IN THE UNITED STATES DISTRICT COURT
FOR THE DISTRICT OF DELAWARE

STRAGENT, LLC, **Plaintiff,** **v.** **BMW OF NORTH AMERICA, LLC, and BMW MANUFACTURING CO., LLC,** **Defendants.**	**C.A. No. 20-510-LPS** **JURY TRIAL DEMANDED**

FIRST SUPPLEMENTAL COMPLAINT FOR PATENT INFRINGEMENT

This is an action for patent infringement in which Plaintiff Stragent, LLC ("Plaintiff" or "Stragent") complains against Defendants BMW of North America, LLC and BMW Manufacturing Co., LLC, all upon information and belief, as follows:

THE PARTIES

1. Plaintiff Stragent is a limited liability company existing under the laws of the State of Texas.

2. Defendant BMW of North America, LLC ("BMW NA") is a limited liability company organized under the laws of Delaware, with its registered office located at The Corporation Trust Company, Corporation Trust Center, 1209 Orange Street, Wilmington, Delaware 19801, and having a place of business at 300 Chestnut Ridge Road, Woodcliff Lake, New Jersey 07677. BMW NA is a wholly owned subsidiary of BMW (US) Holding Corp., a holding entity, which in turn is a wholly owned subsidiary of Bayerische Motoren Werke AG ("BMW AG"). MINI USA, the distributor for the Mini brand of automobiles, is a division of BMW NA.

3. Defendant BMW Manufacturing Co., LLC ("BMW MC") is a corporation

organized under the laws of Delaware, with its registered office located at The Corporation Trust Company, Corporation Trust Center, 1209 Orange Street, Wilmington, Delaware 19801, and having a place of business at 1400 Highway 101 South, Greer, South Carolina 29651. BMW MC is a wholly owned subsidiary of BMW (US) Holding Corp., which in turn is a wholly owned subsidiary of BMW AG.

4. BMW AG is based in Munich, Germany, and is the parent company of the BMW Group, which comprises BMW AG itself and all subsidiaries over which BMW AG has either direct or indirect control. The BMW Group encompasses entities in the Automotive, Motorcycles and Financial Services operating segments operating in about 140 countries and employing a workforce of about 134,682 people. BMW NA and BMW MC are part of the BMW Group.

5. BMW NA markets all BMW-branded and Mini-branded automobiles in this country. BMW MC manufactures certain BMW cars in South Carolina, which it then transfers to BMW NA to market in this country, or to export to other countries.

6. The automobiles which are manufactured by BMW MC, whether sold in this country or exported, and all BMW-branded and Mini-branded automobiles used, offered for sale, sold or imported by BMW NA, since July 11, 2017, the date Patent No. 9,705,765 issued, are collectively referred to here as “BMW Autos” or as “Accused Instrumentalities.”

JURISDICTION AND VENUE

7. This is an action for patent infringement arising under the patent laws of the United States of America, 35 U.S.C. § 1, et seq., including 35 U.S.C. § 271. This Court has subject matter jurisdiction pursuant to 28 U.S.C. §§ 1331 and 1338(a).

8. This Court has general and specific personal jurisdiction over Defendants because

both Defendants reside in the State of Delaware.

9. Venue is proper in this judicial district as to each Defendant under 28 U.S.C. § 1400(b) because both Defendants reside in this judicial district.

DEFENDANTS' USE OF AUTOSAR TECHNOLOGY

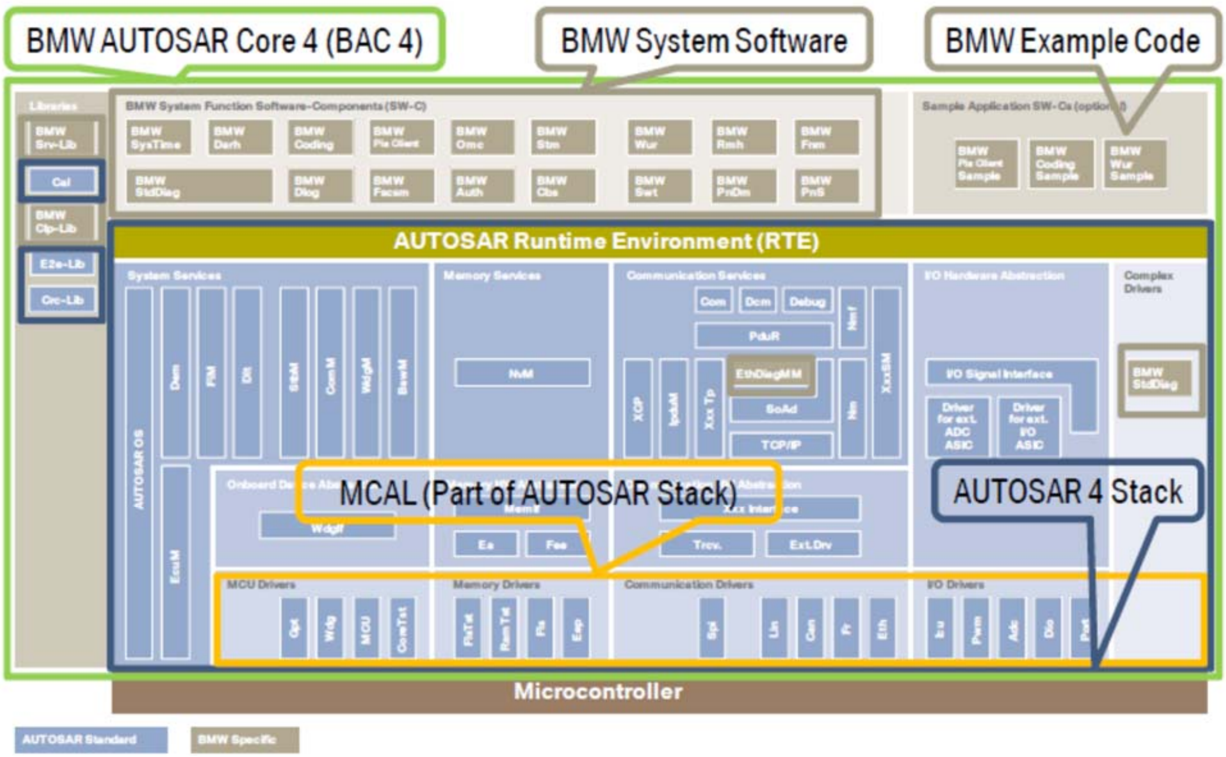
10. AUTOSAR (AUTomotive Open System ARchitecture) is an enabling technology, and, particularly, a layered system for sharing information in an automobile vehicle, which has been standardized by a worldwide development partnership of vehicle manufacturers, suppliers and other companies from the electronics, semiconductor and software industry. AUTOSAR comprises, among other things, a set of specifications describing software architecture components and defining their interfaces. The AUTOSAR standards facilitate the exchange and update of software and hardware over the service life of a vehicle by providing a common software infrastructure for automotive systems of all vehicle domains based on standardized interfaces for the different software layers. E.g. the software in a control module is stored at different levels. Highest up is the actual application that is unique to the different car manufacturers. The application corresponds to a certain function and means that something must finally be carried out. Execution is performed by the processor (Micro controller) generating an electrical output signal. But before the applications request reaches the processor, it (i.e. the request) must be managed by the different parts of the software. Communication between these software parts is now standardized according to guidelines from AUTOSAR. This means that the supplier supplies the control module with software that works directly with the car manufacturer's application. All control modules connected to FlexRay and CAN networks contain software components corresponding to AUTOSAR's specifications. AUTOSAR's specifications also apply to communication that occurs via FlexRay and CAN networks.

11. The BMW Group is a Core Partner in the AUTOSAR consortium. The first BMW automobiles incorporating the Autosar technology were the 2009 model year BMW Series 7. Since then, Autosar has become the standard in all BMW automobiles. Simon Furst, General Manager Software Development and Software Infrastructure for BMW AG, confirmed that AUTOSAR has been implemented as the core common requirement of all electronic control units of all BMW automobiles, or, in Mr. Furst's words: "ONE ARCHITECTURE. ONE STANDARD. AUTOSAR." Dr. Furst's presentation indicates that BMW has incorporated the entirety of the Autosar standard relevant to, at least, the Controller Area Network ("CAN"), FlexRay and Ethernet networks. There have been several releases of AUTOSAR. The AUTOSAR Releases are backward-compatible, and any differences in the releases do not affect the application of the Autosar standard to the BMW Autos.

12. In the period from July 11, 2017, all BMW Autos that have been made, used, offered for sale, sold or imported into this country (including the X1, X2, X3, X4, X5, X6, X7, i3, i8, The BMW 2, 3, 4, 5, 7, 8 Series, Z4 and M Models, as well as BMW's Mini-branded automobiles) have incorporated the Autosar technology.

13. The Autosar systems in BMW Autos include a number of automotive electronic control units ("ECUs"), at least many of which are provided by third-party manufacturers. The ECUs comprise Autosar-related software provided by the BMW Group and software that is specified by the BMW Group, and specifically (a) the Autosar stack, which is the Autosar basic software ("BSW") including the microcontroller abstraction layer and run time environment according to Autosar ver. 4.x.x; (b) BMW Autosar Core 4, which consists of the BMW System Software and a compatible Autosar Stack; and (c) BMW System Software, which denotes the BMW specific parts of the BMW Autosar Core 4 as well as related software specifications,

documentations, software tools, exemplary build environment and tests. Dr. Furst stated in his presentation: "Common requirements and features to all OEMs shall be standardized in Autosar," and that "BMW takes the AUTOSAR stack as base to develop and integrate all BMW specific functionalities, called 'BMW system functions', on top," displaying an image of the BMW Autosar implementation:



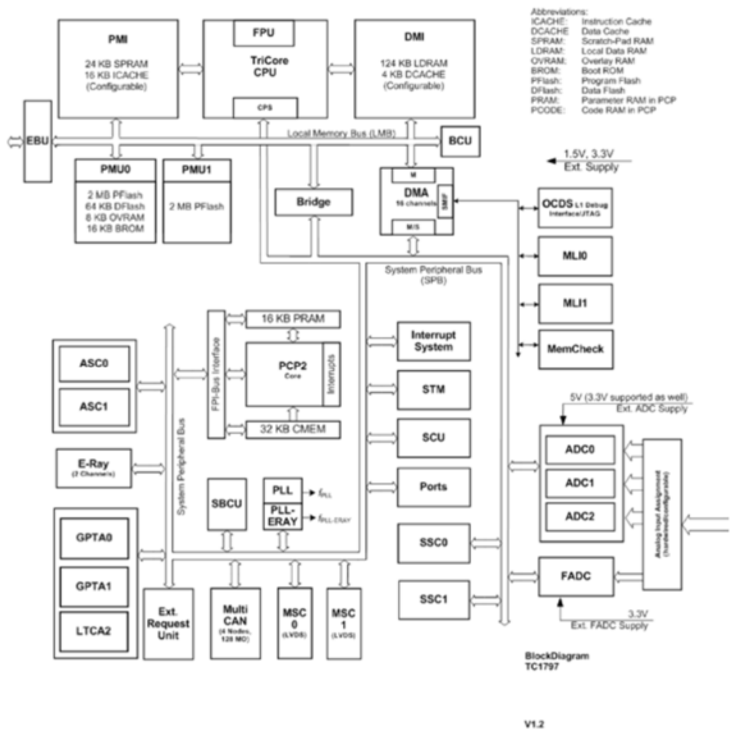
Dr. Furst continued: "BMW [i] delivers all BMW specific platform software parts as source code for series development; [ii] delivers an example configuration and a free license for the make/build system; [and iii] delivers the SW-Cs in serial production quality and supports them during the whole lifecycle."

14. Thereafter, the ECU manufacturer will substitute its own Autosar Stack, but BMW requires that the Stack be compliant with the Autosar 4.x.x. standard, or as Dr. Furst stated, the ECU provider "Has to ensure that 'his' stack is compliant to the required AUTOSAR version." The ability to transmit data by way of AUTOSAR is an essential feature and function

of BMW Autos.

15. Although, as noted above, it is apparent that all BMW Autos comprise the Autosar systems specified by the BMW Group, it is impossible to determine the Autosar detail for any given BMW Auto. The reason is that the Autosar details are embedded in the firmware of the various ECUs of the BMW Auto, and at least many of the ECUs in BMW Autos are VIN-specific. That firmware information cannot be extracted from the ECUs. Further, the BMW Group has maintained its implementation of Autosar as a trade secret, and it is not possible to determine from publicly-available information the actual implementations of Autosar in BMW Autos, at least beyond the information that the BMW Group has generally implemented the full Autosar Release 4.x.x. Thus, for example, one can consider BMW Part MEVD172G, which is the Basic Control Unit DME:

16. The part is manufactured by Bosch, pursuant to the BMW Group's specifications, and is apparently incorporated into certain models. Part MEVD172G has included the Infineon TriCore TC1797 32-Bit Single-Chip Microcontroller:



The chip is designed to be compatible with Autosar for use in engine and transmission control units, and includes, for example, both flash and Ram memory, both CAN and FlexRay channels/modules, serial interfaces, and on-chip debug support. Nevertheless, it is impossible to credibly deconstruct the chip in the ECU to determine the software embedded in the chip. This illustrates the impossibility of determining the exact software that the BMW Group had specified for any particular ECU or automobile.

COUNT I

INFRINGEMENT OF U.S. PATENT NO. 10,248,477

17. Plaintiff realleges and incorporates by reference the paragraphs 1 through 16 of this Complaint, as if fully set forth herein.

18. Plaintiff is the owner by assignment of United States Patent No. 10,248,477 entitled "System, Method and Computer Program Product for Sharing Information in a Distributed Framework" ("the '477 Patent"). The '477 Patent was duly and legally issued on

April 2, 2019. A true and correct copy of the '477 Patent is attached as Exhibit A. Plaintiff holds the exclusive rights to bring suit with respect to any past, present, and future infringement of the '477 Patent.

19. The '477 Patent claims are materially different from the claims that had been considered in *Inter Partes* Reviews IPR2017-00676, IPR2017-00677, IPR2017-00457, IPR2017-00458, IPR2017-01503, IPR2017-01502, IPR2017-01504, 2017-01519, IPR2017-01520, IPR2017-01521; 2017-01522, which collectively involved Patent 8,209,705 claims 1-20, and Patent 8,566,843 claims 1-59 (collectively "Prior IPRs"). As one simple example, the '477 Patent claim 1 specifies a "layered system for sharing information," wherein the information is stored and, thereafter, the system "share[s] <u>the stored information</u> with at least one of a plurality of heterogeneous processes including at least one process associated with a second physical network selected from the group consisting of FlexRay, Controller Area Network, and Local Interconnect Network, utilizing a network protocol different from a protocol of the first physical network," which limitations were not included in the claims considered in the Prior IPRs.

20. During the prosecution of the applications leading to the '477 Patent, applicant cited to the Patent Office all the prior art that was raised during the course of the Prior IPRs, and advised the Patent Office of the Prior IPR proceedings, including that "The above cited *inter partes* reviews have received final written decisions rendering all challenged claims unpatentable."

21. On or about January 21, 2021, Defendant BMW Of North America, LLC, together with Mercedes-Benz USA, LLC, filed a "Petition For *Inter Partes* Review Of U.S. Patent No. 10,248,477," which is Case IPR2021-00417. On July 19, 2021, the United States Patent And Trademark Office Patent Trial And Appeal Board Granted Institution of *Inter Partes*

Review of claims 1–30.

22. Defendants have directly infringed independent claim 23 and its dependent claims 26, 27 and 28 of the '477 Patent, literally and/or under the doctrine of equivalents, by making, using, offering for sale, selling, and/or importing Accused Instrumentalities, on and after April 2, 2019, without license or authority. Any Accused Instrumentality containing at least one ECU that operates according to, or in compliance with, Autosar 4.x.x standards inherently and necessarily infringes the above claims of the '477 Patent. Infringement is demonstrated by the initial claim charts served by Plaintiff on Defendants on July 28, 2021.

23. Plaintiff and Defendants have previously litigated U.S. Patents 8,209,705 and 8,566,843, which are patents which issued from a common application and are related to the '477 Patent. In that prior litigation, Defendants never produced any evidence that the Accused Instrumentalities were not practicing Autosar and/or were not infringing the claims of U.S. Patents 8,209,705 and 8,566,843. If such evidence had existed, Defendants would have disclosed such evidence to Plaintiff, because such proof would have been sufficient to avoid the cost of the prior or future litigations. Thus, Defendants' inability to identify any limitation in the Plaintiff's asserted patents that is not found in Autosar or in Defendants' automotive vehicles is an eloquent admission by Defendants that the implementation of Autosar infringes the claims of Plaintiff's asserted patents and that Defendants are knowingly infringing.

24. Defendants' acts of infringement have caused and continue to cause damage to Plaintiff. Plaintiff is entitled to recover from Defendants the damages sustained by Plaintiff as a result of Defendants' wrongful acts.

COUNT II

DIRECT INFRINGEMENT OF U.S. PATENT NO. 10,031,790

25. Plaintiff realleges and incorporates by reference prior paragraphs 1 through 16 of this Complaint, as if fully set forth herein.

26. Plaintiff is the owner by assignment of United States Patent No. 10,031,790 ("the '790 Patent") entitled "System, Method and Computer Program Product for Sharing Information in a Distributed Framework." The '790 Patent was duly and legally issued on July 24, 2018. A true and correct copy of the '790 Patent is attached as Exhibit B. Plaintiff holds the exclusive rights to bring suit with respect to any past, present, and future infringement of the Patent.

27. The '790 Patent claims are materially different from the claims that had been considered in the Prior IPRs. Further, during the prosecution of the applications leading to the '790 Patent, applicant cited to the Patent Office all the prior art that was raised during the course of the Prior IPRs, and advised the Patent Office of the Prior IPR proceedings, including that "The above cited *inter partes* reviews have received final written decisions rendering all challenged claims unpatentable."

28. On or about January 21, 2021, Defendant BMW Of North America, LLC, together with Mercedes-Benz USA, LLC, filed a "Petition For *Inter Partes* Review Of U.S. Patent No. 10,031,790," which is Case IPR2021-00418. On July 19, 2021, the United States Patent And Trademark Office Patent Trial And Appeal Board Rejected Institution of *Inter Partes* Review.

29. Defendants have directly infringed independent claim 1 and its dependent claims 2, 4, 5, 6, 8 and 9, and independent claim 15 and its dependent claims 16, 18, 19, 20, 22, 23 and 24 of the '790 Patent, literally and/or under the doctrine of equivalents, by making, using,

offering for sale, selling, and/or importing BMW Autos without license or authority. Any Accused Instrumentality containing at least one ECU that operates according to, or in compliance with, Autosar 4.x.x standards inherently and necessarily infringes the above claims of this Patent. Infringement is demonstrated by the initial claim charts served by Plaintiff on Defendants on July 28, 2021.

30. Defendants have been aware of the '790 Patent and its application to BMW Autos since at least June 22, 2018, when Plaintiff delivered a notice letter to Defendants' counsel, advising Defendants of the about-to-issue application for the '790 Patent, together with a claim chart demonstrating how BMW Autos fell within the scope of at least the allowed application claim 50, which is now claim 15 of the '790 Patent.

31. Defendants' response to the above letter of June 22, 2018 never identified any limitation of the '790 Patent that was missing or could not be found in the Accused Instrumentalities. Further, Plaintiff and Defendants have previously litigated U.S. Patents 8,209,705 and 8,566,843, which are patents which issued from a comm on application and are related to the '790 Patent. In that prior litigation, Defendants never produced any evidence that the Accused Instrumentalities were not practicing Autosar and/or were not infringing the claims of U.S. Patents 8,209,705 and 8,566,843. If such evidence had existed, Defendants would have disclosed such evidence to Plaintiff, because such proof would have been sufficient to avoid the cost of the prior or future litigations. Thus, Defendants' inability to identify any limitation in the Plaintiff's patents that is not found in Autosar or in Defendants' automotive vehicles is an eloquent admission by Defendants that the implementation of Autosar infringes the claims of Plaintiff's asserted patents and that Defendants are knowingly infringing. Defendants' acts of infringement have been willful.

32. Defendants' acts of infringement have caused and continue to cause damage to Plaintiff. Plaintiff is entitled to recover from Defendants the damages sustained by Plaintiff as a result of Defendants' wrongful acts.

COUNT III

DIRECT INFRINGEMENT OF U.S. PATENT NO. 10,002,036

33. Plaintiff realleges and incorporates by reference prior paragraphs 1 through 16 of this Complaint, as if fully set forth herein.

34. Plaintiff is the owner by assignment of United States Patent No. 10,002,036 ("the '036 Patent") entitled "System, Method and Computer Program Product for Sharing Information in a Distributed Framework." The '036 Patent was duly and legally issued on June 19, 2018. A true and correct copy of the '036 Patent is attached as Exhibit C. Plaintiff holds the exclusive rights to bring suit with respect to any past, present, and future infringement of the Patent.

35. The '036 Patent claims are materially different from the claims that had been considered in the Prior IPRs. Further, during the prosecution of the applications leading to the '036 Patent, applicant cited to the Patent Office the Prior IPRs and all the prior art that was raised during the course of the Prior IPRs.

36. On or about January 21, 2021, Defendant BMW Of North America, LLC, together with Mercedes-Benz USA, LLC, filed a series of petitions for *Inter Partes* Review of U.S. Patent No. 10,002,036, which are IPR Cases 2021-00425; 2021-00426 and 2021-00427. On July 19, 2021, in a series of Decisions, the United States Patent And Trademark Office Patent Trial And Appeal Board Rejected Institution of *Inter Partes* Review. Other petitions for *Inter Partes* Review are pending.

37. Defendants have directly infringed independent claim 1 and its dependent claims

9, 19, 34, 38, 72, 79, 82 and 98; and independent claim 102 and its dependent claims 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 117, 121 and 122, of the ‘036 Patent, literally and/or under the doctrine of equivalents, by making, using, offering for sale, selling, and/or importing the Accused Instrumentalities without license or authority. Any Accused Instrumentality containing at least one ECU that operates according to, or in compliance with, Autosar 4.x.x standards inherently and necessarily infringes the above claims of the ‘036 Patent. Infringement is demonstrated by the initial claim charts served by Plaintiff on Defendants on July 28, 2021.

38. Defendants have been aware of the ‘036 Patent and its application to the Accused Instrumentalities since at least June 22, 2018, when Plaintiff delivered a notice letter to Defendants’ counsel, advising Defendants of the ‘036 Patent, together with Exhibit G which demonstrated how the Accused Instrumentalities fell within the scope of claim 1 of the ‘036 Patent.

39. Defendants’ response to the above letter of June 22, 2018 never identified any limitation of the ‘036 Patent that was missing or could not be found in the Accused Instrumentalities. Further, Plaintiff and Defendants have previously litigated U.S. Patents 8,209,705 and 8,566,843, which are patents which issued from a common application and are related to the ‘036 Patent. In that prior litigation, Defendants never produced any evidence that the Accused Instrumentalities were not practicing Autosar and/or were not infringing the claims of U.S. Patents 8,209,705 and 8,566,843. If such evidence had existed, Defendants would have disclosed such evidence to Plaintiff, because such proof would have been sufficient to avoid the cost of the prior or future litigations. Thus, Defendants’ inability to identify any limitation in the Plaintiff’s asserted patents that is not found in Autosar or in Defendants’ automotive vehicles is

an eloquent admission by Defendant that the implementation of Autosar infringes the claims of Plaintiff's asserted patents and that Defendant is knowingly infringing. Defendants' acts of infringement have been willful.

40. Defendants' acts of infringement have caused and continue to cause damage to Plaintiff. Plaintiff is entitled to recover from Defendants the damages sustained by Plaintiff as a result of Defendants' wrongful acts.

COUNT IV

DIRECT INFRINGEMENT OF U.S. PATENT NO. 9,705,765

41. Plaintiff realleges and incorporates by reference prior paragraphs 1 through 9 of this Complaint, as if fully set forth herein.

42. Plaintiff is the owner by assignment of United States Patent No. 9,705,765 ("the '765 Patent") entitled "System, Method and Computer Program Product for Sharing Information in a Distributed Framework." The '765 Patent was duly and legally issued on July 11, 2017. A true and correct copy of the '765 Patent is attached as Exhibit D. Plaintiff holds the exclusive rights to bring suit with respect to any past, present, and future infringement of the Patent.

43. The '765 Patent claims are materially different from the claims that had been considered in the Prior IPRs. Further, during the prosecution of the applications leading to the '765 Patent, applicant cited to the Patent Office the Prior IPRs and all the prior art that was raised during the course of the Prior IPRs.

44. On or about January 21, 2021, Defendant BMW Of North America, LLC, together with Mercedes-Benz USA, LLC, filed a petition for *Inter Partes* Review of Patent 9,705,765, which is Case IPR2021-00419. On July 19, 2021, the United States Patent And Trademark Office Patent Trial And Appeal Board Rejected Institution of *Inter Partes* Review.

45. Defendants have directly infringed independent claims 12 and 24, and claims 26, 27, 28 and 31 dependent on claim 24 of the '765 Patent, literally and/or under the doctrine of equivalents, by making, using, offering for sale, selling, and/or importing the Accused Instrumentalities without license or authority. Any Accused Instrumentality containing at least one ECU that operates according to, or in compliance with, Autosar 4.x.x standards inherently and necessarily infringes the above claims of this Patent. Infringement is demonstrated by the initial claim charts served by Plaintiff on Defendants on July 28, 2021.

46. Defendants have been aware of the '765 Patent and its application to the Accused Instrumentalities since at least June 22, 2018, when Plaintiff delivered a notice letter to counsel, advising Defendants of the '765 Patent, together with a claim chart demonstrating how the Accused Instrumentalities fell within the scope of claim 1 of the '765 Patent.

