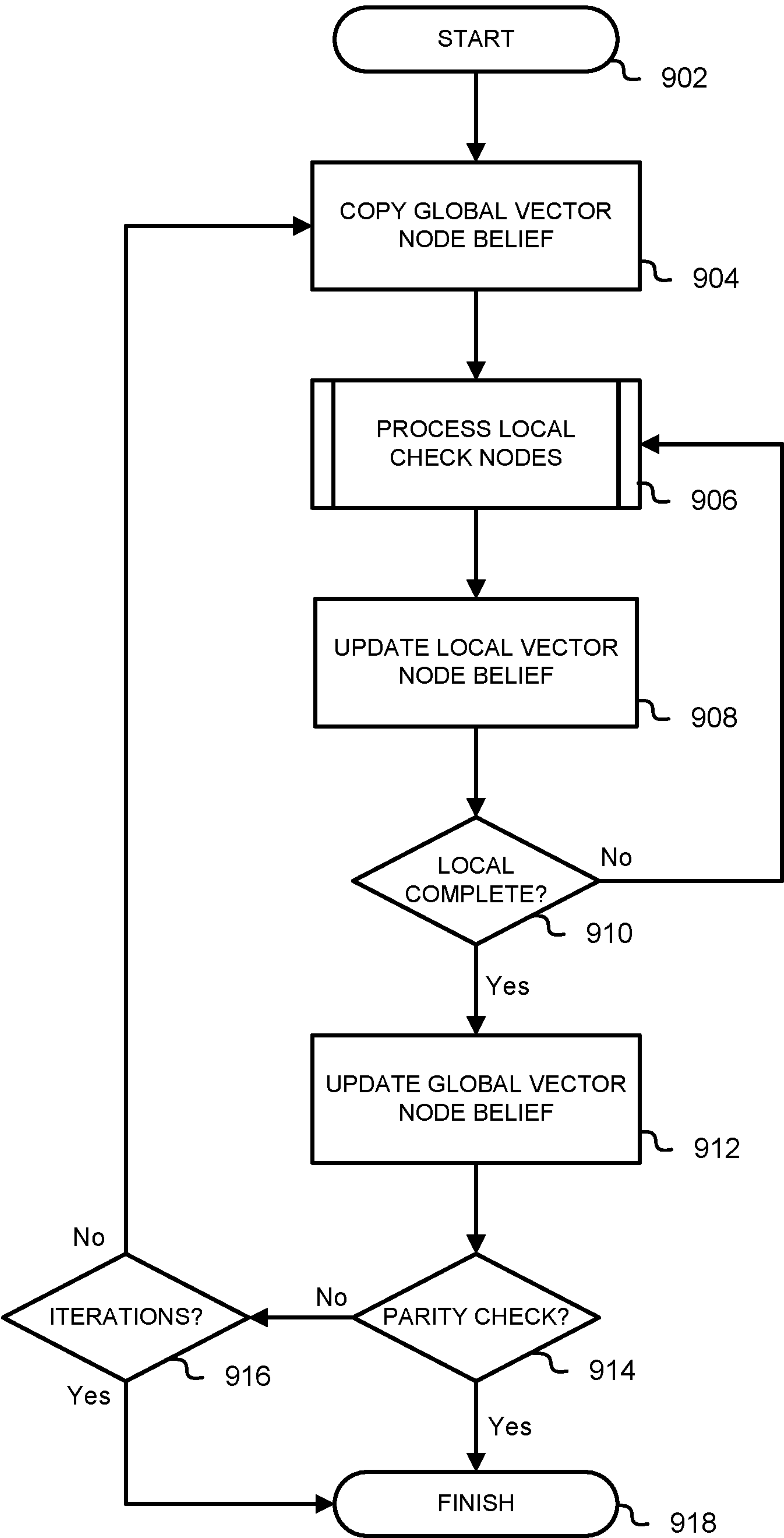


FIG. 9

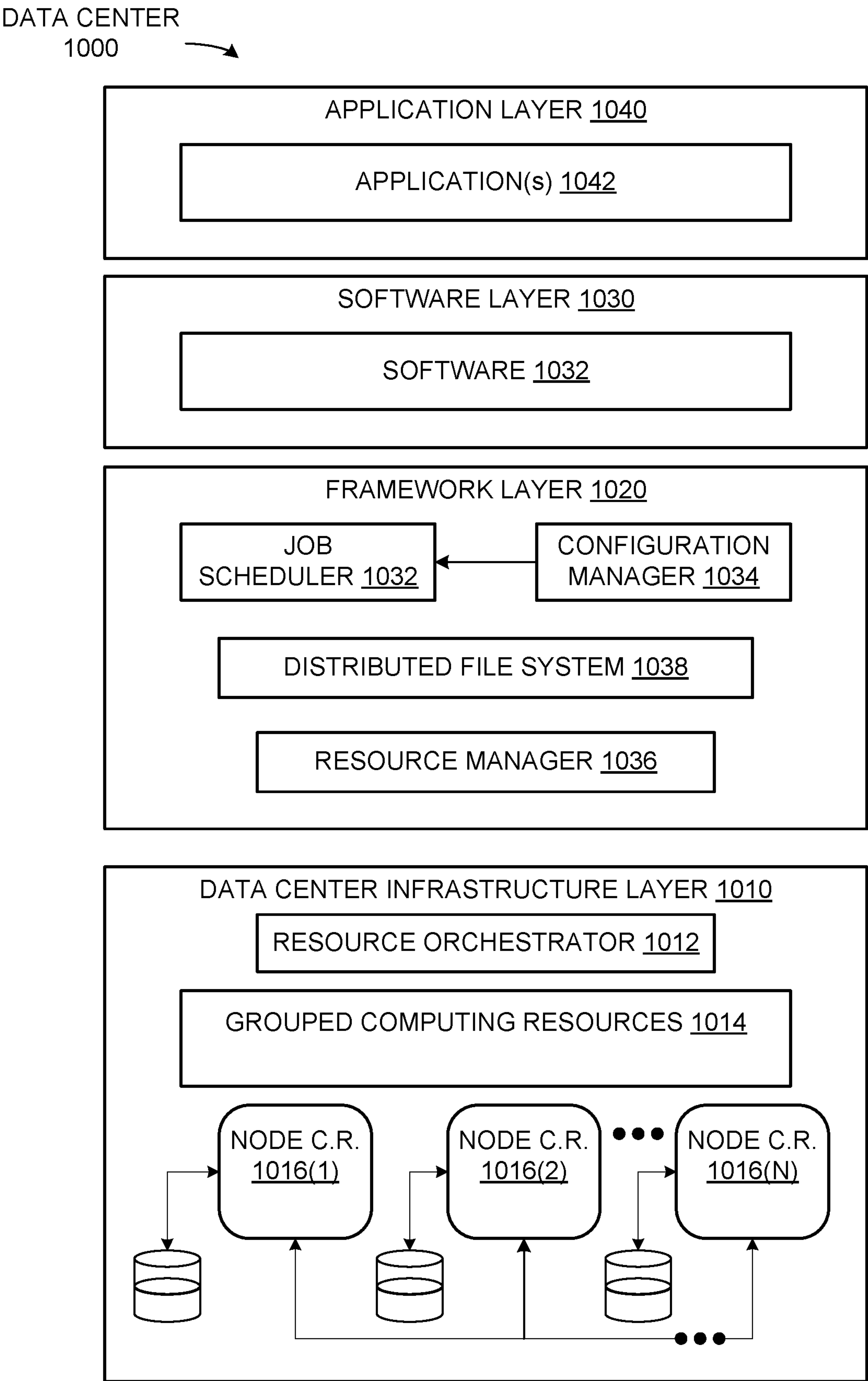


FIG. 10

1100

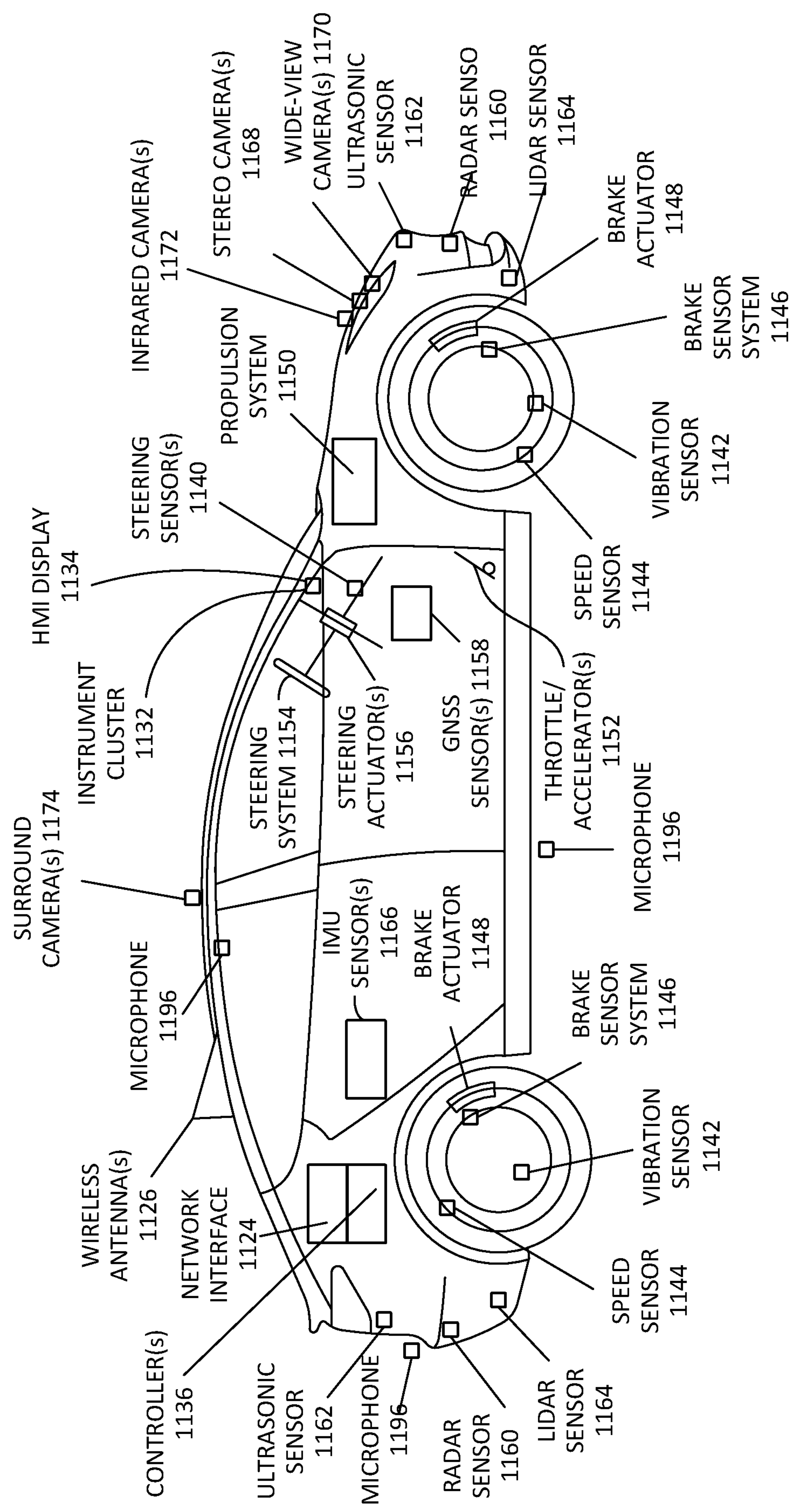


FIG. 11A

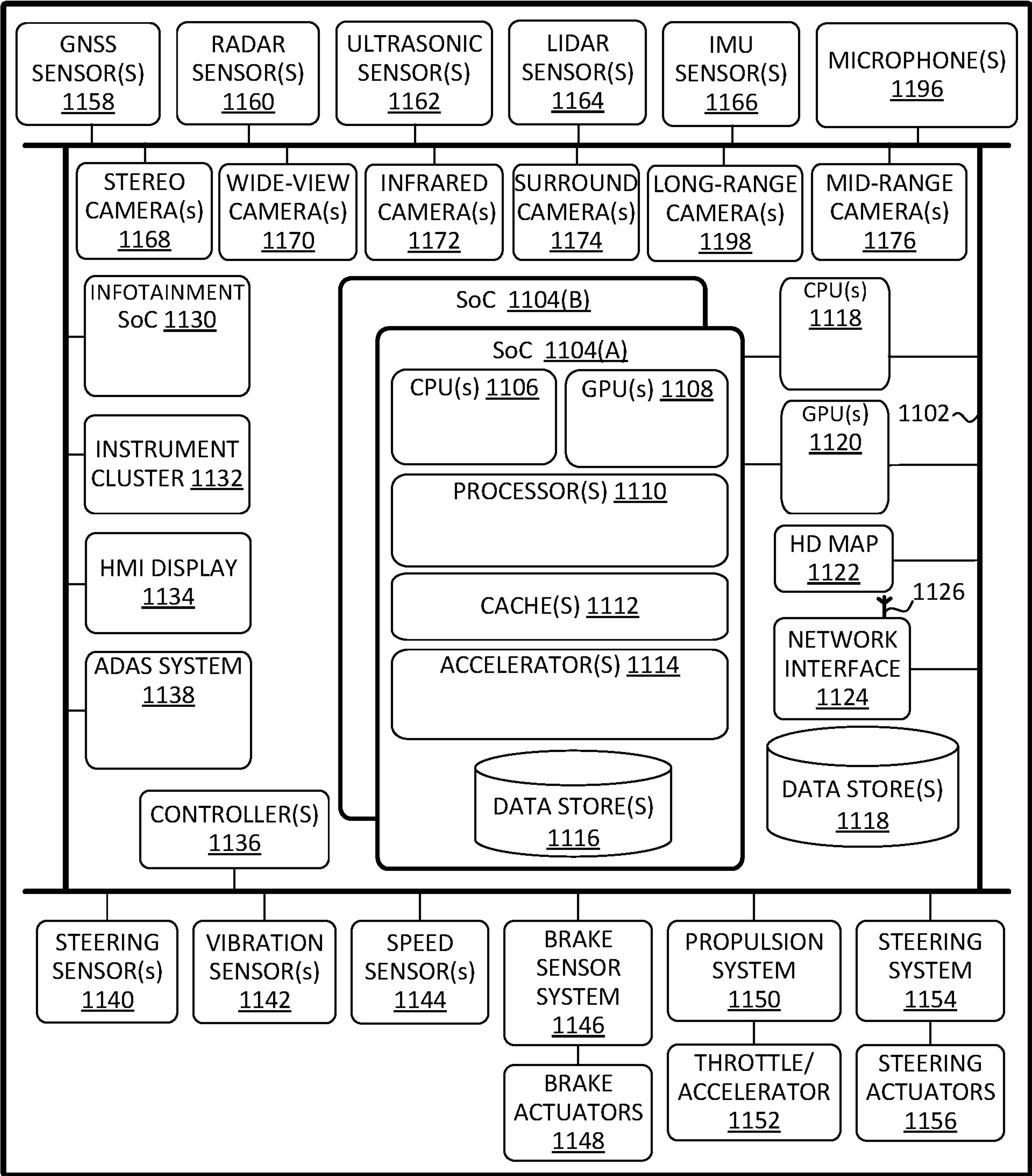
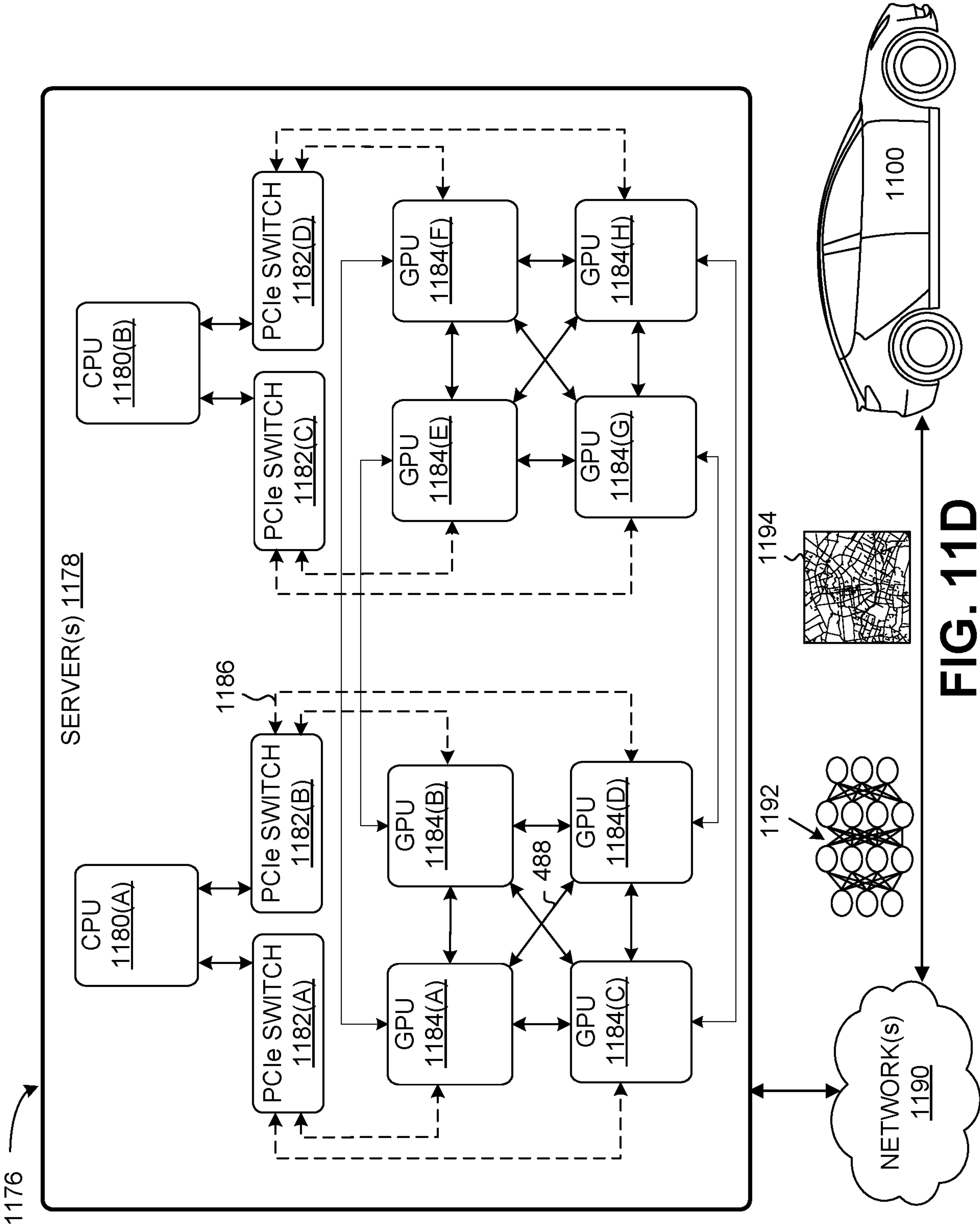