47. Defendants' response to the above letter of June 22, 2018 never identified any limitation of the '765 Patent that was missing or could not be found in the Accused Instrumentalities. Further, Plaintiff and Defendants had previously litigated U.S. Patents 8,209,705 and 8,566,843, which are patents which issued from a common application and are related to the '765 Patent. In that prior litigation, Defendants never produced any evidence that the Accused Instrumentalities were not practicing Autosar and/or were not infringing the claims of U.S. Patents 8,209,705 and 8,566,843. If such evidence had existed, Defendants would have disclosed such evidence to Plaintiff, because such proof would have been sufficient to avoid the cost of the prior or future litigations. Thus, Defendants' inability to identify any limitation in the Plaintiff's asserted patents that is not found in Autosar or in Defendants' automotive vehicles is an eloquent admission by Defendants that the implementation of Autosar infringes the claims of Plaintiff's asserted patents and that Defendants are knowingly infringing. Defendant's acts of

infringement have been willful.

48. Defendants' acts of infringement have caused and continue to cause damage to Plaintiff. Plaintiff is entitled to recover from Defendants the damages sustained by Plaintiff as a result of Defendants' wrongful acts.

PRAYER FOR RELIEF

WHEREFORE, Plaintiff Stragent requests that this Court enter:

A. A judgment in favor of Stragent that Defendants have infringed the '477 Patent, the '790 Patent, the '036 Patent and the '765 Patent;

B. A judgment and order requiring Defendants to pay Stragent its damages, costs, expenses, prejudgment and post-judgment interest, and post-judgment royalties for Defendants' infringement of the Patents as provided under 35 U.S.C. § 284;

C. A judgment and order holding that Defendants' infringement was willful, and awarding treble damages and attorney fees and expenses;

D. Judgment that this is an exceptional case, and, thus, awarding attorney fees and expenses to Plaintiff; and

E. Any and all other relief to which the Court may deem Stragent entitled.

DEMAND FOR JURY TRIAL

Plaintiff Stragent, under Rule 38 of the Federal Rules of Civil Procedure, requests a trial by jury of any issues so triable by right.

Dated: August 4, 2021

Respectfully submitted,

/s/ George Pazuniak

George Pazuniak (DE Bar 478)
O'KELLY & O'ROURKE, LLC
824 N. Market St.
Suite 1001A
Wilmington, DE 19801
Direct: 207-359-8576
Email: GP@del-iplaw.com

Counsel for Plaintiff Stragent, LLC.

Of Counsel:

Thomas F. Meagher
Alan Christopher Pattillo
Meagher Emanuel Laks Goldberg & Liao, LLP
One Palmer Square
Suite 325
Princeton, New Jersey 08542
(609) 454-3500
tmeagher@meagheremanuel.com
cpattillo@meagheremanuel.com

US010248477B2

(12) **United States Patent**
Fuchs et al.

(10) **Patent No.: US 10,248,477 B2**
(45) **Date of Patent: *Apr. 2, 2019**

(54) **SYSTEM, METHOD AND COMPUTER PROGRAM PRODUCT FOR SHARING INFORMATION IN A DISTRIBUTED FRAMEWORK**

(71) Applicant: **Stragent, LLC**, Longview, TX (US)

(72) Inventors: **Axel Fuchs**, San Jose, CA (US); **Scott Sturges Andrews**, Petaluma, CA (US)

(73) Assignee: **Stragent, LLC**, Longview, TX (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **16/148,949**

(22) Filed: **Oct. 1, 2018**

(65) **Prior Publication Data**

US 2019/0034250 A1 Jan. 31, 2019

Related U.S. Application Data

(63) Continuation of application No. 16/042,159, filed on Jul. 23, 2018, and a continuation of application No.
(Continued)

(51) **Int. Cl.**
G06F 9/54 (2006.01)
H04L 12/26 (2006.01)
(Continued)

(52) **U.S. Cl.**
CPC ***G06F 9/546*** (2013.01); ***B60R 16/0231*** (2013.01); ***G06F 9/54*** (2013.01);
(Continued)

(58) **Field of Classification Search**
CPC G06F 9/542; G06F 9/545; G06F 9/546; H04L 43/10; H04L 67/10; B60R 16/0231
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,956,489 A * 9/1999 San Andres G06F 11/1662
709/221

OTHER PUBLICATIONS

Notice of Allowanced dated Jun. 27, 2018 for U.S. Appl. No. 15/919,201.

* cited by examiner

Primary Examiner — Charles E Anya

(57) **ABSTRACT**

A system, method and computer program product are provided for sharing information in an automobile vehicle comprising: receiving information in the form of a packet data unit representing datum information carried by an overall message from a first physical network selected from the group consisting of FlexRay, Controller Area Network, Local Interconnect Network and Media Oriented Systems Transport; in response to the receipt of the information, issuing a storage resource request in connection with a storage resource; determining whether the storage resource is available for storing the information; determining whether a threshold has been reached in association with the storage resource request; in the event the storage resource is not available and the threshold associated with the storage resource request has not been reached, issuing another storage resource request in connection with the storage resource; in the event the storage resource is available, storing the information in the storage resource; and sharing the stored information with at least one of a plurality of heterogeneous processes including at least one process associated with a second physical network selected from the group consisting of FlexRay, Controller Area Network, Local Interconnect Network and Media Oriented Systems Transport, utilizing a network protocol different from a protocol of the first physical network.

30 Claims, 17 Drawing Sheets

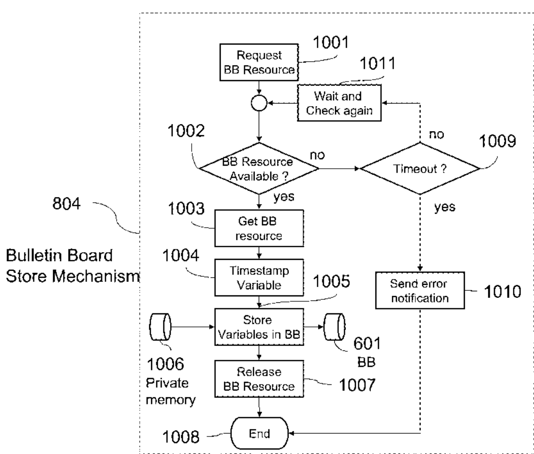


EXHIBIT A

Related U.S. Application Data

15/919,201, filed on Mar. 12, 2018, now Pat. No. 10,031,790, which is a continuation of application No. 15/405,110, filed on Jan. 12, 2017, now Pat. No. 10,002,036, which is a continuation of application No. 14/011,705, filed on Aug. 27, 2013, now Pat. No. 9,575,817, which is a continuation of application No. 13/531,319, filed on Jun. 22, 2012, now Pat. No. 8,566,843, which is a continuation of application No. 12/182,570, filed on Jul. 30, 2008, now Pat. No. 8,209,705, which is a continuation of application No. 10/737,690, filed on Dec. 15, 2003, now Pat. No. 7,802,263.

(60) Provisional application No. 60/434,018, filed on Dec. 17, 2002.

(51) **Int. Cl.**
H04L 29/06 (2006.01)
H04L 29/08 (2006.01)
B60R 16/023 (2006.01)

(52) **U.S. Cl.**
CPC ***G06F 9/542*** (2013.01); ***G06F 9/545*** (2013.01); ***H04L 43/04*** (2013.01); ***H04L 43/06*** (2013.01); ***H04L 43/08*** (2013.01); ***H04L 43/10*** (2013.01); ***H04L 67/02*** (2013.01); ***H04L 67/10*** (2013.01); ***H04L 67/12*** (2013.01); *G06F 2209/547* (2013.01); *H04L 65/102* (2013.01)

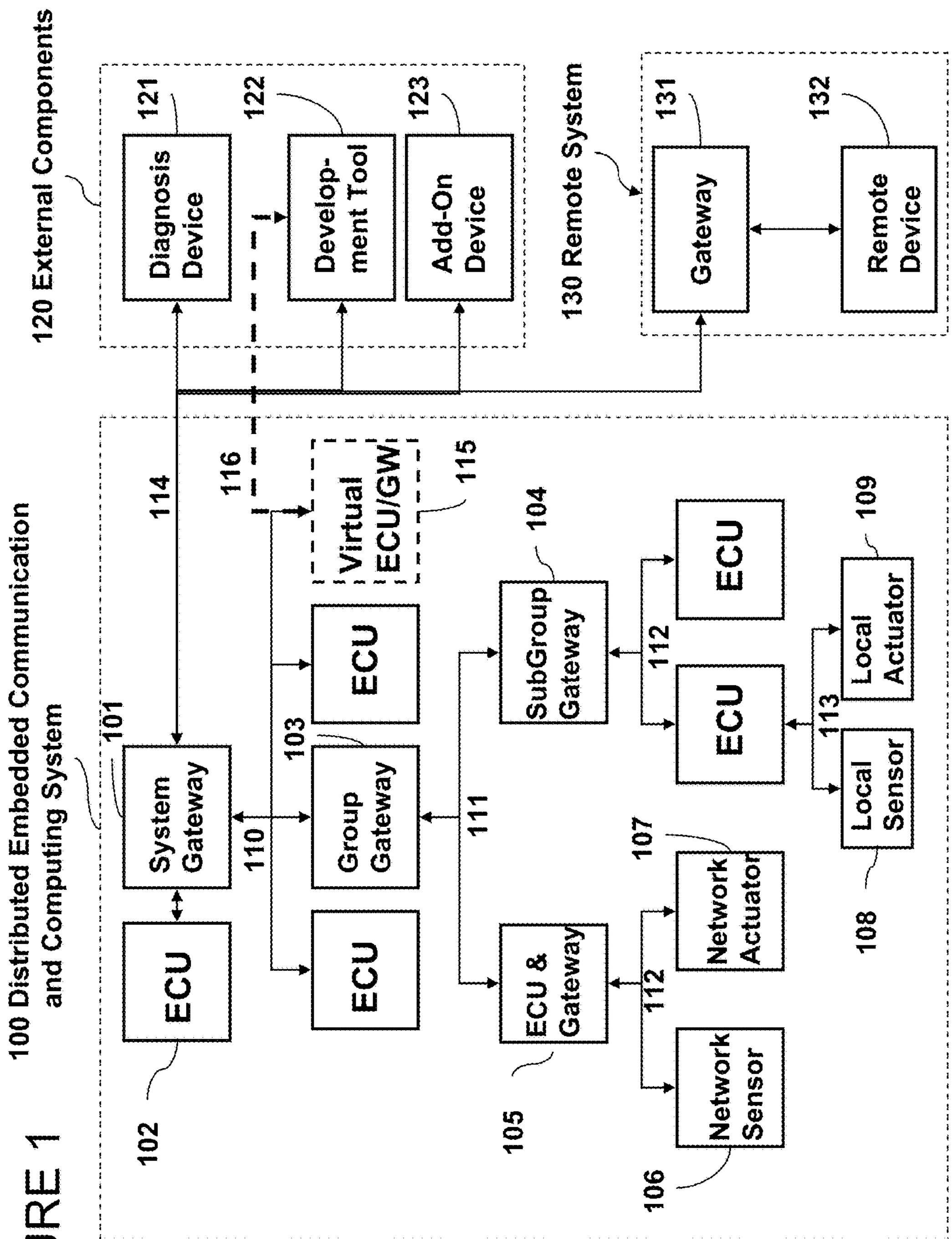

120 External Components
121
Diagnosis Device
122
Develop-ment Tool
123
Add-On Device
130 Remote System
131
Gateway
132
Remote Device
100 Distributed Embedded Communication and Computing System
101
System Gateway
102
ECU
114
110
103
Group Gateway
ECU
ECU
116
Virtual ECU/GW
115
111
104
SubGroup Gateway
105
ECU & Gateway
112
ECU
ECU
112
113
109
Local Actuator
Local Sensor
108
107
Network Actuator
Network Sensor
106

FIGURE 1

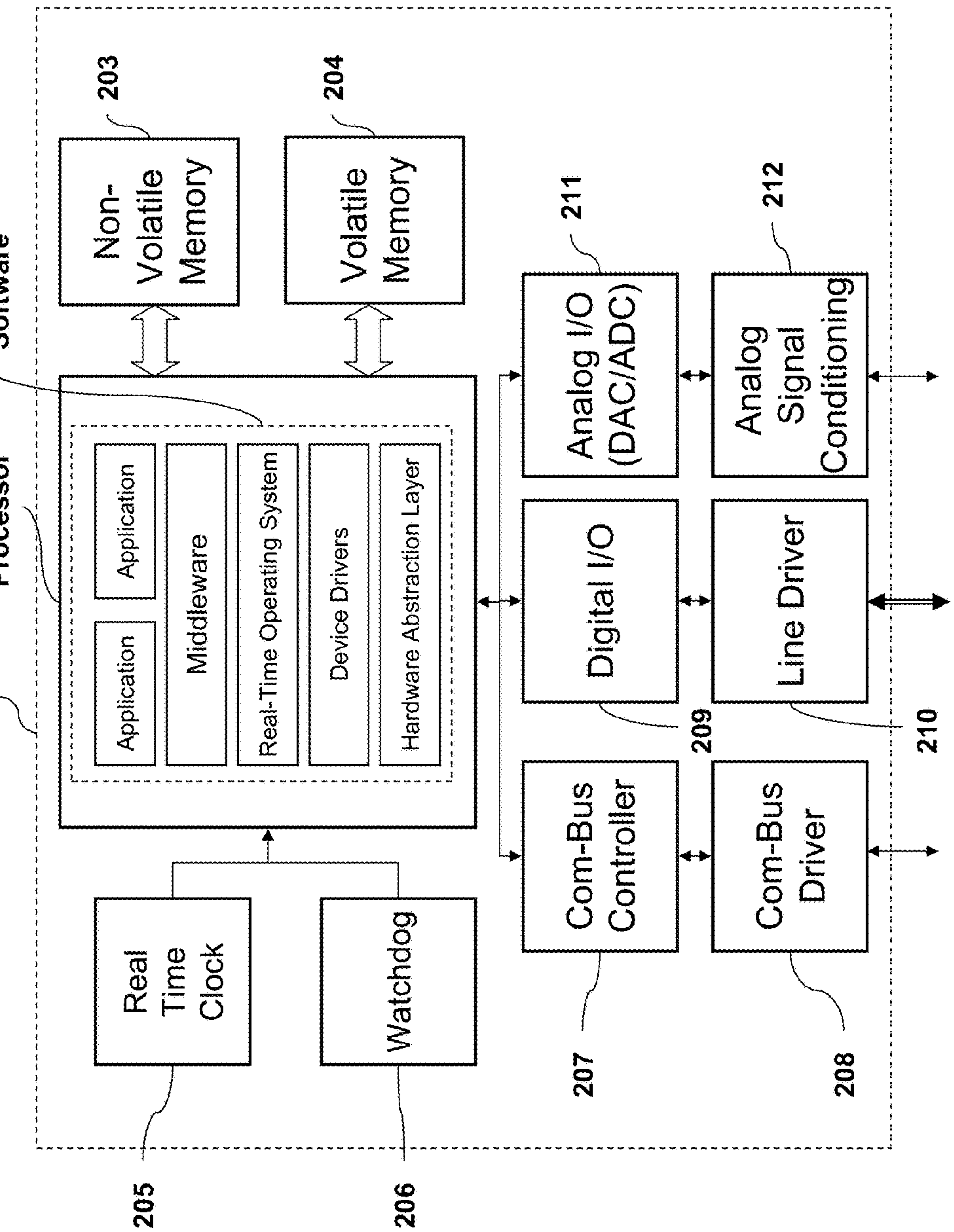

200 ECU
201 Micro Processor
202 Software
Application
Application
Middleware
Real-Time Operating System
Device Drivers
Hardware Abstraction Layer
203
Non-Volatile Memory
204
Volatile Memory
Real Time Clock
205
Watchdog
206
Com-Bus Controller
207
Com-Bus Driver
208
Digital I/O
209
Line Driver
210
Analog I/O (DAC/ADC)
211
Analog Signal Conditioning
212

FIGURE 2

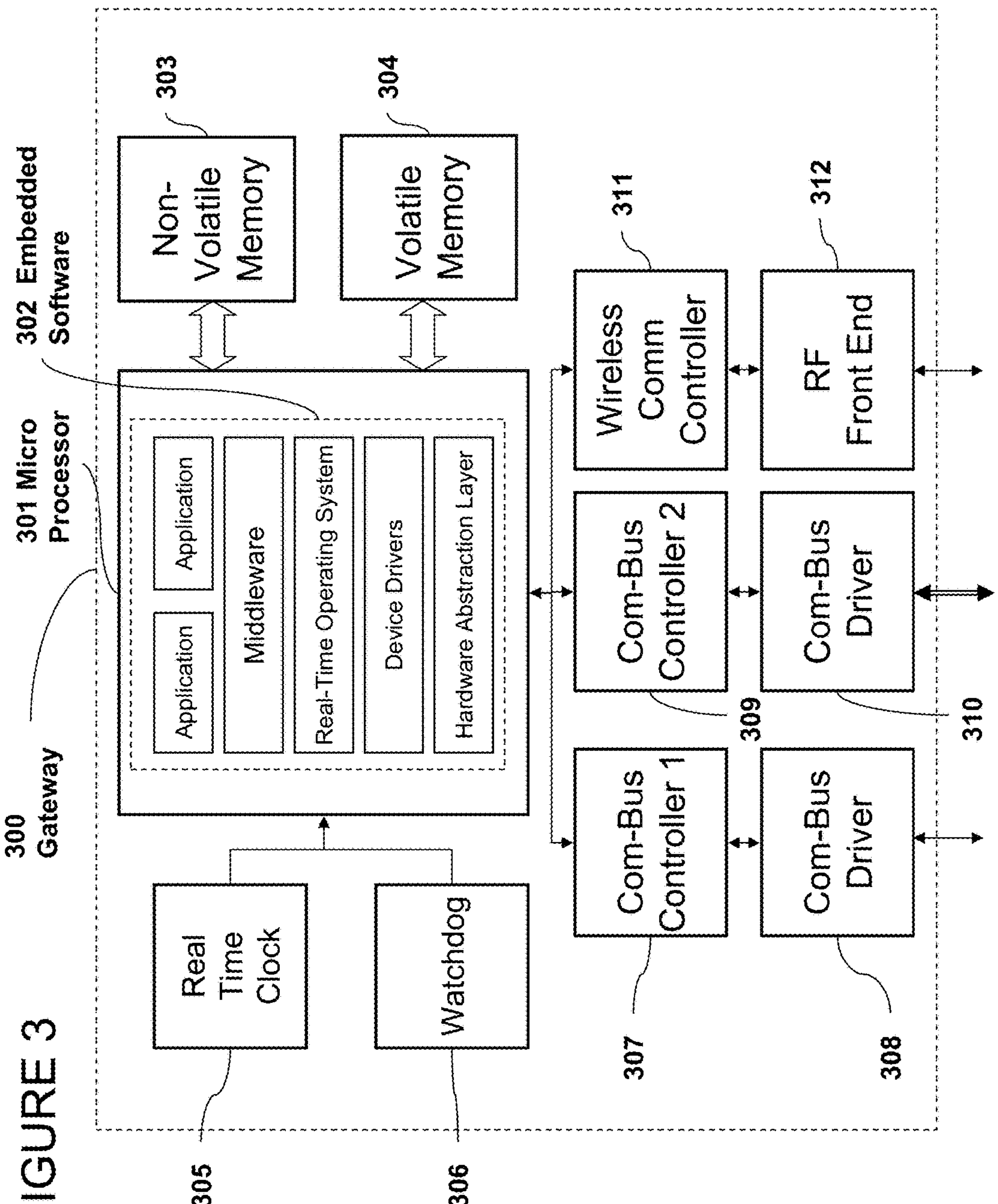

300 Gateway
301 Micro Processor
302 Embedded Software
Application
Application
Middleware
Real-Time Operating System
Device Drivers
Hardware Abstraction Layer
303
Non-Volatile Memory
304
Volatile Memory
305
Real Time Clock
306
Watchdog
307
Com-Bus Controller 1
308
Com-Bus Driver
309
Com-Bus Controller 2
310
Com-Bus Driver
311
Wireless Comm Controller
312
RF Front End

FIGURE 3

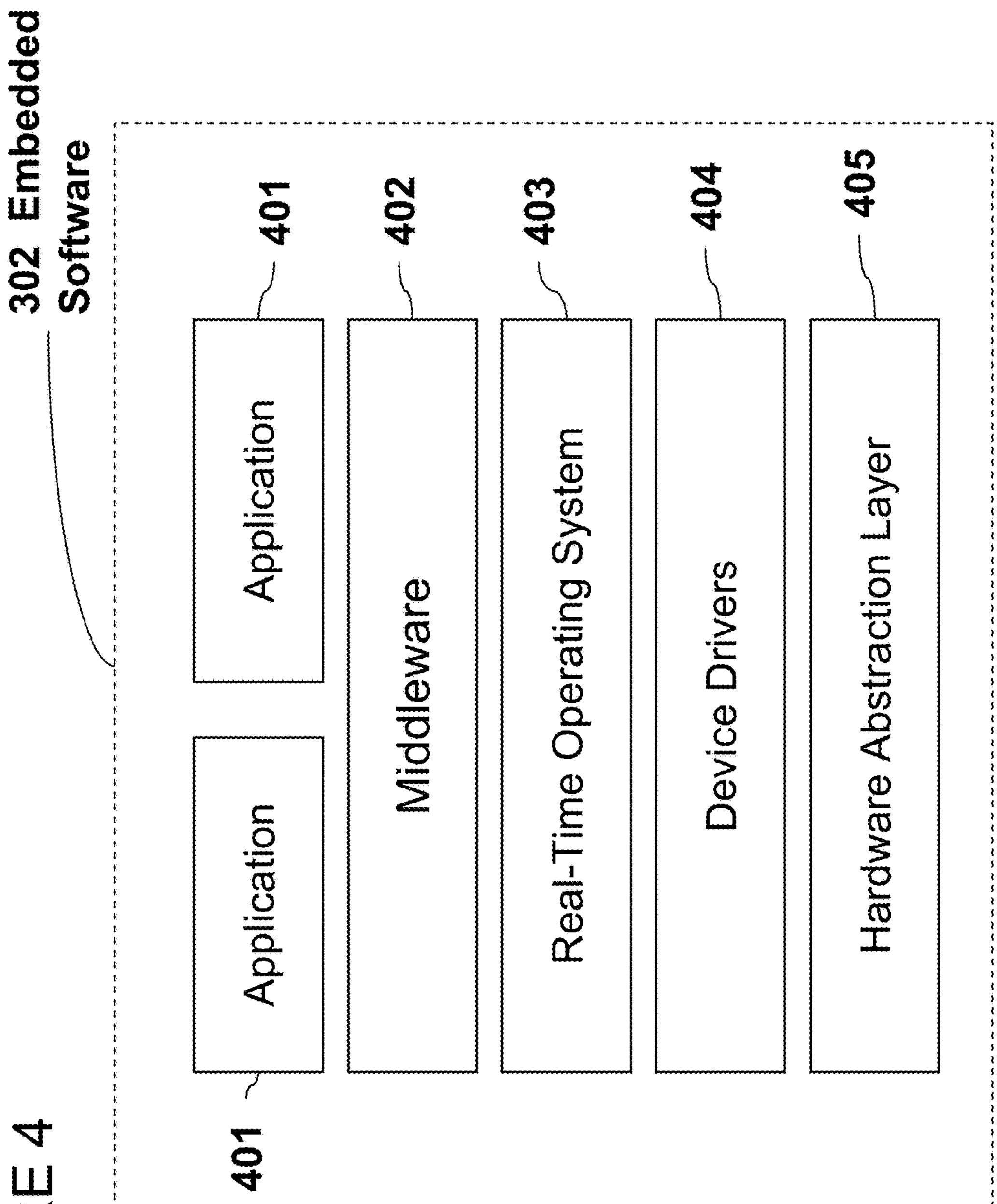

302 Embedded Software
401
402
403
404
405
Application
Application
Middleware
Real-Time Operating System
Device Drivers
Hardware Abstraction Layer
401

FIGURE 4

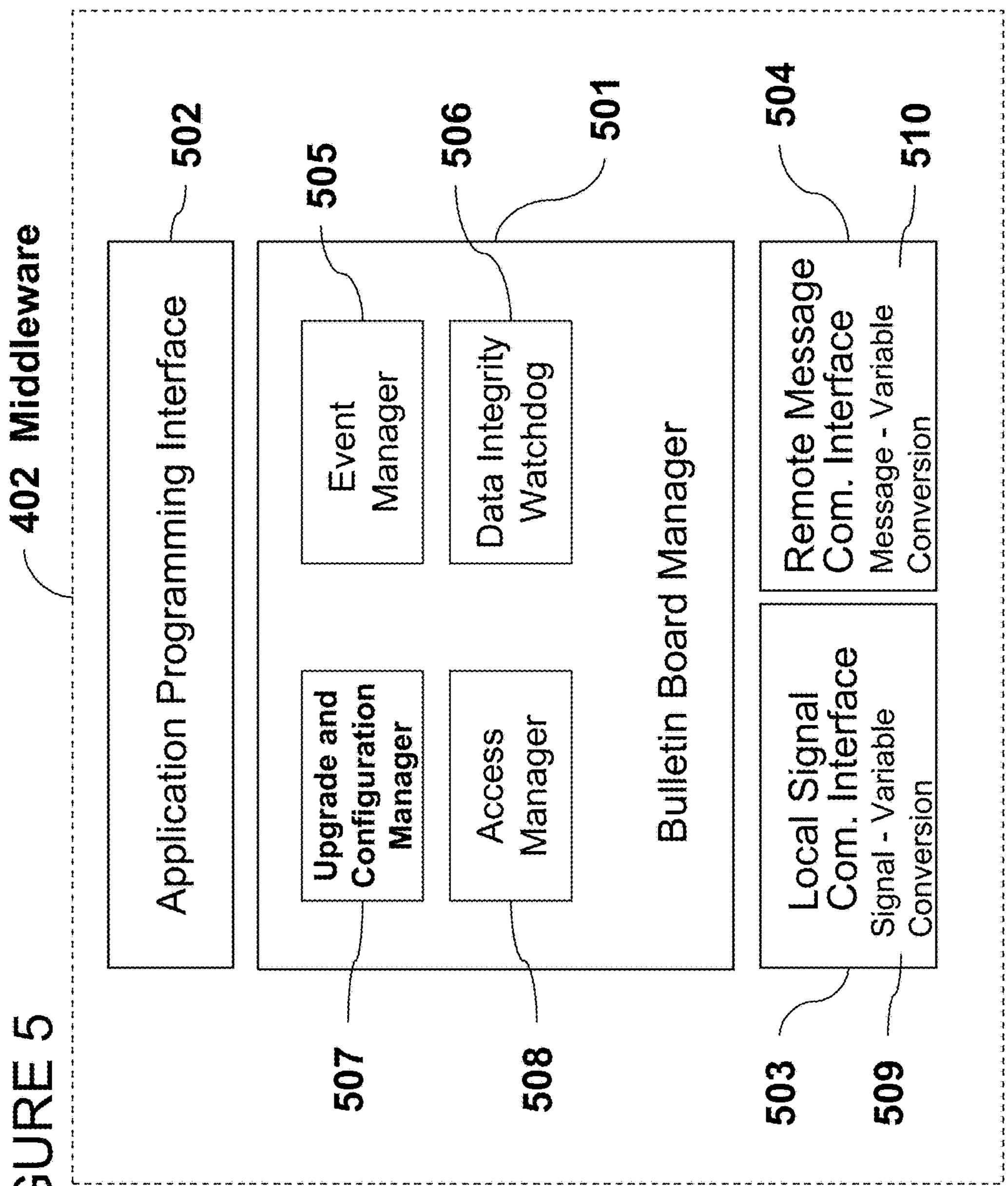

402 Middleware
Application Programming Interface
502
Bulletin Board Manager
501
Event Manager
505
Data Integrity Watchdog
506
Upgrade and Configuration Manager
507
Access Manager
508
Remote Message Com. Interface
Message - Variable Conversion
504
510
Local Signal Com. Interface
Signal - Variable Conversion
503
509

FIGURE 5

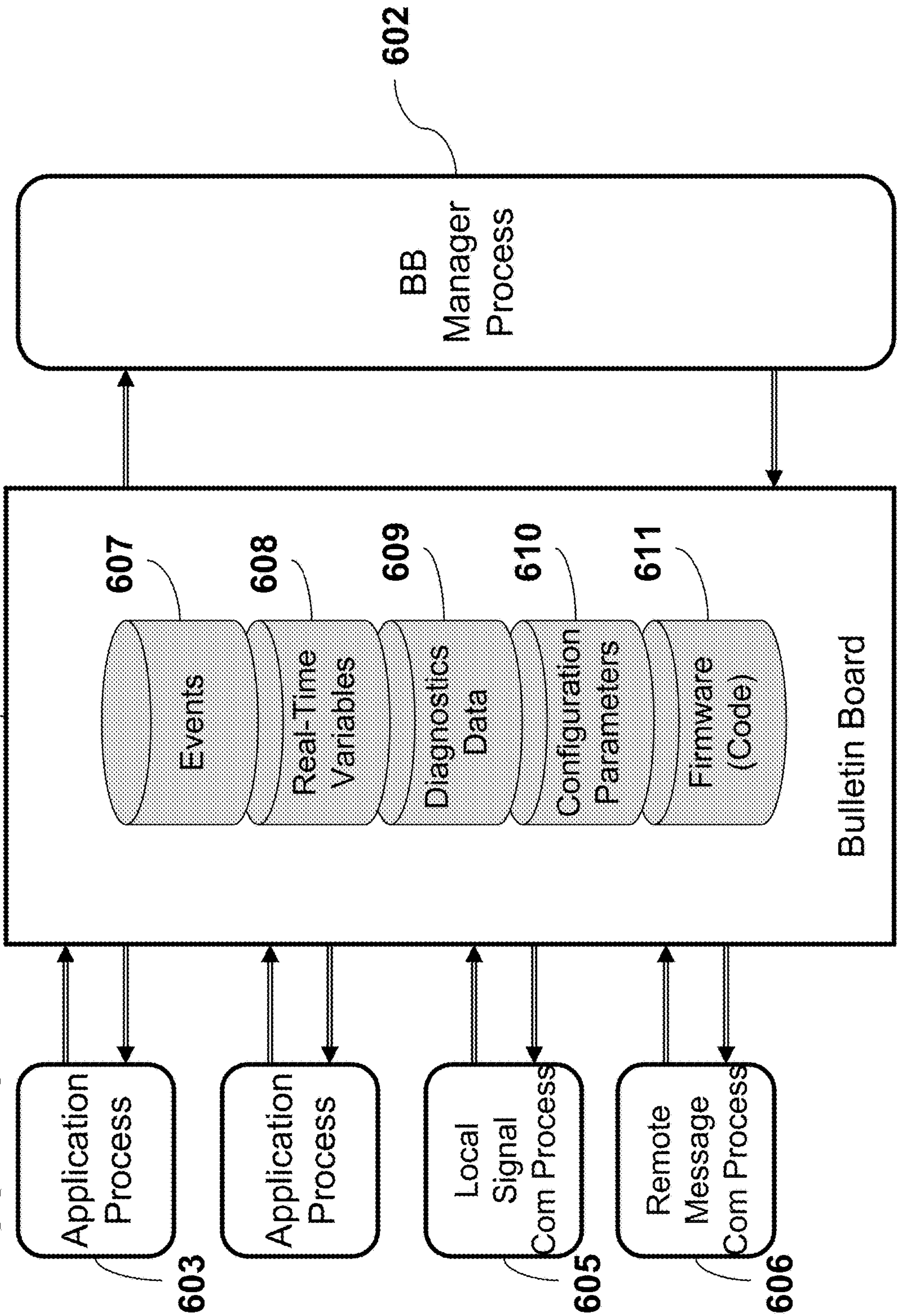

602
BB Manager Process
601
607
608
609
610
611
Events
Real-Time Variables
Diagnostics Data
Configuration Parameters
Firmware (Code)
Bulletin Board
Application Process
Application Process
Local Signal Com Process
Remote Message Com Process
603
605
606

FIGURE 6

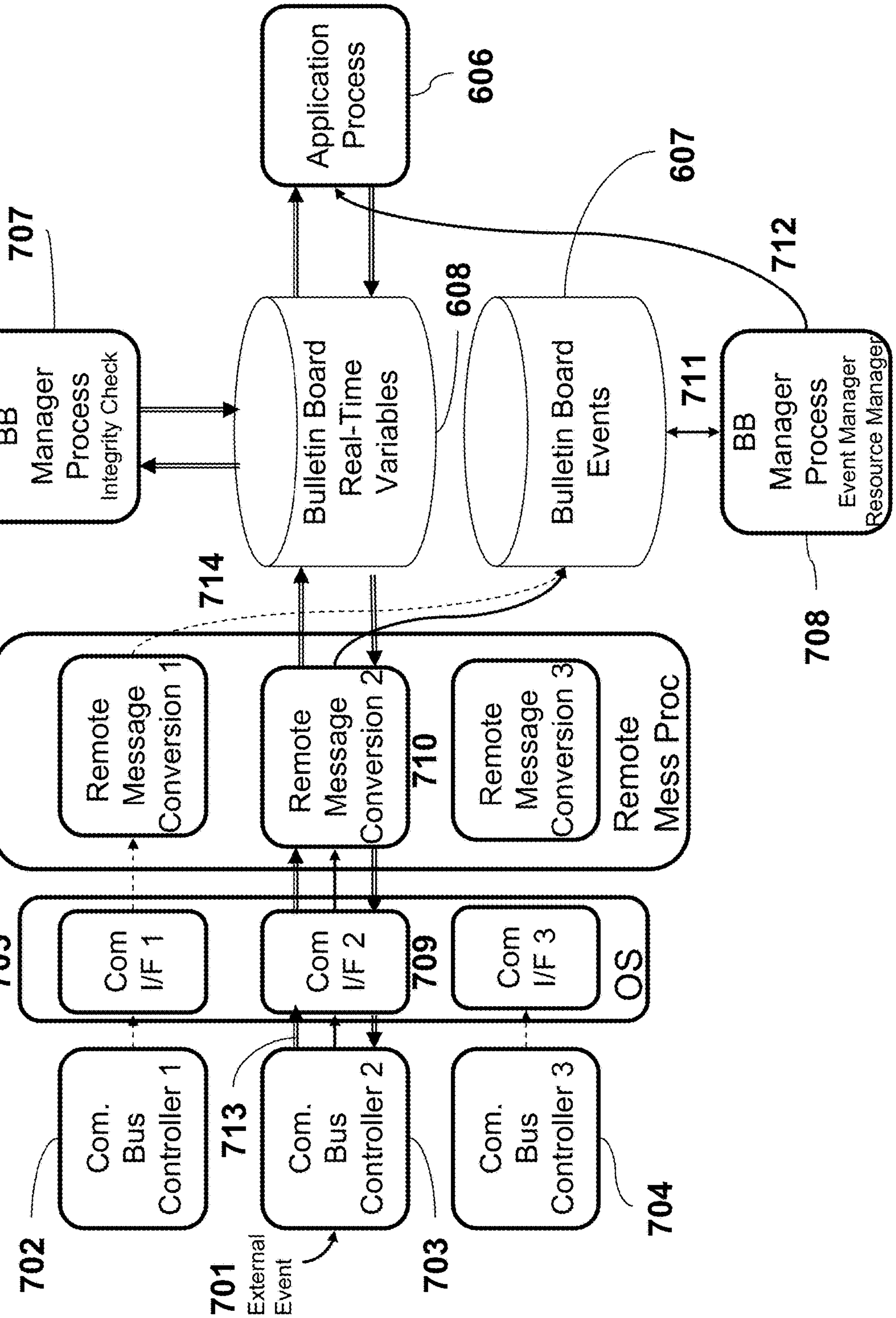

Application Process
606
607
BB Manager Process Integrity Check
707
Bulletin Board Real-Time Variables
608
Bulletin Board Events
711
712
BB Manager Process Event Manager Resource Manager
708
714
706
Remote Message Conversion 1
Remote Message Conversion 2
710
Remote Message Conversion 3
Remote Mess Proc
705
Com I/F 1
Com I/F 2
709
Com I/F 3
OS
713
Com. Bus Controller 1
Com. Bus Controller 2
Com. Bus Controller 3
702
701
External Event
703
704

FIGURE 7

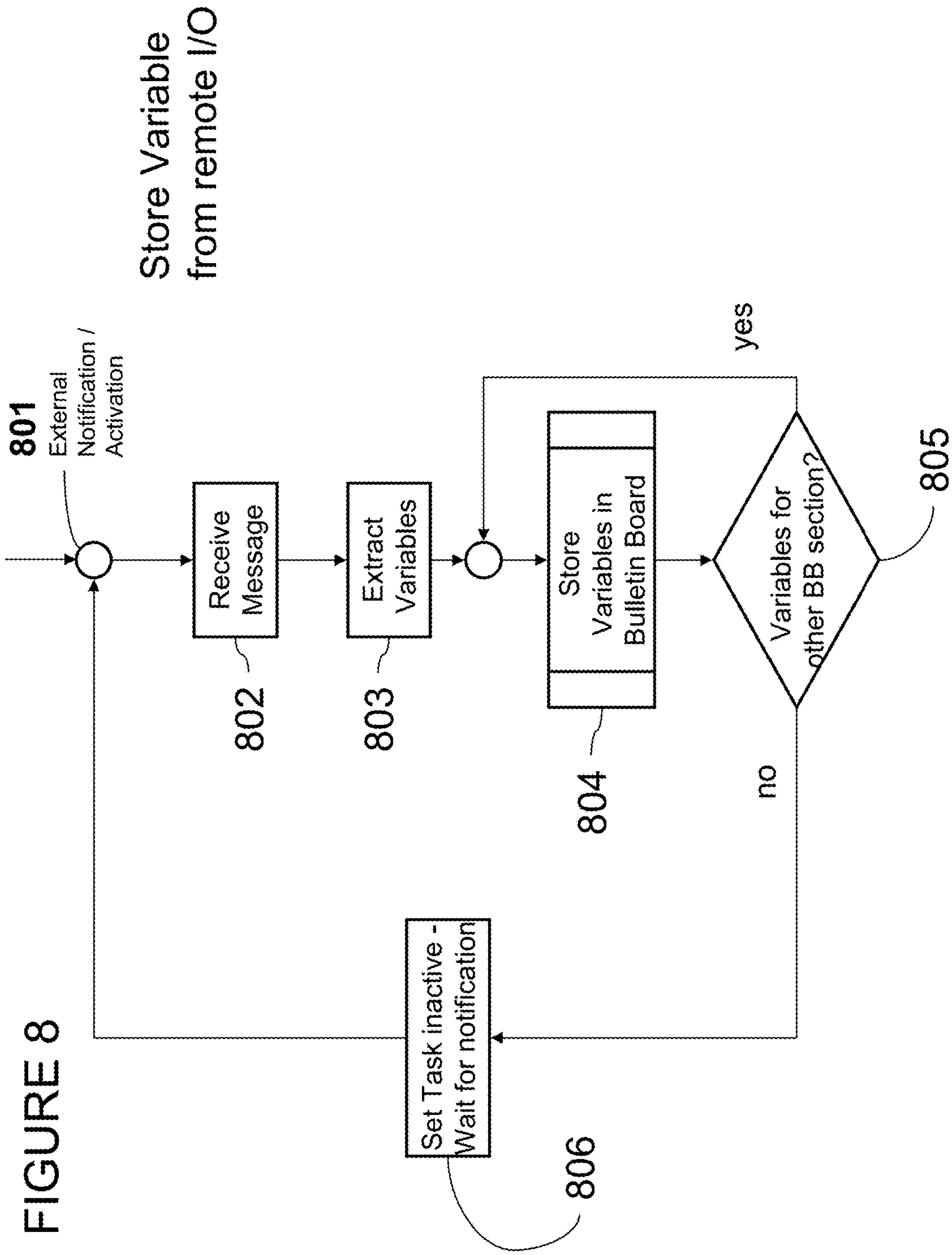

Store Variable
from remote I/O
801
External
Notification /
Activation
Receive
Message
802
Extract
Variables
803
Store
Variables in
Bulletin Board
804
Variables for
other BB section?
805
yes
no
Set Task inactive -
Wait for notification
806

FIGURE 8

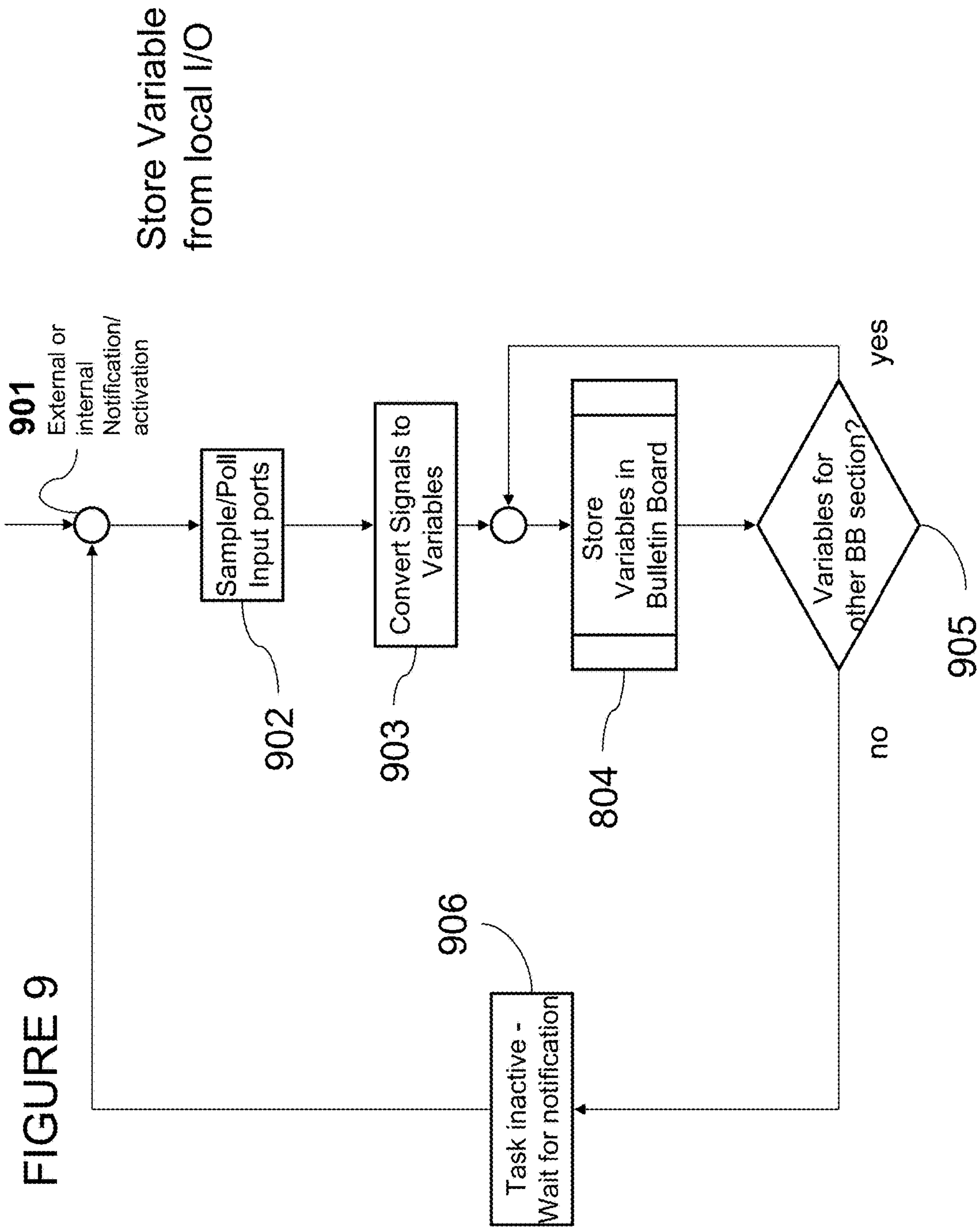

Store Variable
from local I/O
901
External or
internal
Notification/
activation
Sample/Poll
Input ports
902
Convert Signals to
Variables
903
Store
Variables in
Bulletin Board
804
Variables for
other BB section?
905
yes
no
Task inactive -
Wait for notification
906

FIGURE 9

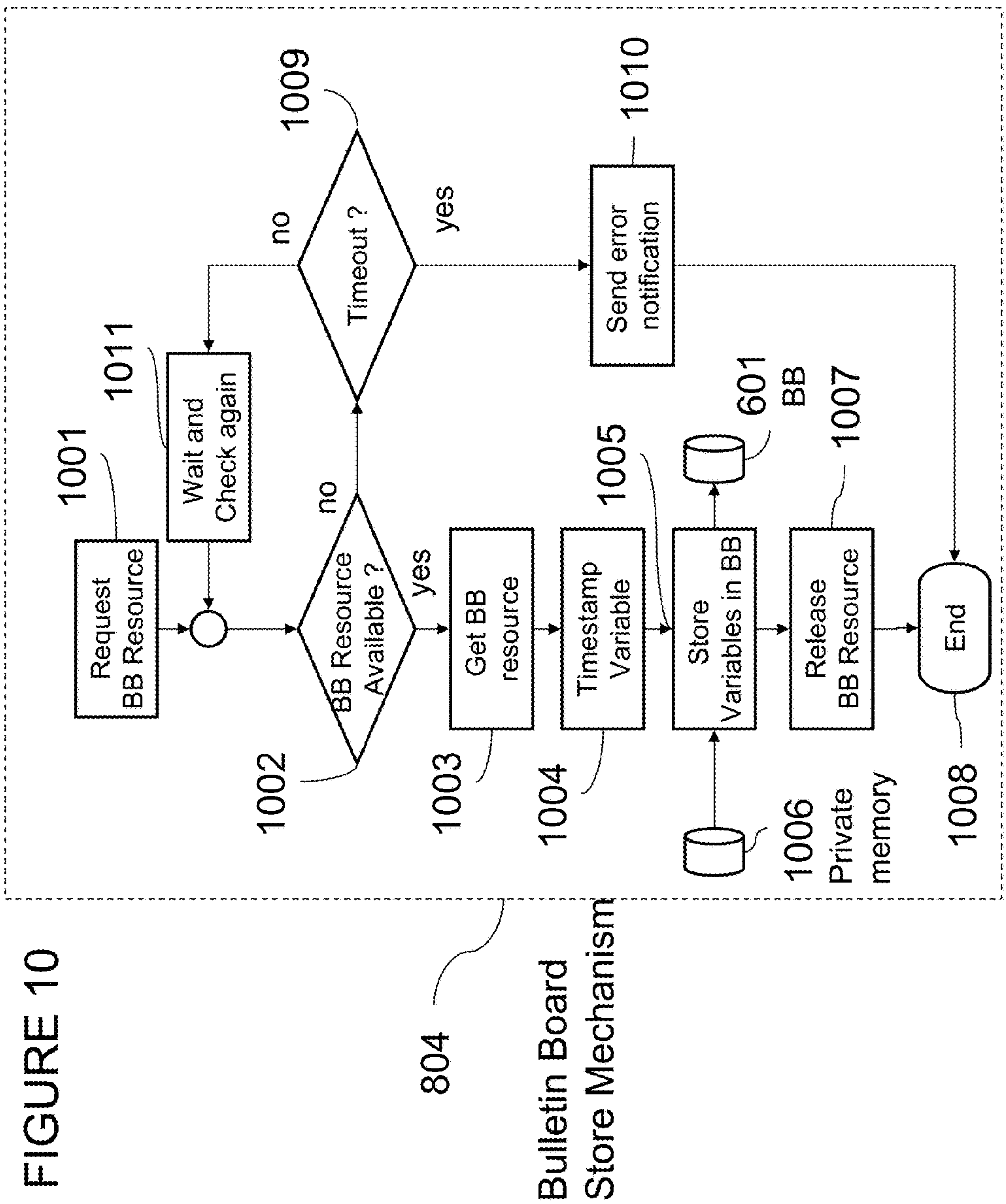

1001
Request BB Resource
1011
Wait and Check again
1002
BB Resource Available ?
no
yes
1009
Timeout ?
no
yes
1010
Send error notification
1003
Get BB resource
1004
Timestamp Variable
1005
Store Variables in BB
601
BB
1006
Private memory
1007
Release BB Resource
1008
End
804
Bulletin Board Store Mechanism

FIGURE 10

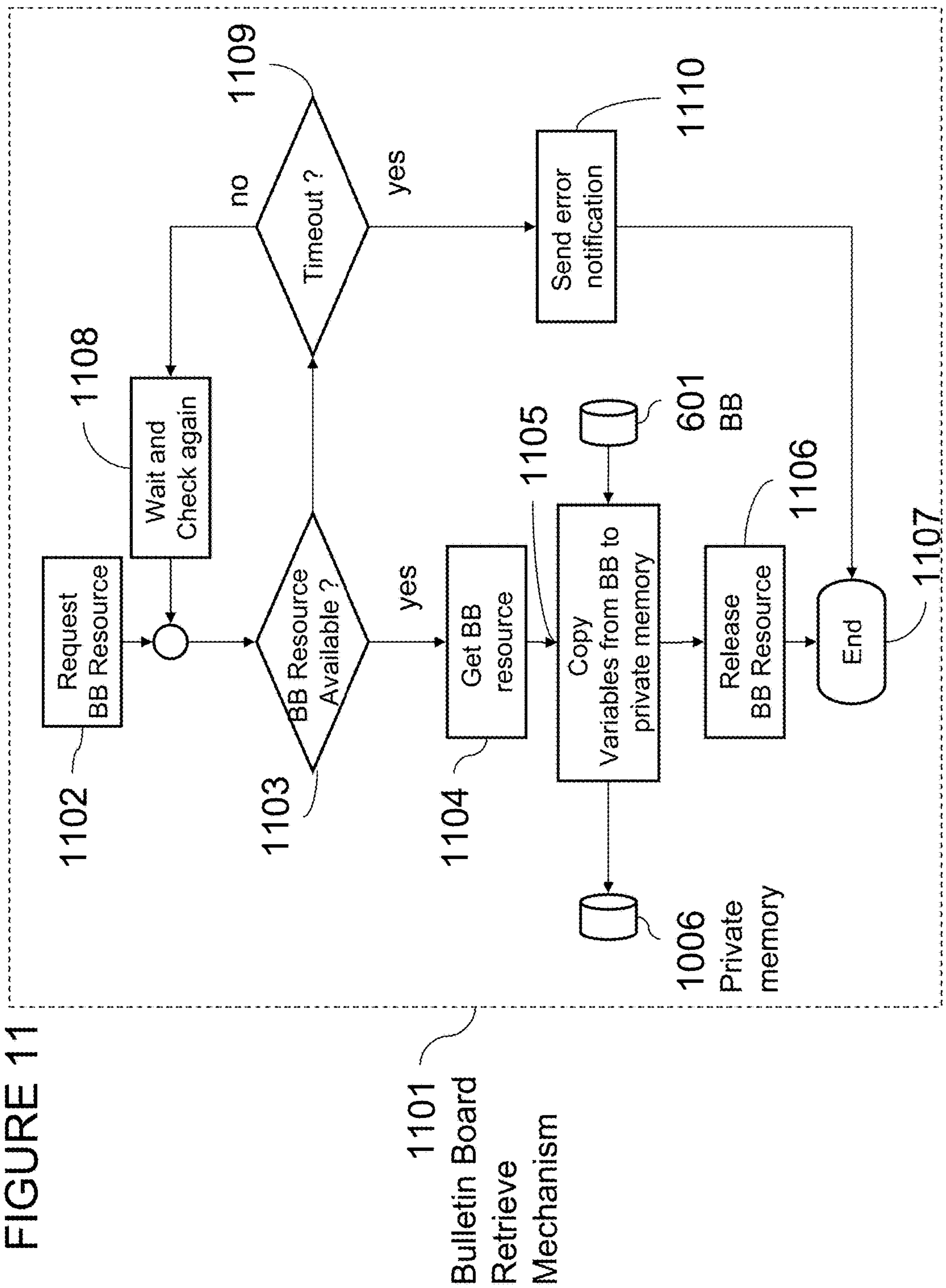

1101
Bulletin Board
Retrieve
Mechanism
1102
Request
BB Resource
1103
BB Resource
Available ?
yes
1104
Get BB
resource
1105
Copy
Variables from BB to
private memory
601
BB
1006
Private
memory
1106
Release
BB Resource
1107
End
1108
Wait and
Check again
1109
Timeout ?
no
yes
1110
Send error
notification