FIG. 11C



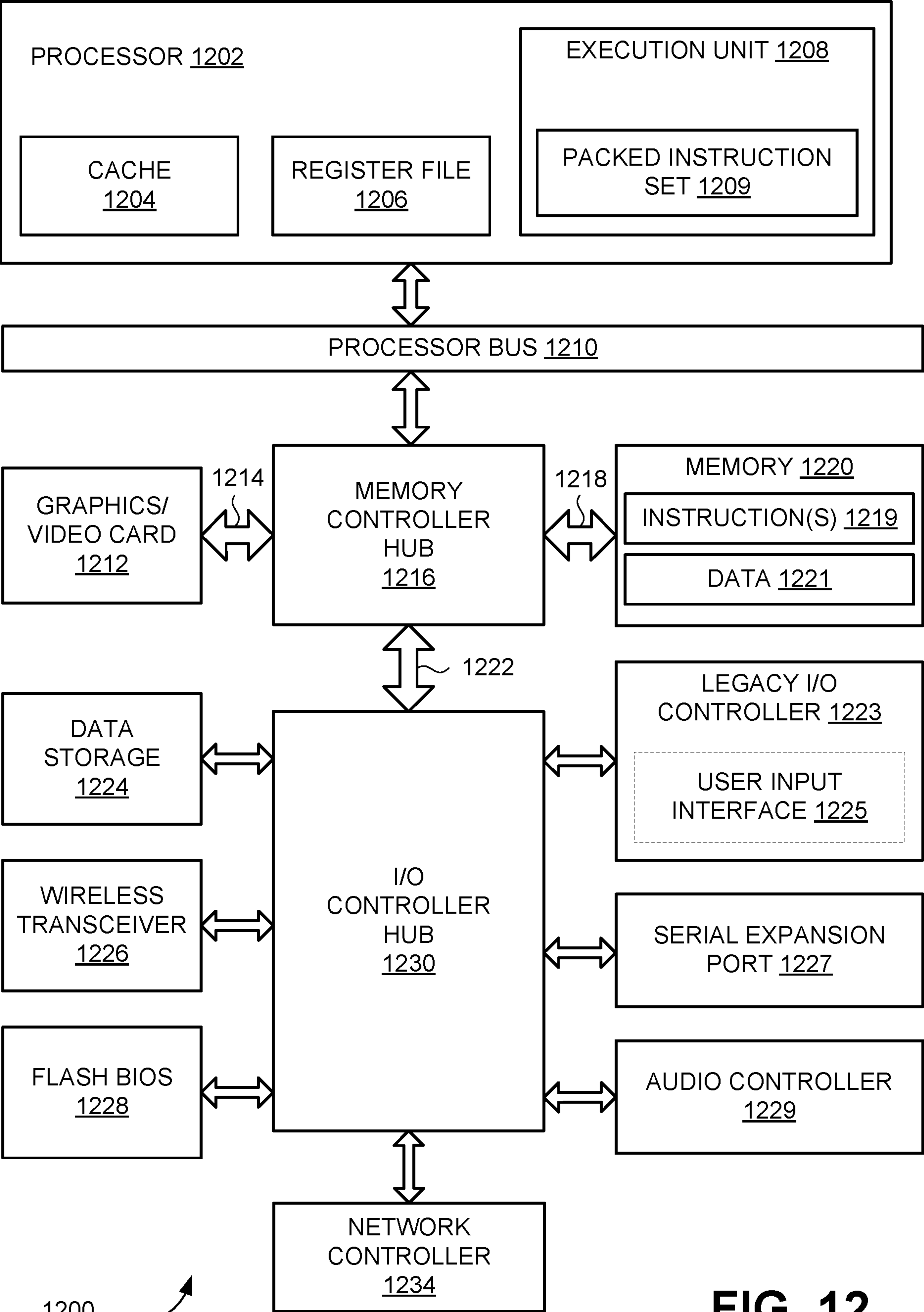


FIG. 12

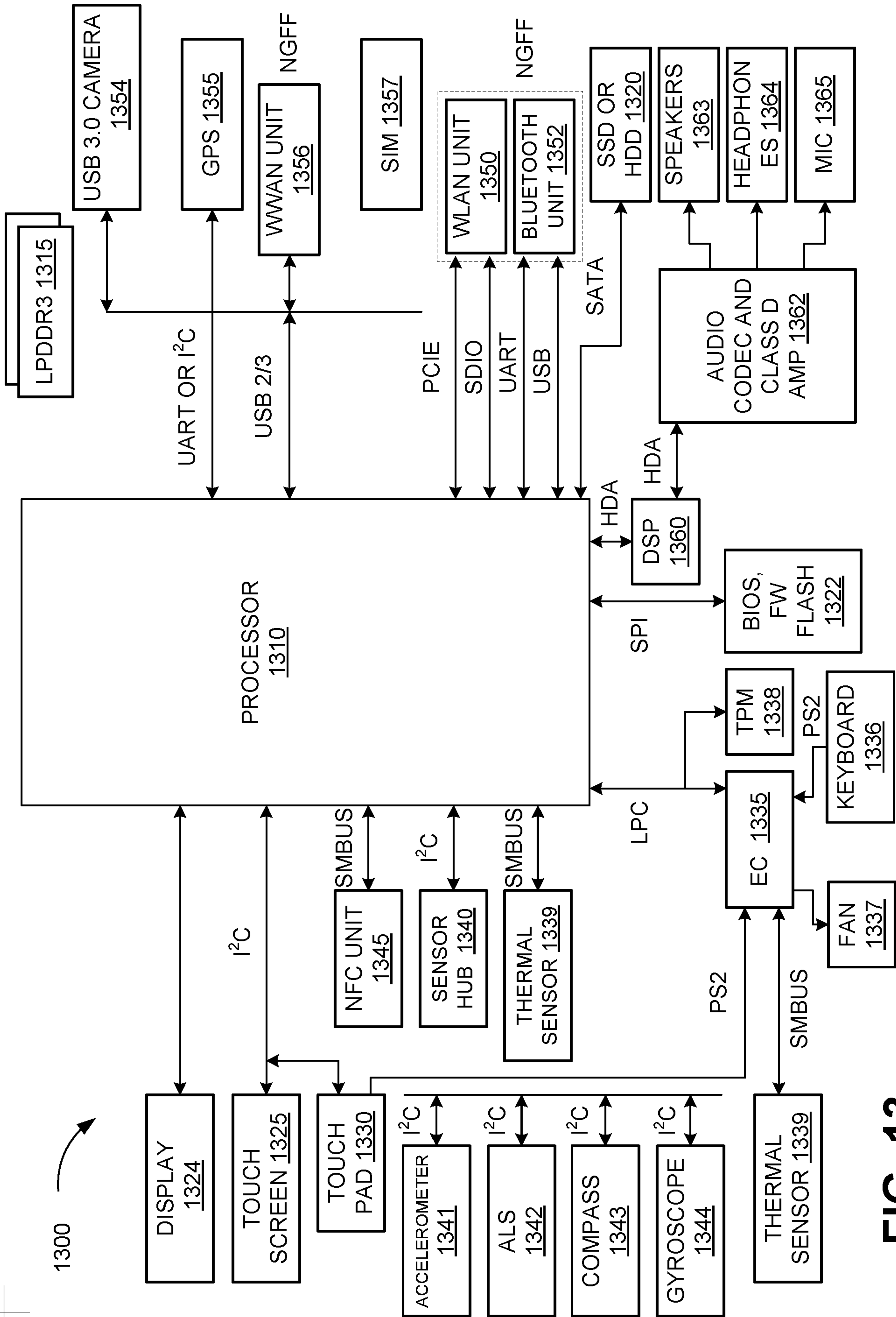


FIG. 13

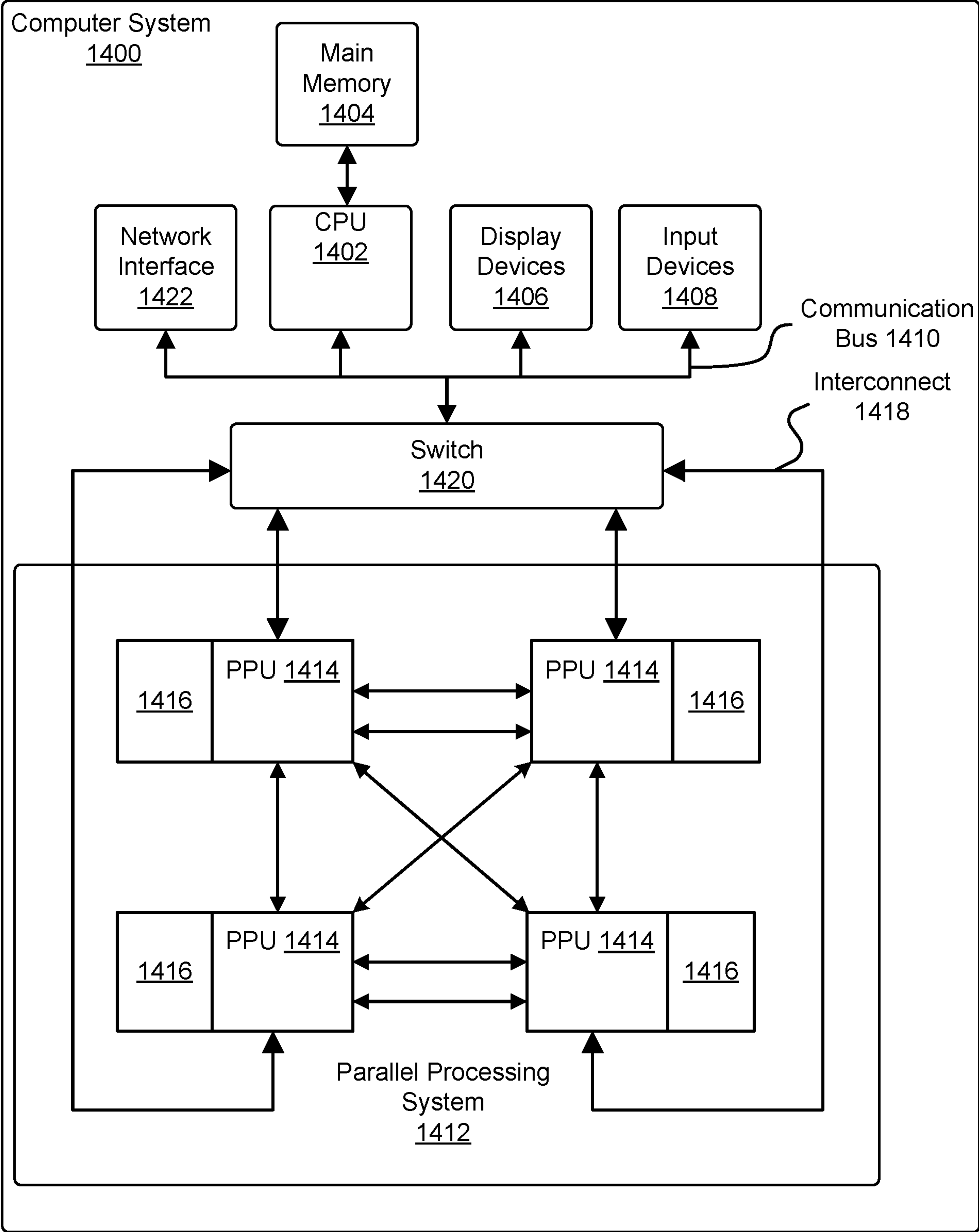


FIG. 14

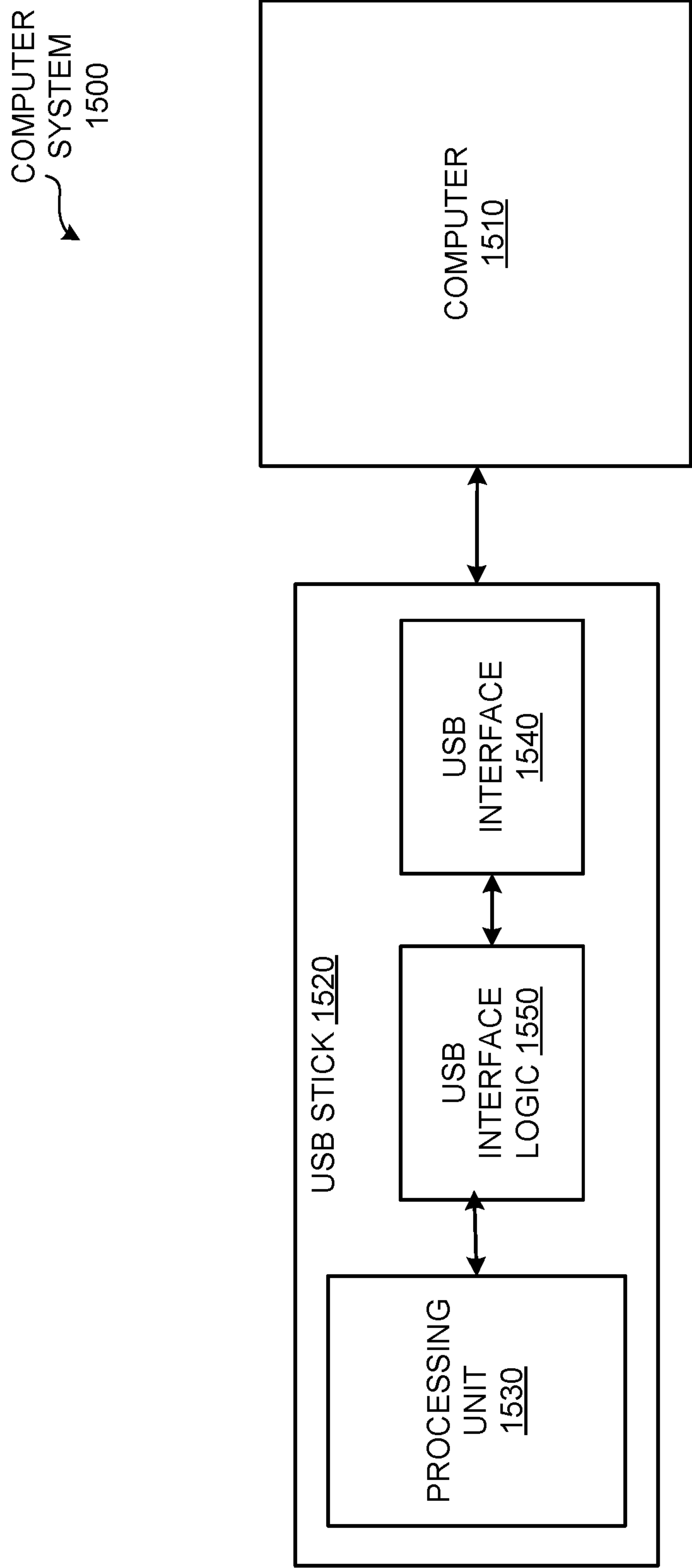


FIG. 15

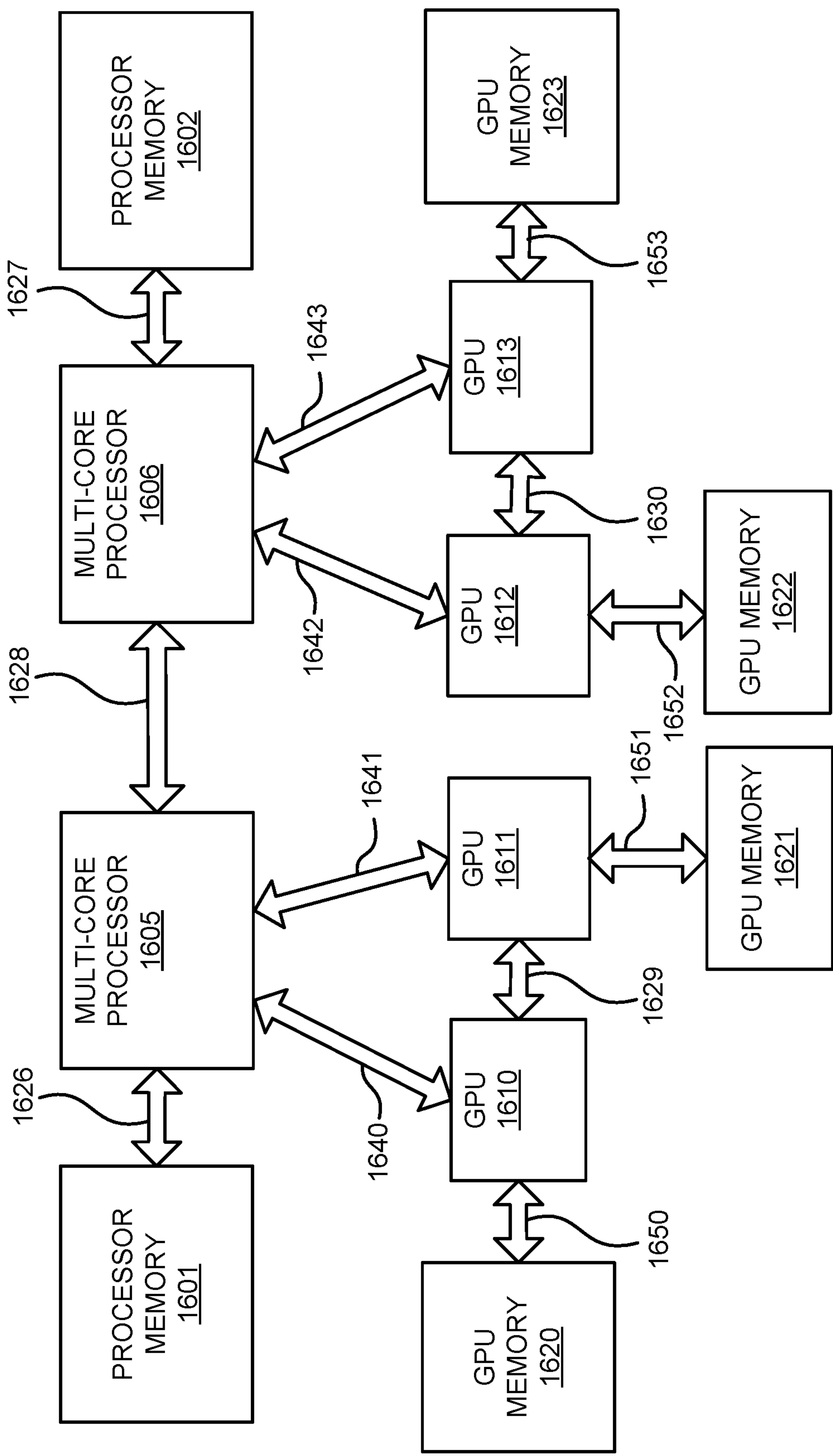


FIG. 16A

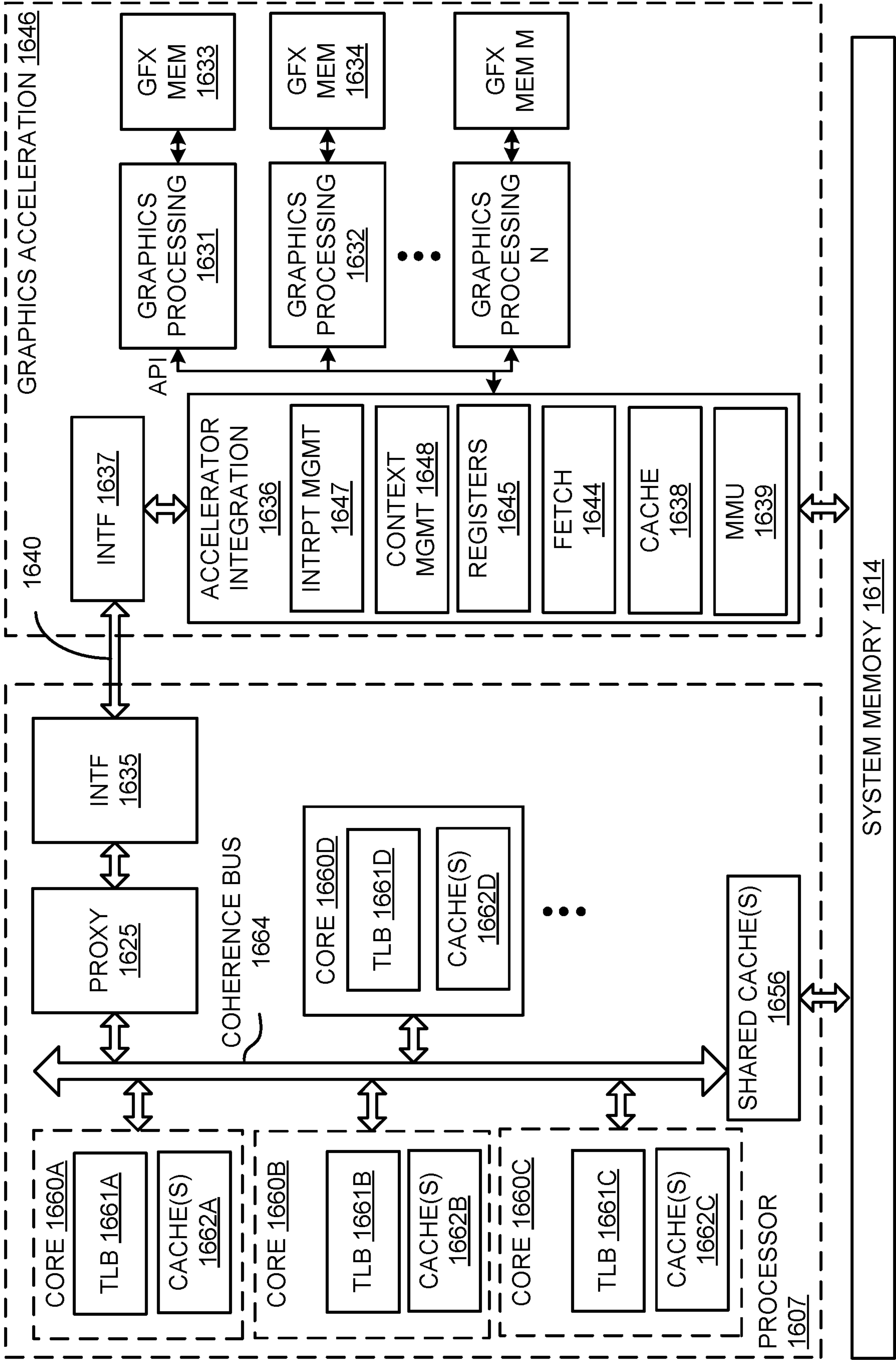


FIG. 16B

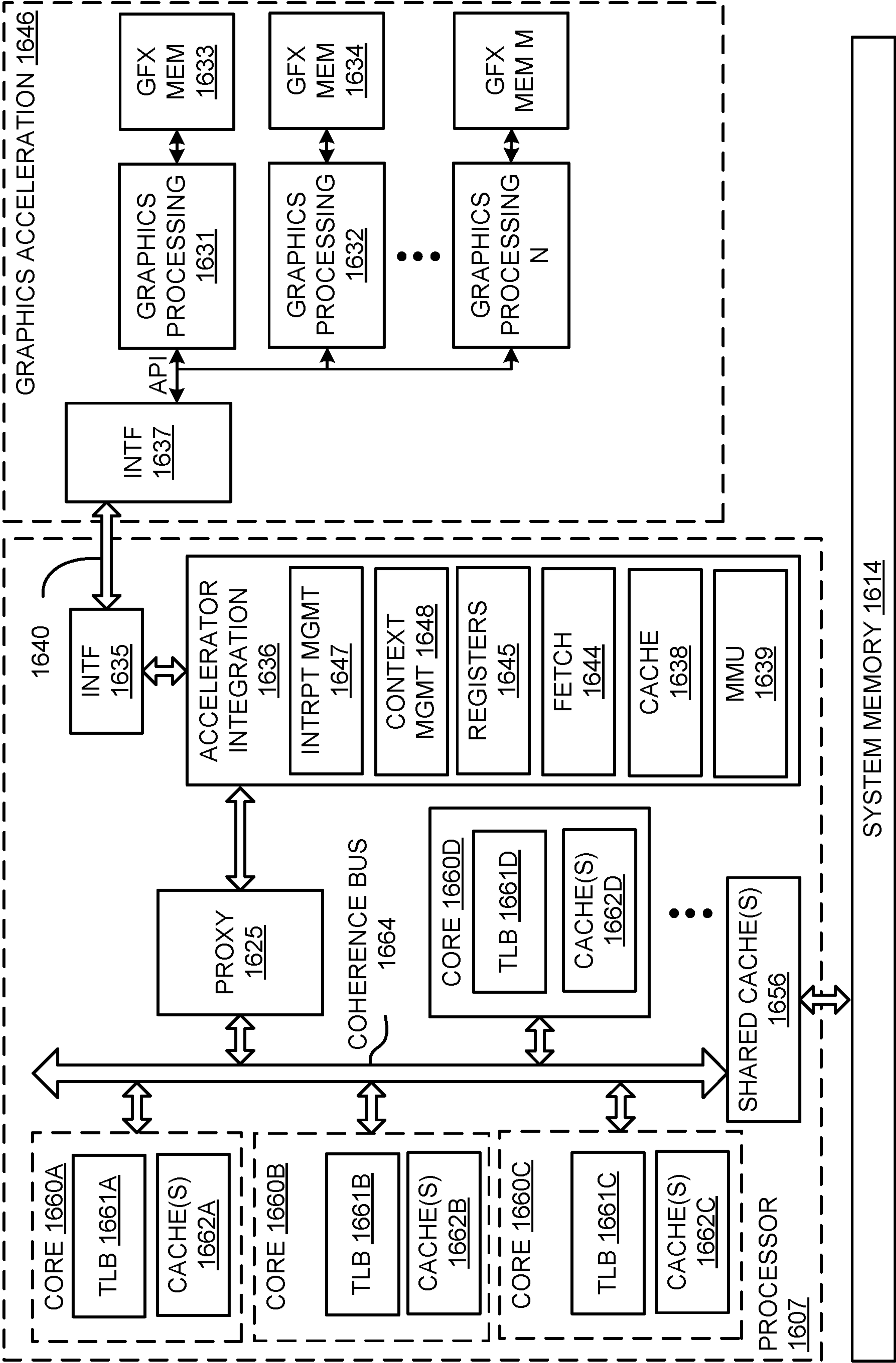


FIG. 16C

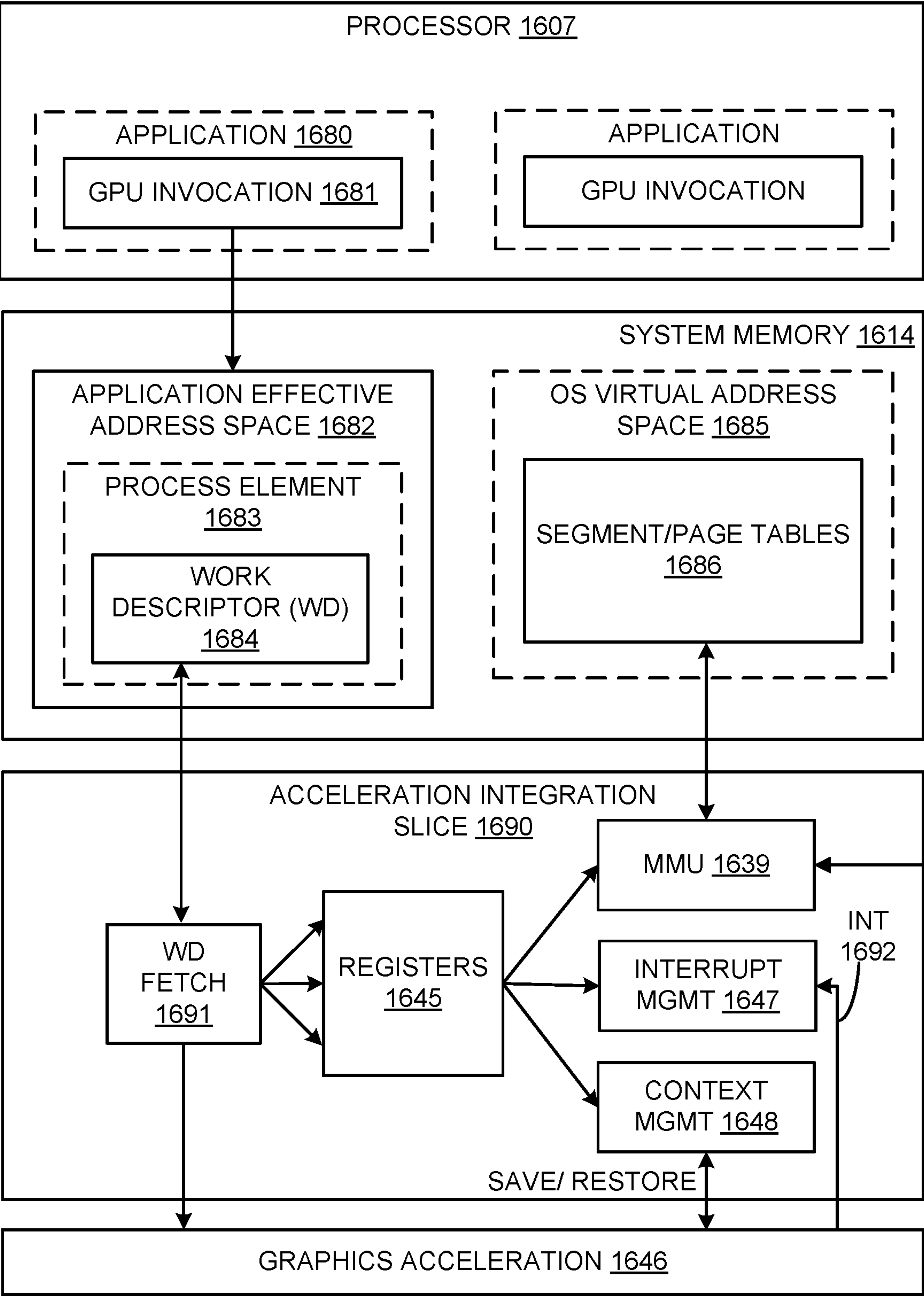


FIG. 16D

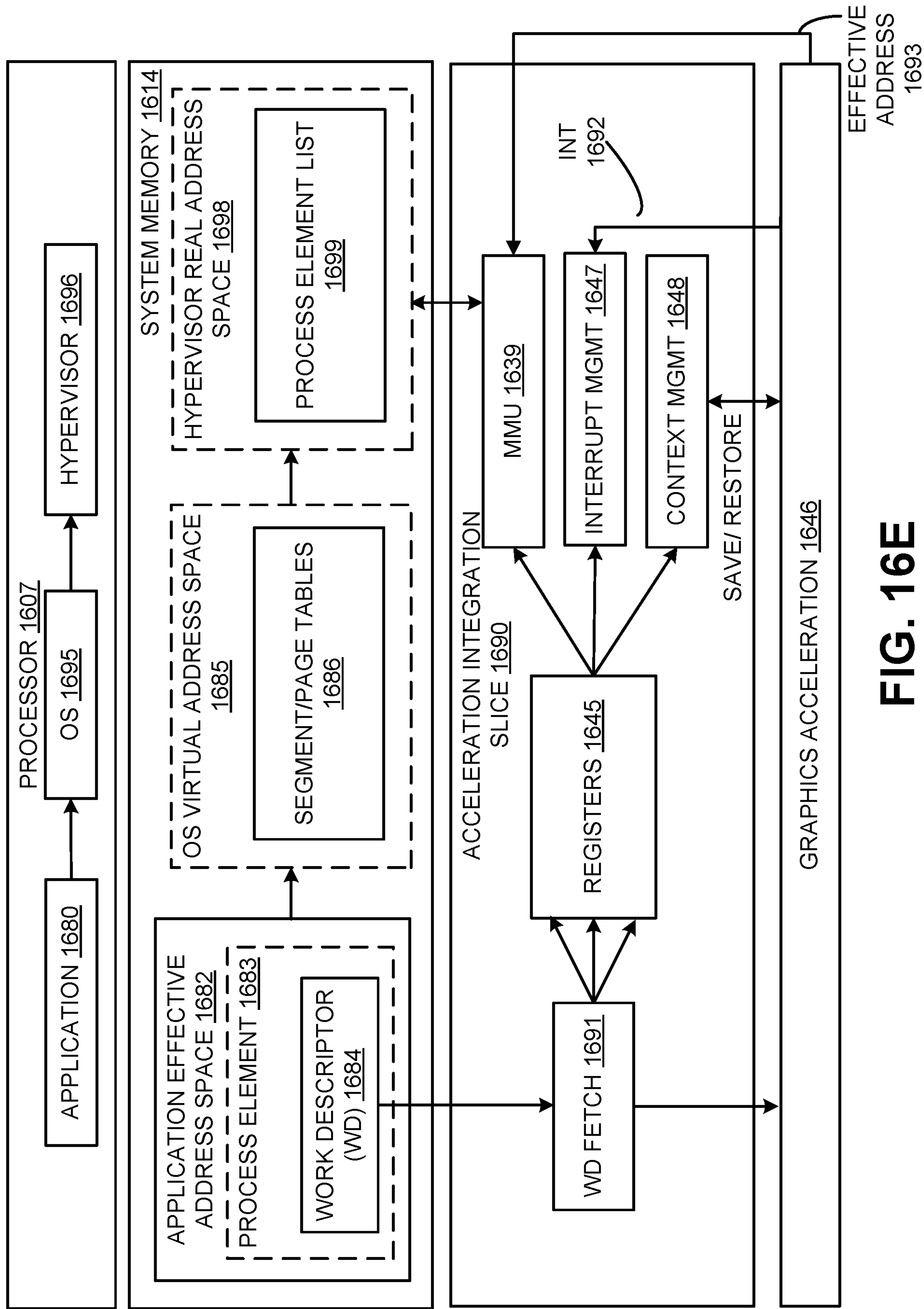


FIG. 16E

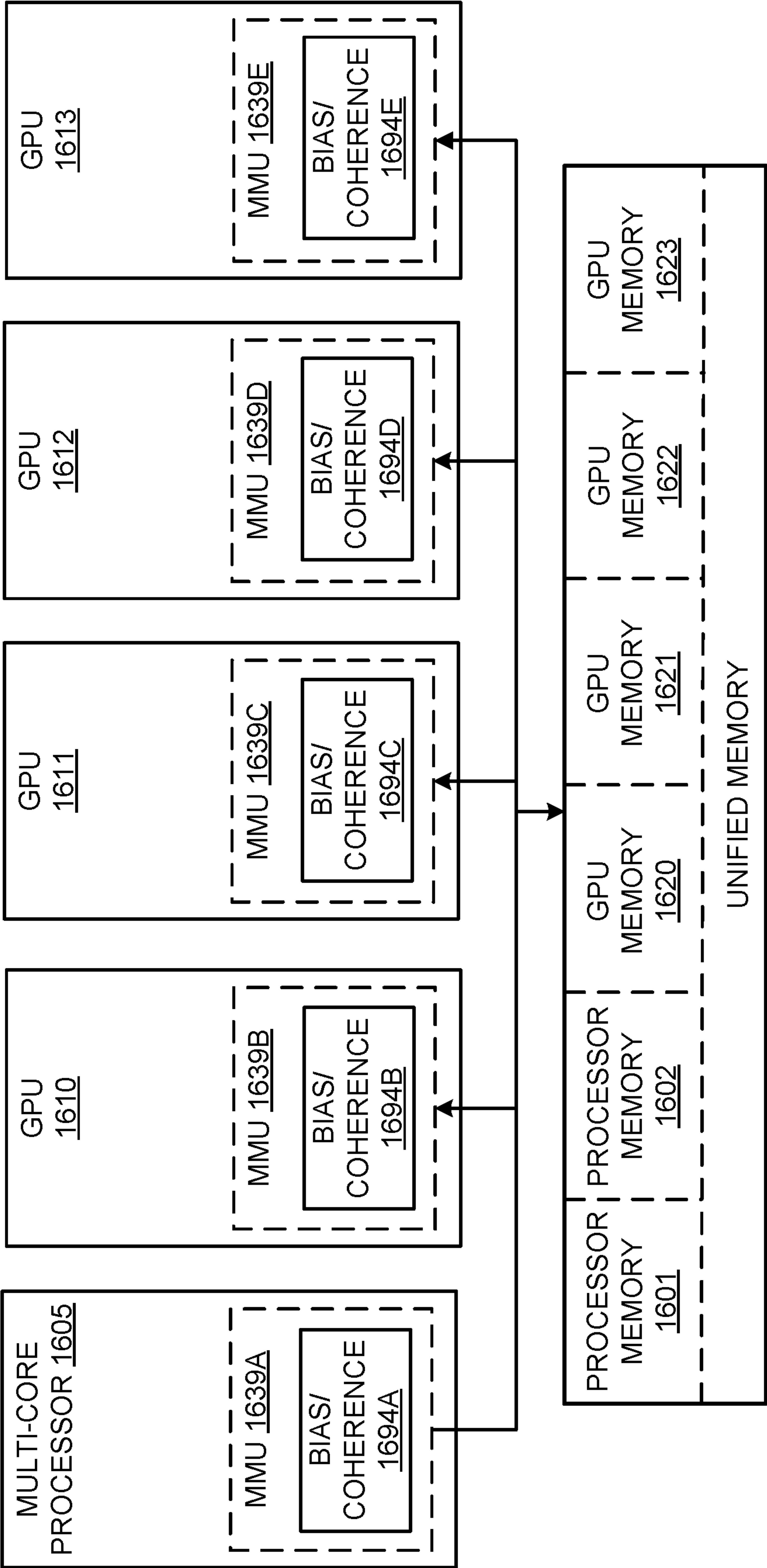


FIG. 16F

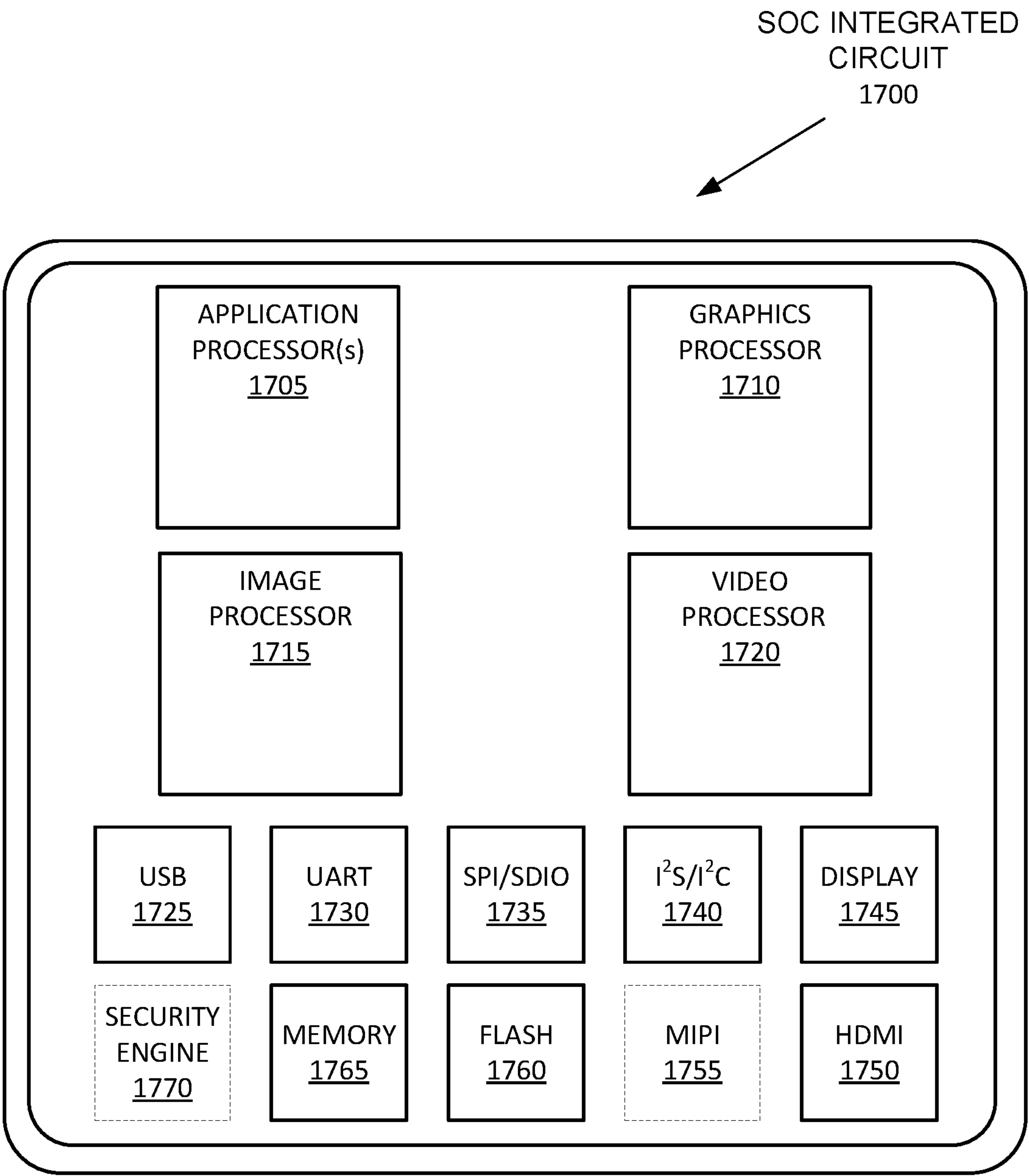


FIG. 17

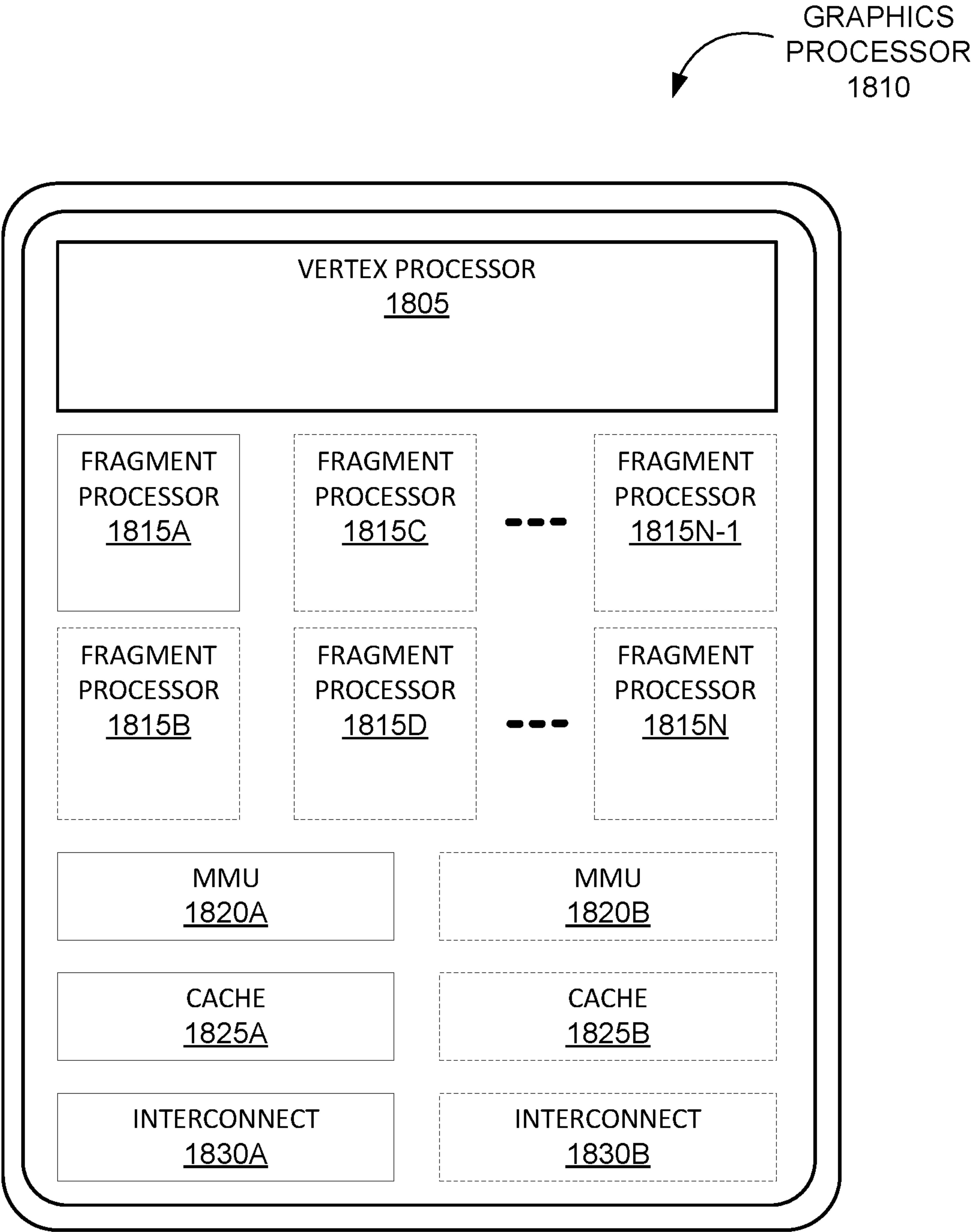


FIG. 18A

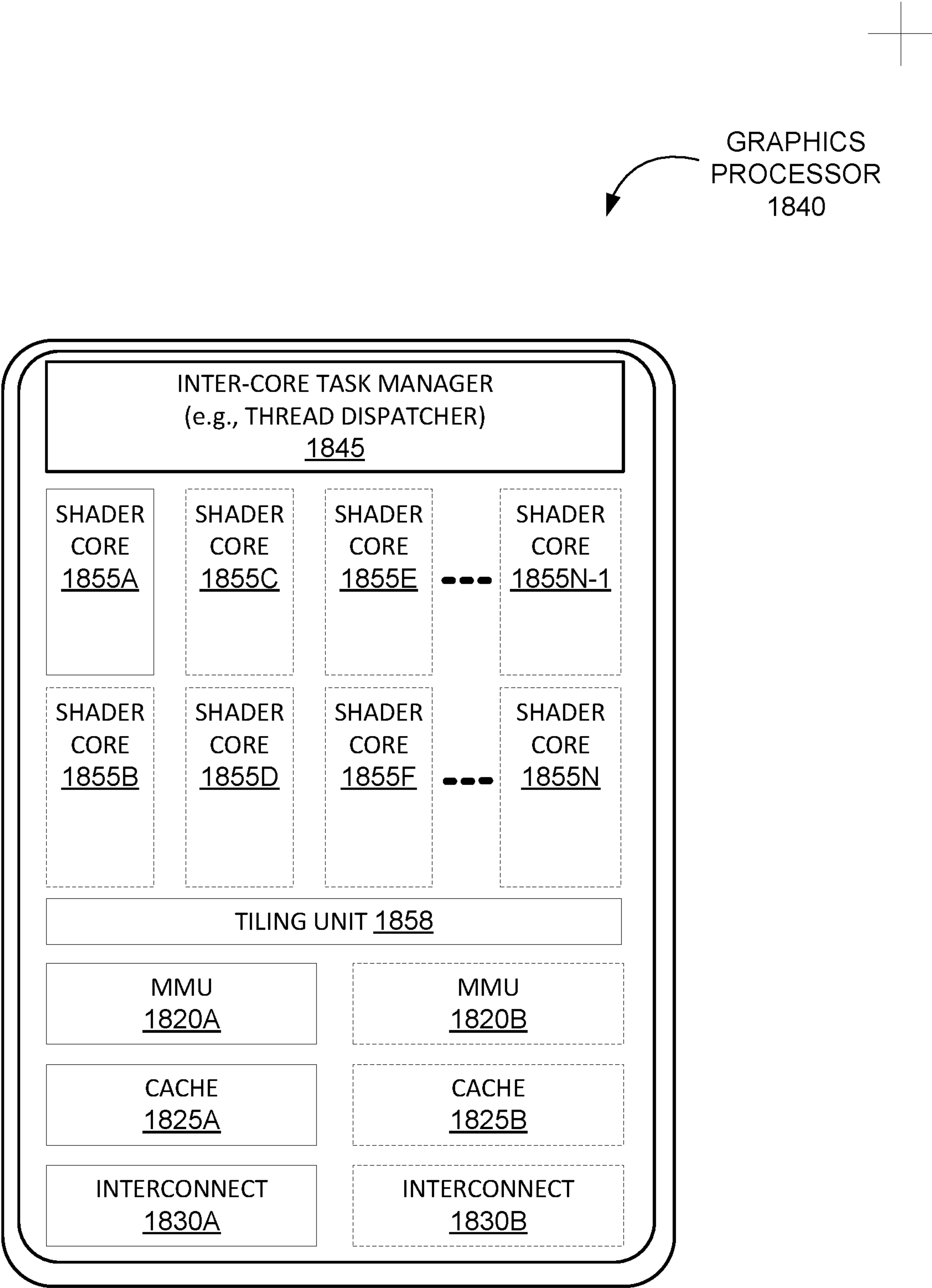


FIG. 18B

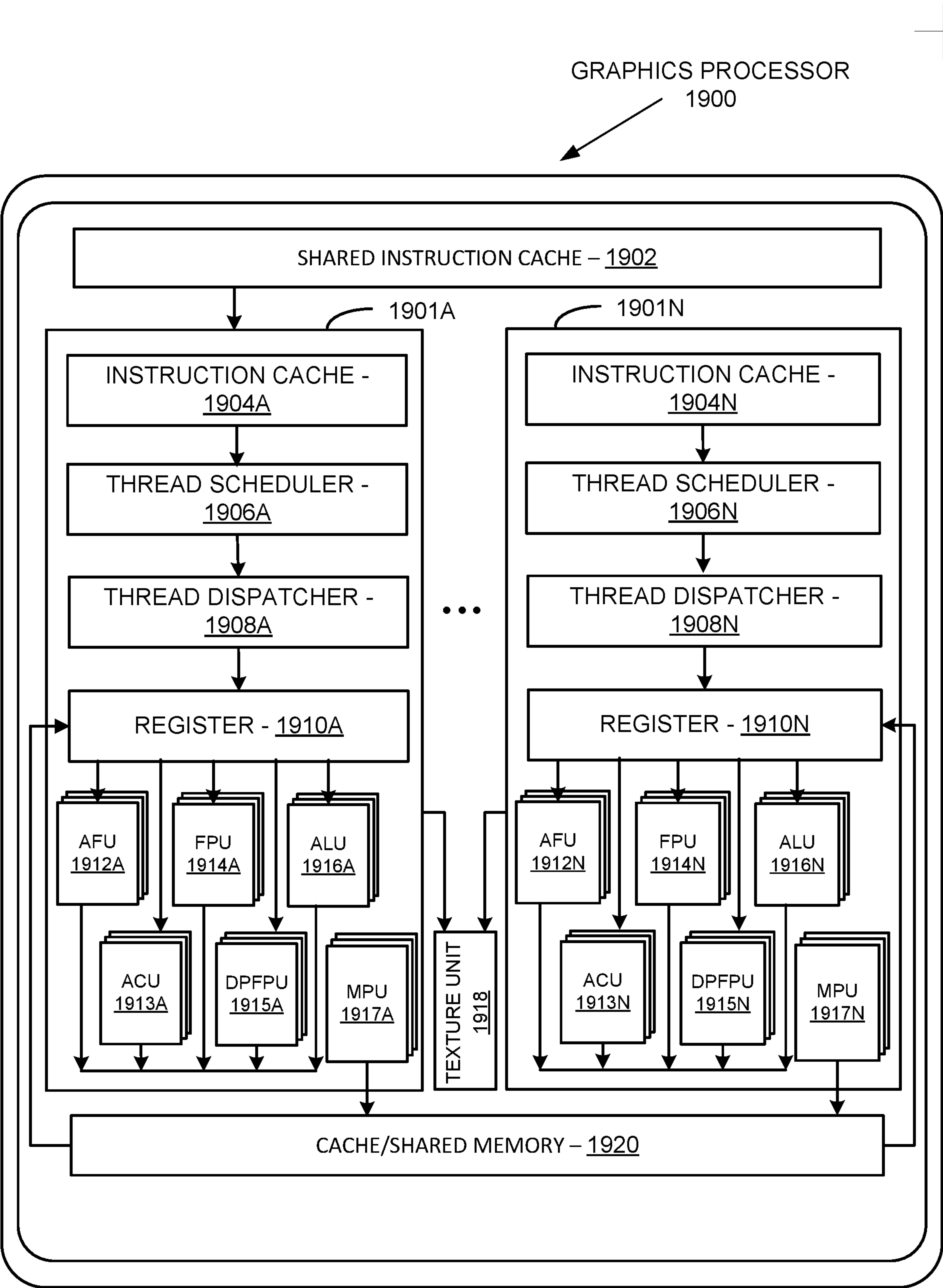