FIGURE 11

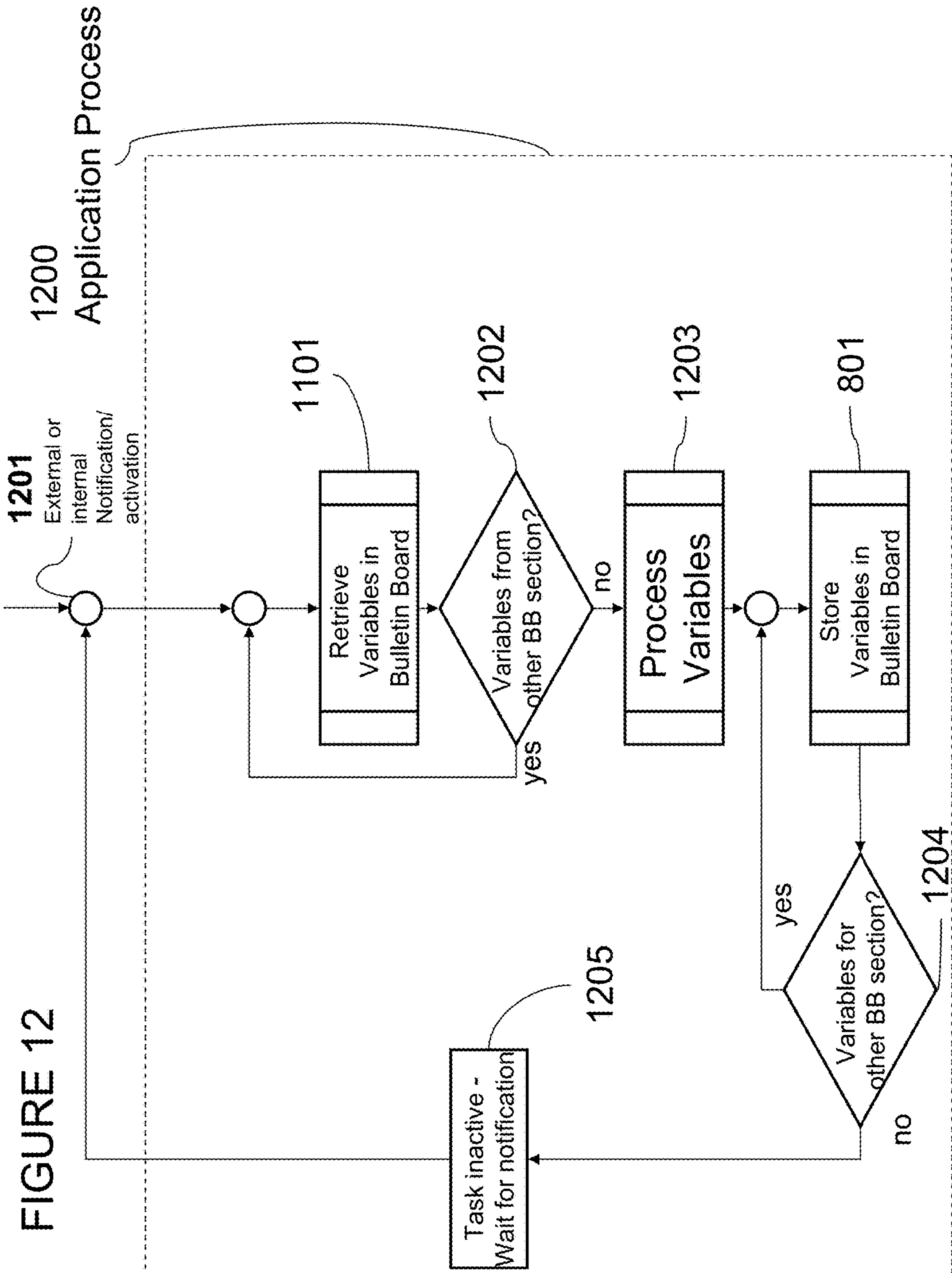

1200
Application Process
1201
External or internal Notification/ activation
1101
Retrieve Variables in Bulletin Board
1202
Variables from other BB section?
yes
no
1203
Process Variables
801
Store Variables in Bulletin Board
1204
Variables for other BB section?
yes
no
1205
Task inactive - Wait for notification

FIGURE 12

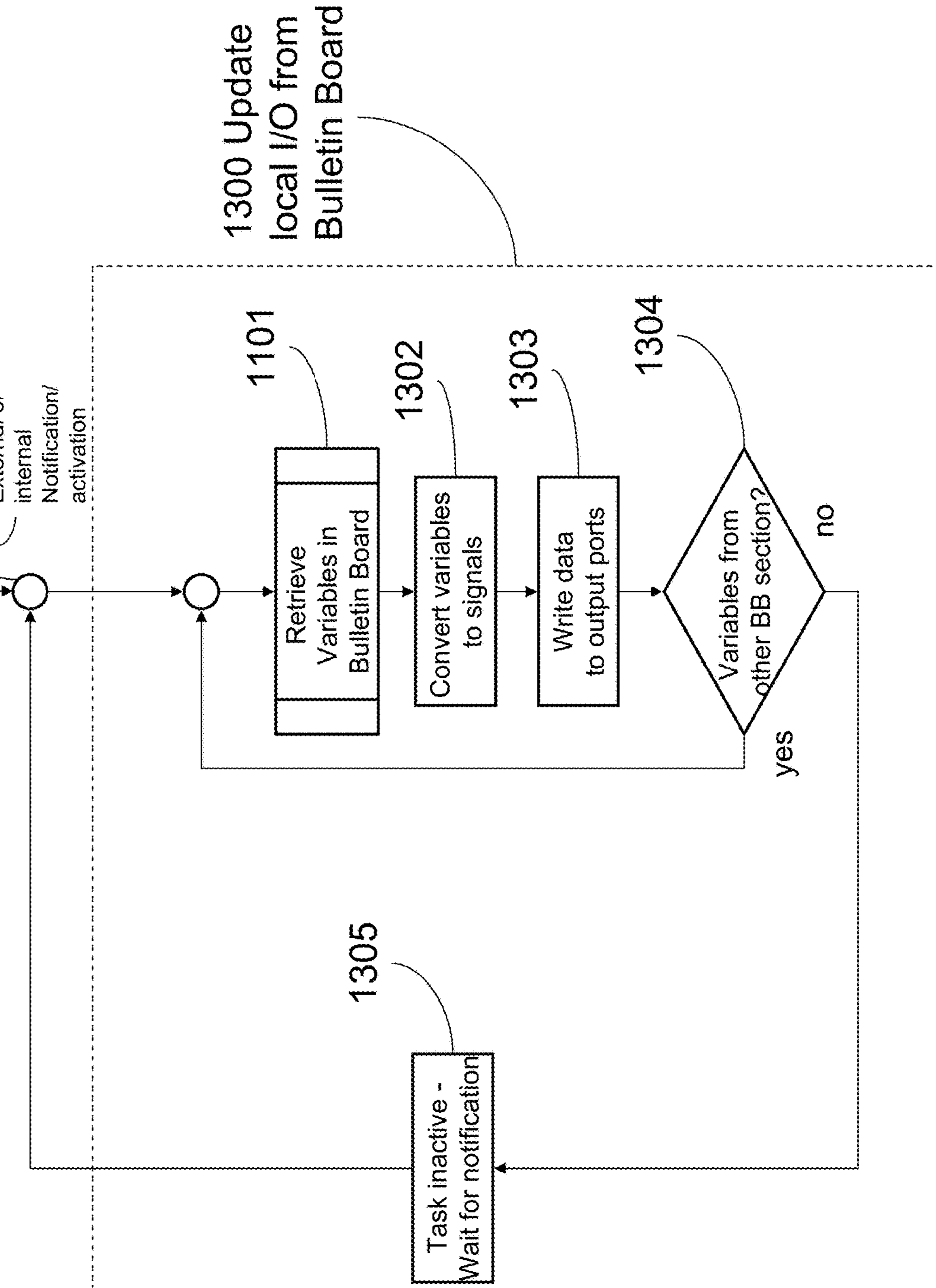

1300 Update local I/O from Bulletin Board
1301 External or internal Notification/ activation
1101
Retrieve Variables in Bulletin Board
1302
Convert variables to signals
1303
Write data to output ports
1304
Variables from other BB section?
yes
no
1305
Task inactive - Wait for notification

FIGURE 13

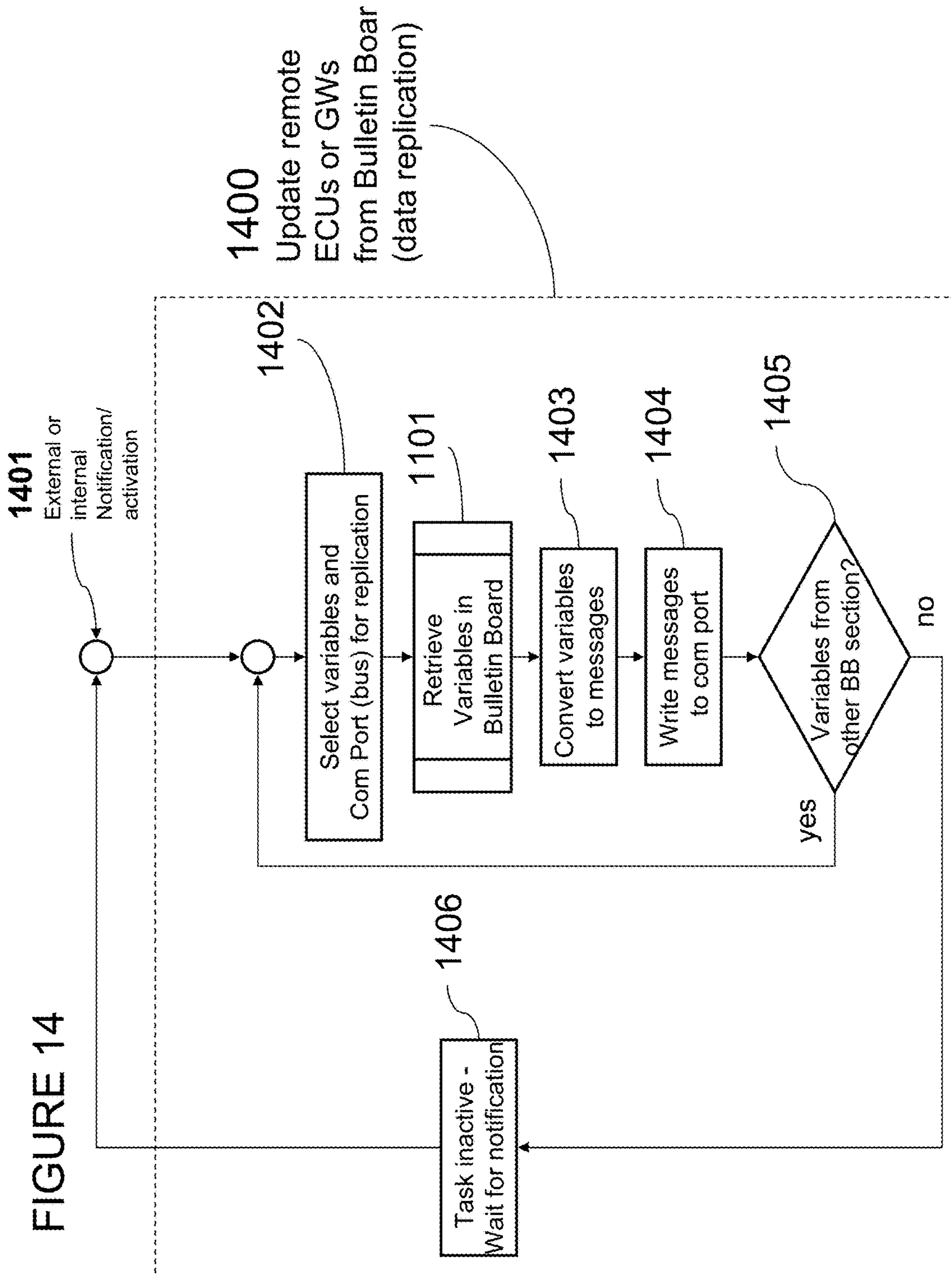

1400
Update remote ECUs or GWs from Bulletin Boar (data replication)
1401
External or internal Notification/ activation
1402
Select variables and Com Port (bus) for replication
1101
Retrieve Variables in Bulletin Board
1403
Convert variables to messages
1404
Write messages to com port
1405
Variables from other BB section?
yes
no
1406
Task inactive - Wait for notification

FIGURE 14

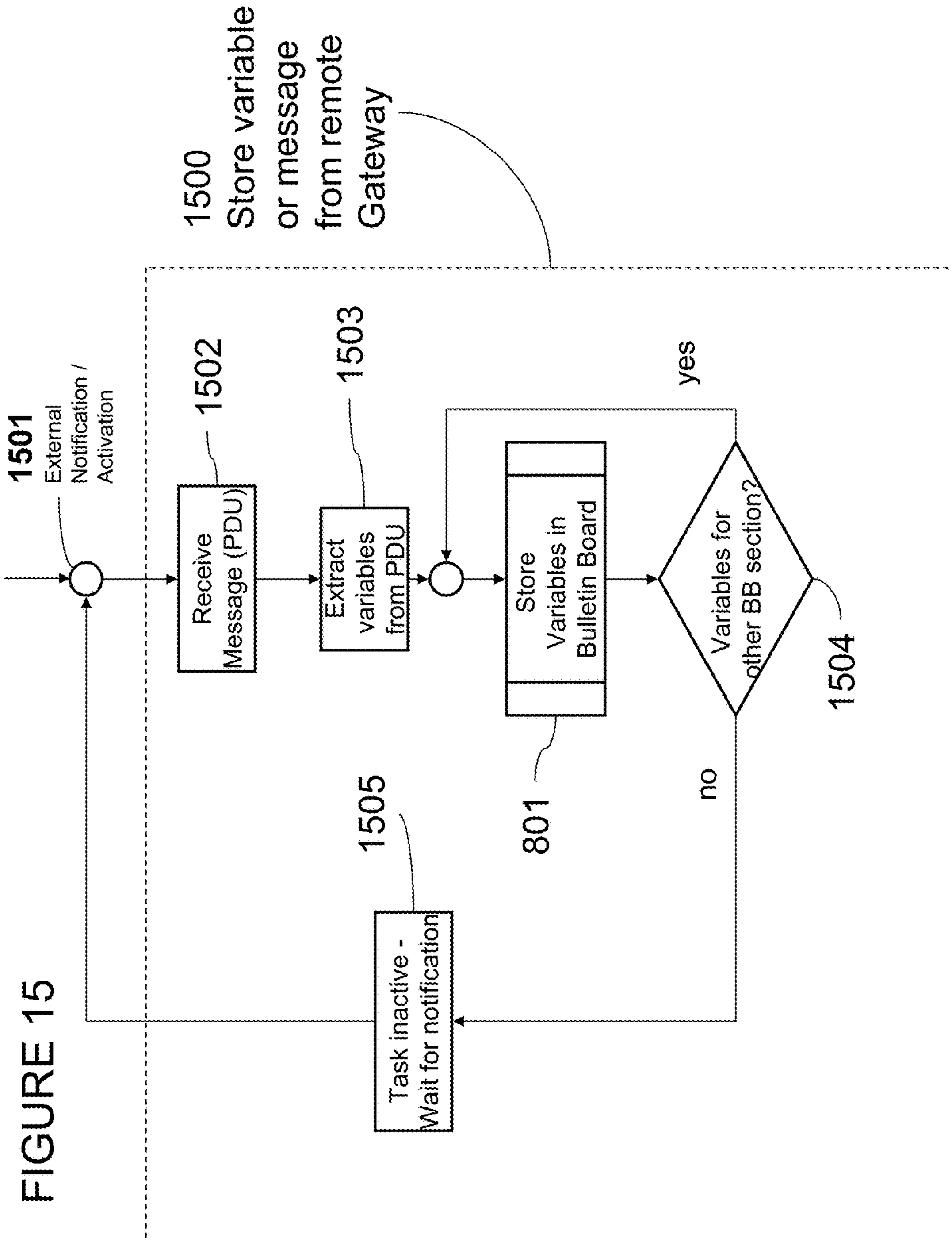

1500
Store variable or message from remote Gateway
1501
External Notification / Activation
1502
Receive Message (PDU)
1503
Extract variables from PDU
Store Variables in Bulletin Board
801
Variables for other BB section?
1504
yes
no
1505
Task inactive - Wait for notification

FIGURE 15

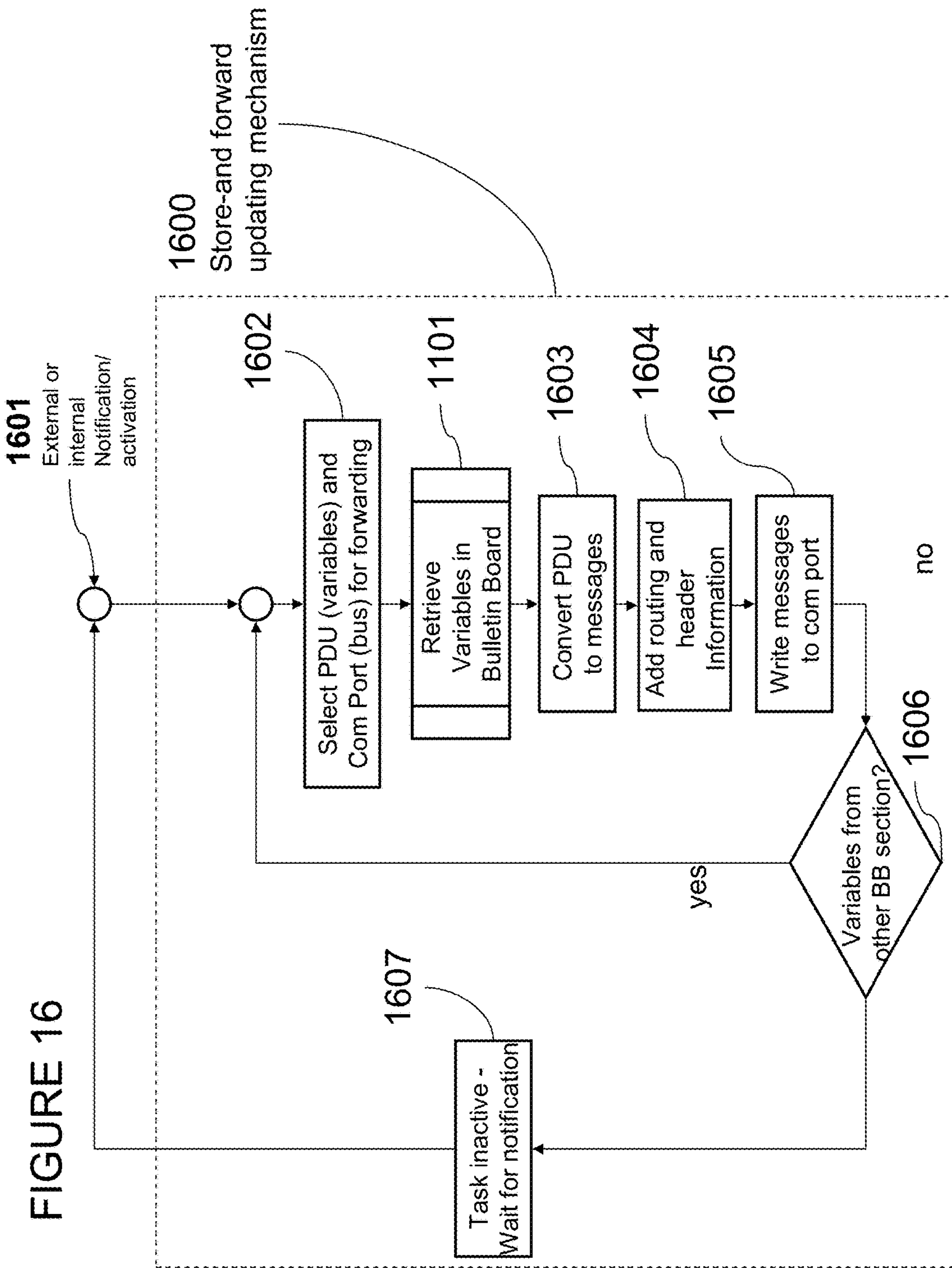

1600
Store-and forward
updating mechanism
1601
External or
internal
Notification/
activation
1602
Select PDU (variables) and
Com Port (bus) for forwarding
1101
Retrieve
Variables in
Bulletin Board
1603
Convert PDU
to messages
1604
Add routing and
header
Information
1605
Write messages
to com port
1606
Variables from
other BB section?
yes
no
1607
Task inactive -
Wait for notification

FIGURE 16

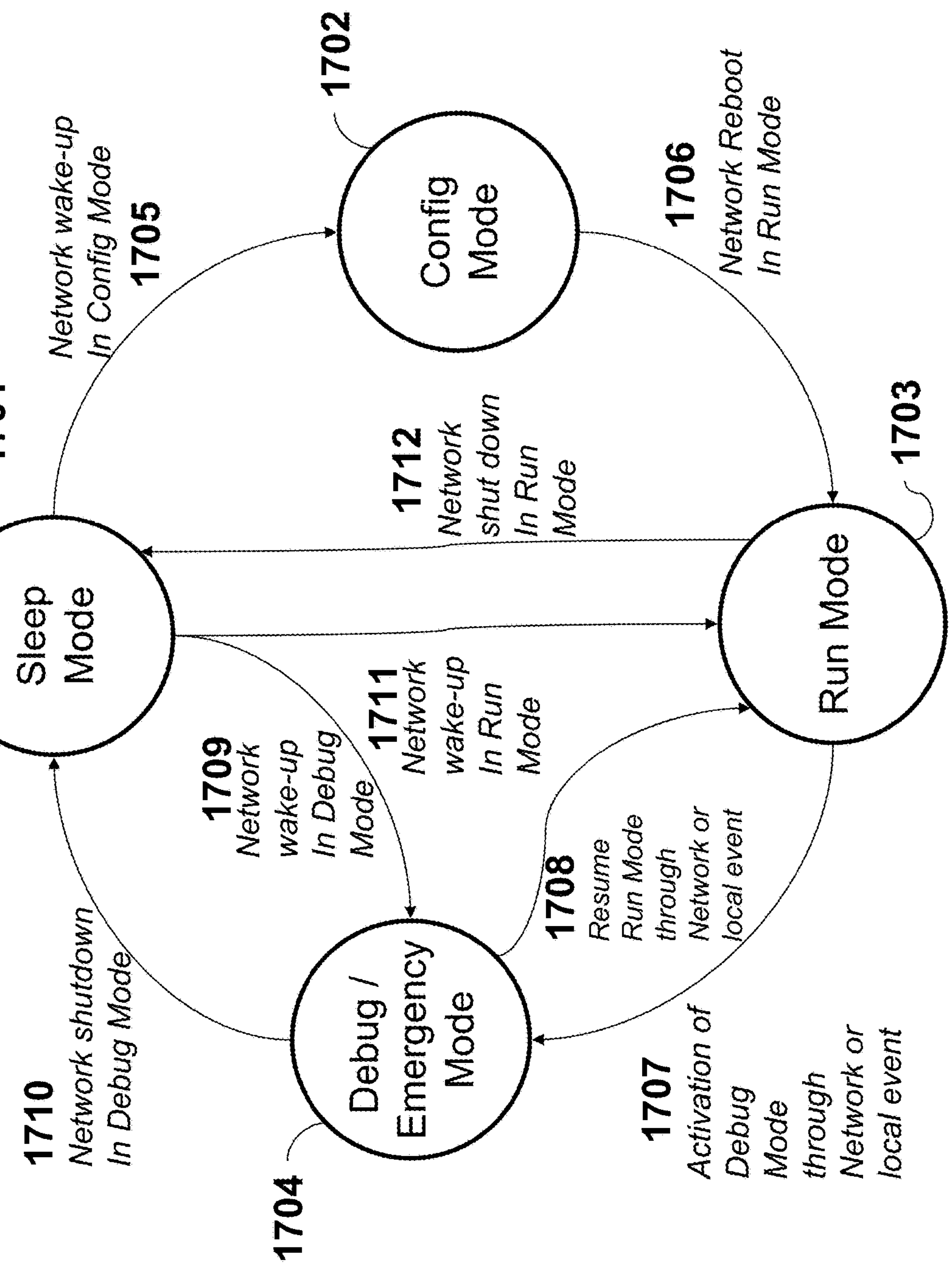

1701
Sleep Mode
1702
Config Mode
1703
Run Mode
1704
Debug / Emergency Mode
1705
Network wake-up In Config Mode
1706
Network Reboot In Run Mode
1707
Activation of Debug Mode through Network or local event
1708
Resume Run Mode through Network or local event
1709
Network wake-up In Debug Mode
1710
Network shutdown In Debug Mode
1711
Network wake-up In Run Mode
1712
Network shut down In Run Mode

FIGURE 17

SYSTEM, METHOD AND COMPUTER PROGRAM PRODUCT FOR SHARING INFORMATION IN A DISTRIBUTED FRAMEWORK

RELATED APPLICATION(S)

This application is a continuation of U.S. patent application Ser. No. 16/042,159 filed Jul. 23, 2018, which is a continuation of U.S. patent application Ser. No. 15/919,201 filed Mar. 12, 2018, now U.S. Pat. No. 10,031,790, which is a continuation of U.S. patent application Ser. No. 15/405,110 filed Jan. 12, 2017, now U.S. Pat. No. 10,002,036, which is continuation of U.S. patent application Ser. No. 14/011,705 filed Aug. 27, 2013, now U.S. Pat. No. 9,575,817, which is a continuation of U.S. patent application Ser. No. 13/531,319 filed Jun. 22, 2012, now U.S. Pat. No. 8,566,843, which is a continuation of U.S. patent application Ser. No. 12/182,570 filed Jul. 30, 2008, now U.S. Pat. No. 8,209,705, which is a continuation of U.S. patent application Ser. No. 10/737,690 filed Dec. 15, 2003, now U.S. Pat. No. 7,802,263, which, in turn, claims priority under 35 U.S.C. § 119 based on U.S. Provisional Application No. 60/434,018 filed Dec. 17, 2002, all of which are incorporated herein by reference.