FIG. 19A

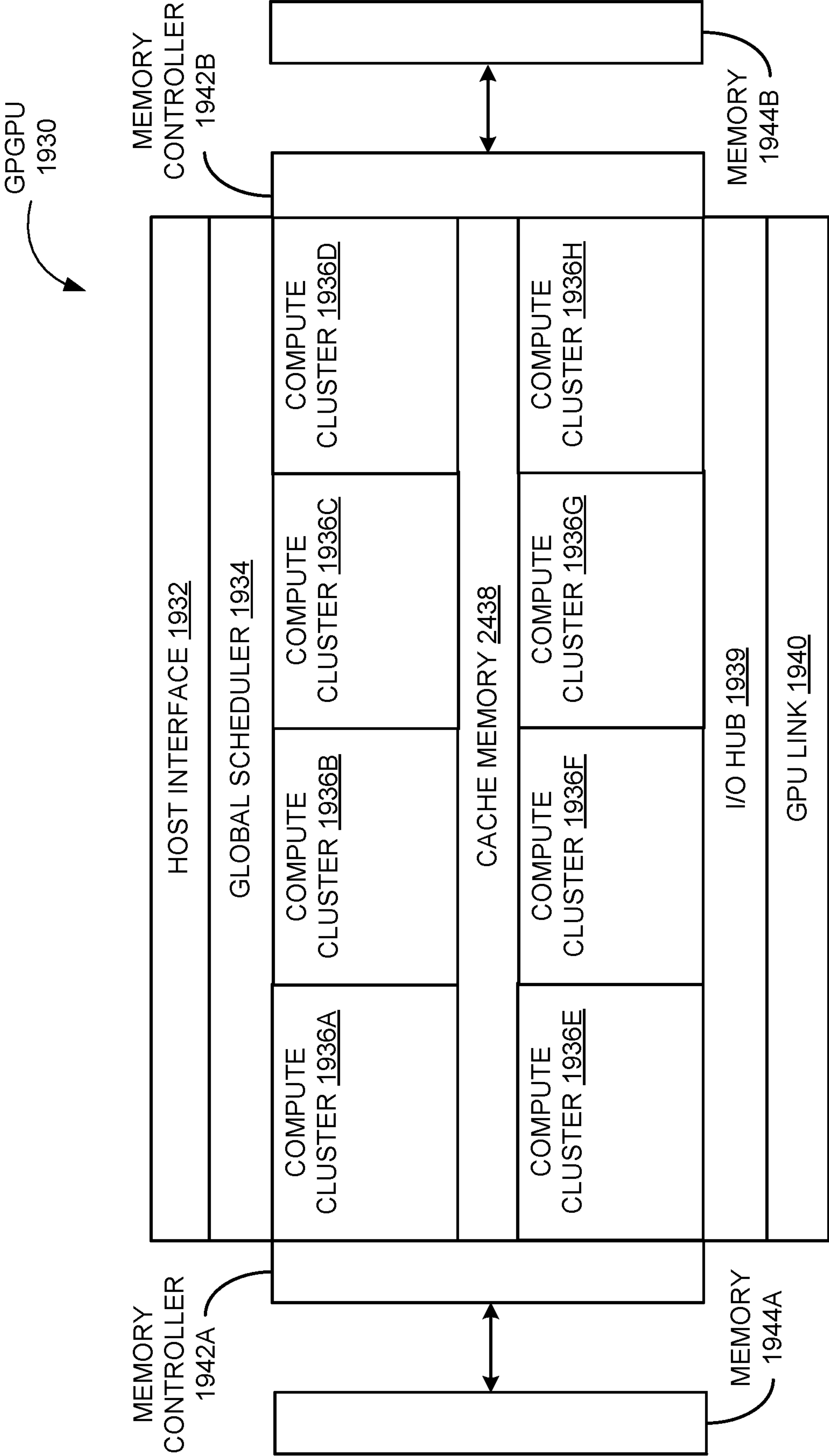


FIG. 19B

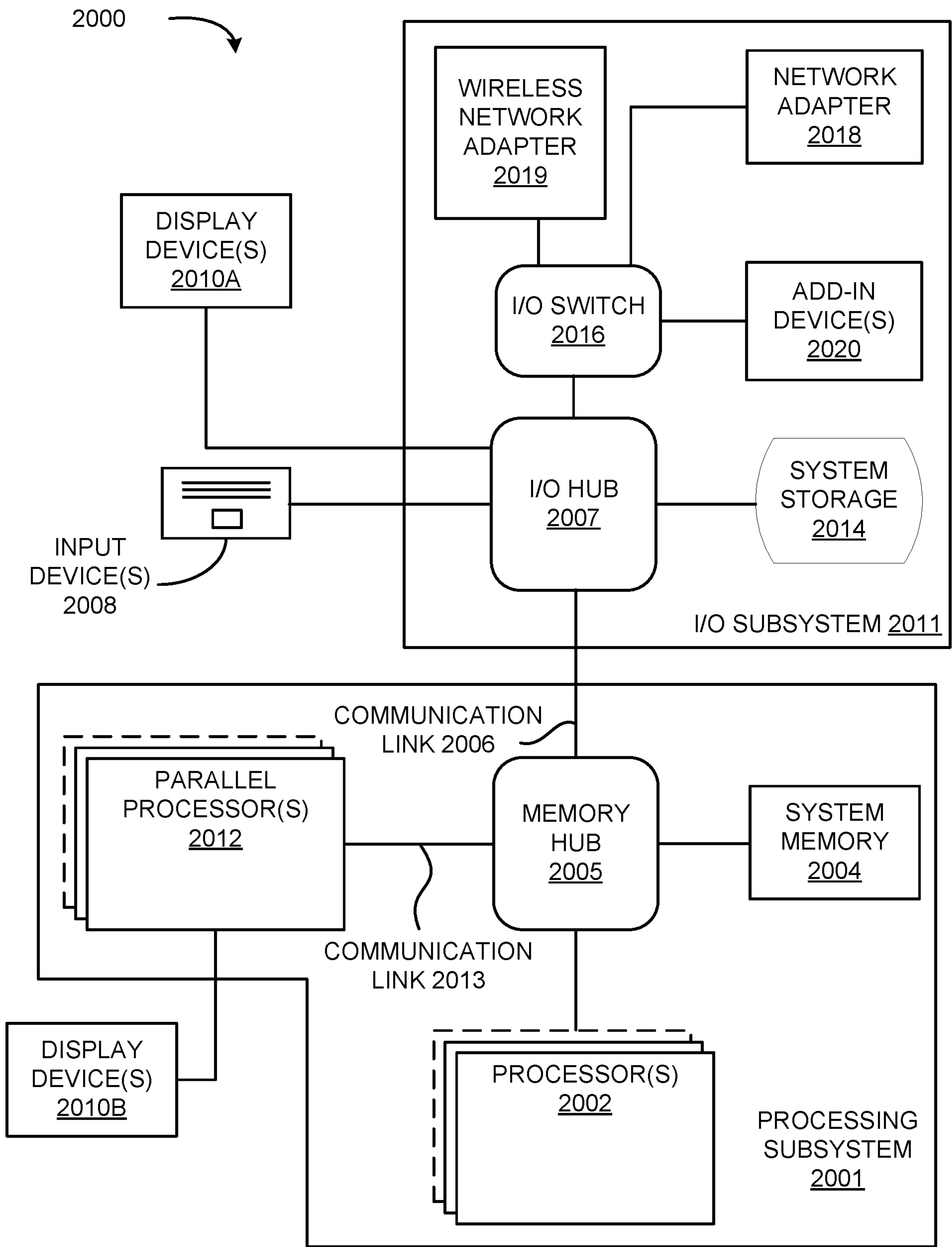


FIG. 20

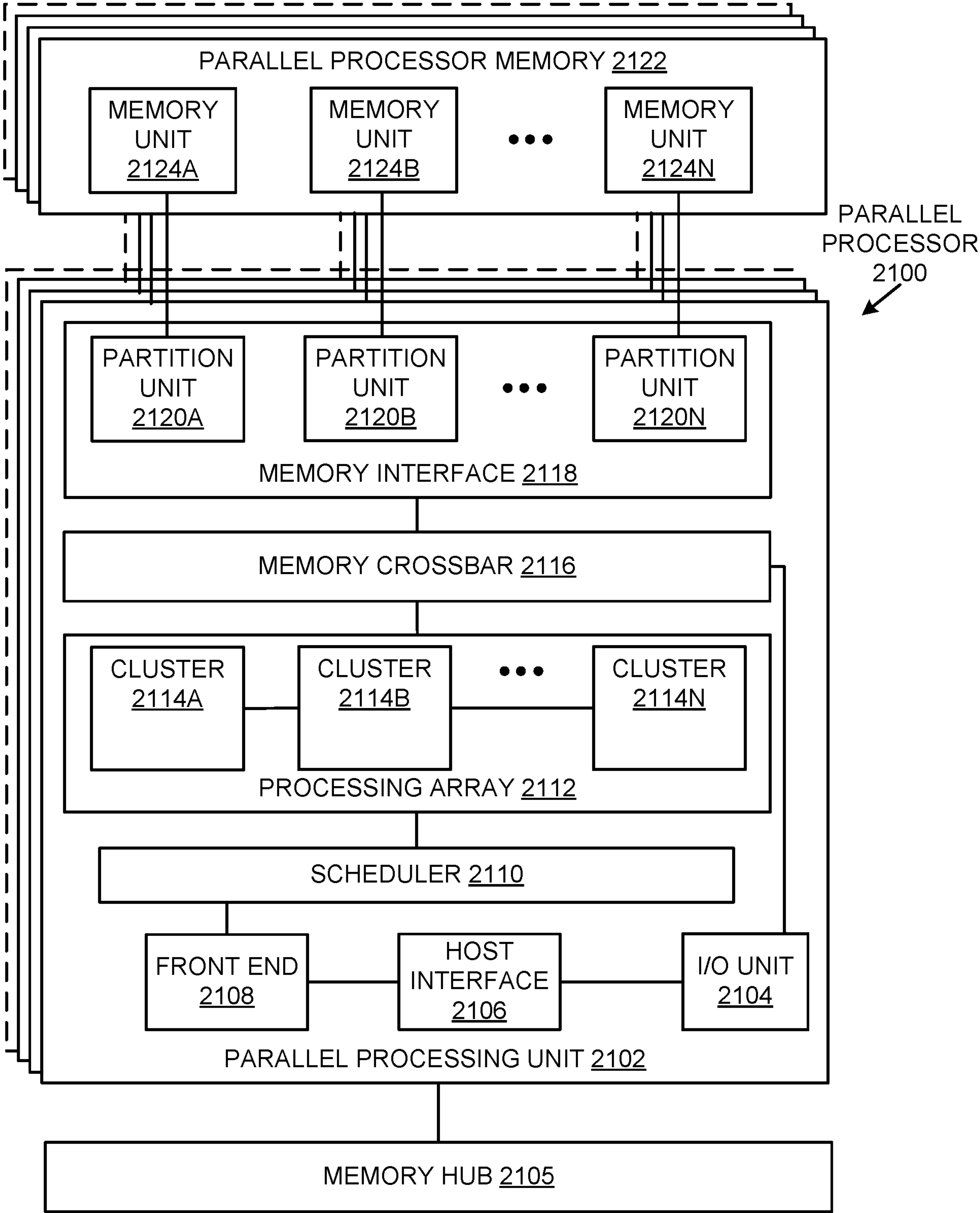


FIG. 21A

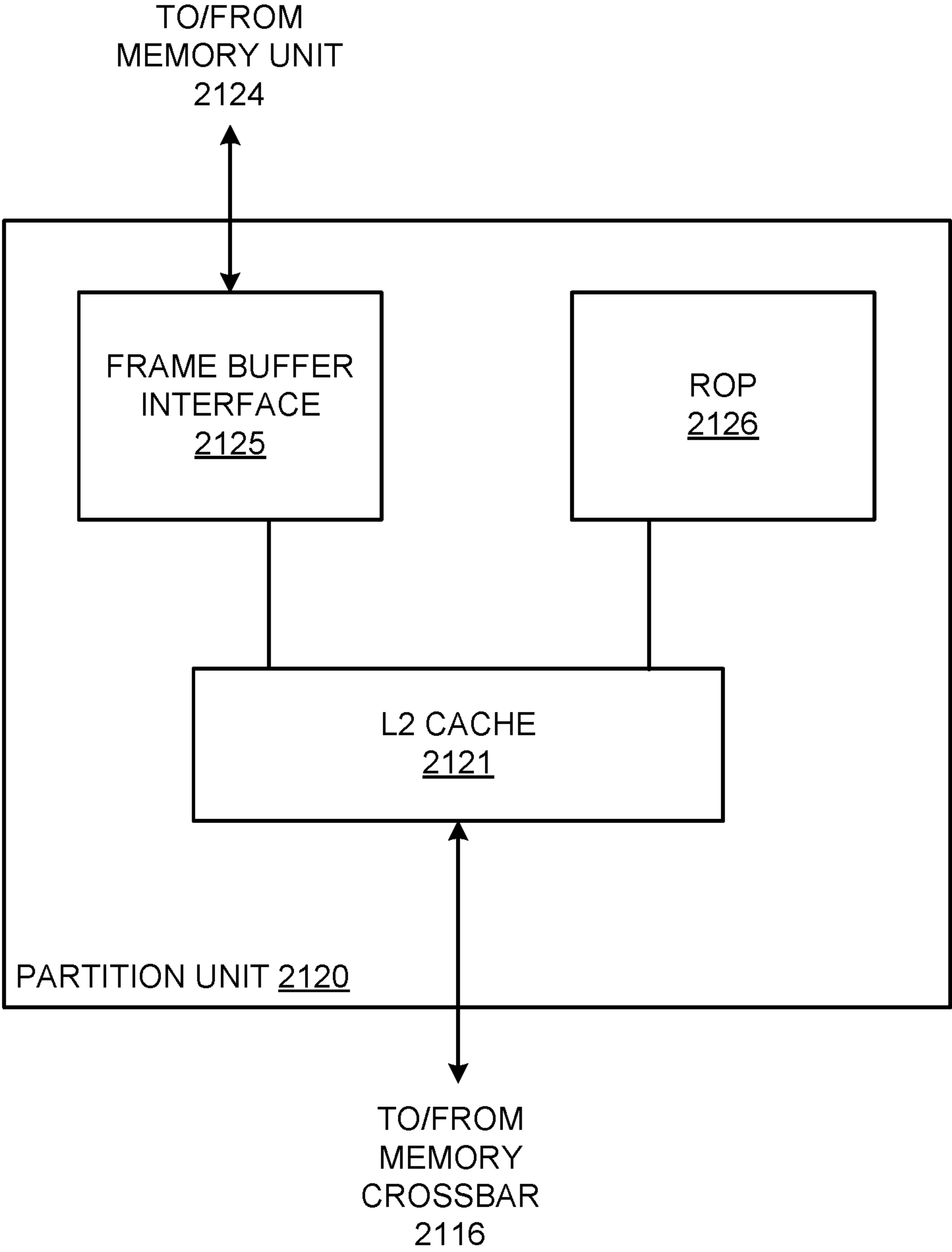


FIG. 21B

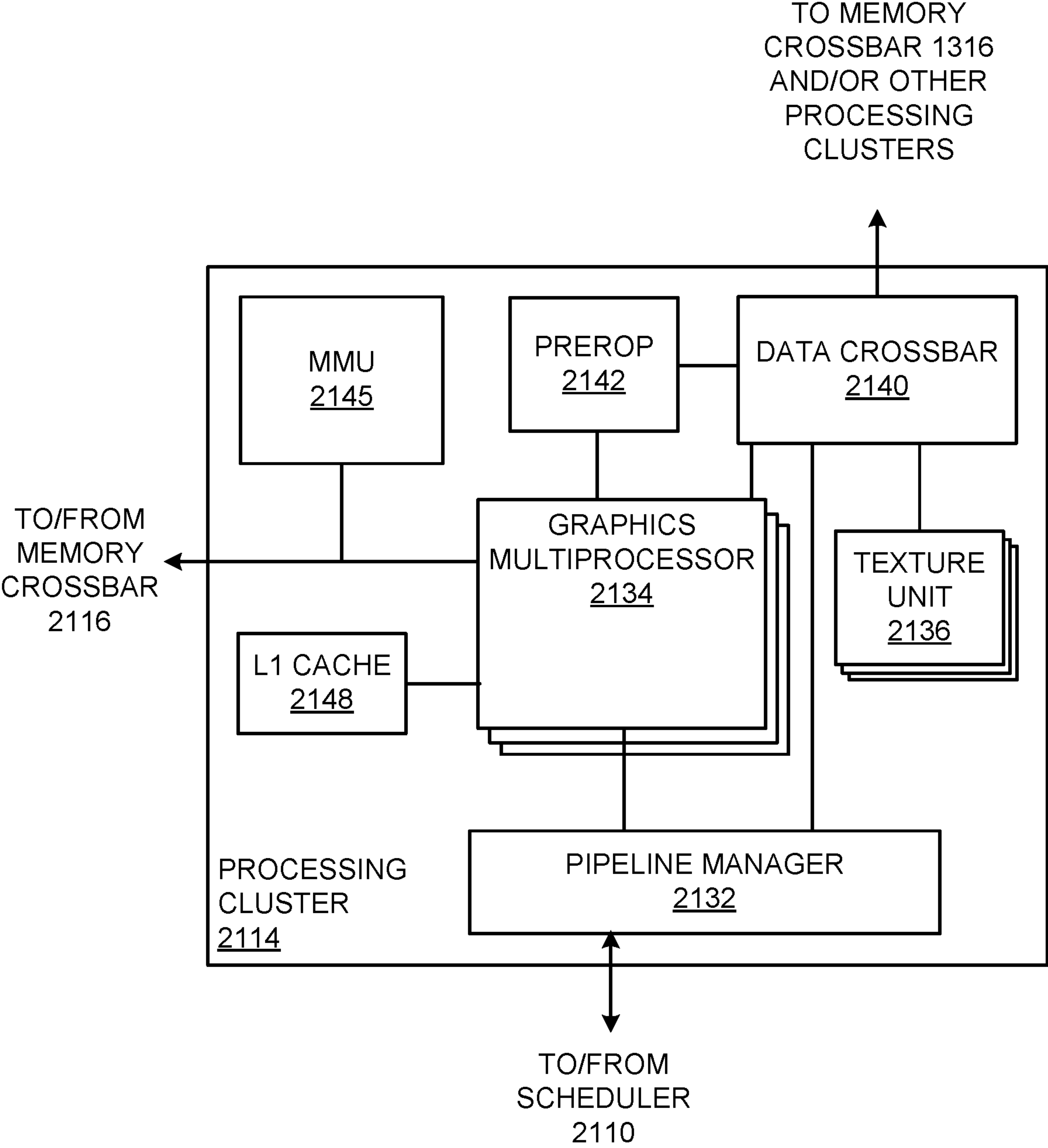


FIG. 21C

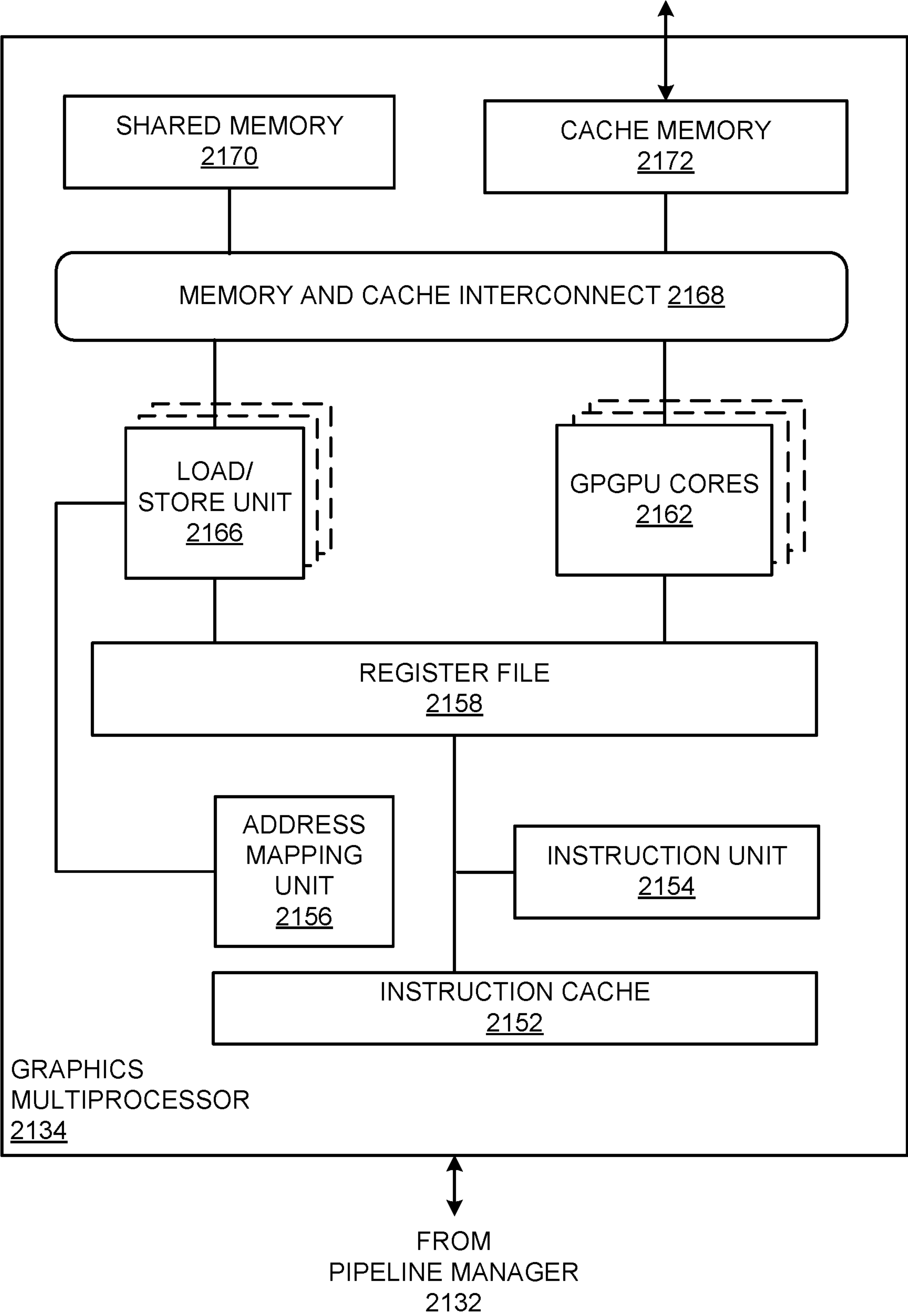


FIG. 21D

2200

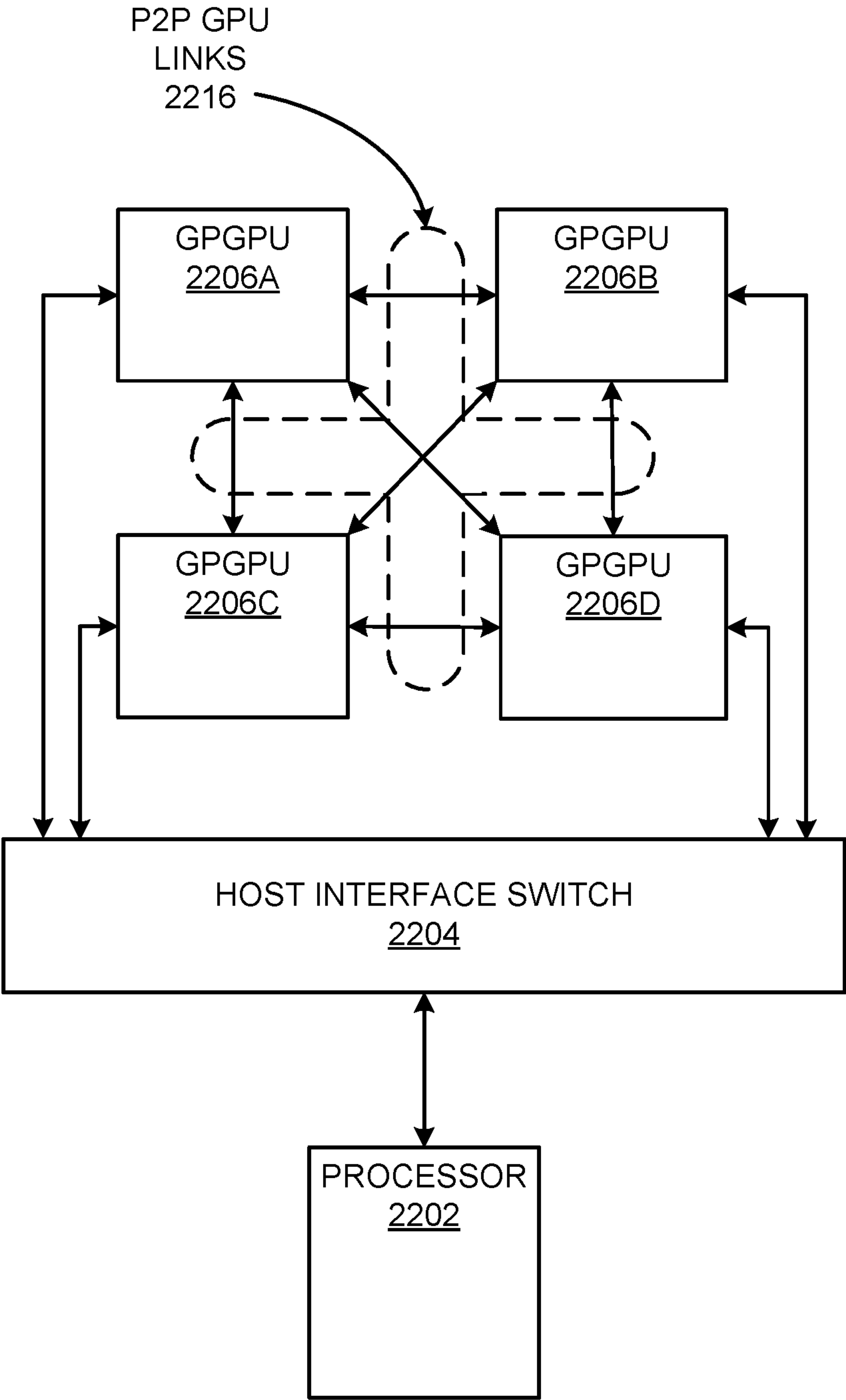