FIELD AND BACKGROUND OF THE INVENTION

The present invention relates to the field of distributed control and monitoring systems that may include certain temporal behavior.

Such technology may optionally apply to electronic vehicle communication and control systems, real-time monitoring systems, industrial automation and control systems, as well as any other desired system.

SUMMARY OF THE INVENTION

A system, method and computer program product are provided for sharing information in an automobile vehicle comprising: receiving information in the form of a packet data unit representing datum information carried by an overall message from a first physical network selected from the group consisting of FlexRay, Controller Area Network, Local Interconnect Network and Media Oriented Systems Transport; in response to the receipt of the information, issuing a storage resource request in connection with a storage resource; determining whether the storage resource is available for storing the information; determining whether a threshold has been reached in association with the storage resource request; in the event the storage resource is not available and the threshold associated with the storage resource request has not been reached, issuing another storage resource request in connection with the storage resource; in the event the storage resource is available, storing the information in the storage resource; and sharing the stored information with at least one of a plurality of heterogeneous processes including at least one process associated with a second physical network selected from the group consisting of FlexRay, Controller Area Network, Local Interconnect Network and Media Oriented Systems Transport, utilizing a network protocol different from a protocol of the first physical network.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** is a block diagram of an embodiment of a system of one embodiment;

FIG. **2** is a block diagram generally depicting an embodiment of an ECU as part of the system illustrated in FIG. **1**;

FIG. **3** is a block diagram generally depicting an embodiment of a Gateway device as part of the system illustrated in FIG. **1**;

FIG. **4** is a block diagram of an embodiment of the software architecture assumed for one embodiment.

FIG. **5** is a block diagram of an embodiment of the middleware that contains the methods of one embodiment.

FIG. **6** is a block diagram of an embodiment of the bulletin board that describes the process interaction of one embodiment.

FIG. **7** is a block diagram of an embodiment of the bulletin board that describes the process interaction with multiple external communication buses as part of one embodiment.

FIG. **8** is a flow chart diagram of an embodiment of the variable store from remote I/O method of one embodiment.

FIG. **9** is a flow chart diagram of an embodiment of the variable store from local I/O method of one embodiment.

FIG. **10** is a flow chart diagram of an embodiment of the variable method of one embodiment.

FIG. **11** is a flow chart diagram of an embodiment of the variable retrieve method of one embodiment.

FIG. **12** is a flow chart diagram of an embodiment of the application process using the method of one embodiment.

FIG. **13** is a flow chart diagram of an embodiment of the local I/O update from bulletin board method of one embodiment.

FIG. **14** is a flow chart diagram of an embodiment of the variable replication method of one embodiment.

FIG. **15** is a flow chart diagram of an embodiment of the message store from remote gateway method of one embodiment.

FIG. **16** is a flow chart diagram of an embodiment of the message forward to remote ECU or Gateway method of one embodiment.

FIG. **17** is a state transition diagram of an embodiment of the mode switching method of one embodiment.

DETAILED DESCRIPTION

FIG. **1** is a block diagram generally depicting elements of an embodiment of the present distributed embedded communication and computing system. The system architecture may be situated in automotive electronics or industrial control and monitoring systems. In an automotive environment, the various Electronic Control Units (ECUs, **102**) control complex applications such as engine control, brake control, or diagnostics. They are either connected to sensors and actuators via discrete links or simple standard functions such as sensors and actuators are organized into separate sub networks.

These complex functions such as braking, engine-control, etc. are then grouped into the backbone system functions for the car, such as body control, power train and chassis. The backbone also includes the vehicle's high level functions such as diagnostics, telematics and entertainment systems.

Therefore the system is typically hierarchically organized and includes a variety of gateways (**101**,**104**,**105**), which relay messages up and down through the system layers. Each layer may contain multiple electronic control units (ECU, **102**) that are connected through wired serial multiplexing bus-systems such as Controller Area Network (CAN or ISO11898), Flexray, LIN, J1850, J1708, MOST, IEEE1394, and other similar serial multiplexing buses or through wireless multiplexing systems such as IEEE802.11, IEEE802.15, Bluetooth, Zigbee, or similar other wireless links.

Typically, functions provided by an ECU (**102**) are bound to hard real-time temporal behavior. In the context of the present description, real-time may include any response time that may be measured in milli- or microseconds, and/or is less than 1 second.

The ECU may receive a set of real-time input variables from local sensors (**108**), which are connected via discrete signal lines (**113**), or from networked sensors (**106**), which are connected through a multiplexing bus-system (**112**). The ECU may also share variables with other ECUs (**102**) that are either connected on the same physical multiplexing bus or that it can reach through a gateway (**101**,**103**,**104**).

Then the ECU (**102**) processes the input variables and generates a set of output variables that are either shared with other ECUs (**102**) as described above, or which are output to local actuators (**109**), which are connected via discrete signal lines (**113**), or to networked actuators, which are connected through a multiplexing bus (**112**). ECUs (**102**) typically share information with devices that are connected on the same physical multiplexing system. This method of information sharing is called horizontal information sharing in a hierarchical system. Gateways (**101**,**103**,**104**) link multiple physical multiplexing systems together. In the context of the present description, such information may include data, a signal, and/or anything else capable of being stored and shared.

The highest level in the hierarchical system is the system level. The system level gateway (**101**) may be connected to ECUs on the system level multiplexing bus (**117**), to subsequent gateways (**103**) that also link to subsequent communication buses (**110**), and to external components (**120**) that may contain diagnostics devices (**121**), development tools (**122**), other add-on devices (**123**) or other instances of distributed embedded communication and computing systems (**100**). In addition, the system gateway (**101**) may also be connected to an external gateway (**131**) that may link the system to a remote device (**132**) through wireless or wired wide-area-networks such as the Internet, using standard protocols such as UDP/IP, TCP/IP,RTP,HTTP, SOAP, JAVA, etc. or nonstandard proprietary protocols.

Subsequent to the system level may be several layers of groups and subgroups that are link to the higher levels via gateways (**101**,**103**,**104**,**105**).

During the design-time of the system, not all ECUs may exist. Therefore, the development tool (**122**) may provide a plug-in component or virtual ECU/GW (**115**) that directly links into the wired multiplexing bus or wireless network (**110**) and also allows for separate control functions via a tool-link (**116**).

The block diagram in FIG. **2** depicts the detailed elements within a generic ECU (**200**) that is one embodiment of ECU (**102**). The ECU (**200**) typically contains a micro-processor (**201**), volatile memory (**204**) such as RAM, S-RAM or similar, non-volatile memory (**203**) such as EEPROM, FLASH, etc., a real time clock for internal timing of processes (**205**), a watchdog (**206**) to maintain the health of the system, one or more communication bus controllers (**207**) with associated drivers (**208**), digital I/O (**209**) with line drivers (**210**), and analog I/O (**211**) with associated analog signal conditioning (**212**).

In an alternate embodiment, the ECU (**200**) may also contain a wireless communication controller (**311**) and a RF-Front-end (**312**) as outlined in FIG. **3**. The software (**202**) can either be stored in local non-volatile memory (**203**) or partially downloaded via the communication link (**207**,**208**) and stored in the volatile memory. The software is then executed in the microprocessor (**201**).

The block diagram FIG. **3** depicts the detailed elements within a generic gateway (**300**) that is one embodiment of Gateway (**101**,**103**,**104**,**105**) in FIG. **1**.

FIG. **4** outlines one embodiment of the software architecture in an embedded system. The hardware abstraction layer (**405**) allows the system developer to adapt a standard operating system to a specific hardware as used in an ECU (**200**) or gateway (**300**). The hardware abstraction layer (**405**) adapts the real-time operating system (**403**) and the device drivers (**404**) to a specific hardware implementation.

One embodiment includes the middleware (**402**) that has direct access to the real-time operating system (**403**), the device drivers (**404**) and the hardware abstraction layer (**405**). The middleware isolates the application from input/output functions and allows multiple applications to share common variables locally. In addition, the middleware lets applications share variables with remote applications/processes. In the context of the present description, a process may refer to any hardware and/or software operation, etc.

In one embodiment, the middleware can directly interface with the input/output mechanisms of the hardware without utilizing an operating system (**403**) or hardware abstraction layer (**405**).

Another embodiment of the middleware utilizes a preemptive multitasking operating system with explicit control of resources. In an alternate embodiment, the middleware can be built with a static multitasking scheme with implicit resource management or be part of a single task system.

Referring now to FIG. **5**, the middleware (**402**) contains the bulletin board manager (**501**), a local signal communication interface (**503**), a remote message communication interface (**504**), and an application programming interface (**502**). The application interface (**502**) provides methods and data interfaces to a plurality of applications. In one embodiment, the application interface is an object library that can be linked to an application at design time.

The bulletin board manager (**501**) contains an upgrade and configuration manager (**507**), an event manager (**505**), a data access manager (**508**), and a data integrity watchdog (**506**). The upgrade and configuration manager (**507**) is necessary to configure the data structure of the bulletin board and to make executable code available to individual processing nodes. In the context of the present description, the bulletin board may refer to any database that enables users to send and/or read electronic messages, files, and/or other data that are of general interest and/or addressed to no particular person/process.

The access manager provides access control mechanisms for the code update and configuration mode. It also may control access rights for individual applications at execution time in the run mode.

The event manager (**505**) captures input-output events as variables and generates new events, flags, or signals based on operations on state variables in the bulletin board. Such operations may include test of maximum values, the occurrence of logically combined events, the result of an integrity check, or events and signals that are created based on any other logical or arithmetic computation on the state variables that are stored in the bulletin board. The actual processing of data and manipulation of data may be done in the application that uses the middleware (**402**). The data integrity watchdog analyses the stored state variables for its integrity and generates events or flags if any problem occurs.

The local signal communication interface (**503**) interfaces with the local discrete input/output hardware to update the bulletin board with new variables and to update the input/output interfaces with the state variables from the bulletin board. It also converts state variables to input/output signals and input/output signals to state variables that can be stored in the bulletin board. The conversion process may contain scaling of signals as well as offset compensation. Typically this processing helps to convert I/O signals that are measured in Volt to a physical entity and vice versa. The communication with the local discrete input output system can be triggered by events or signals can be sampled time-triggered based on a cyclic global or local time base.

The remote message communication interface (**504**) interfaces to serial multiplexing interfaces (buses) that are connected to the specific processing node (ECU or Gateway). It extracts variables from a plurality of messaging protocols and stores them in the database. It also replicates local bulletin-board state variables to the associated processing nodes by composing the appropriate messages for each communication link. The message transfer can be initiated triggered by a bus event, by a local event, or by a time-triggered mechanism that uses a cyclic local or global time base.

FIG. **6** shows the concept of an extended bulletin board or an embedded real-time database (**601**). In this embodiment the ECU (**102**) or the Gateway (**101**) hosts one or multiple bulletin boards with relational links between the variables in the bulletin boards. The relations are defined by data processing functions that the gateway can operate on bulletin boards to obtain new information that can be stored in yet another bulletin board.

The bulletin board (**601**) may contain but is not limited to events (**607**), real-time variables (**608**), diagnostics data (**609**), configuration parameters (**610**), and firmware (**611**) to upgrade individual components of the executable code or the entire software of a processing node. Each type of information may include one or more sections so that individual processes are not blocked if they access separate sections of data.

The memory of the bulletin board is subdivided into areas that nodes on each external network can read from and write into and other areas that an external network may only read from. The data contained in the bulletin board may be stored in volatile or non-volatile memory. Each data entry may consist of one value or an array of values that also may represent a time series.

In one embodiment, each application process (**603**), local signal communication process (**605**), remote message communication process, and the bulletin manager (**602**) can individually access the bulletin board using operating system functions for resource management that may include semaphores, events, signals, call-back routines, flags, etc. in an alternate embodiment of the system the bulletin-board manager controls all interaction with the bulletin-board and all applications have to pass data to the bulletin-board manager. This approach simplifies the interaction with the bulletin board, but adds delay time and jitter to the state variables.

At design time, various hierarchies of memory management can be applied. In practice it is more efficient to allow each sub network and subsystem to place system variable data into local bulletin boards. This is because many system variables are primarily used only within their subsystem or sub network. By placing local information in a shared memory (local bulletin board), it can be used by multiple processes on this processor node. A group bulletin board allows devices on a sub-network to share information with a minimum of network traffic. A system bulletin board allows access to system-wide variables and information.

FIG. **7** illustrates the logical architecture of the interconnection between three heterogeneous network controllers (**702**, **703**, **704**), the associate Operating System interfaces (**705**), the remote message communication process (**706**), the bulletin board (**608**), and the application process (**606**). The connection to each communication controller is fundamentally implemented at the physical interface (the wire, fiber or electromagnetic wireless interface). Each of the higher level layers (data link, network, etc) in the communication interface (**705**) deals with specific features of the individual communication process. In practice these layers are typically represented in a message by "header" bits that contain information about that layer of the network being used to send the message.

Using this model, each communicated message may be processed at each layer to remove (and use) the associated header information for that level. Once all layers are processed the remaining packet data unit (PDU) represents the datum or core information carried by the overall message. In one embodiment, each communication controller has an associated communication interface and an associated remote message conversion mechanism. For instance communication bus controller **2** (**703**) has an associated communication interface **2** (**709**), and an associated remote message conversion **2** (**710**).

This arrangement allows the remote message process (**706**) to directly access information at the data link layer and interface it with the bulletin board. A network layer is not necessary. The remote message communication process (**706**) has a multi-network access interface (essentially a processing capability that can interpret and apply the header information for a variety of networks) and the bulletin board read/write memory access. Now, the individual processing nodes do not need to know about the existence of multiple networks. Each variable can be accessed from all connected physical networks in their proprietary format. Thus the normalization of the information has only to be handled at the gateway through replication of stored data to multiple attached networks.

Continuing with FIG. **7**, the communication procedure is described. In the given example, an external event (**701**) on communication controller **2** (**703**) triggers the operating system to notify the remote message communication process (**706**) that data is available. The notification may be a flag, a call-back routine, an event, or any other operating signal. The associated remote message conversion method **2** (**710**) extracts the data (e.g. real time variables) from the message PDU and stores the data in the bulletin board (**608**). It may also store the associated event as variable in the bulletin board and signal the bulletin-board event manager that new data is available.

The bulletin event manager then notifies the application process (**606**) with the appropriate mechanism. In addition, the event manager may trigger the sampling of local signals using the local signal communication process (**605**) described in FIG. **6**. Finally the bulletin event manager may trigger the bulletin board manager (**707**) to perform integrity checks or generate additional events based on the change of the state variables.

One embodiment provides a new mechanism for creating an information interconnection between two or more heterogeneous communication networks. In the context of the present description, heterogeneous networks may refer to any different communication networks with at least one aspect that is different.

The approach uses a common, or shared storage system that is connected to all of the system networks through network interfaces. A critically important feature of the bulletin board approach is that the complexity of the bulletin board grows linearly with the number of networks (as opposed to as N(N−1) for the gateway approach), and in one-to-many situations the number of message transformations is half that of the standard networking approach.

In an alternate embodiment of the remote message communication process (**706**) any remote process can access data via a single network interface. This approach requires a network layer in each processing node and therefore adds overhead to communications. To communicate between two heterogeneous networks, this process may then be repeated in reverse by adding back the header information for the various layers of the second network, and eventually putting the message onto the second network's physical link. The remote message communication manager (**706**) then can be simplified to only one message assembly and disassembly mechanism.

FIGS. **8-17** illustrate the method of operation of one embodiment of the present system, and also refer to aspects and elements one embodiment shown in FIGS. **1** through **7**.

FIG. **8** details the remote messaging process (**706**) described in FIG. **7**. Referring now to FIG. **8**, the core process of storing data from remote processes that are communicated through multiplexed communication links, into the bulletin board is described. An external notification or task activation starts the process. Then a message (**802**) is received from the operating system layer.

In an alternate embodiment, the message is directly copied form the input register of the communication controller. Then the process extracts variables from the message. Additional signal adaptation may be necessary. The sub-process **804** stores the variables in the bulletin board. If the process only updates one section of the bulletin board it waits for the next message notification (**806**). If variables in multiple sections need to be updated, the process repeats (**804**).

FIG. **9** shows the data update from local input/output peripherals. The process starts with an internal or external notification or task activation. Typically this process is repeated cyclic triggered by an internal or external real-time clock. When the process is activated, it samples or polls the local input ports that may include analog and digital signals (**902**). Then it converts these signals to real-time variables by using the conversion parameters stored in the bulletin board. The signal conditioning parameters van either be defined at design time or adaptively updated by the application process. Then the process stored the new state variables in the bulletin board using the sub-process (**804**) described above.

FIG. **10** describes the bulletin board store procedure (**804**) in more detail. Before new data can be stored in the bulletin board, the procedure has to request the access right to the common resource, a section of the non-volatile or volatile memory, from the operating system (**1001**). This is called explicit resource management.

If the resource is available, the process gets the resource. If the resource is not available, it may try it again after a waiting period (**1011**) until the resource is available. After a certain time has elapsed (**1009**) beyond a configurable threshold, the temporal behavior of the state variable can't be captured any longer and the middle-ware may send an error notification to the associated process.

After reserving the resource (**1003**), the bulletin board store mechanism (**804**) timestamps the state variable for future temporal reference (**1004**). Then, the bulletin board store procedure (**804**) copies the variables or parameters from its private memory (**1006**) to the shared bulletin-board memory (**601**). Then it releases the bulletin board resource.

In an alternate embodiment, the bulletin board store procedure (**804**) has exclusive access to the bulletin board (**601**) and does not need operations **1002**, **1003**, **1007**, **1009**, **1010**, and **1011** because the resource access is realized through implicit resource management. This can be achieved with either static task scheduling or by allowing only the bulletin board store procedure (**804**) to access the bulletin board (**601**).

FIG. **11** describes the bulletin board retrieve procedure (**1101**) in more detail. Before data can be retrieved from the bulletin board, the procedure has to request the access right to the common resource, a section of the non-volatile or volatile memory, from the operating system (**1102**).

If the resource is available, the process gets the resource. If the resource is not available, it may try it again after a waiting period (**1108**) until the resource is available. After a certain time has elapsed (**1109**) beyond a configurable threshold, the temporal behavior of the state variable can't be captured any longer and the middle-ware may send an error notification to the associated process (**1110**).

After reserving the resource (**1104**), the bulletin board retrieve mechanism (**1101**) copies the variables or parameters from the shared bulletin-board memory (**601**) to its private memory (**1006**). Then, it releases the bulletin board resource. In an alternate embodiment the bulletin board retrieve procedure (**1101**) has exclusive access to the bulletin board (**601**) and does not need operations **1103**, **1104**, **1106**, **1108**, **1109**, and **1110**. Because the resource access is realized through implicit resource management, this can be achieved with either static task scheduling or by allowing only the bulletin board retrieve procedure (**1101**) to access the bulletin board (**601**).

Referring to FIG. **12**, the application process (**1200**) utilizes the bulletin board retrieve mechanism (**1101**) to access all parameters, events, and real-time variables from the bulletin board. Thus the application process is decoupled from the temporal behavior of the input/output variables and can be triggered by a plurality of events (**1201**).

The application process may retrieve one or multiple sets of variables stored in a plurality of memory sections. Then the application process processes the variables (**1203**) with its method. Because the method is not tied to the location of the input/output variables, the application process can be moved or replicated to a plurality of processing nodes (ECUs or Gateways). After processing the input variables and generating a set of output variables, the application process uses the bulletin board store method (**801**) to update one or a plurality of memory sections in the bulletin board. If the application process is a cyclic procedure, it waits until the next activation occurs (**1205**).

Continuing with FIG. **13**, the update local I/O from bulletin board process (**1300**) utilizes the bulletin board retrieve mechanism (**1101**) to access real-time variables from the bulletin board and convert them to output signals (**1302**) that can be written to the output port (**1303**). The I/O update process may retrieve one or multiple sets of variables stored in a plurality of memory sections. If the I/O update process is a cyclic procedure, it waits until the next activation occurs (**1305**).

FIG. **14** describes the data replication process (**1400**). This process can be triggered by a plurality of notification mechanisms, such as events, alarm signals, internal and external timers, and flags set in the bulletin board. It then selects a subset of variables to be replicated and a communication port (**1402**). Next it retrieves the variables from the bulletin board with mechanism (**1401**) and assembles the messages for the specific communication link (**1403**). The message may include an address or identification number for all bulletin boards and associated processing nodes (ECUs and Gateways).

Finally, it writes the messages to the communication port (**1404**). In an alternate embodiment, it handles the messages to the associated interface procedure of the operating system. Then it repeats the procedure, until all variables are updated on all communication ports. If the data replication process is a cyclic procedure, it waits until the next activation occurs (**1405**).

Referring now to FIG. **15**, the store message from remote processing node (gateway or ECU) process (**1500**) describes how replicated data is stored in the bulletin board. This process can be triggered by a plurality of notification mechanisms, such as internal or remote events, alarm signals, internal and external timers, and flags set in the bulletin board. The communication bus may also issue these notifications. The process (**1500**) then reads a message from the communication port (**1502**), selects a subset of variables to be replicated (**1503**), and stores the variables in the bulletin board with procedure (**801**). In an alternate embodiment, this procedure may also be used to store a packet data unit (PDU) in the bulletin board for later replication on the same or a different communication link.

This store and forward networking mechanism can be implemented without the need for complex networking protocols and is therefore well suited for limited processing power and memory environments. It also works in soft-real time environments when no strict temporal behavior is required. The data store operation (**801**) may be repeated for a plurality of bulletin board sections. If the data replication process is a cyclic procedure, it waits until the next activation occurs (**1505**).

Continuing now with FIG. **16**, the store and forward updating mechanism (**1600**) replicates messages from remote processing nodes to other processing nodes from stored packet data units in the bulletin board. This process can be triggered by a plurality of notification mechanisms (**1601**), such as internal or remote events, alarm signals, internal and external timers, and flags set in the bulletin board. The communication bus may also issue these notifications.

The process (**1600**) then selects a message to be forwarded (**1602**) and the appropriate communication link and retrieves the PDU with the bulletin board retrieve mechanism (**1101**). It then adds the appropriate messages header for the communication link (**1603**) and may add routing information (**1604**). Finally the update process (**1600**) writes the messages to the communication port (**1605**). If the updating process is a cyclic procedure, it waits until the next activation occurs (**1607**).

FIG. **17** describes the various modes that the distributed communications and computing system (**100**) can be operated in. In one embodiment, the system operates in various distinct modes in order to preserve the integrity of the system and still allow for changing the architecture and behavior of the network or the roles of the individual nodes. When the distributed computing and communication system wakes up from the sleep mode (**1701**), it can enter a configuration and upgrade mode (**1702**), an emergency or debug mode (**1704**), or the normal real-time run mode (**1703**). The root node or system gateway in a distributed communication and computing system defines the mode based on the existence of external events, such as an external control command, internal events, a system failure, or failed integrity check.

Referring now to FIG. **1**, the external commands may be generated from a development tool (**122**) or a remote device (**132**) that is connected via a remote gateway (**131**). In an alternate embodiment, each ECU (**102**) or virtual ECU (**115**) can trigger the system to enter a different operating mode.

Continuing with FIG. **17**, in the configuration mode (**1702**), the system software and the information-sharing configuration can be updated via a secure communication link with encrypted commands Each processing node (ECU or gateway) may have security mechanisms such as a certificate that allows it to identify and authorize another entity (remote gateway, remote ECU, or development tool) to make changes to its bulletin board parameters.

The remote entity may also download a new firmware to the bulletin board. The ECU or gateway can store this new firmware in its non-volatile memory while it backs up the original image on the bulletin board for the case that the new software is not functional. In the update mode, the distributed system can also reconfigure the communication and computing infrastructure based on a new set of parameters that need to be stored in the individual bulletin boards.

In the normal run mode (**1703**), the system operates in the real-time information sharing mode and network configuration and certain parameters can't be changed. That protection allows defining deterministic temporal behavior on all communication links. But any processing node may enter a debug/emergency mode (**1704**) if a failure or other qualifying event occurs.