FIG. 22

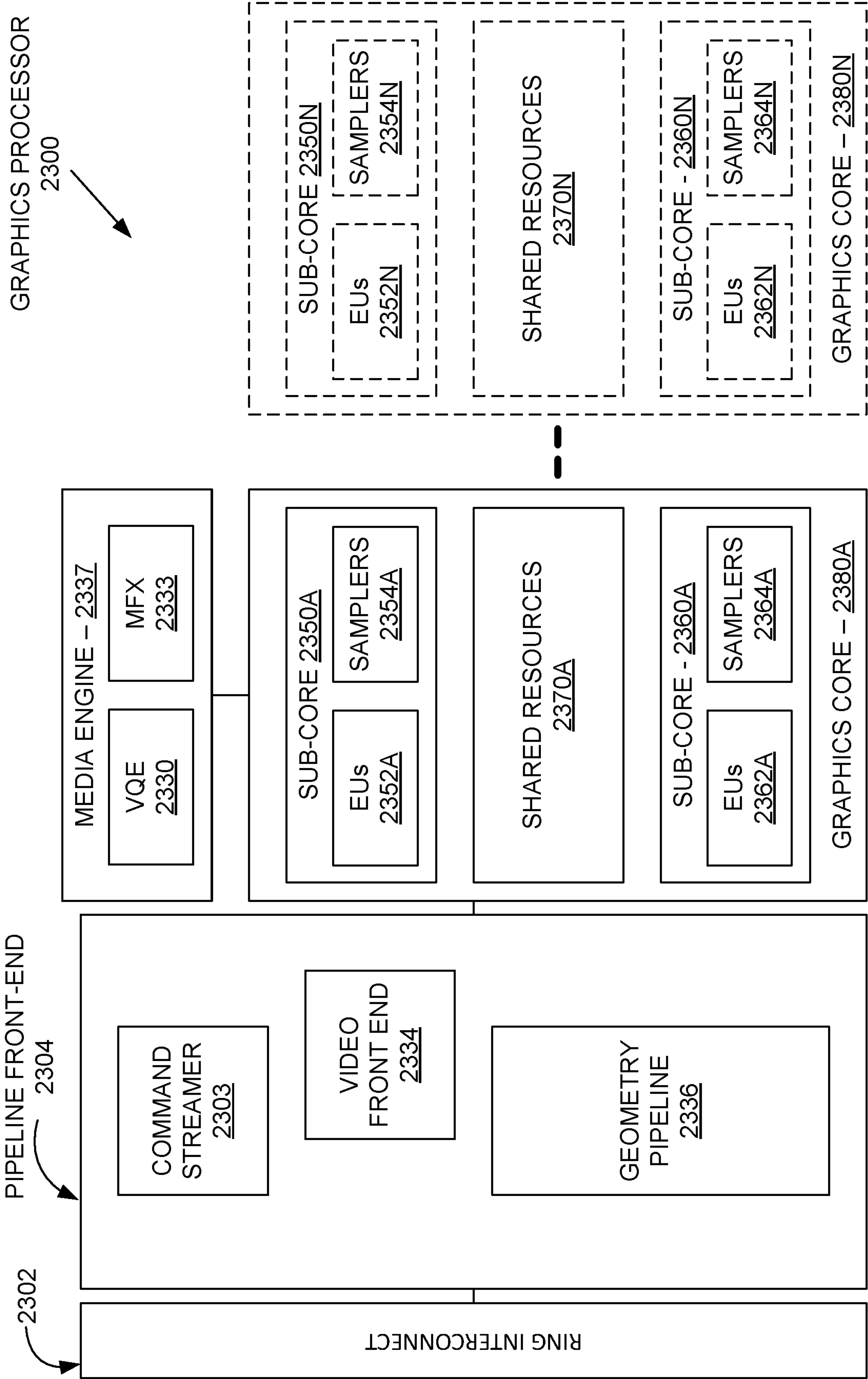


FIG. 23

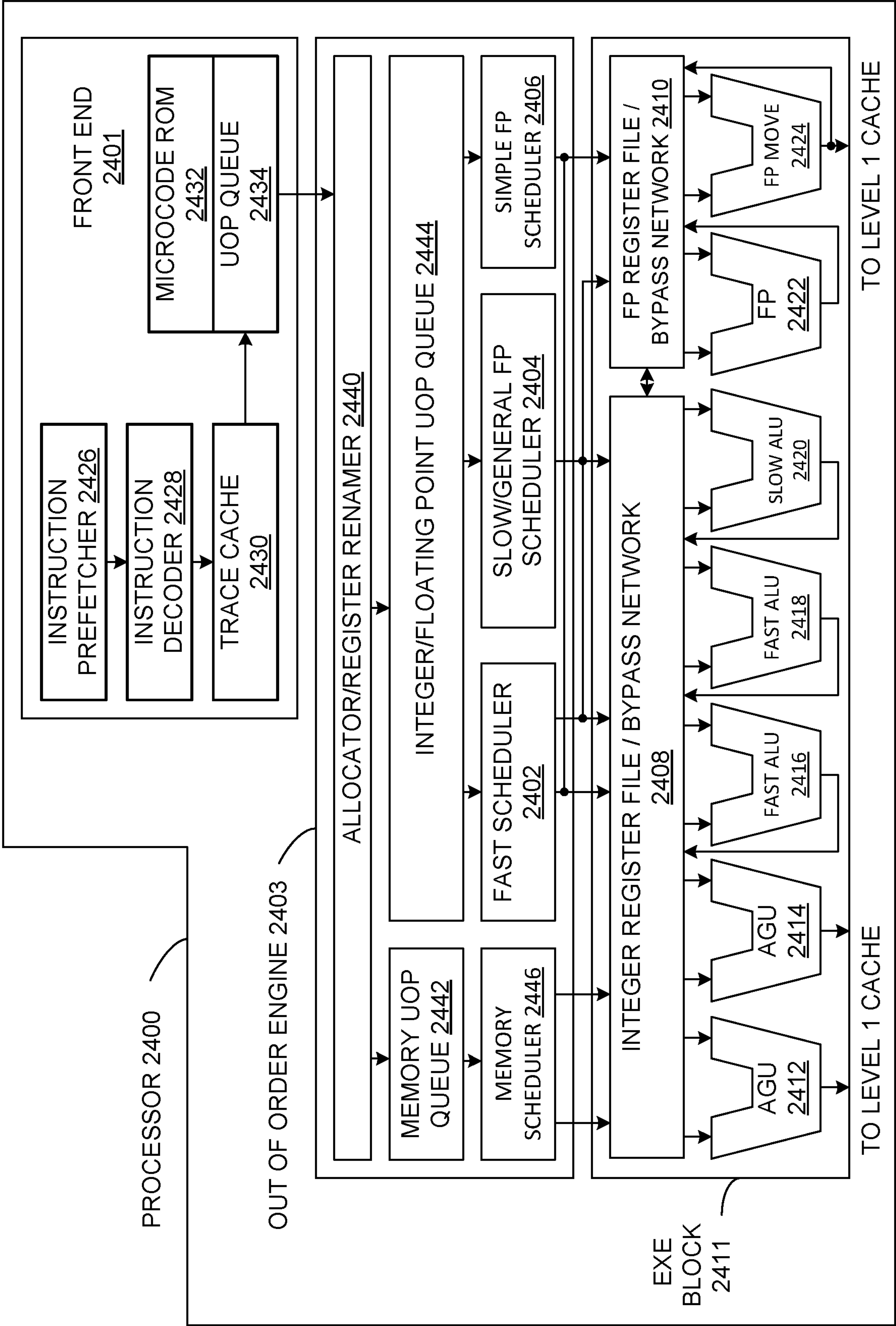


FIG. 24

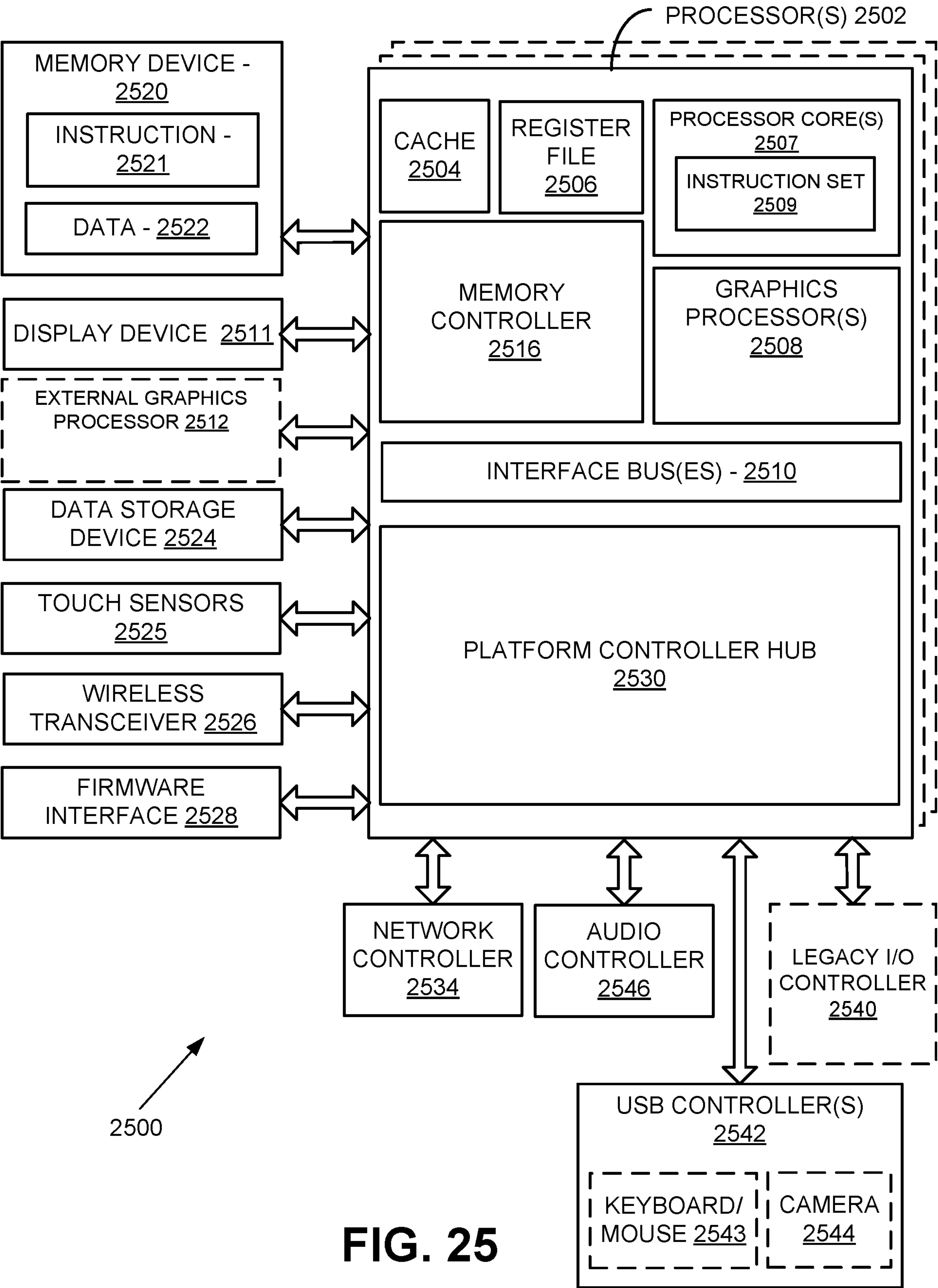


FIG. 25

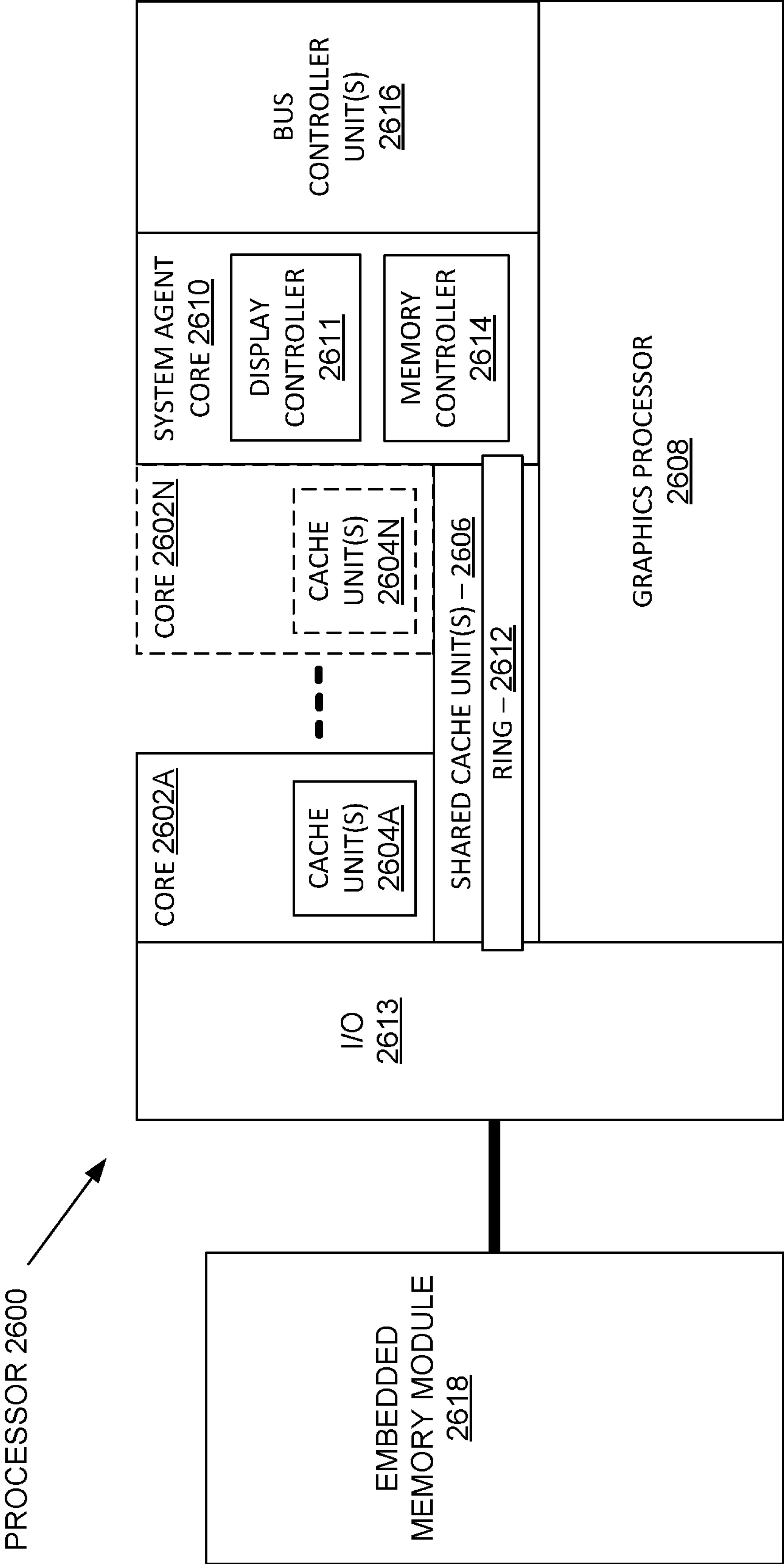


FIG. 26

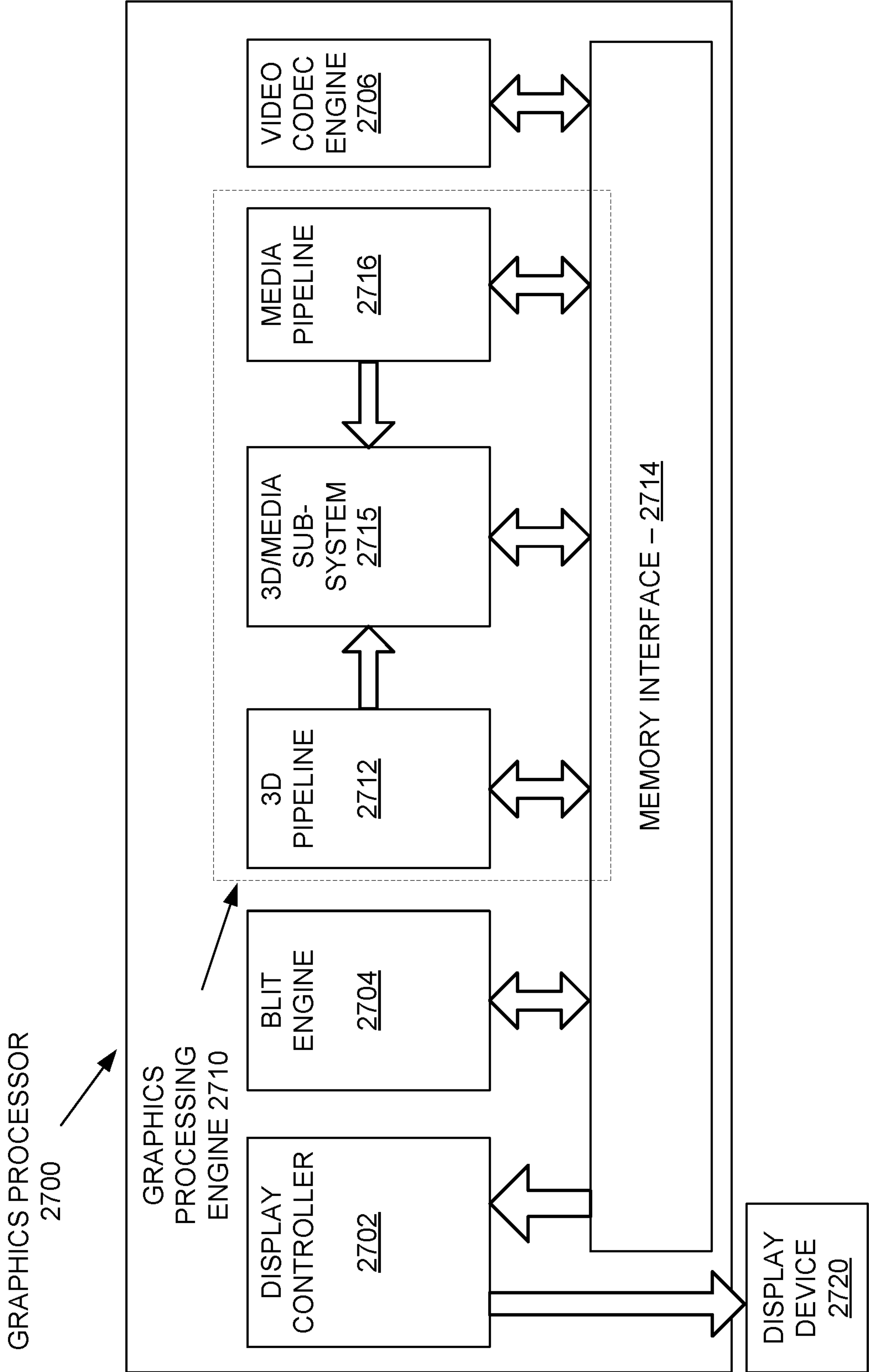


FIG. 27

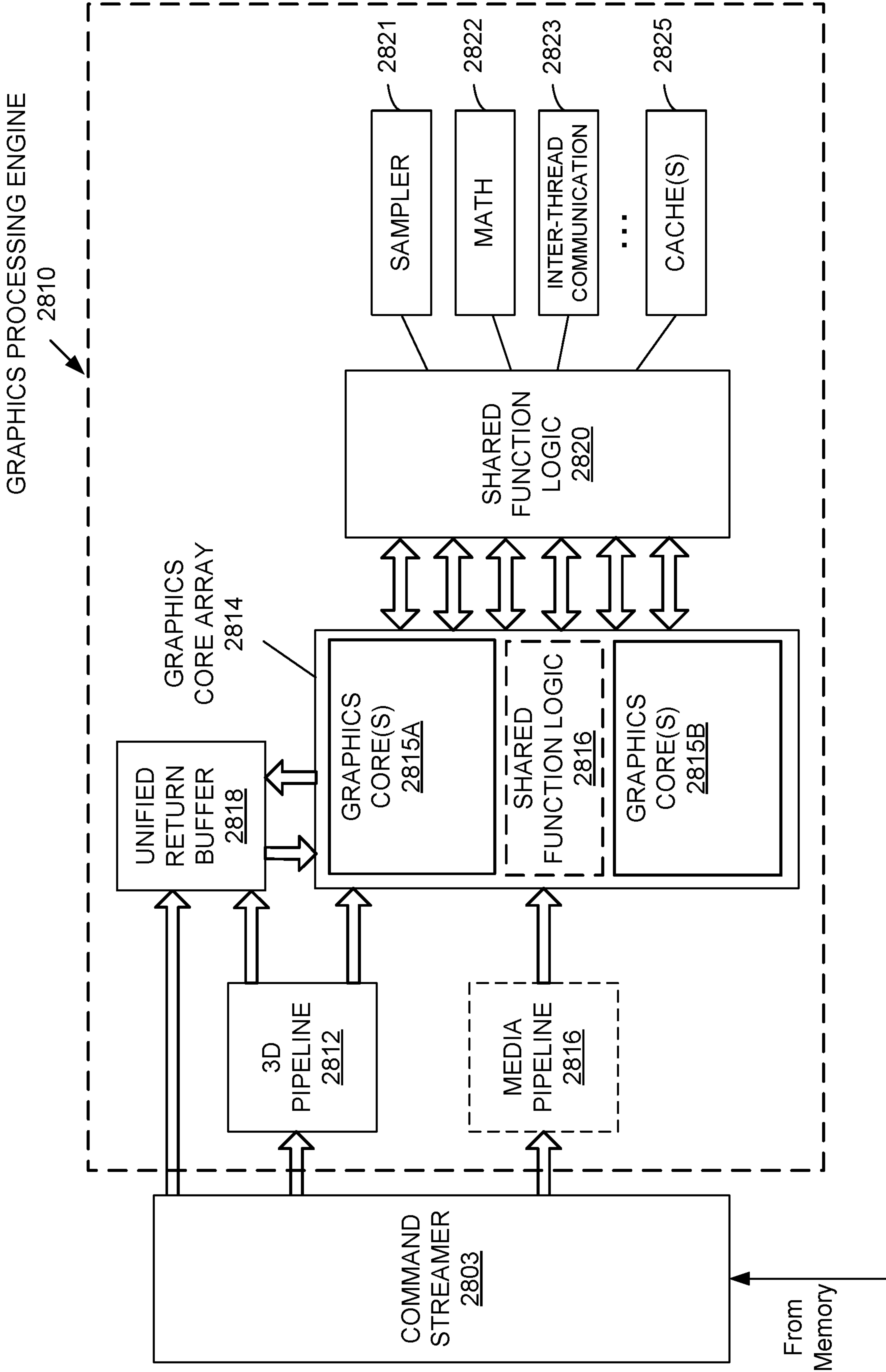
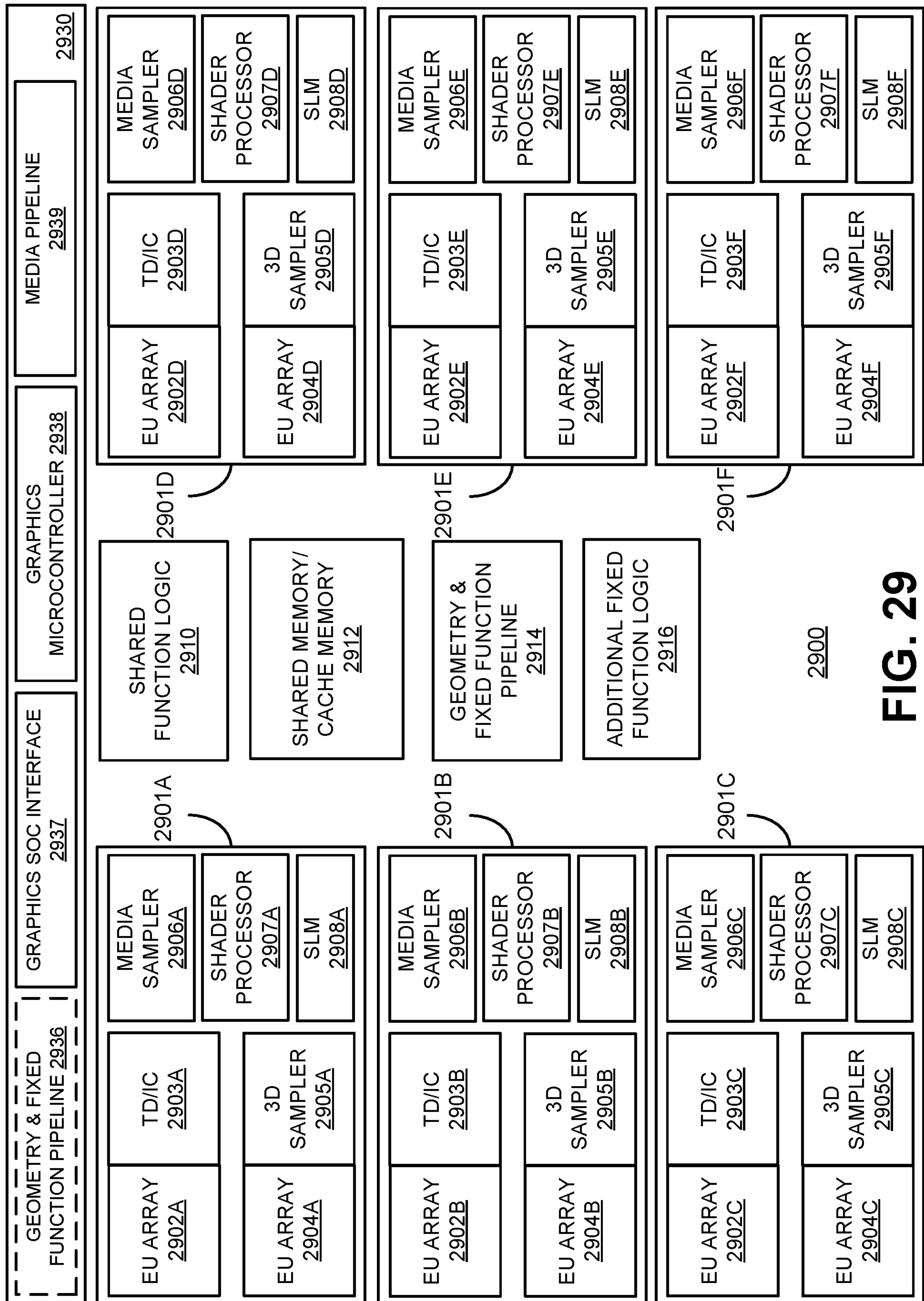