In the emergency mode, a processor executes an alternate procedure that maintains the temporal behavior on the communication links but may reduce or increase the amount of information shared with other processors. It also lets other processing nodes check on the integrity of sensors and actuators. In the maintenance and upgrade mode, an external system can upgrade executable code images and the bulletin-board configuration via secure communication links.

A system and method are thus provided for sharing information within a distributed embedded communications and computing system and with components outside the embedded system. The information sharing mechanism relies on a bulletin board that may include a small database operating under hard real-time conditions with minimal delays, communication latency, and jitter. The embedded database or bulletin board isolates a real-time application in a Electronic Control Unit (ECU) from various other real time applications and from input output signals in the same module (local information sharing), from event-triggered communications with applications in other modules, and from time-triggered communications with applications in other modules.

One design criteria of the database is that the temporal behavior of communications does not impact the real-time computing task and provides enough information access performance at peak time demand Typically, distributed embedded systems consist of a static structure that can be analyzed at design time. In addition to the real-time operation, the proposed method for information sharing also provides access to the parameters of the embedded system and allows for software upgrades of certain modules.

The present embodiment addresses the shortcomings of traditional computer networks with following enhancements:

1) The concept of multi-mode storage that links two or more communication networks via a bulletin board. The

bulletin board is a multi-mode storage that can be thought of an extension to shared memory that can be accessed by local and remote processes at attached networks. There may be multiple hierarchical layers of bulletin boards depending on the topology of the communication system. The bulletin board increases the network efficiency by reducing the number of transactions needed to access remote variables.

2) The concept of a direct-access bulletin board that does not require a network layer translation of messages on each node of the network. Even though this approach restricts the reach of each node to only adjacent nodes and the next gateway, this still allows cross-network variable sharing though vertical real-time replication of data.

3) The concept of hierarchical bulletin board management that allows restriction of information access to certain levels in a network, but still allows the replication of information to other nodes in the network. This paradigm follows the path of reducing the information amount from the leaves of the network to central control and diagnosis hubs.

4) The concept that a gateway can host an assembly of bulletin boards or embedded database that allows operations on bulletin boards to generate events for associated processes. This extension allows definition of a set of data processing operations that would be done once in a network and would be instantly available for connected nodes. Examples for operations are sensor data state observers, diagnostics, integrity checks, fail-safe mechanisms, etc.

5) The concept that an embedded communication and computing network can run in multiple modes in order to provide for a guaranteed deterministic behavior of the system. This property can be achieved by only allowing change to the configuration and/or the functions (SW code) in a secured configuration and upgrade mode. If the network is booted in the normal operating mode, all processors execute the existing code and only allow data sharing through the bulletin boards. The emergency or debug mode lets the network run in a fail-safe reduced operation mode or in a diagnostic mode that allows inspection of the system, while it is running. For each operating mode, the gateway can store a processing image on the bulletin board. The advantage of this procedure is that only the communication hubs need to deal with secure data transfer and encryption while the peripheral nodes in the network can be relative simple in design.

6) The concept of designing the topology of a distributed computing and communication system independent of the definition of the individual functions that the network performs. Each processing task is only associated with a bulletin board, but isolated from I/O processing.

Of course, these are all optional embodiments/enhancements.

While various embodiments have been described above, it should be understood that they have been presented by the way of example only, and not limitation. Thus, the breadth and scope of a preferred embodiment should be not limited by any of the above described exemplary embodiments, but should be defined only in accordance with the following claims and their equivalents.

What is claimed is:

1. A layered system for sharing information in an automobile vehicle, said system comprising:

an automotive electronic control unit comprising a microprocessor and an operating system;

a hardware abstraction layer within the electronic control unit allowing the operating system to be adapted to a specific hardware implementation as used in the electronic control unit;

non-volatile memory comprising a database with a data structure;

a memory manager associated with the non-volatile memory, said memory manager comprising an upgrade and configuration manager to configure the data structure of the non-volatile memory, an event manager to capture input-output events as variables and generate new events, flags or signals, a data access manager to control code update and configuration of the memory and access rights for individual applications at execution, and a data integrity component to analyze stored state variables for integrity and generate events or flags if any problem occurs;

the non-volatile memory further comprising instructions to:

receive information in the form of a packet data unit representing datum information carried by an overall message from a first physical network selected from the group consisting of FlexRay, Controller Area Network, and Local Interconnect Network;

in response to the receipt of the information, issue a storage resource request in connection with a storage resource;

determine whether the storage resource is available for storing the information;

determine whether a threshold has been reached in association with the storage resource request;

in the event the storage resource is not available and the threshold associated with the storage resource request has not been reached, issue another storage resource request in connection with the storage resource;

in the event the storage resource is available, store the information in the storage resource; and

share the stored information with at least one of a plurality of heterogeneous processes including at least one process associated with a second physical network selected from the group consisting of FlexRay, Controller Area Network, and Local Interconnect Network, utilizing a network protocol different from a protocol of the first physical network;

interfaces for communication with each of FlexRay, Controller Area Network, and Local Interconnect Network networks, with each physical network in communication with a component including at least one of a sensor, an actuator, or a gateway, and with each of the FlexRay, Controller Area Network, and Local Interconnect Network interfaces comprising a corresponding network communication bus controller including a corresponding network communication bus driver;

the interfaces including a first communication interface for interfacing with the first physical network, the first communication interface including a first communication interface-related data link layer component, said first communication interface configured to extract variables from the overall message communicated by the first physical network employing a first protocol and storing the packet data unit representing the datum information carried by the overall message from a first physical network in the database; and

a second communication interface for interfacing with the second physical network utilizing a protocol different than the protocol of the first physical network, the second communication interface including a second communication interface-related data link layer component;

wherein the automotive electronic control unit is configured such that the stored information may be shared with the second physical network by replicating the packet unit data obtained from the first physical network by composing another message configured to be communicated using the different protocol of the second physical network.

2. The system of claim **1**, wherein the memory comprises at least one bulletin board that stores information that may be addressed by any process of any network having access to the storage resource of the electronic control unit.

3. The system of claim **2**, wherein a complexity of the bulletin board grows linearly with a number of networks.

4. The system of claim **1**, wherein the electronic control unit comprises code for the electronic control unit to run in a diagnostic mode that allows inspection of the system, while the electronic control unit is running.

5. The system of claim **4**, wherein the electronic control unit comprises code for the electronic control unit to additionally run in a debugging mode.

6. The system of claim **1**, wherein at least one of the first and the second physical networks is a multiplexing bus.

7. The system of claim **1**, wherein each of the first and the second physical networks is a multiplexing bus.

8. The system of claim **1**, wherein neither the first nor the second physical network is a multiplexing bus.

9. The system of claim **1**, wherein the automotive electronic control unit comprises digital input/output, an analog to digital converter, and a digital to analog converter.

10. The system of claim **1**, wherein the automotive electronic control unit comprises a real time clock and a watchdog.

11. The system of claim **2**, wherein the electronic control unit comprises:

code for the electronic control unit to run in a diagnostic mode that allows inspection of the system, while the electronic control unit is running;

code to run in a debugging mode;

code to interface with each of a FlexRay multiplexing bus, a Controller Area Network multiplexing bus, and a Local Interconnect Network multiplexing bus;

digital input/output, an analog to digital converter, and a digital to analog converter;

a real time clock;

a watchdog; and

wherein in the event the storage resource is not available and the threshold associated with the storage resource request has been reached, the electronic control unit is configured to send a notification to a process associated with the information.

12. A layered system for sharing information in an automobile vehicle, said system comprising:

an automotive electronic control unit comprising a microprocessor and an operating system;

a hardware abstraction layer within the electronic control unit allowing the operating system to be adapted to a specific hardware implementation as used in the electronic control unit;

non-volatile memory comprising a database with a data structure;

a memory manager associated with the non-volatile memory, said memory manager comprising an upgrade and configuration manager to configure the data structure of the non-volatile memory, an event manager to capture input-output events as variables and generate new events, flags or signals, a data access manager to control code update and configuration of the memory and access rights for individual applications at execution, and a data integrity component to analyze stored state variables for integrity and generate events or flags if any problem occurs;

the non-volatile memory further comprising instructions to:

receive information in the form of a packet data unit representing datum information carried by an overall message from a first physical network selected from the group consisting of FlexRay, Controller Area Network, Local Interconnect Network and Media Oriented Systems Transport;

in response to the receipt of the information, issue a storage resource request in connection with a storage resource;

determine whether the storage resource is available for storing the information;

determine whether a threshold has been reached in association with the storage resource request;

in the event the storage resource is not available and the threshold associated with the storage resource request has not been reached, issue another storage resource request in connection with the storage resource;

in the event the storage resource is available, store the information in the storage resource; and

share the stored information with at least one of a plurality of heterogeneous processes including at least one process associated with a second physical network selected from the group consisting of FlexRay, Controller Area Network, Local Interconnect Network and Media Oriented Systems Transport, utilizing a network protocol different from a protocol of the first physical network;

interfaces for communication with each of FlexRay, Controller Area Network, and Local Interconnect Network networks, with each physical network in communication with a component including at least one of a sensor, an actuator or a gateway, and with each of the FlexRay, Controller Area Network, Local Interconnect Network and Media Oriented Systems Transport interfaces comprising a corresponding network communication bus controller including a corresponding network communication bus driver;

the interfaces including a first communication interface for interfacing with the first physical network, the first communication interface including a first communication interface-related data link layer component, said first communication interface configured to extract variables from the overall message communicated by the first physical network employing a first protocol and storing the packet data unit representing the datum information carried by the overall message from a first physical network in the database; and

a second communication interface for interfacing with the second physical network utilizing a protocol different than the protocol of the first physical network, the second communication interface including a second communication interface-related data link layer component;

wherein the automotive electronic control unit is configured such that the stored information may be shared with the second physical network by replicating the packet unit data obtained from the first physical network by composing another message configured to be communicated using the different protocol of the second physical network.

13. The system of claim **12**, wherein the memory comprises at least one bulletin board that stores information that may be addressed by any process of any network having access to the storage resource of the electronic control unit.

14. The system of claim **13**, wherein a complexity of the bulletin board grows linearly with a number of networks.

15. The system of claim **12**, wherein the electronic control unit comprises code for the electronic control unit to run in a diagnostic mode that allows inspection of the system, while the electronic control unit is running.

16. The system of claim **15**, wherein the electronic control unit comprises code for the electronic control unit to additionally run in a debugging mode.

17. The system of claim **12**, wherein at least one of the first and the second physical networks is a multiplexing bus.

18. The system of claim **12**, wherein each of the first and the second physical networks is a multiplexing bus.

19. The system of claim **12**, wherein neither the first nor the second physical network is a multiplexing bus.

20. The system of claim **12**, wherein the automotive electronic control unit comprises digital input/output, an analog to digital converter, and a digital to analog converter.

21. The system of claim **12**, wherein the automotive electronic control unit comprises a real time clock and a watchdog.

22. The system of claim **12**, wherein the electronic control unit comprises:

code for the electronic control unit to run in a diagnostic mode that allows inspection of the system, while the electronic control unit is running;

code to run in a debugging mode;

code to interface with each of a FlexRay multiplexing bus, a Controller Area Network multiplexing bus, a Local Interconnect Network multiplexing bus, and a Media Oriented Systems Transport multiplexing bus;

digital input/output, an analog to digital converter, and a digital to analog converter;

a real time clock;

a watchdog; and

wherein in the event the storage resource is not available and the threshold associated with the storage resource request has been reached, the electronic control unit is configured to send a notification to a process associated with the information.

23. A system, comprising:

a first physical network including at least one of a FlexRay network, a Controller Area Network, and a Local Interconnect Network;

a second physical network including at least one of the FlexRay network, the Controller Area Network, and the Local Interconnect Network; and

an automotive electronic control unit communicatively coupled to the first physical network and the second physical network, the automotive electronic control unit comprising:

a first physical network interface including a first network communication bus controller with a first network communication bus driver,

a second physical network interface including a second network communication bus controller with a second network communication bus driver,

a micro-processor communicatively coupled to the first physical network interface and the second physical network interface,

a hardware abstraction layer,

an operating system configured to be adapted to a specific hardware implementation, utilizing the hardware abstraction layer, and

a non-volatile memory comprising a data structure and instructions, wherein the micro-processor is configured to cause execution of the instructions to cause the automotive electronic control unit to:

configure the data structure of the non-volatile memory;

generate at least one of a new event, a new flag, or a new signal, based on at least one of an input event or an output event;

control at least one of a code update or a configuration in connection with the non-volatile memory;

control one or more access rights for one or more applications;

generate at least one of a problem-related event or a problem-related flag in response to an identification of a problem identified via an analysis of at least one stored state variable;

receive information in the form of a packet data unit including a datum carried by a received message from the first physical network, utilizing a first network protocol;

in response to the receipt of the information, issue a storage resource request in connection with a storage resource;

determine whether the storage resource is available for storing the information;

determine whether a threshold has been reached in association with the storage resource request;

in the event the storage resource is not available and the threshold associated with the storage resource request has not been reached, issue another storage resource request in connection with the storage resource;

in the event the storage resource is available, store the information in the storage resource;

replicate the stored information;

compose an outgoing message including the replicated stored information; and

share the outgoing message with the second physical network, utilizing a second network protocol that is different from the first network protocol.

24. The system of claim **23**, wherein the non-volatile memory comprises at least one bulletin board that includes the stored information that is capable of being accessed by any process of any network having access to the electronic control unit.

25. The system of claim **24**, wherein a complexity of the bulletin board grows linearly with a number of networks.

26. The system of claim **23**, wherein the electronic control unit comprises code to run in a diagnostic mode that allows inspection of the system, while the electronic control unit is running.

27. The system of claim **26**, wherein the electronic control unit comprises code to additionally run in a debugging mode.

28. The system of claim **24**, wherein the electronic control unit comprises:

code to run in a diagnostic mode that allows inspection of the system, while the electronic control unit is running;

code to run in a debugging mode;

code to interface with each of a FlexRay multiplexing bus, a Controller Area Network multiplexing bus, and a Local Interconnect Network multiplexing bus;

a real time clock;

a watchdog; and

wherein the system is configured such that, in the event the storage resource is not available and the threshold associated with the storage resource request has been reached, the electronic control unit is configured to send a notification to a process associated with the information.

29. The system of claim **24**, wherein the system is configured such that at least one of:

the first physical network includes the FlexRay network;

the first physical network includes the Controller Area Network;

the first physical network includes the Local Interconnect Network;

the second physical network includes the FlexRay network;

the second physical network includes the Controller Area Network;

the second physical network includes the Local Interconnect Network;

the new event, the new flag, or the new signal are all generated based on the input event;

only one of the new event, the new flag, or the new signal is generated based on the input event;

the new signal is generated based on the input event;

the new event, the new flag, or the new signal are all generated based on the output event;

only one of the new event, the new flag, or the new signal is generated based on the output event;

the new signal is generated based on the output event;

a single access right is controlled for a single application;

multiple access rights are controlled for multiple applications;

a single access right is controlled for a single application at execution;

multiple access rights are controlled for multiple applications at execution;

both the problem-related event and the problem-related flag are generated in response to the identification of the problem identified via the analysis of at least one stored state variable;

only one of the problem-related event or the problem-related flag is generated in response to the identification of the problem identified via the analysis of at least one stored state variable;

the problem-related event is generated in response to the identification of the problem identified via the analysis of at least one stored state variable;

at least one stored state variable is a single stored state variable; or the at least one stored state variable includes multiple stored state variables.

30. A system, comprising:

an automobile including:

a first physical network including at least one of a FlexRay network, a Controller Area Network, and a Local Interconnect Network;

a second physical network including at least one of the FlexRay network, the Controller Area Network, and the Local Interconnect Network; and

an automotive electronic control unit communicatively coupled to the first physical network and the second physical network, the automotive electronic control unit comprising:

a first physical network interface including a first network communication bus controller with a first network communication bus driver,

a second physical network interface including a second network communication bus controller with a second network communication bus driver,

a micro-processor communicatively coupled to the first physical network interface and the second physical network interface, a hardware abstraction layer,

an operating system configured to be adapted to a specific hardware implementation, utilizing the hardware abstraction layer, and

a non-volatile memory comprising a data structure and instructions, wherein the micro-processor is configured to cause execution of the instructions to cause the automotive electronic control unit to:

configure the data structure of the non-volatile memory,

generate at least one of a new event, a new flag, or a new signal, based on at least one of an input event or an output event,

control at least one of a code update or a configuration in connection with the non-volatile memory,

control one or more access rights for one or more applications,

generate at least one of a problem-related event or a problem-related flag in response to an identification of a problem identified via an analysis of at least one stored state variable,

receive information in the form of a packet data unit including a datum carried by a received message from the first physical network, utilizing a first network protocol,

in response to the receipt of the information, issue a storage resource request in connection with a storage resource,

determine whether the storage resource is available for storing the information,

determine whether a threshold has been reached in association with the storage resource request,

in the event the storage resource is not available and the threshold associated with the storage resource request has not been reached, issue another storage resource request in connection with the storage resource,

in the event the storage resource is available, store the information in the storage resource,

replicate the stored information,

compose an outgoing message including the replicated stored information, and

share the outgoing message with the second physical network, utilizing a second network protocol that is different from the first network protocol.

* * * * *

US010031790B1

(12) **United States Patent**
Fuchs et al.

(10) **Patent No.: US 10,031,790 B1**
(45) **Date of Patent: *Jul. 24, 2018**

(54) **SYSTEM, METHOD AND COMPUTER PROGRAM PRODUCT FOR SHARING INFORMATION IN A DISTRIBUTED FRAMEWORK**

(71) Applicant: **Stragent, LLC**, Longview, TX (US)

(72) Inventors: **Axel Fuchs**, San Jose, CA (US); **Scott Sturges Andrews**, Petaluma, CA (US)

(73) Assignee: **Stragent, LLC**, Longview, TX (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **15/919,201**

(22) Filed: **Mar. 12, 2018**

Related U.S. Application Data

(63) Continuation of application No. 15/405,110, filed on Jan. 12, 2017, now Pat. No. 10,002,036, which is a continuation of application No. 14/011,705, filed on Aug. 27, 2013, now Pat. No. 9,575,817, which is a continuation of application No. 13/531,319, filed on Jun. 22, 2012, now Pat. No. 8,566,843, which is a continuation of application No. 12/182,570, filed on Jul. 30, 2008, now Pat. No. 8,209,705, which is a continuation of application No. 10/737,690, filed on Dec. 15, 2003, now Pat. No. 7,802,263.

(60) Provisional application No. 60/434,018, filed on Dec. 17, 2002.

(51) **Int. Cl.**
G06F 9/54 (2006.01)
B60R 16/023 (2006.01)
H04L 29/08 (2006.01)
H04L 12/26 (2006.01)

(52) **U.S. Cl.**
CPC ***G06F 9/546*** (2013.01); ***B60R 16/0231*** (2013.01); ***G06F 9/542*** (2013.01); ***G06F 9/545*** (2013.01); ***H04L 43/10*** (2013.01); ***H04L 67/10*** (2013.01); *G06F 2209/547* (2013.01)

(58) **Field of Classification Search**
CPC G06F 9/542; G06F 9/545; G06F 9/546; H04L 43/10; H04L 67/10; B60R 16/0231
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,956,489 A * 9/1999 San Andres G06F 11/1662 709/221

OTHER PUBLICATIONS

Office Action Summary from U.S. Appl. No. 15/405,110 dated May 17, 2018.

* cited by examiner

Primary Examiner — Charles E Anya

(57) **ABSTRACT**

A system, method and computer program product are provided for receiving information associated with a message, issuing a storage resource request in connection with a storage resource and determining whether the storage resource is available. In use, the information is capable of being shared in less than one second, utilizing an automotive electronic control unit which includes a plurality of interfaces.

30 Claims, 17 Drawing Sheets

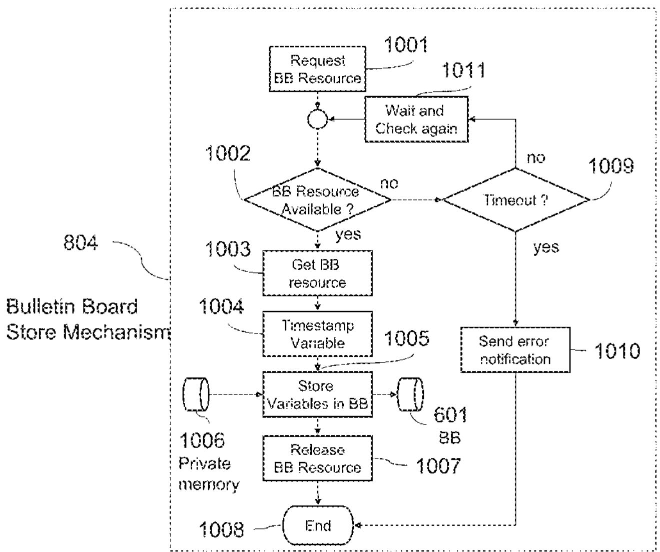


EXHIBIT B

FIGURE 1

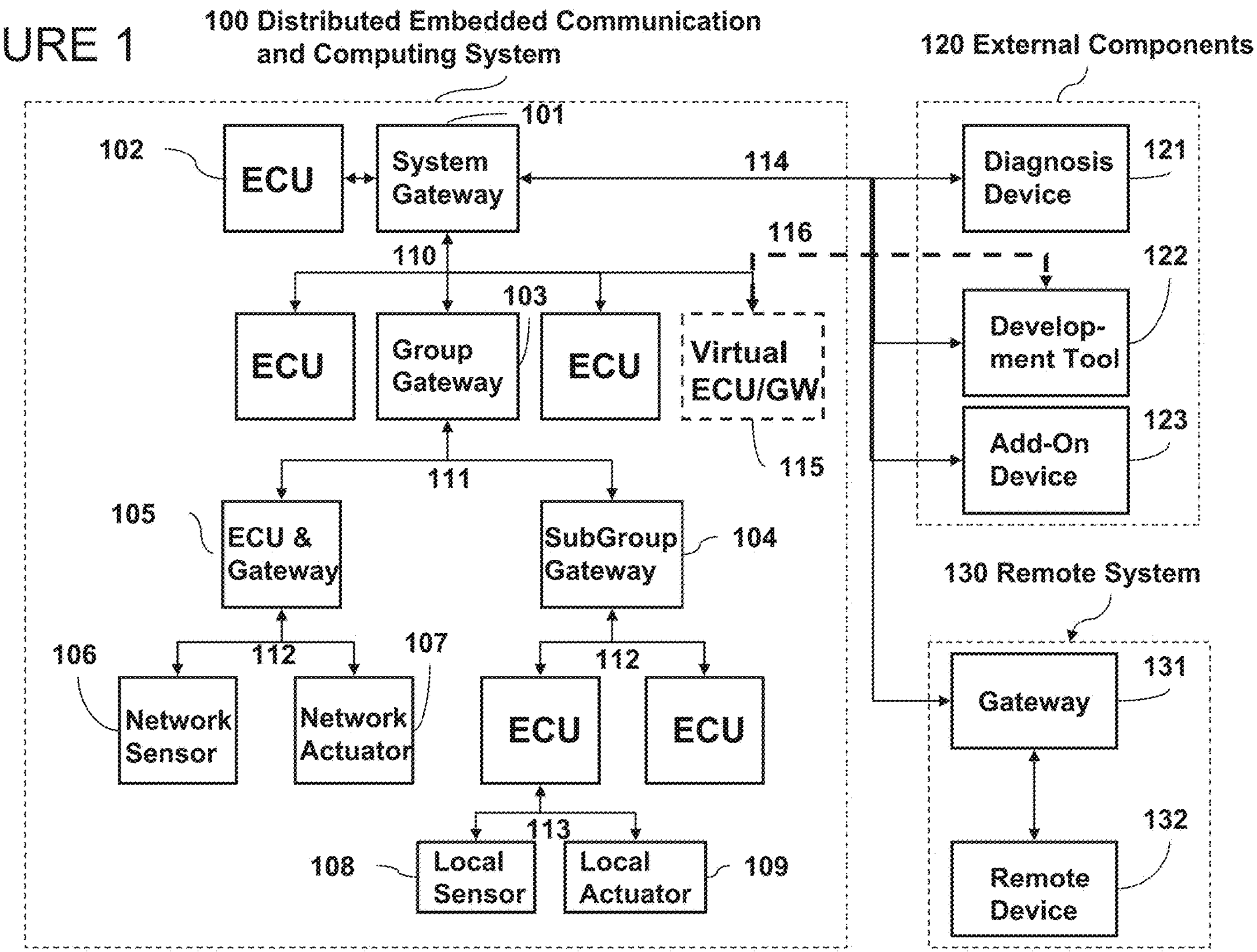
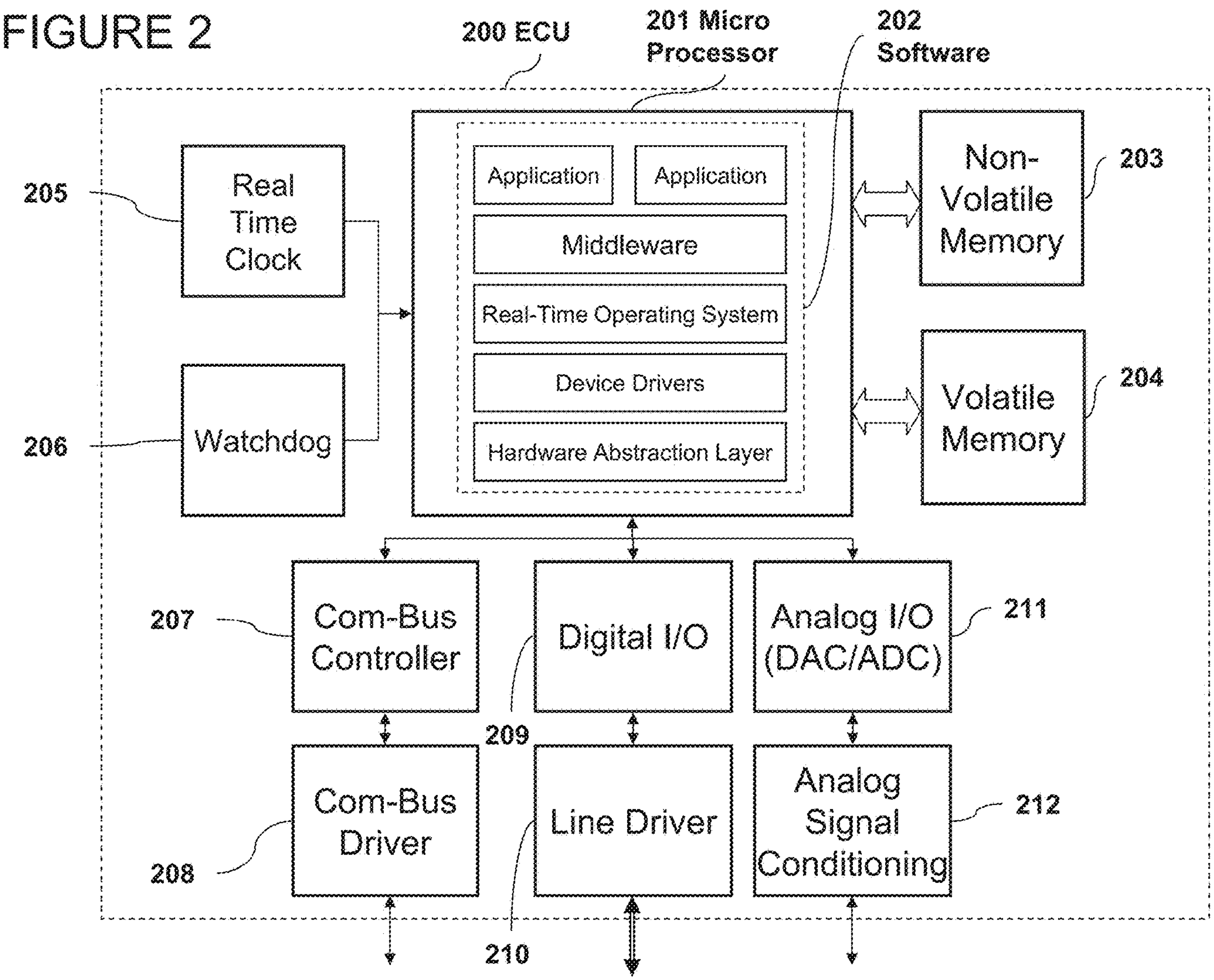

100 Distributed Embedded Communication and Computing System
120 External Components
101
102
ECU
System Gateway
114
Diagnosis Device
121
116
122
110
103
ECU
Group Gateway
ECU
Virtual ECU/GW
Develop-ment Tool
123
Add-On Device
115
111
105
ECU & Gateway
SubGroup Gateway
104
130 Remote System
106
112
107
112
Network Sensor
Network Actuator
ECU
ECU
Gateway
131
132
Remote Device
113
108
Local Sensor
Local Actuator
109

FIGURE 2


200 ECU
201 Micro Processor
202 Software
Application
Application
Middleware
Real-Time Operating System
Device Drivers
Hardware Abstraction Layer
Non-Volatile Memory
203
Volatile Memory
204
205
Real Time Clock
206
Watchdog
207
Com-Bus Controller
Digital I/O
Analog I/O (DAC/ADC)
211
209
208
Com-Bus Driver
Line Driver
Analog Signal Conditioning
212
210

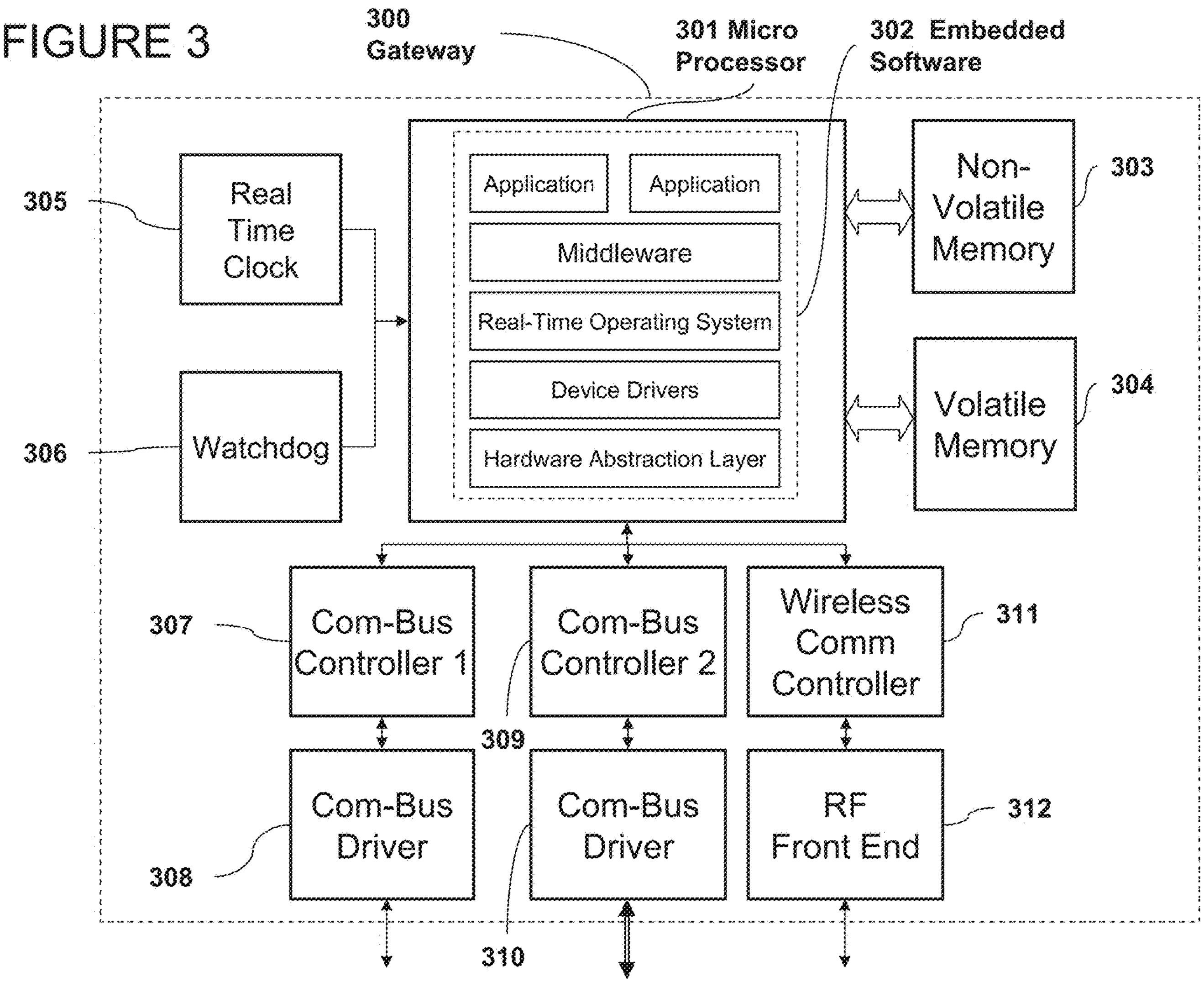

300 Gateway
301 Micro Processor
302 Embedded Software
Application
Application
Middleware
Real-Time Operating System
Device Drivers
Hardware Abstraction Layer
305
Real Time Clock
306
Watchdog
Non-Volatile Memory
303
Volatile Memory
304
307
Com-Bus Controller 1
Com-Bus Controller 2
Wireless Comm Controller
311
309
308
Com-Bus Driver
Com-Bus Driver
RF Front End
312
310

FIGURE 3

FIGURE 4

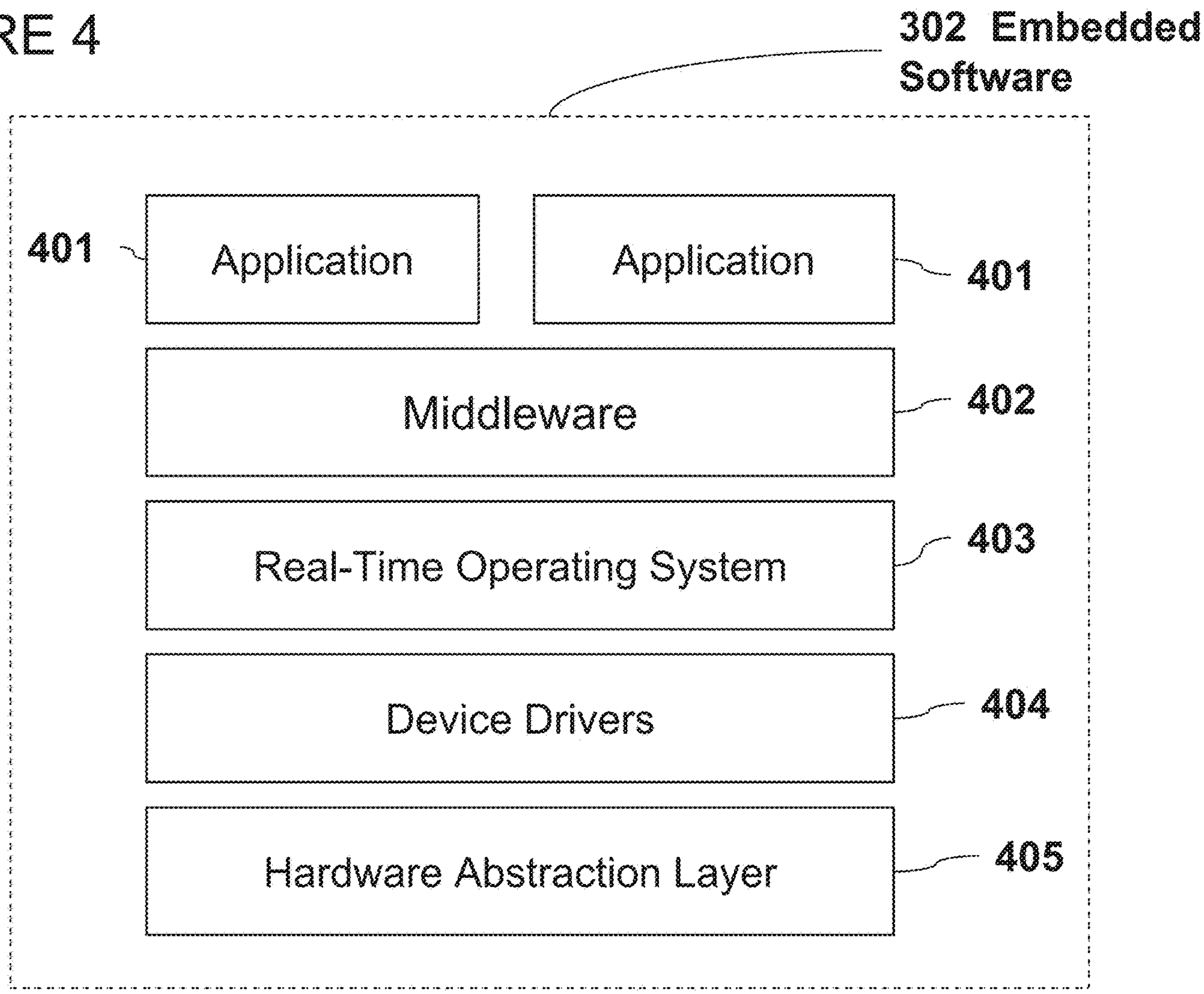

302 Embedded Software
401
Application
Application
401
Middleware
402
Real-Time Operating System
403
Device Drivers
404
Hardware Abstraction Layer
405

FIGURE 5

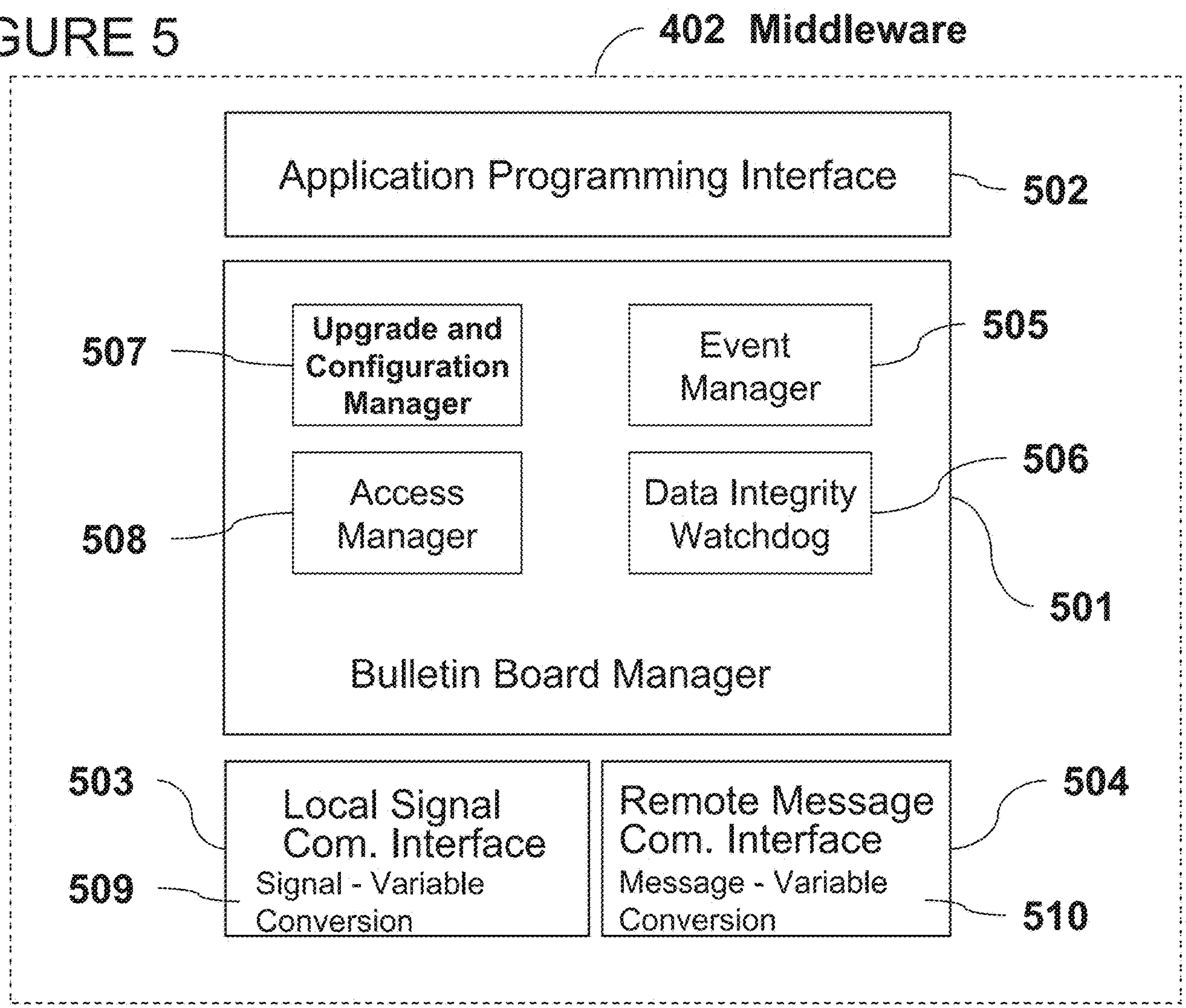

402 Middleware
Application Programming Interface
502
Upgrade and Configuration Manager
507
Event Manager
505
Access Manager
508
Data Integrity Watchdog
506
501
Bulletin Board Manager
503
Local Signal Com. Interface
Signal - Variable Conversion
509
Remote Message Com. Interface
Message - Variable Conversion
504
510

FIGURE 6

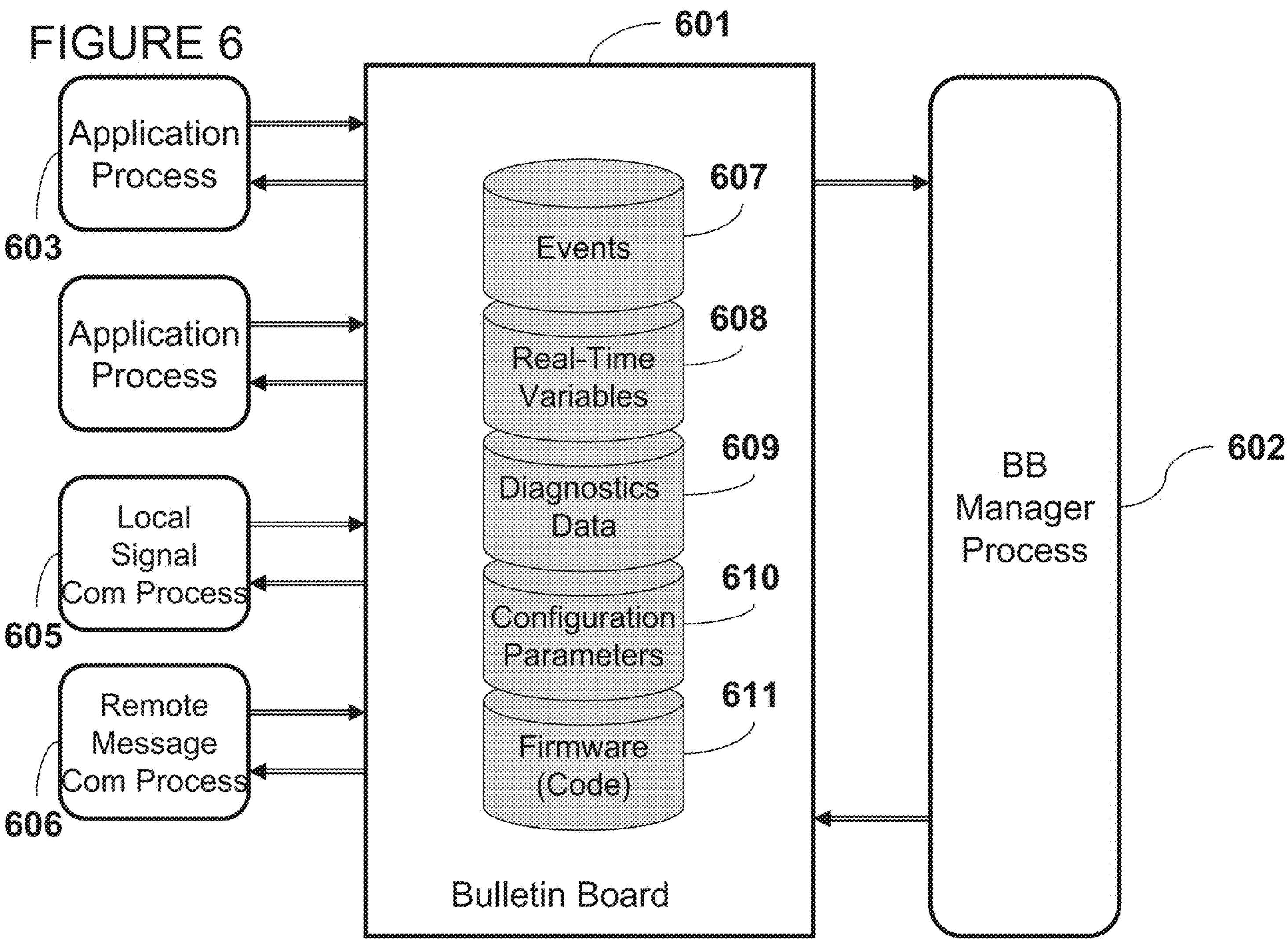
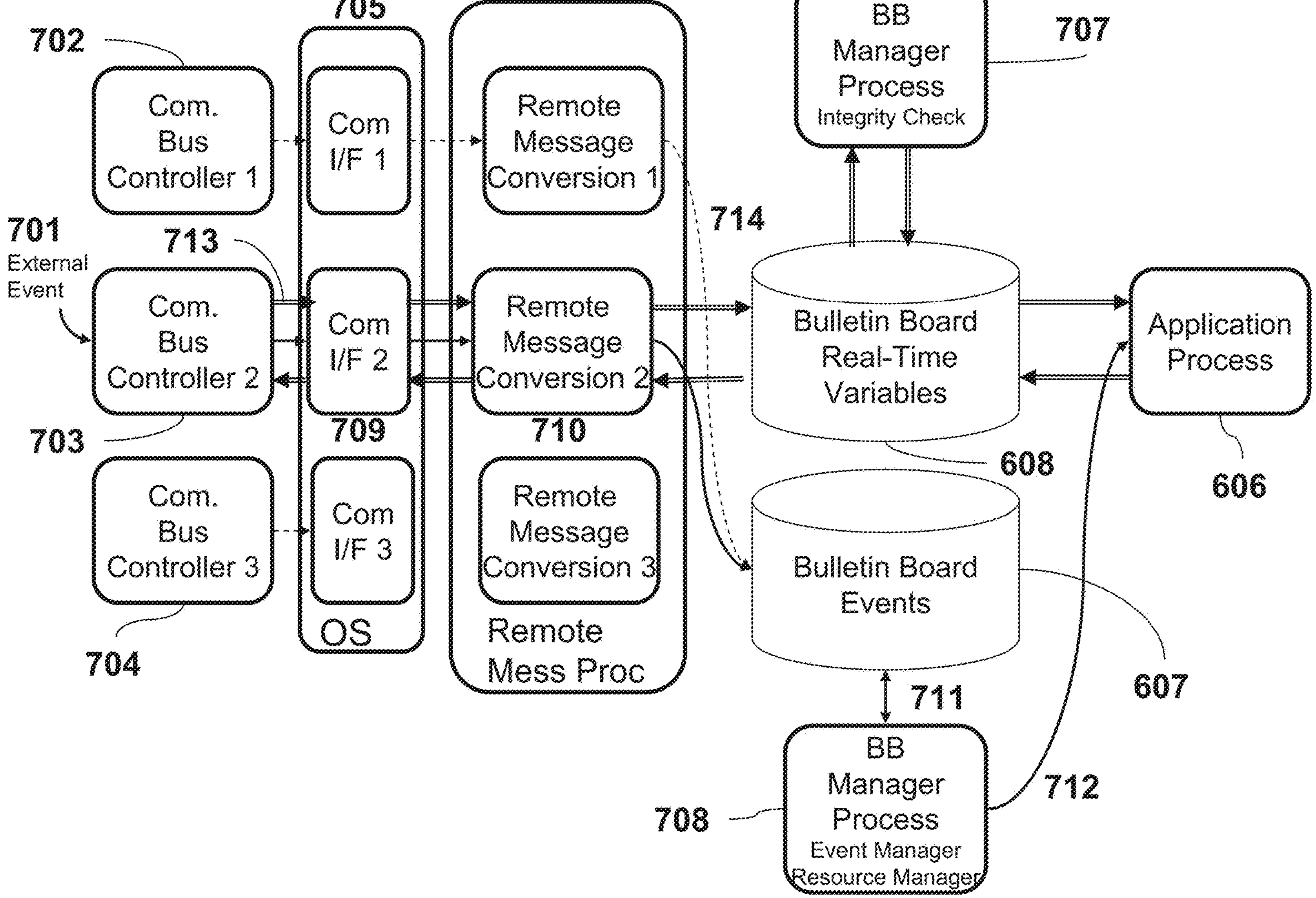

601
Application Process
603
Application Process
Local Signal Com Process
605
Remote Message Com Process
606
Events
607
Real-Time Variables
608
Diagnostics Data
609
Configuration Parameters
610
Firmware (Code)
611
Bulletin Board
BB Manager Process
602

FIGURE 7



702
Com. Bus Controller 1
701
External Event
713
Com. Bus Controller 2
703
Com. Bus Controller 3
704
705
Com I/F 1
Com I/F 2
709
Com I/F 3
OS
706
Remote Message Conversion 1
Remote Message Conversion 2
710
Remote Message Conversion 3
Remote Mess Proc
714
BB Manager Process
Integrity Check
707
Bulletin Board Real-Time Variables
608
Bulletin Board Events
607
711
BB Manager Process
Event Manager
Resource Manager
708
712
Application Process
606