FIG. 28



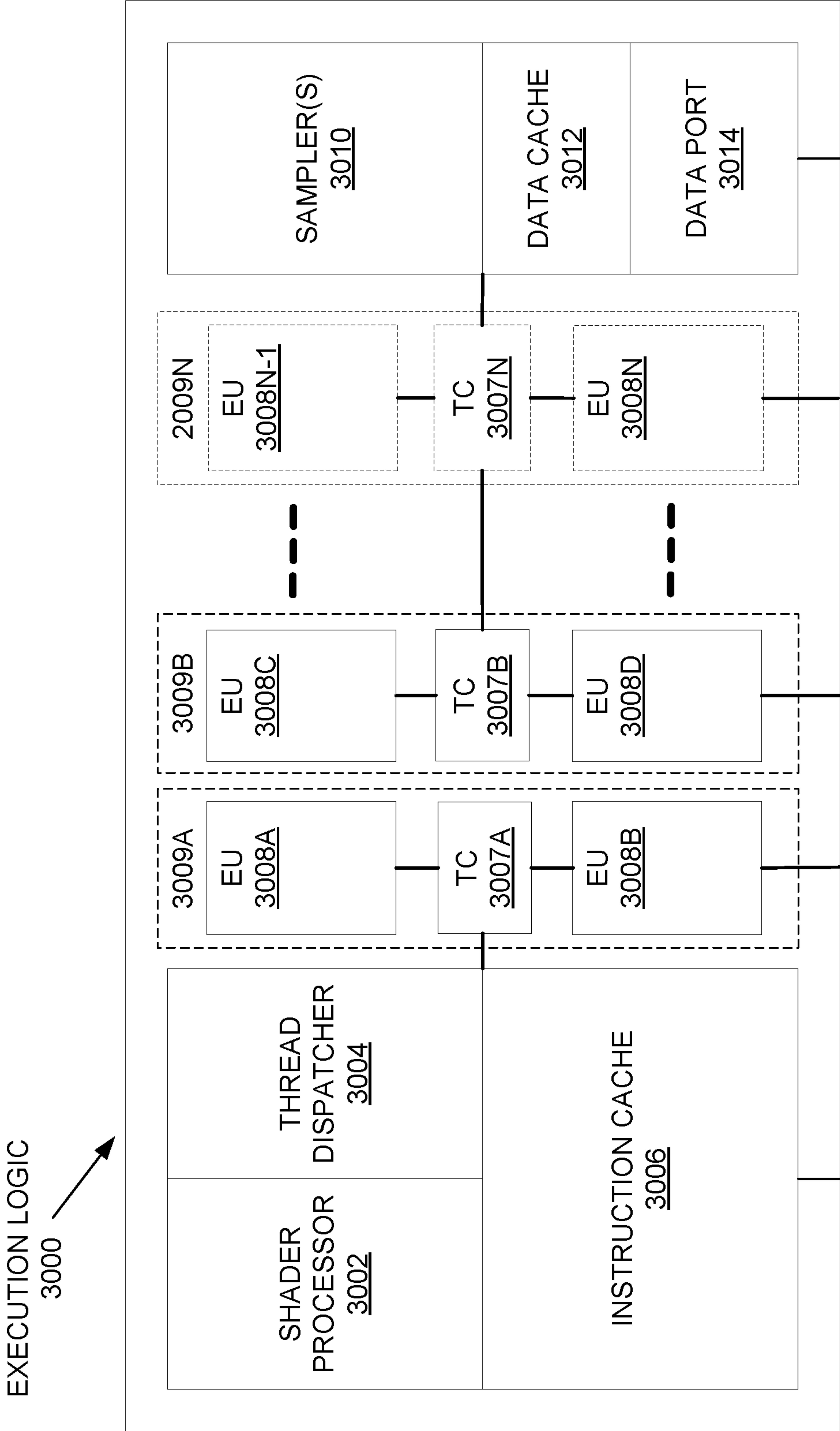


FIG. 30A

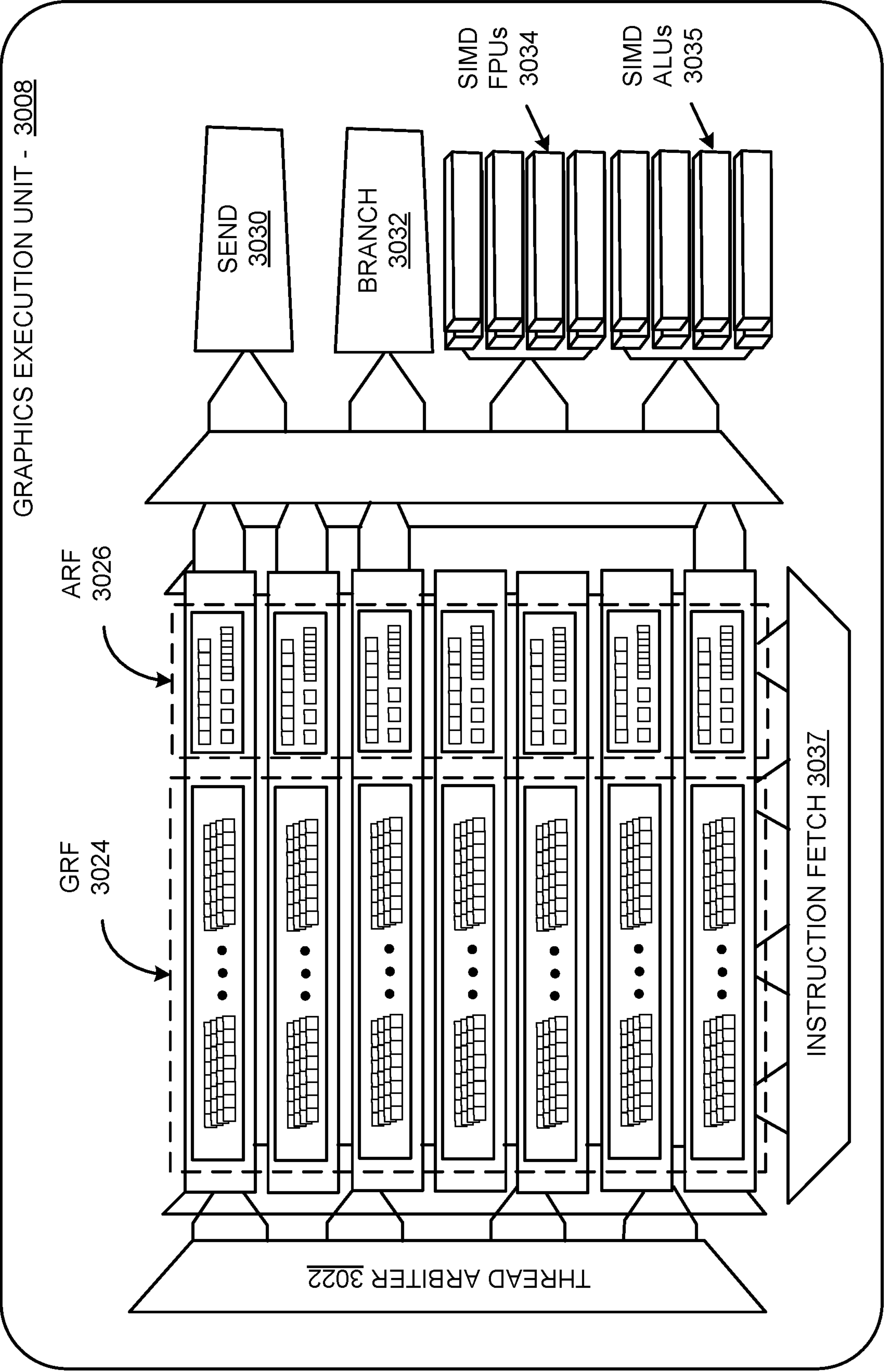


FIG. 30B

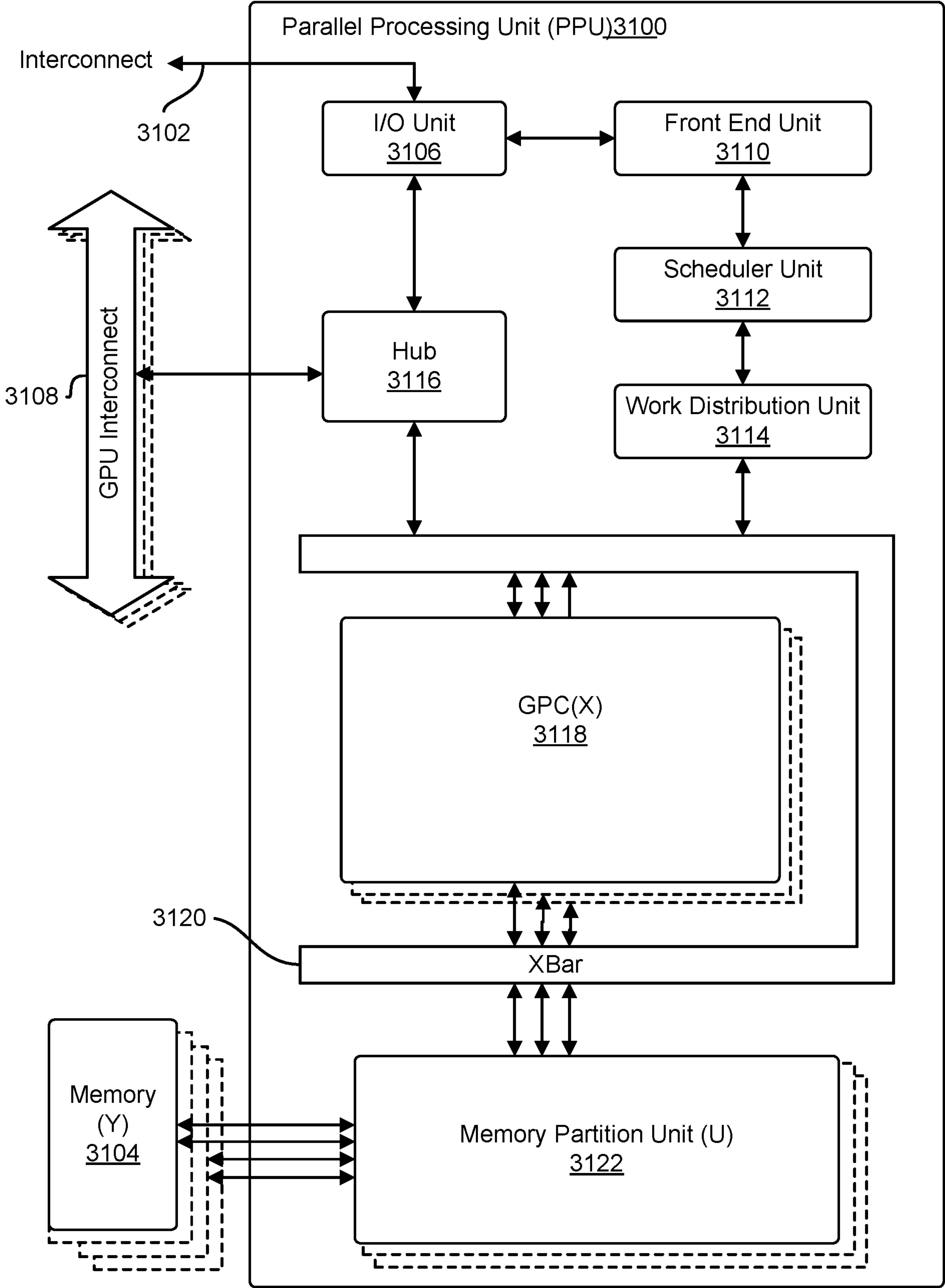


FIG. 31

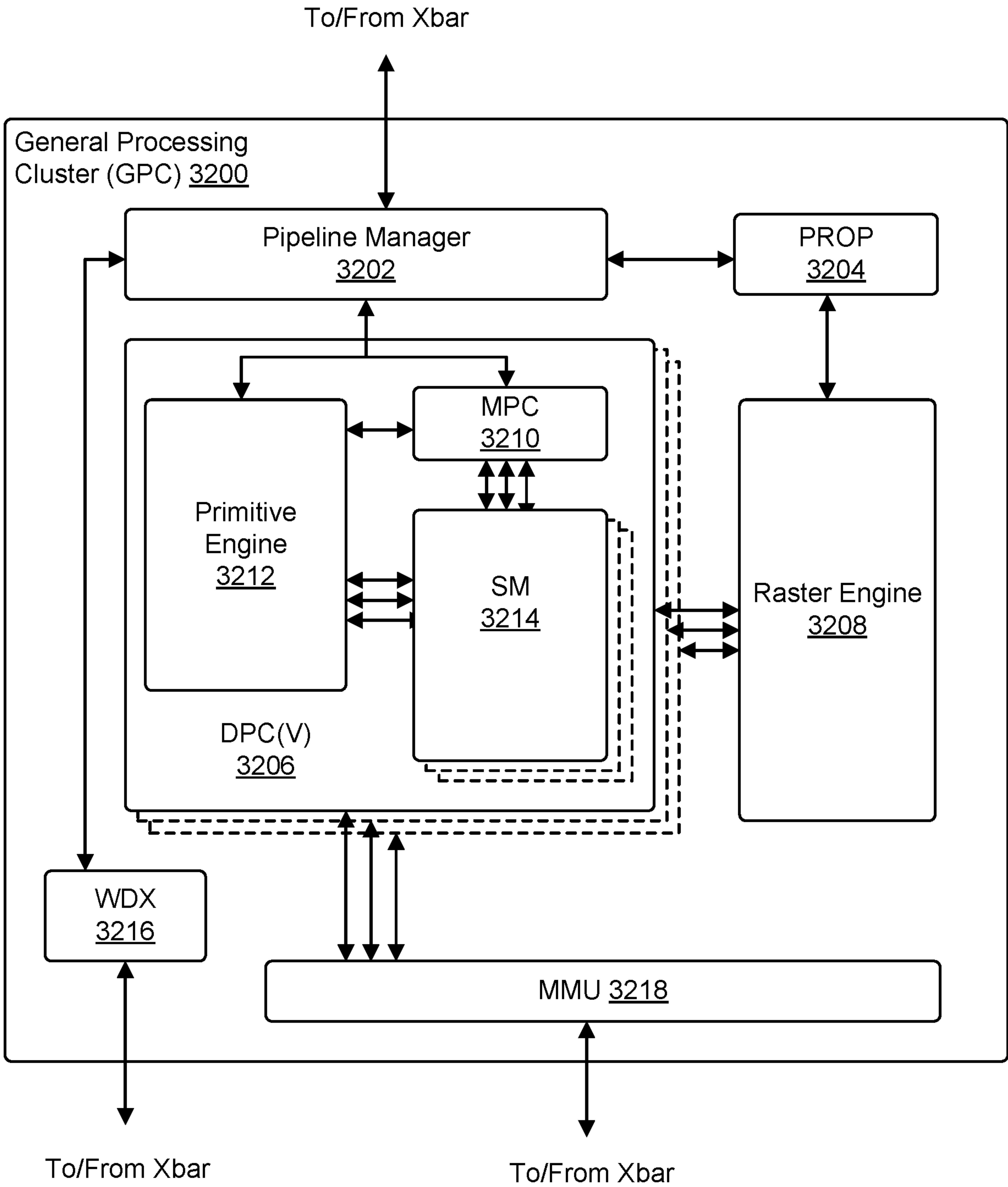


FIG. 32

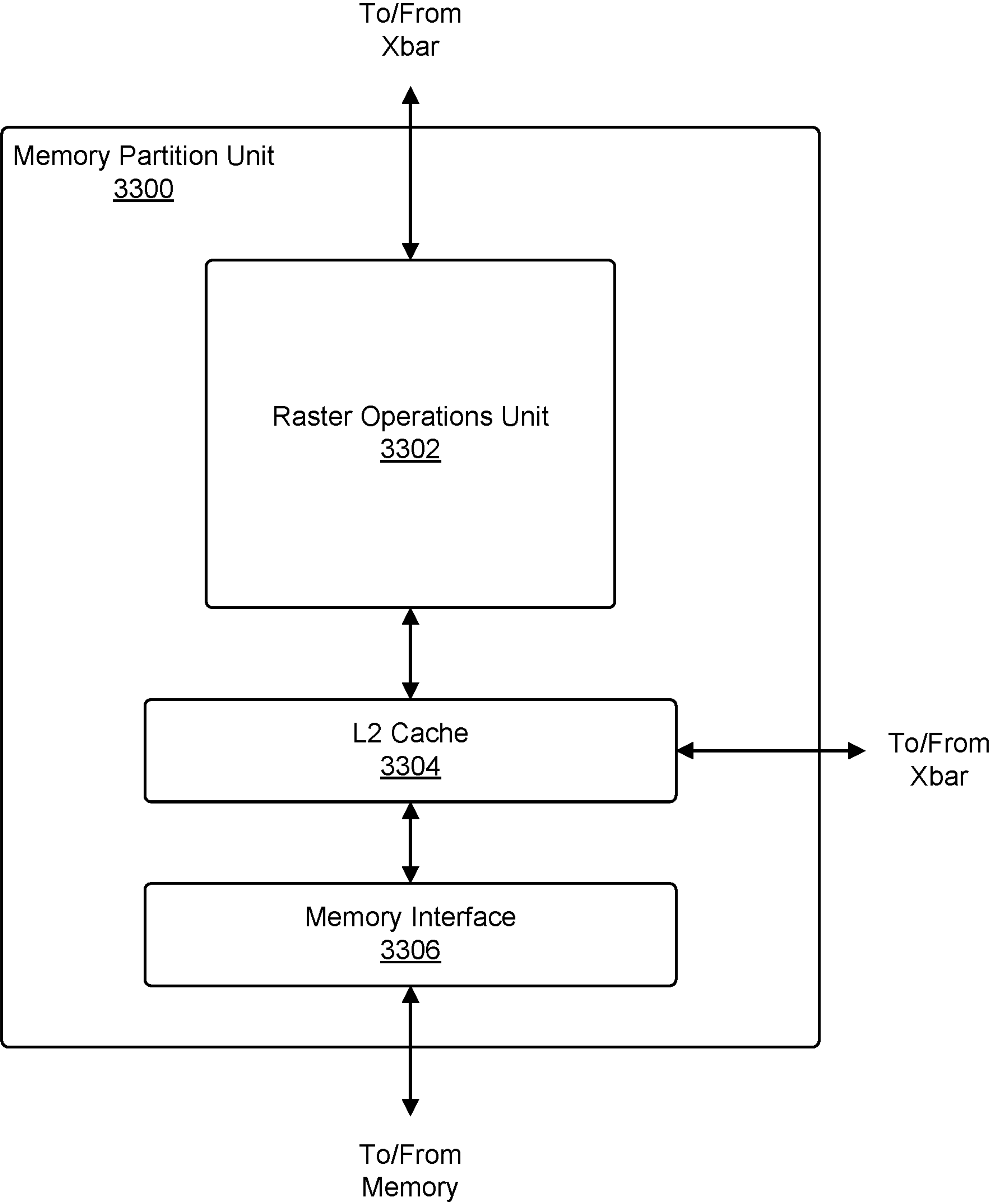


FIG. 33

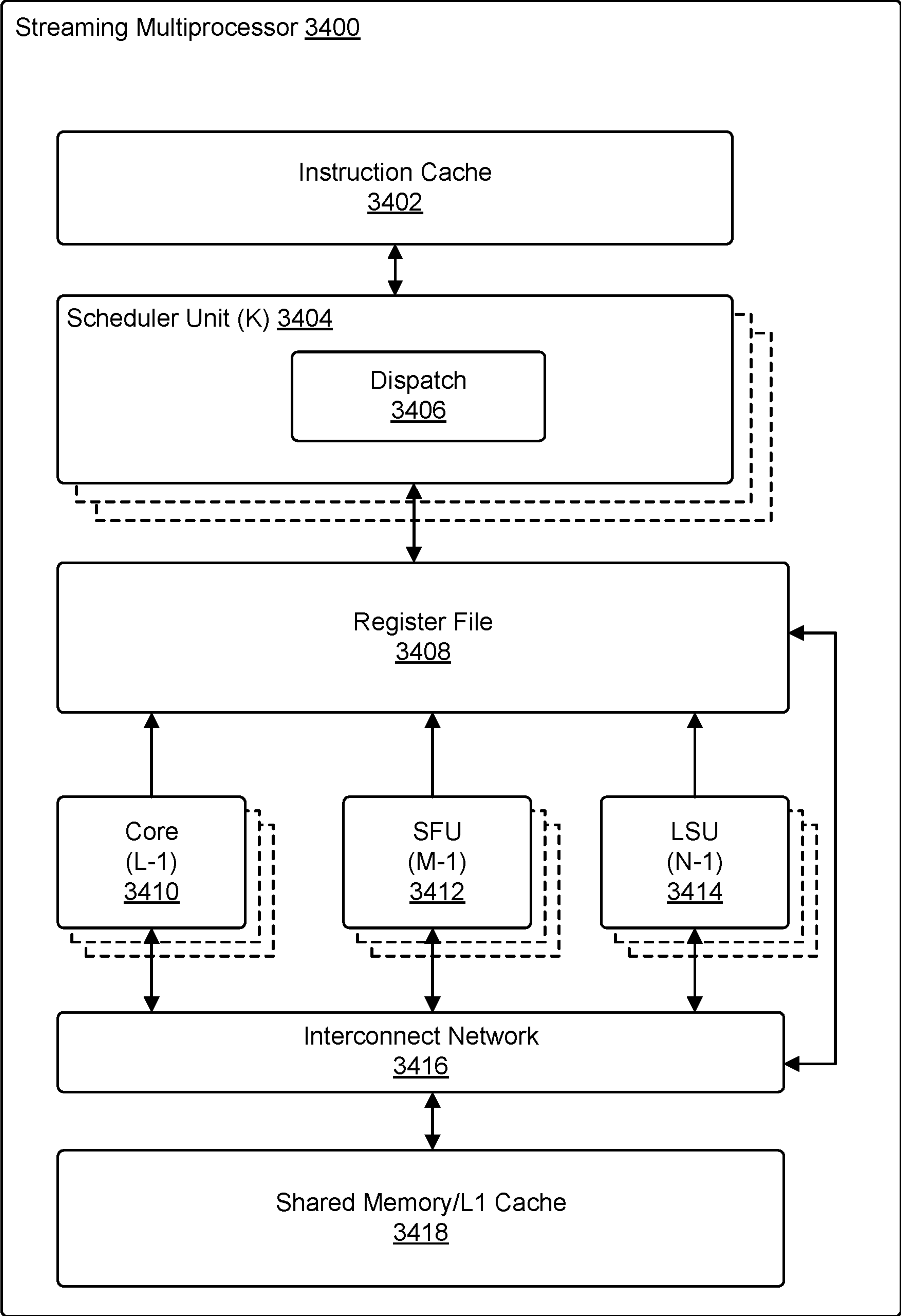


FIG. 34

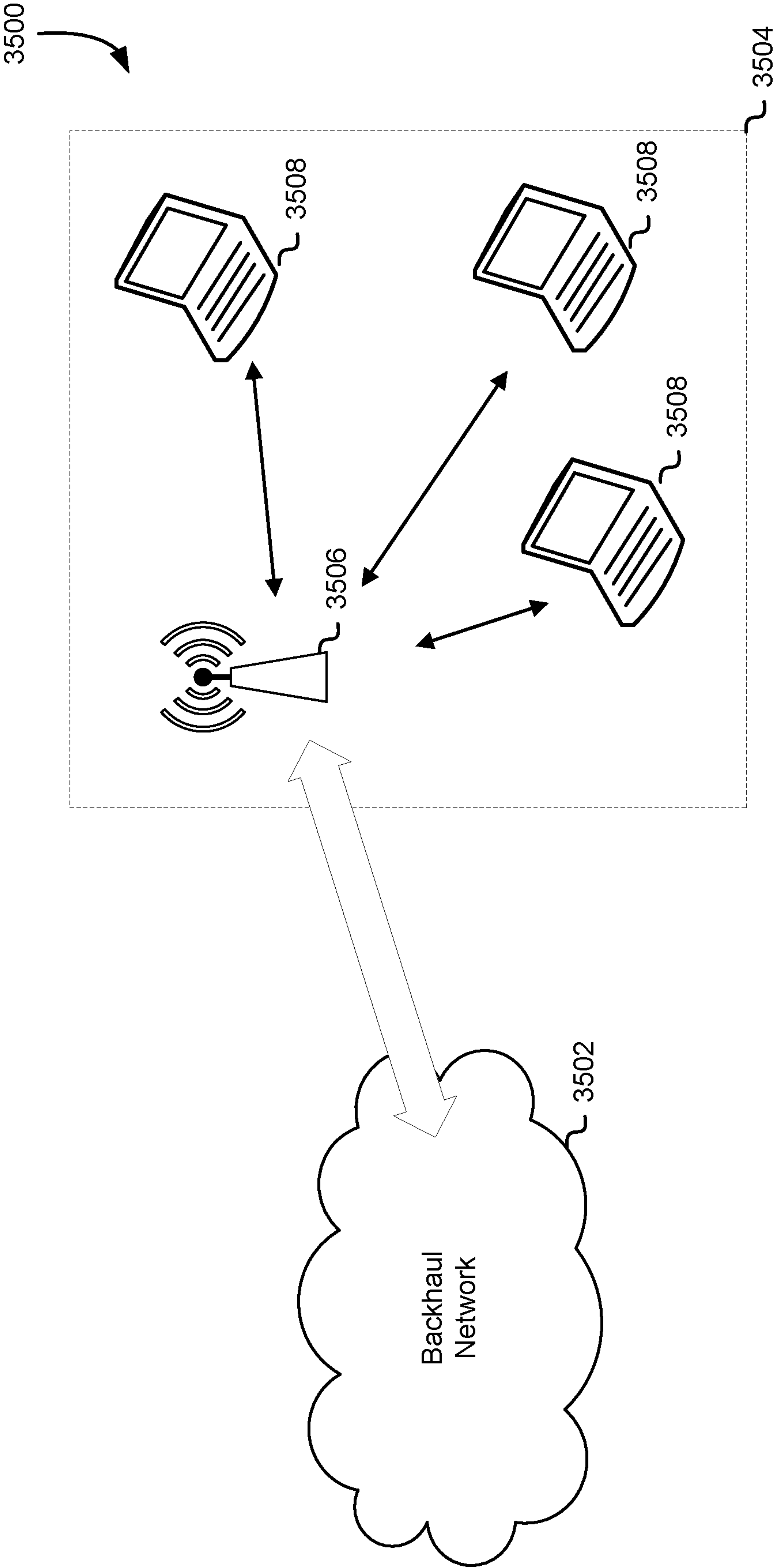


FIG. 35

3600

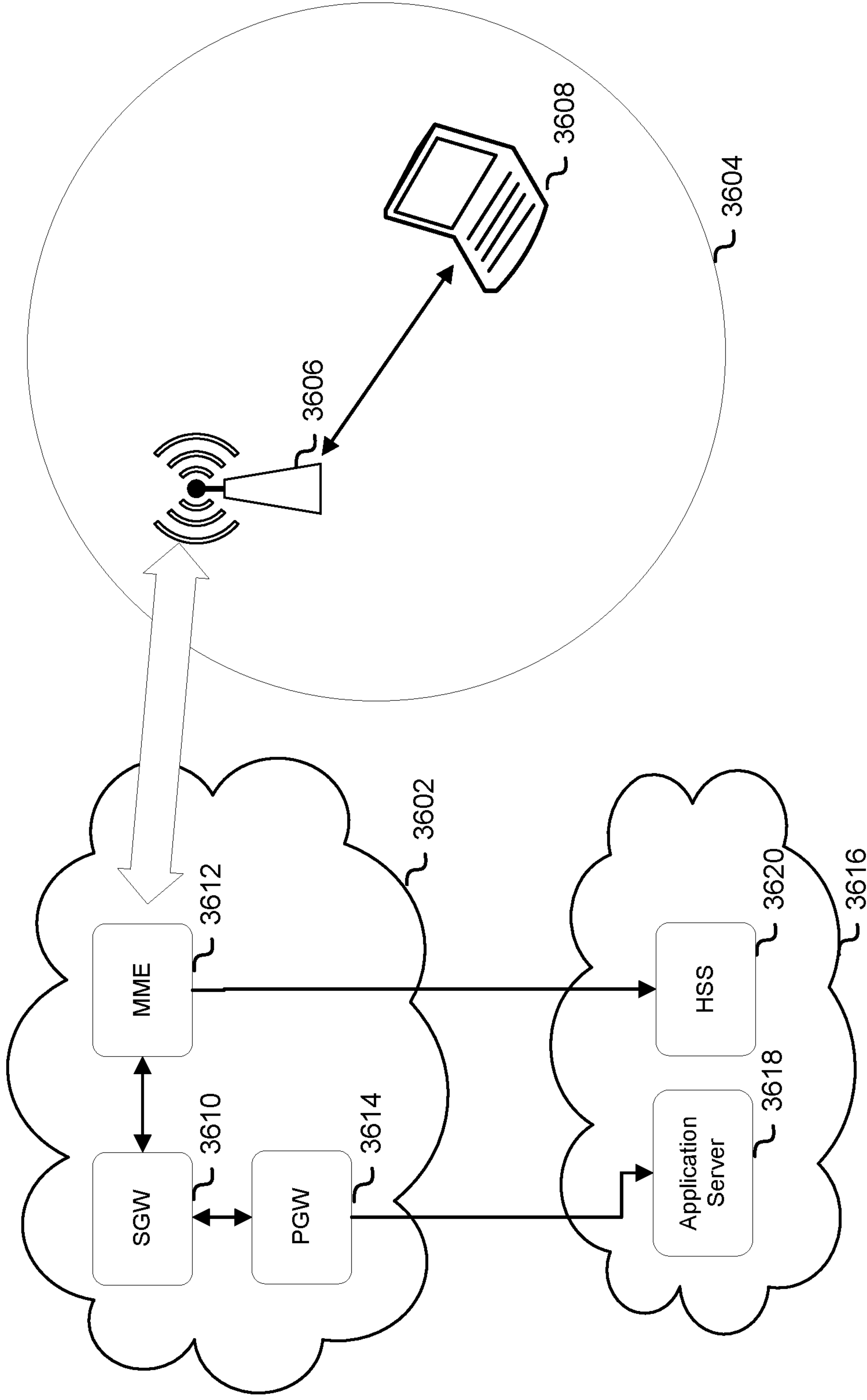


FIG. 36

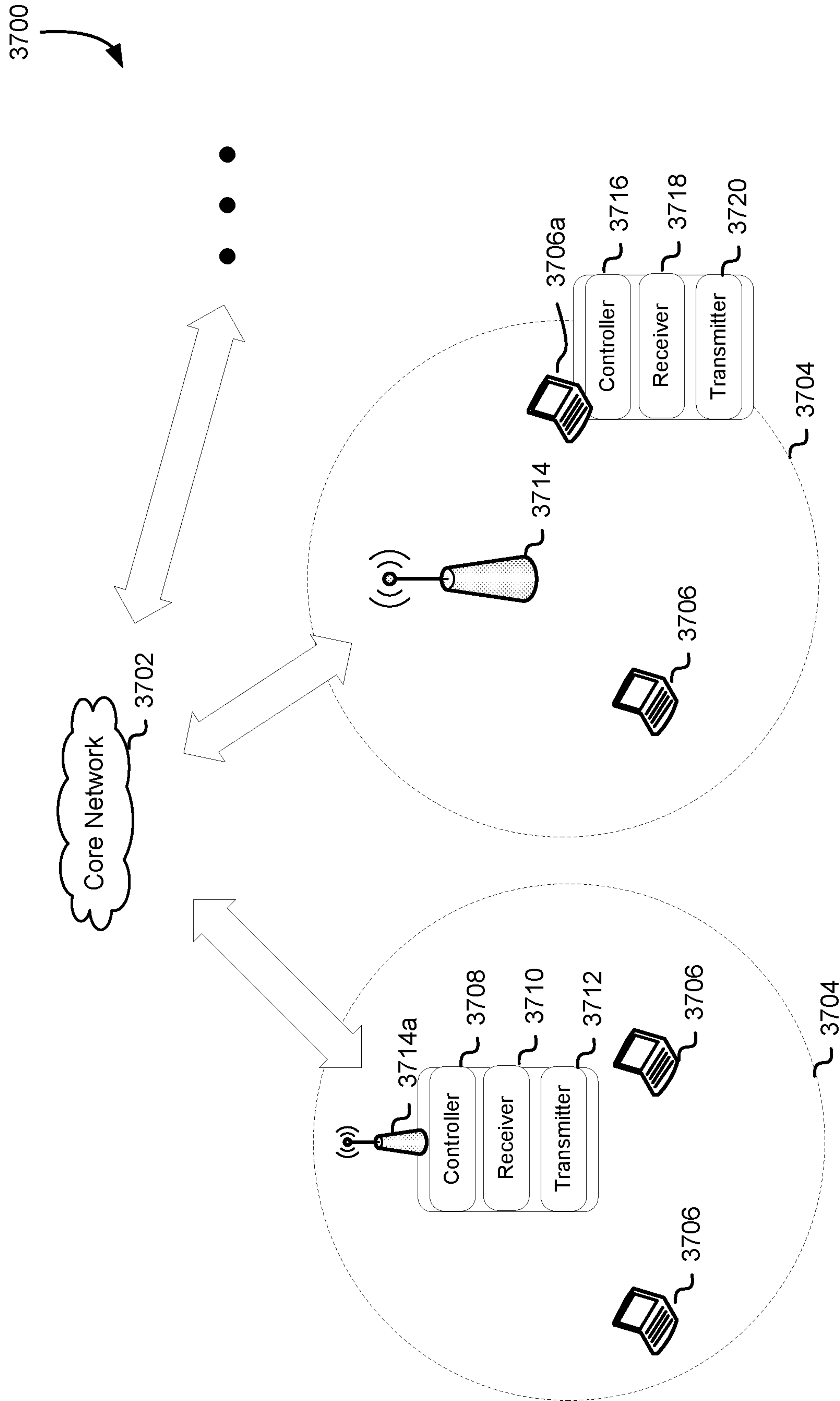


FIG. 37

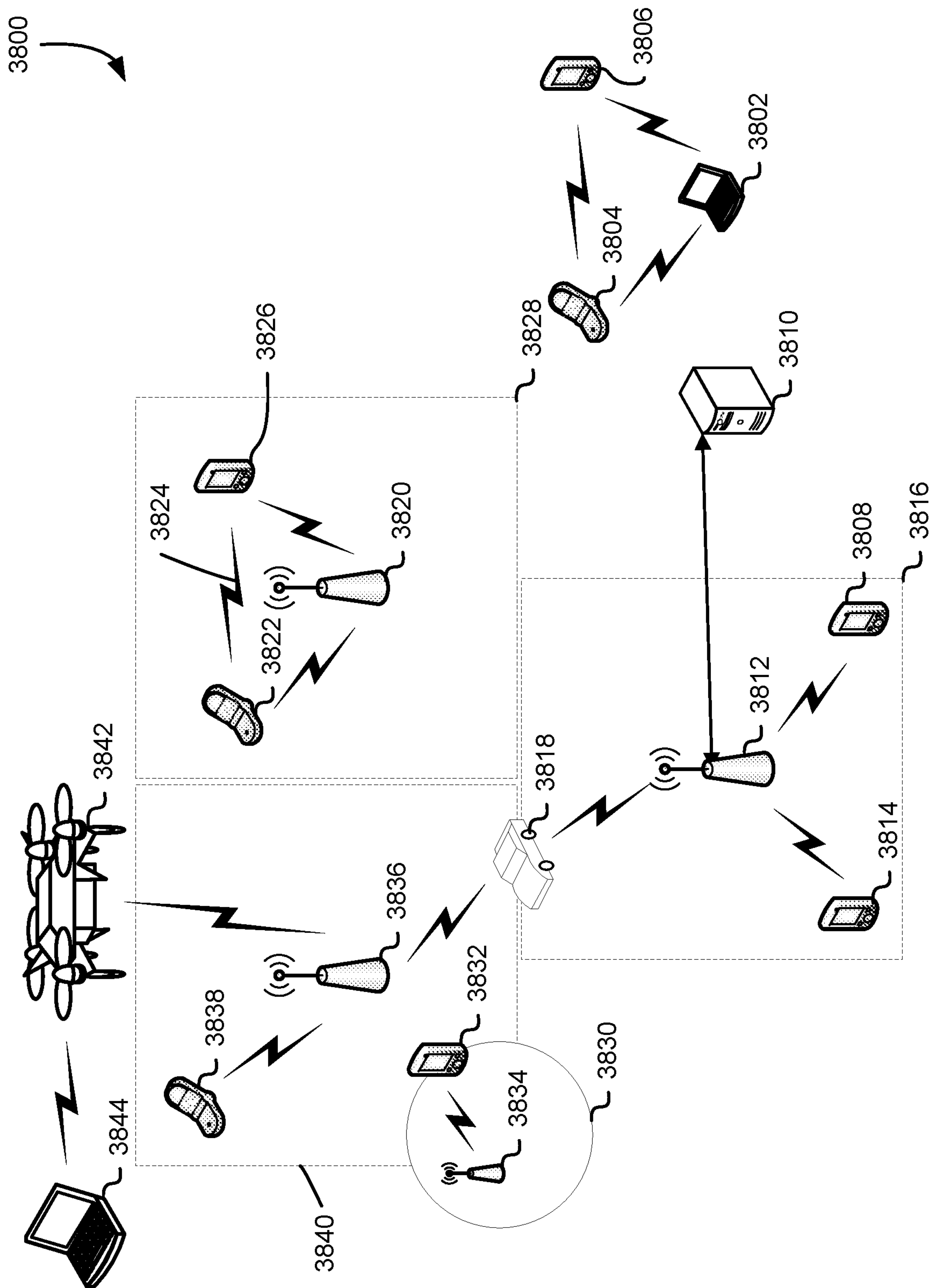


FIG. 38

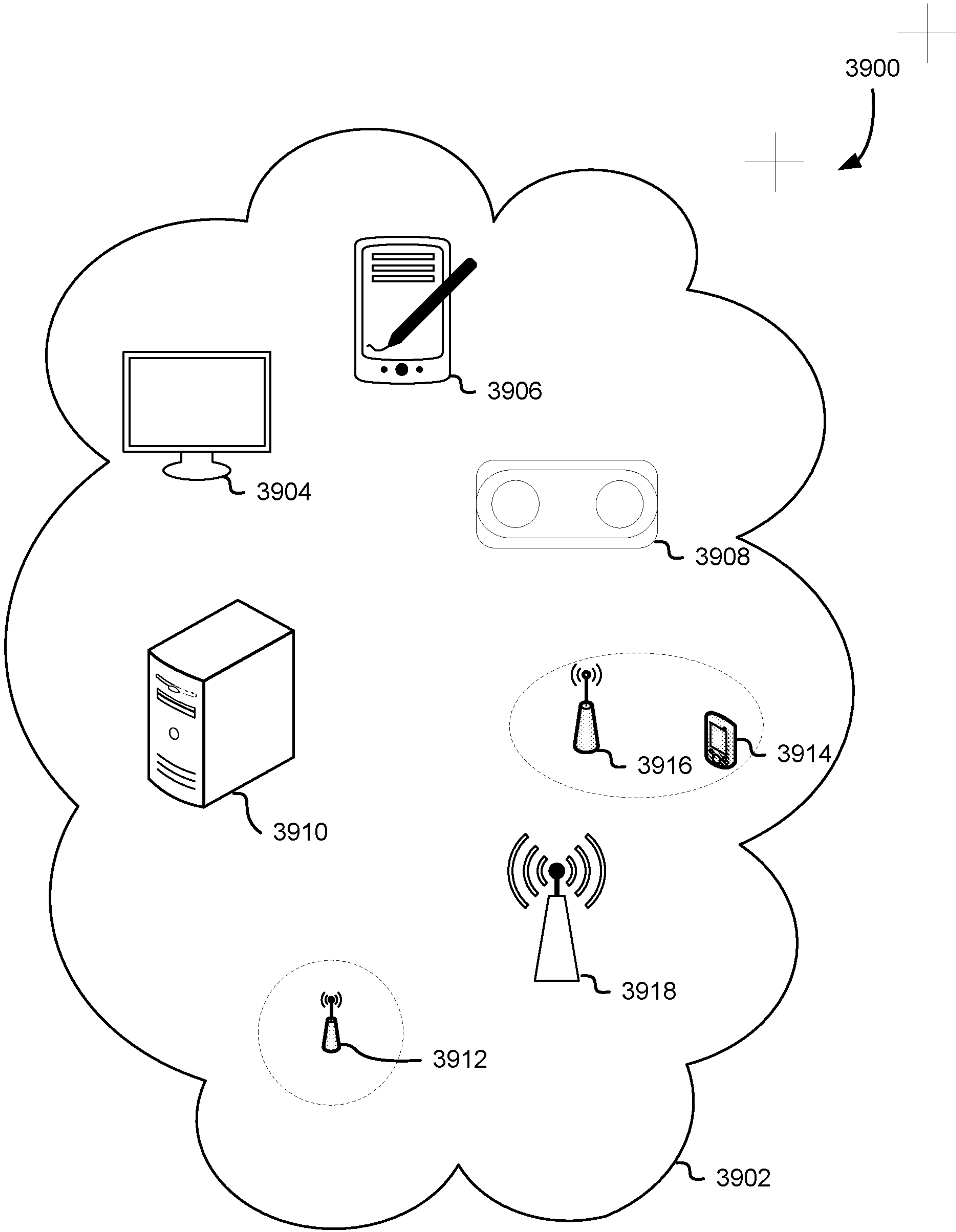


FIG. 39

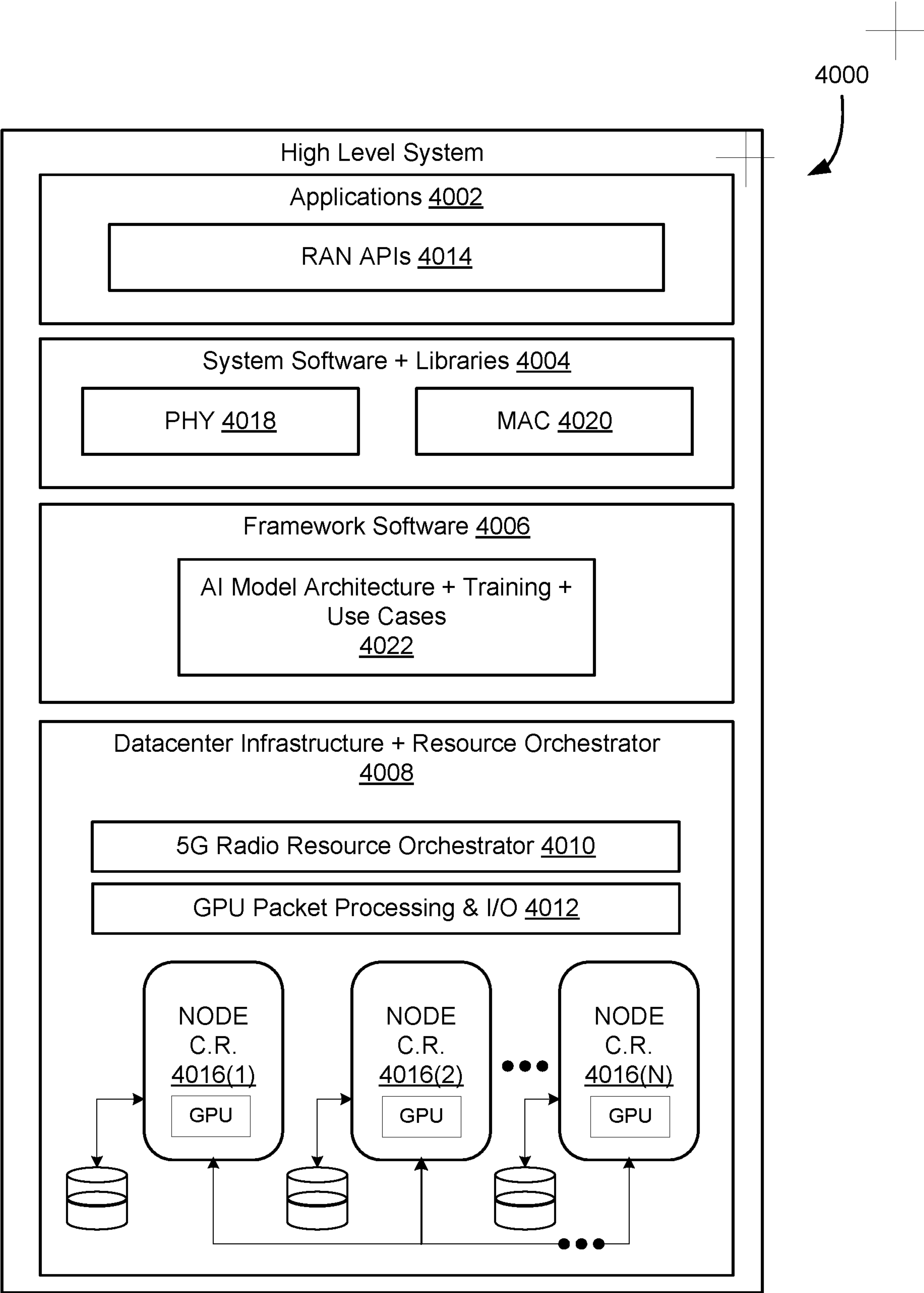


FIG. 40

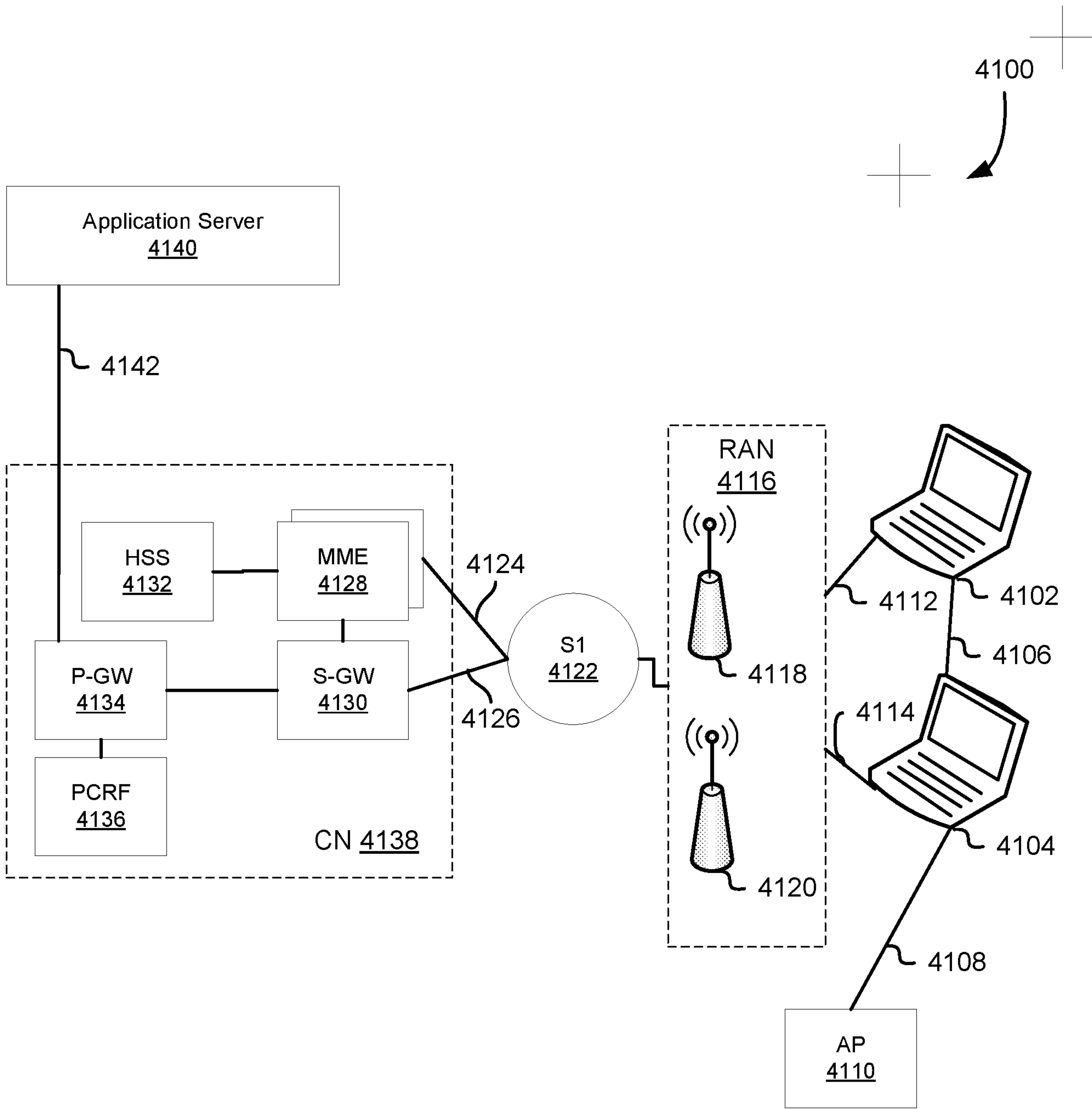


FIG. 41

4200

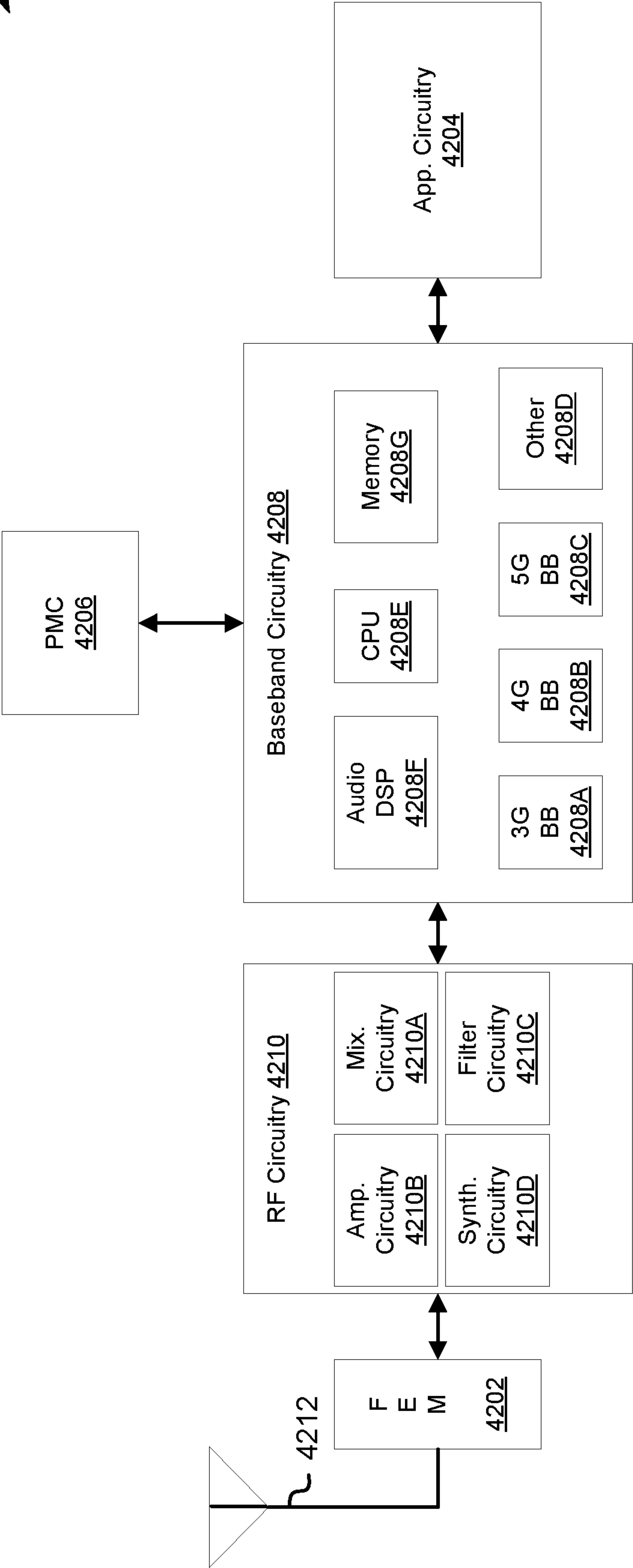


FIG. 42

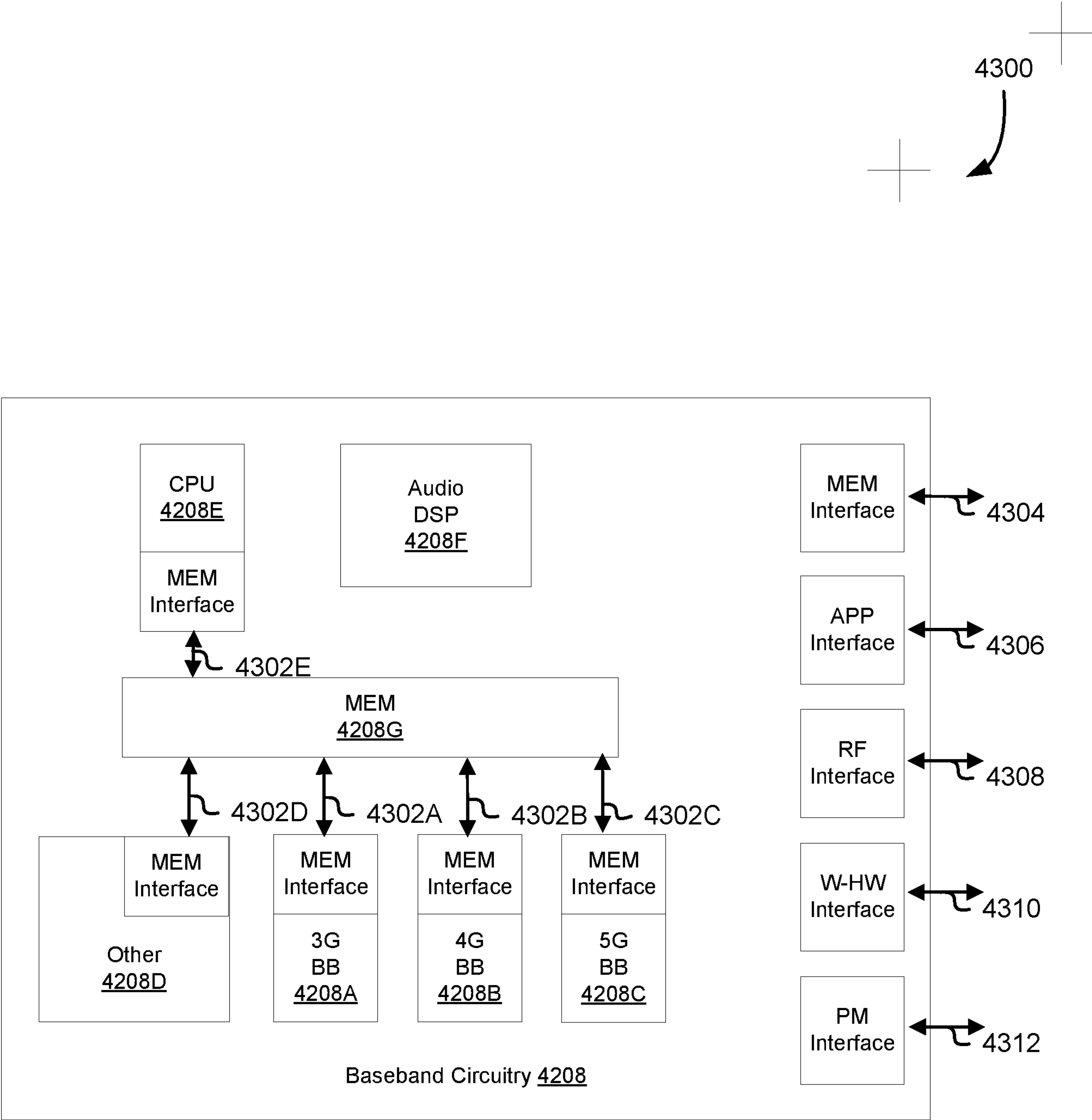


FIG. 43

4400

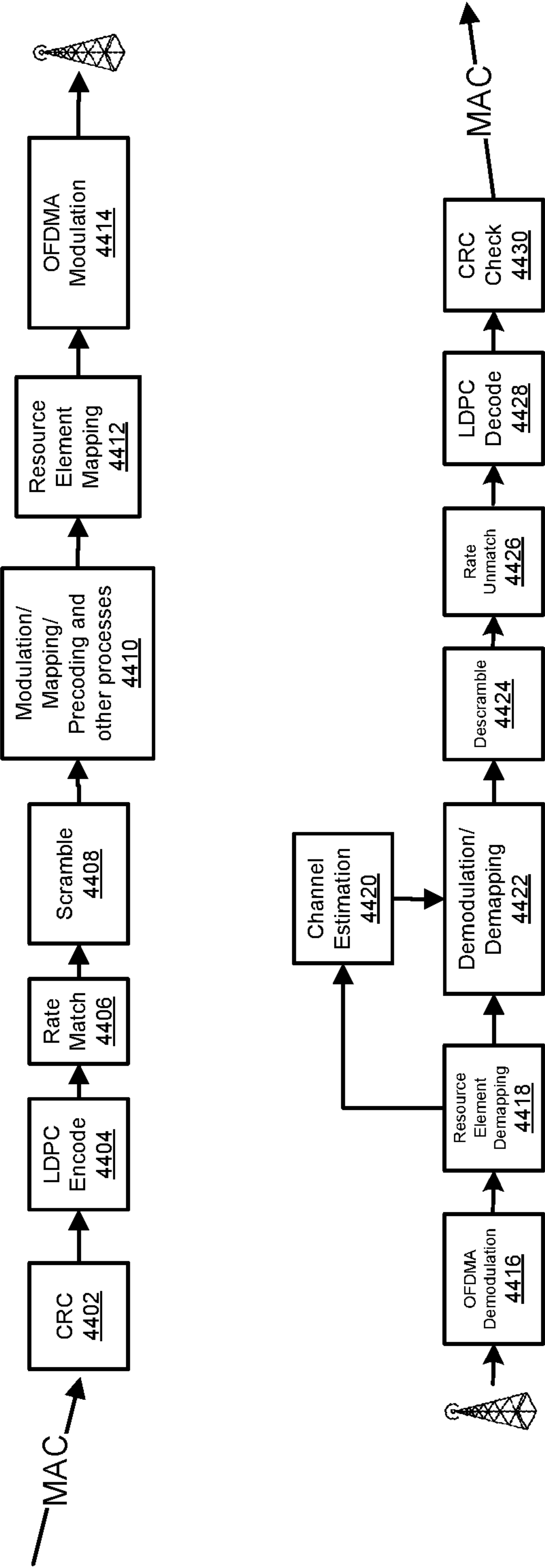


FIG. 44

4500

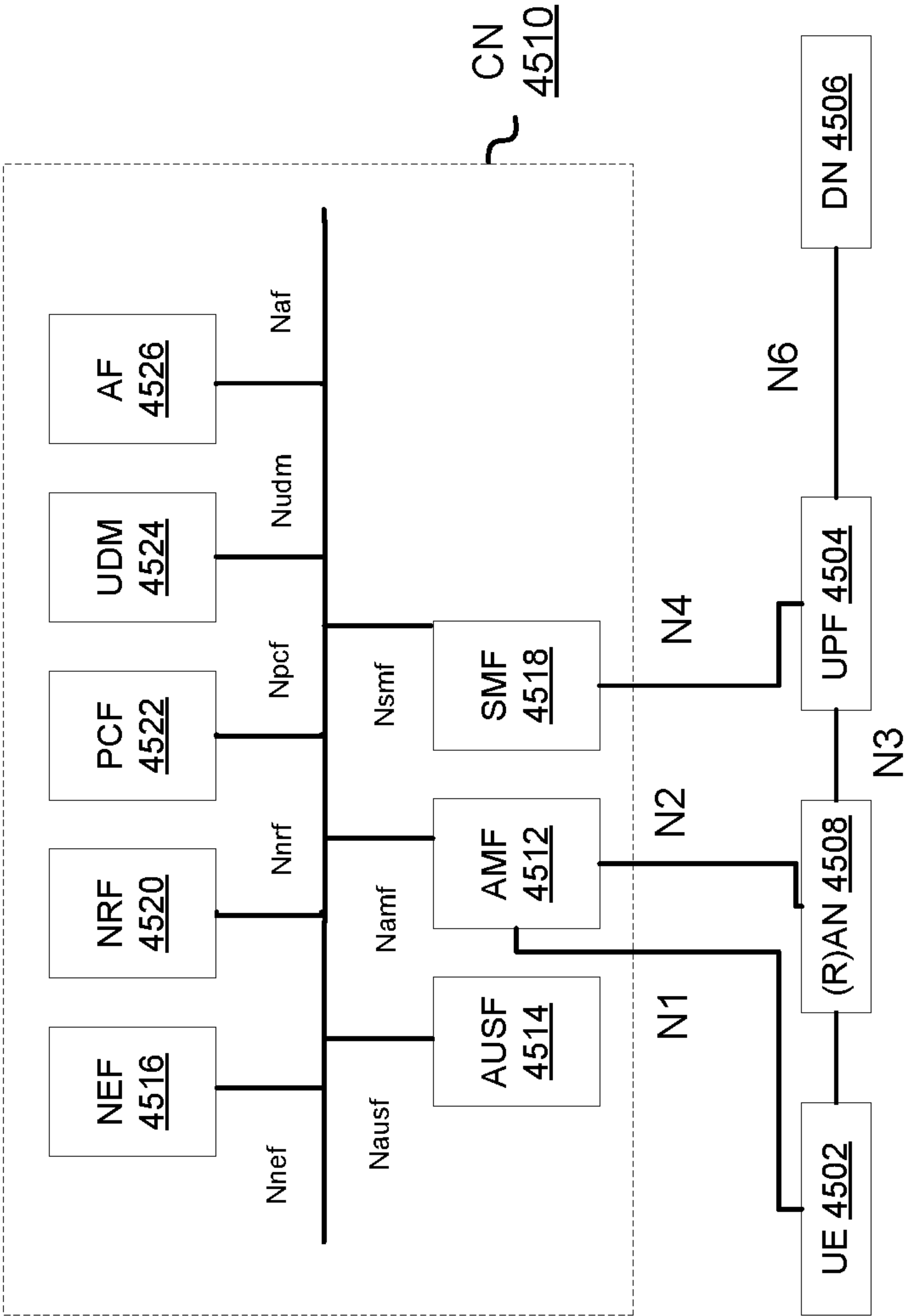


FIG. 45

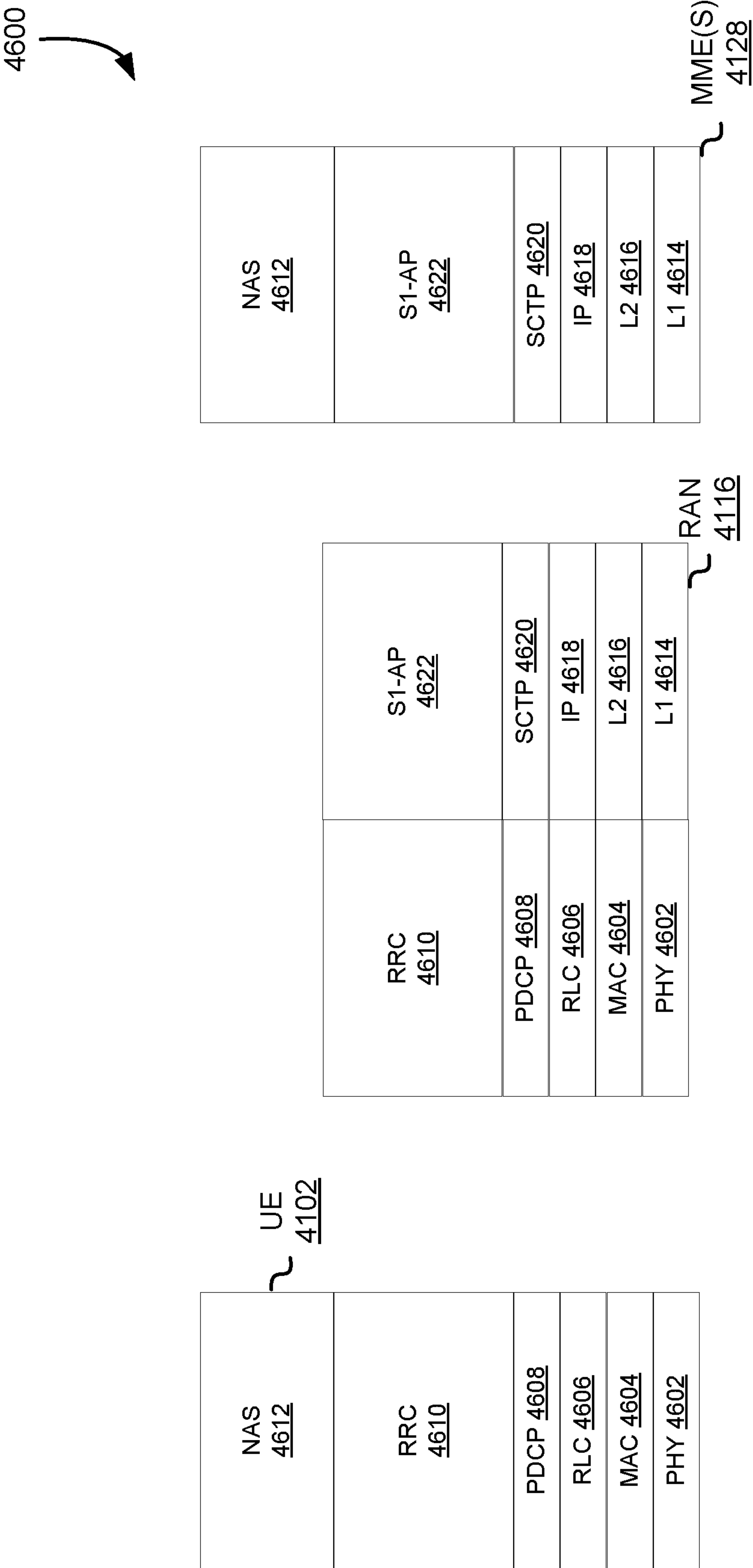
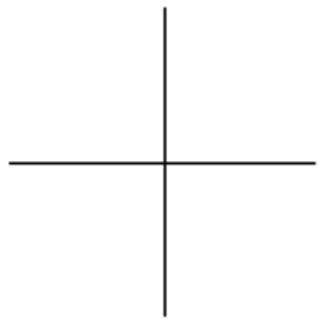
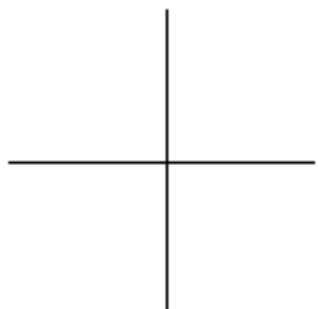
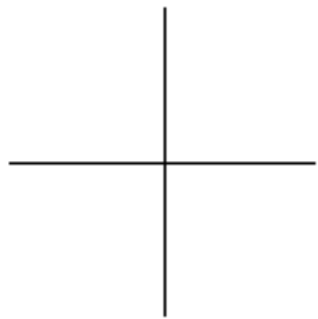
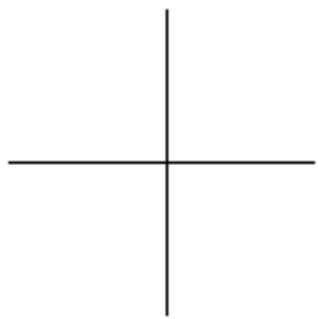


FIG. 46



4700

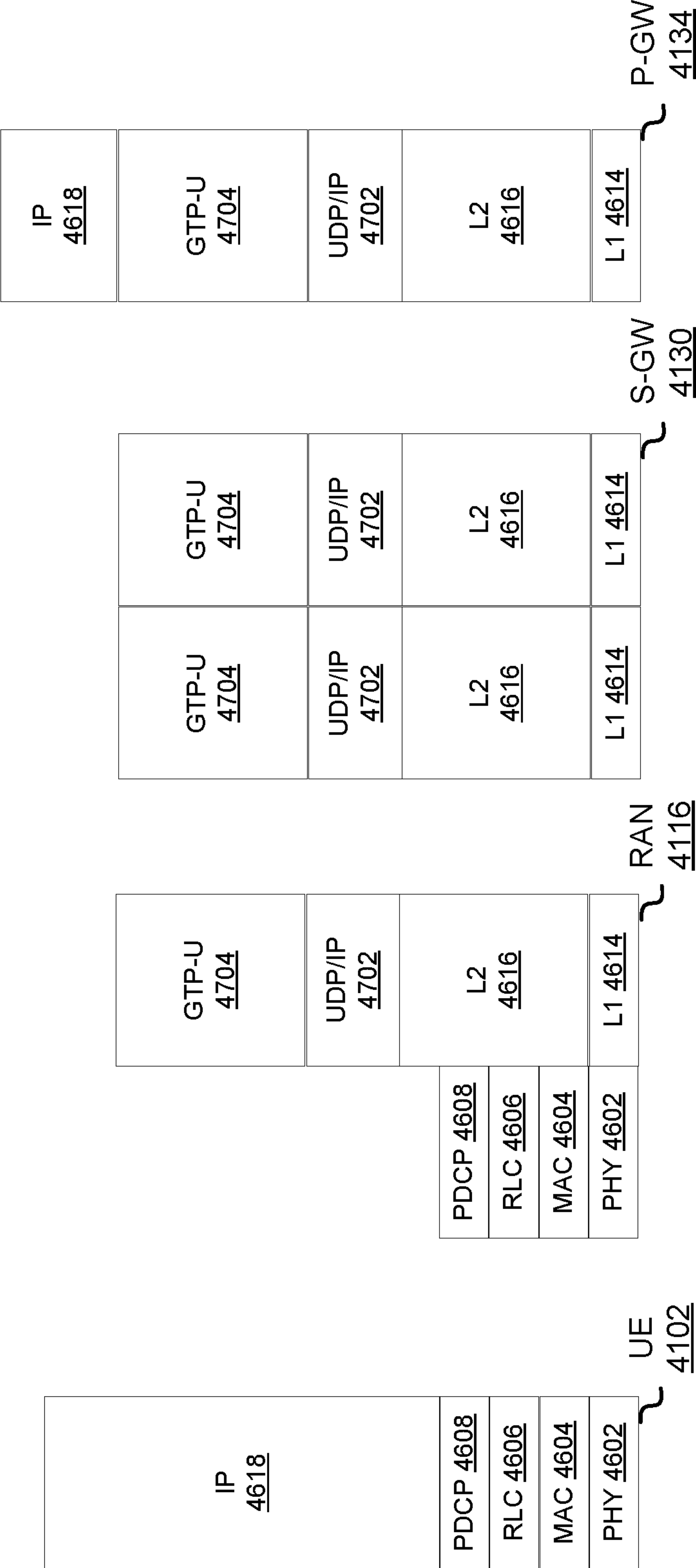


FIG. 47

4800

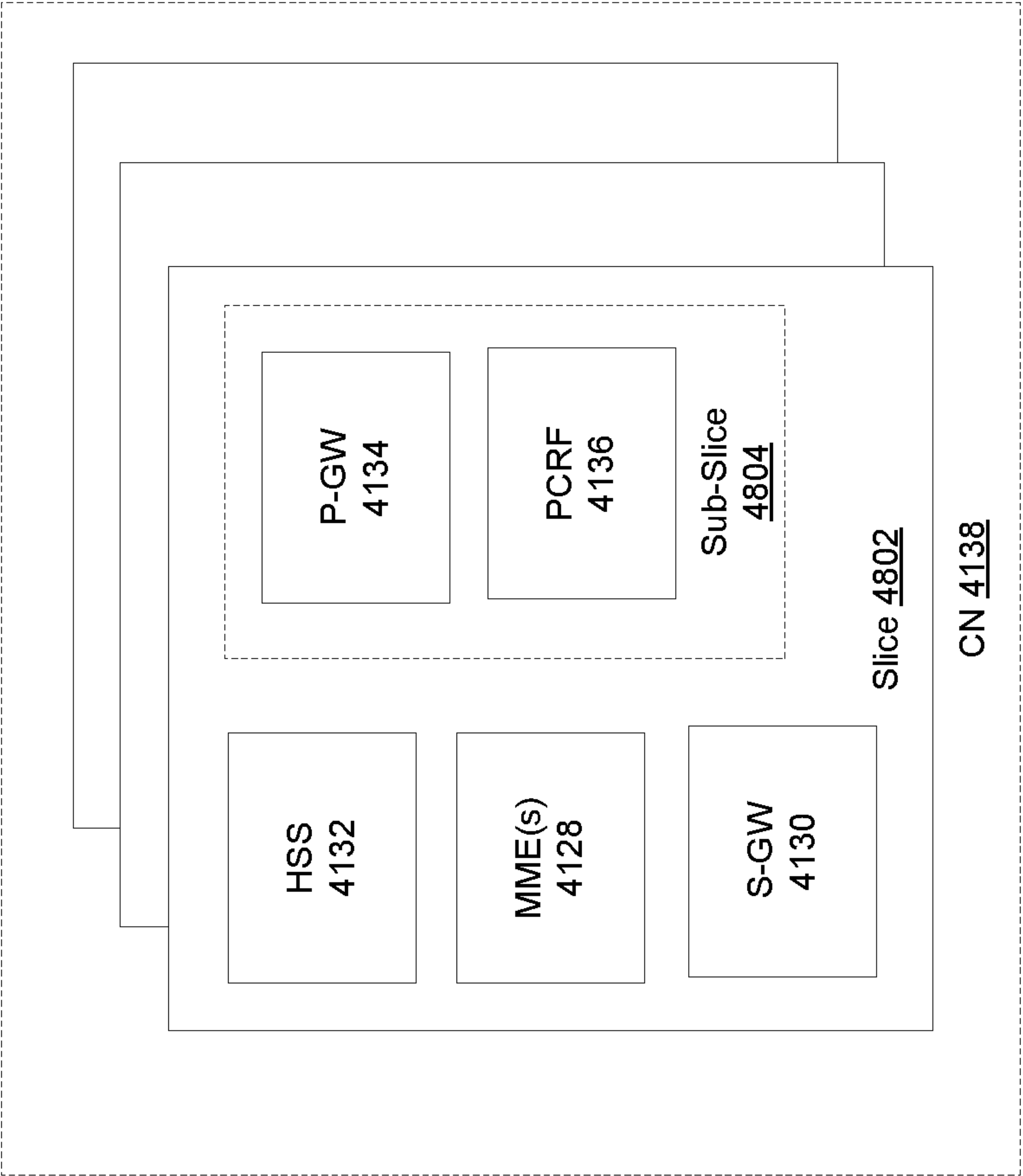


FIG. 48

4900

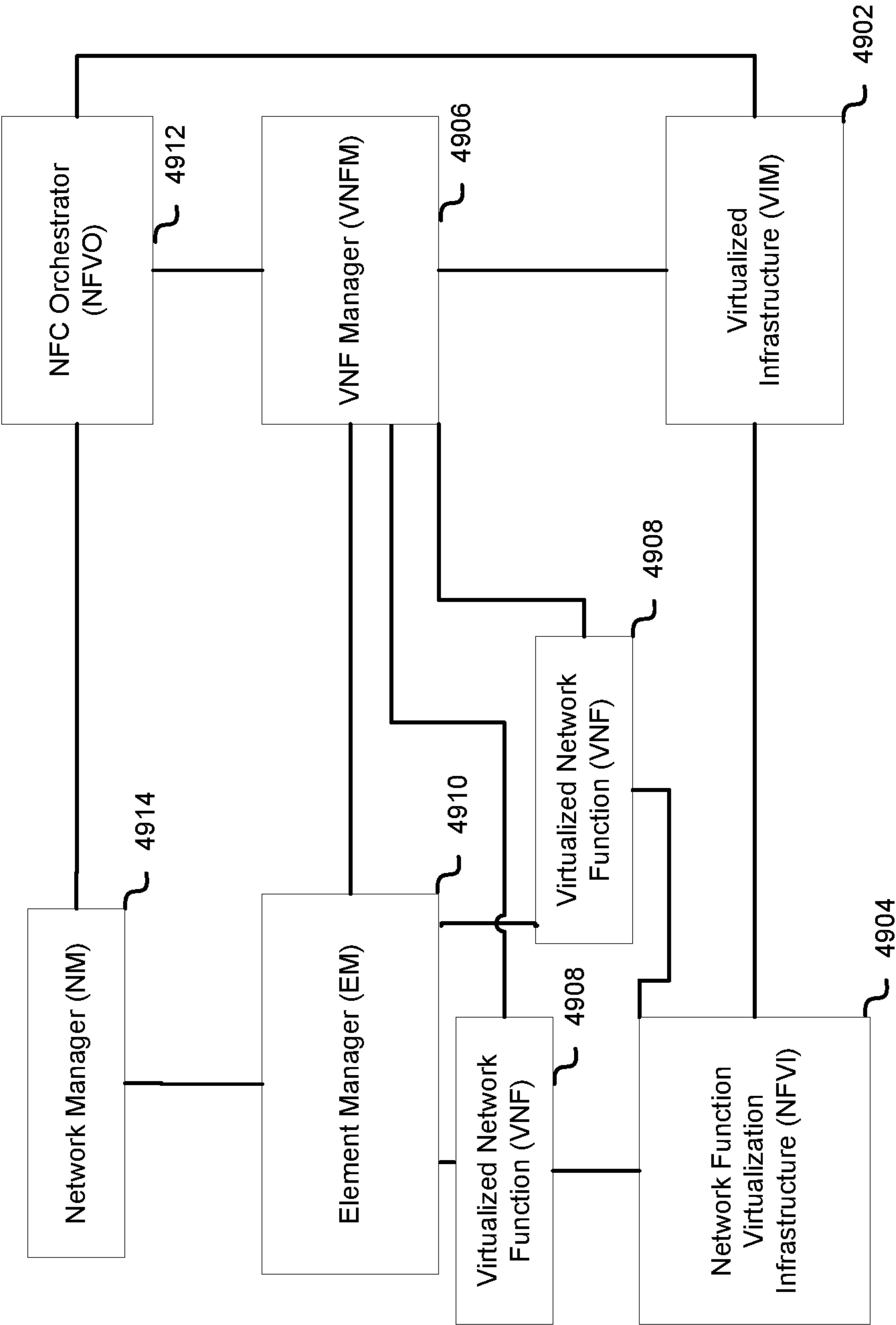


FIG. 49

INTERNATIONAL SEARCH REPORT

International application No

PCT/US2020/060483

A. CLASSIFICATION OF SUBJECT MATTER
 INV. H04L1/00 H03M13/11
 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)
 H04L H03M

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, INSPEC, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2018/123734 A1 (LONCKE VINCENT [US] ET AL) 3 May 2018 (2018-05-03) abstract page 3, paragraph 44 - page 12, paragraph 113 -----	1-31
X	US 2018/109269 A1 (RICHARDSON THOMAS JOSEPH [US] ET AL) 19 April 2018 (2018-04-19) abstract page 1, paragraph 12 - page 2, paragraph 19 page 4, paragraph 43 - page 9, paragraph 86 -----	1-31



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

5 February 2021

Date of mailing of the international search report

15/02/2021

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

Authorized officer

Toumpoulidis, T

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/US2020/060483

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2018123734 A1	03-05-2018	BR 112019008746 A2 CN 109906559 A EP 3535849 A1 JP 2019536316 A KR 20190071728 A SG 11201902036Y A US 2018123734 A1 WO 2018084956 A1	09-07-2019 18-06-2019 11-09-2019 12-12-2019 24-06-2019 30-05-2019 03-05-2018 11-05-2018
US 2018109269 A1	19-04-2018	CN 109804567 A EP 3526900 A1 US 2018109269 A1 WO 2018071834 A1	24-05-2019 21-08-2019 19-04-2018 19-04-2018