FIGURE 8

Store Variable from remote I/O

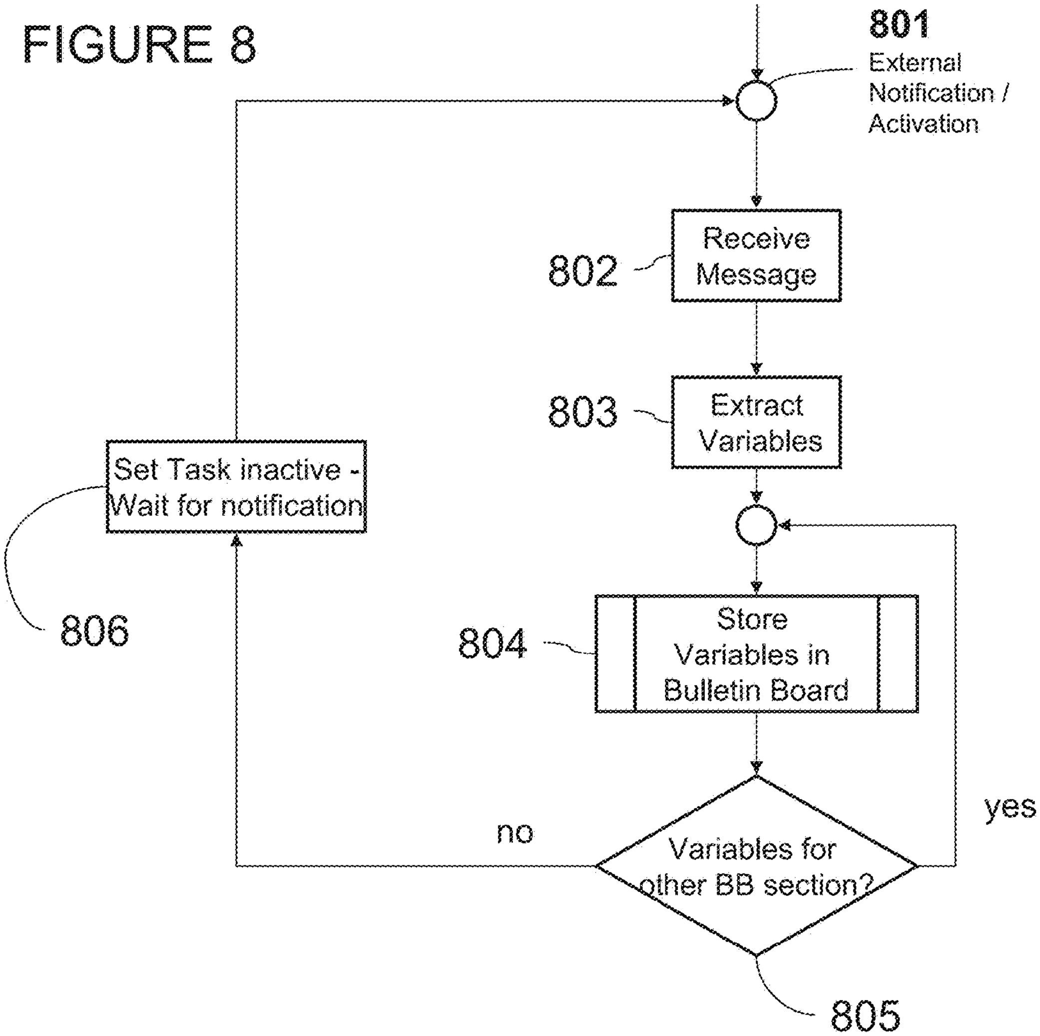


FIGURE 9

Store Variable from local I/O

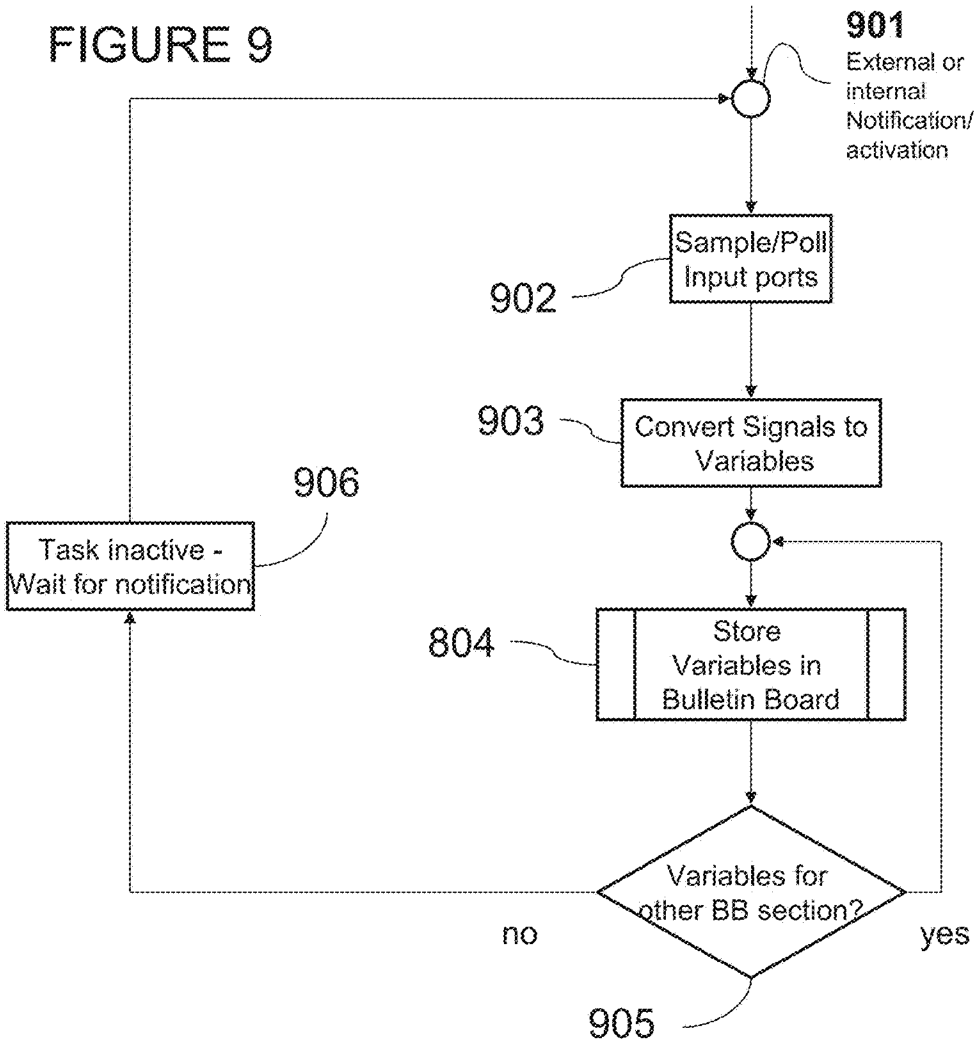
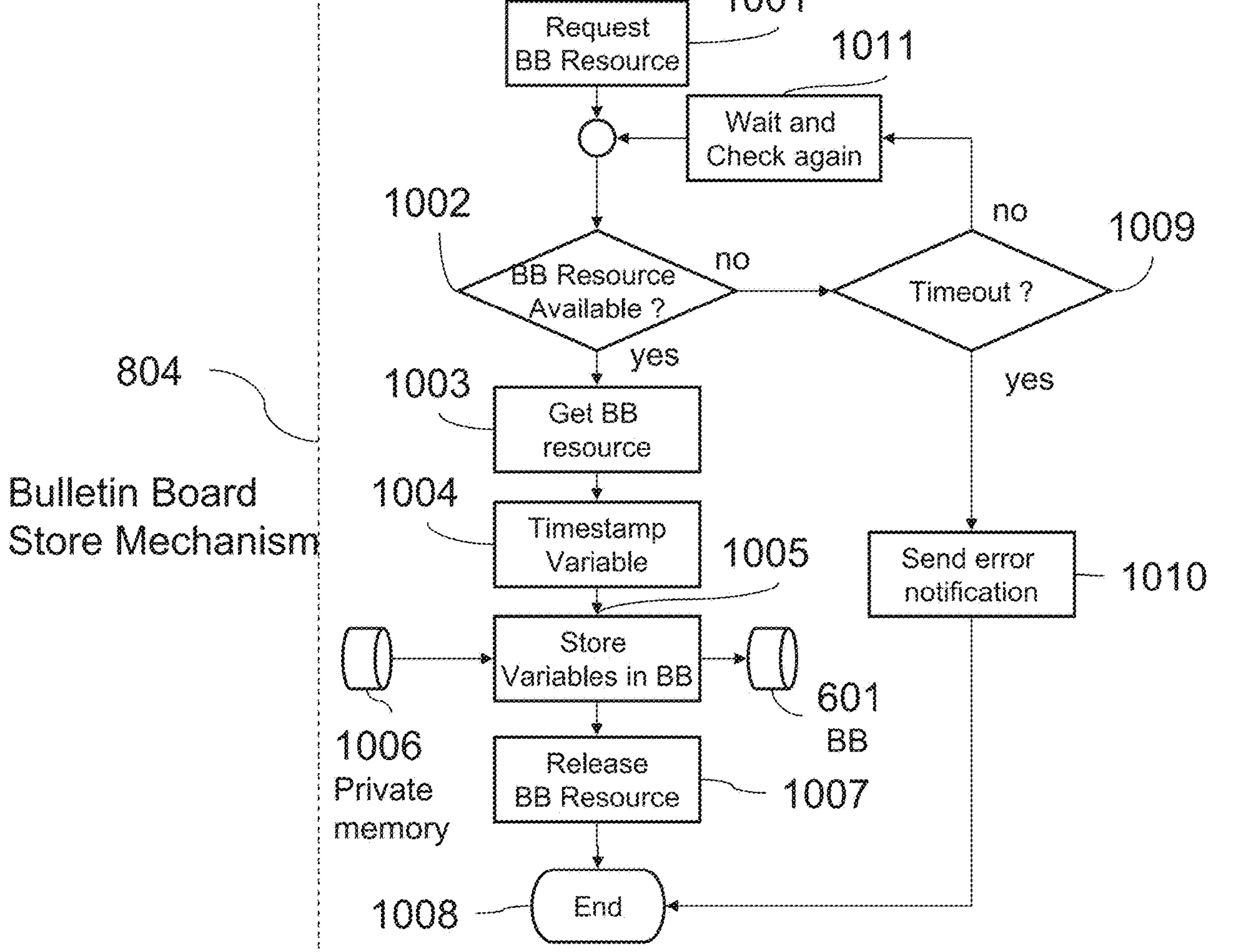


FIGURE 10



804
Bulletin Board Store Mechanism
Request BB Resource
1001
1011
Wait and Check again
1002
BB Resource Available ?
no
yes
no
1009
Timeout ?
yes
1003
Get BB resource
1004
Timestamp Variable
1005
Send error notification
1010
Store Variables in BB
601
BB
1006
Private memory
Release BB Resource
1007
1008
End

FIGURE 11

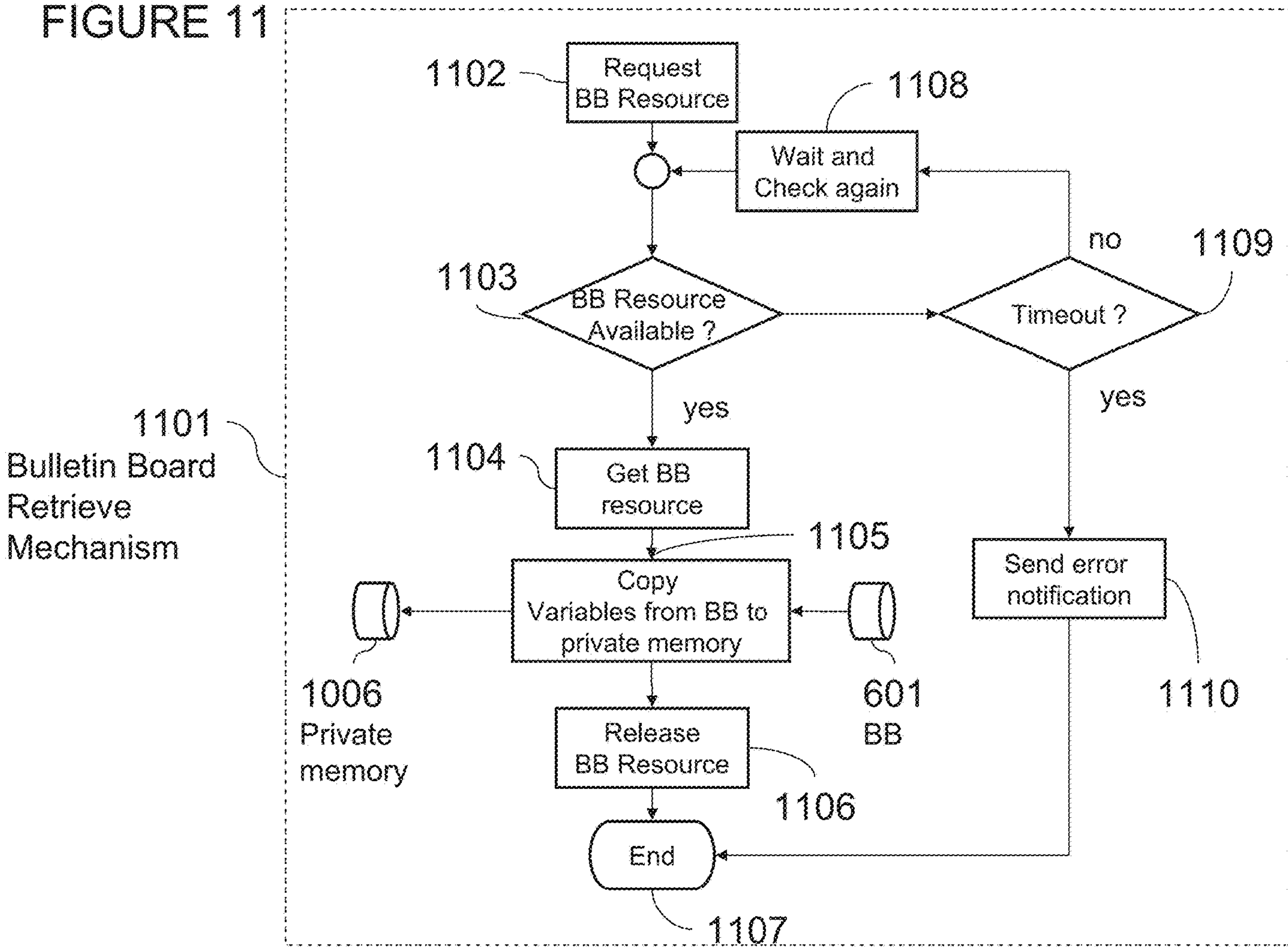

1101
Bulletin Board
Retrieve
Mechanism
1102
Request
BB Resource
1108
Wait and
Check again
1103
BB Resource
Available ?
no
1109
Timeout ?
yes
yes
1104
Get BB
resource
1105
Copy
Variables from BB to
private memory
Send error
notification
1006
Private
memory
601
BB
1110
Release
BB Resource
1106
End
1107

FIGURE 12

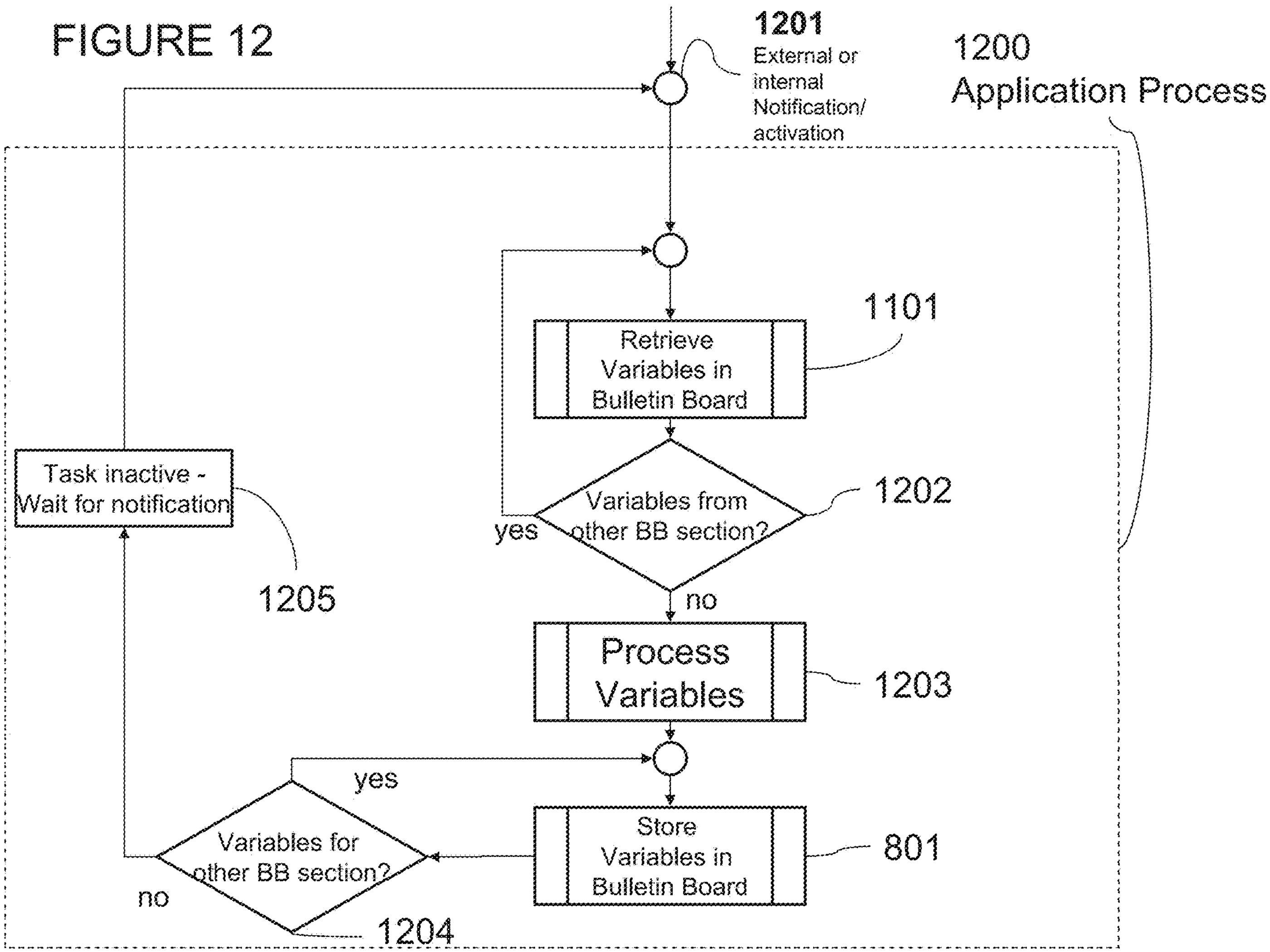

1201
External or internal Notification/ activation
1200
Application Process
Retrieve Variables in Bulletin Board
1101
Variables from other BB section?
yes
no
1202
Task inactive - Wait for notification
1205
Process Variables
1203
yes
Variables for other BB section?
no
1204
Store Variables in Bulletin Board
801

FIGURE 13

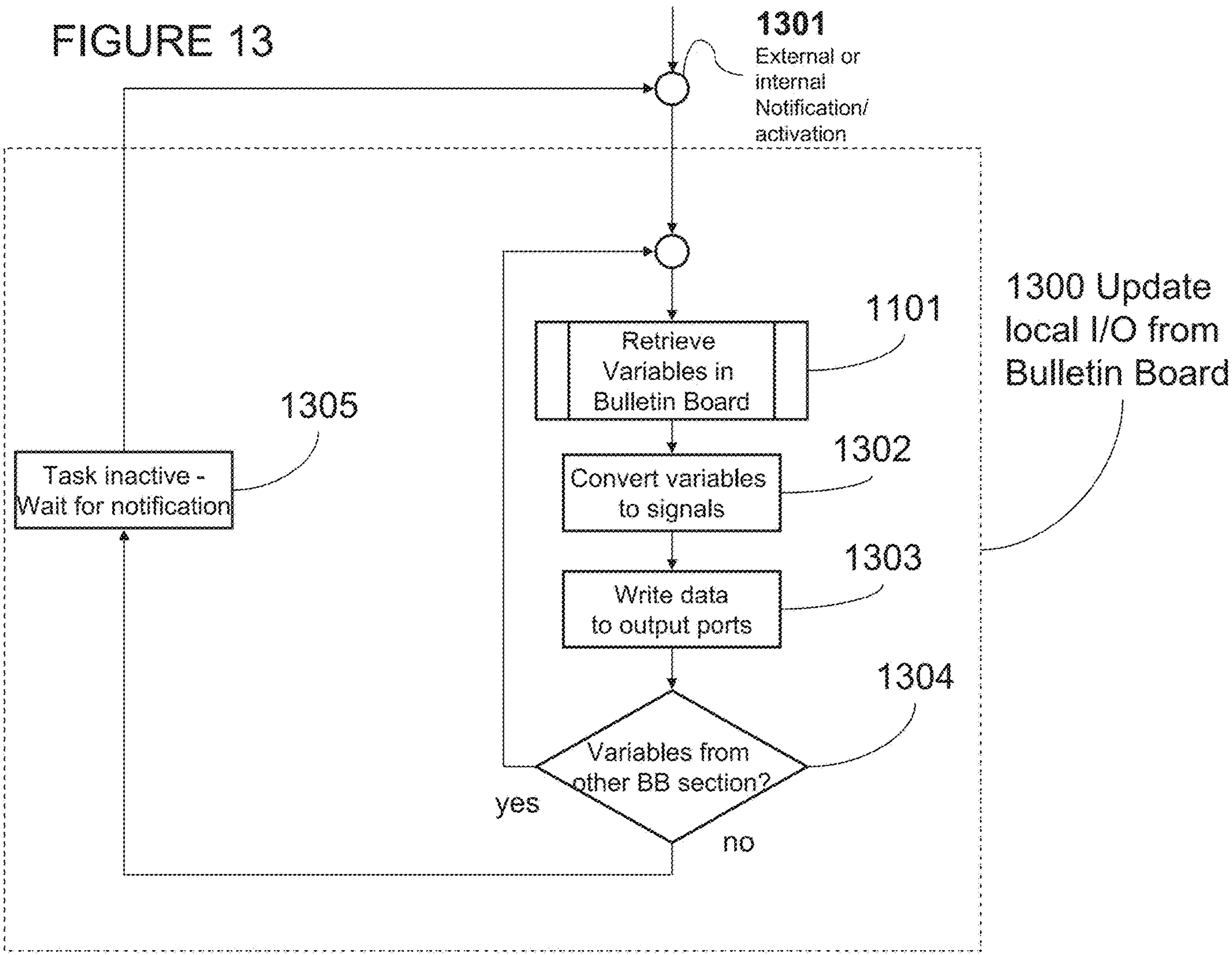
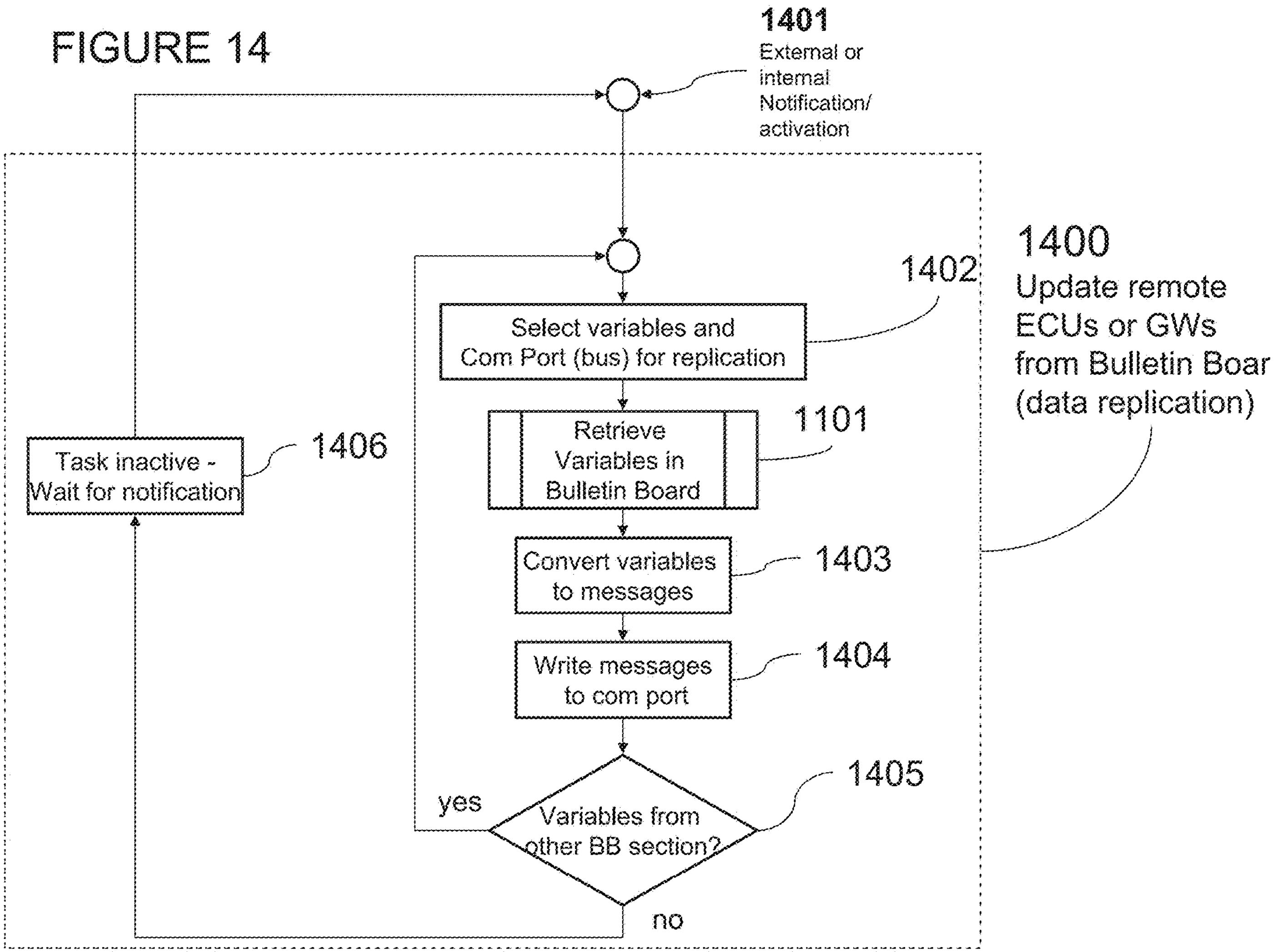

1301
External or
internal
Notification/
activation
1300 Update
local I/O from
Bulletin Board
1101
Retrieve
Variables in
Bulletin Board
1302
Convert variables
to signals
1303
Write data
to output ports
1304
Variables from
other BB section?
yes
no
1305
Task inactive -
Wait for notification

FIGURE 14



1401
External or internal Notification/ activation
1400
Update remote ECUs or GWs from Bulletin Boar (data replication)
1402
Select variables and Com Port (bus) for replication
1101
Retrieve Variables in Bulletin Board
1406
Task inactive - Wait for notification
Convert variables to messages
1403
Write messages to com port
1404
1405
yes
Variables from other BB section?
no

FIGURE 15

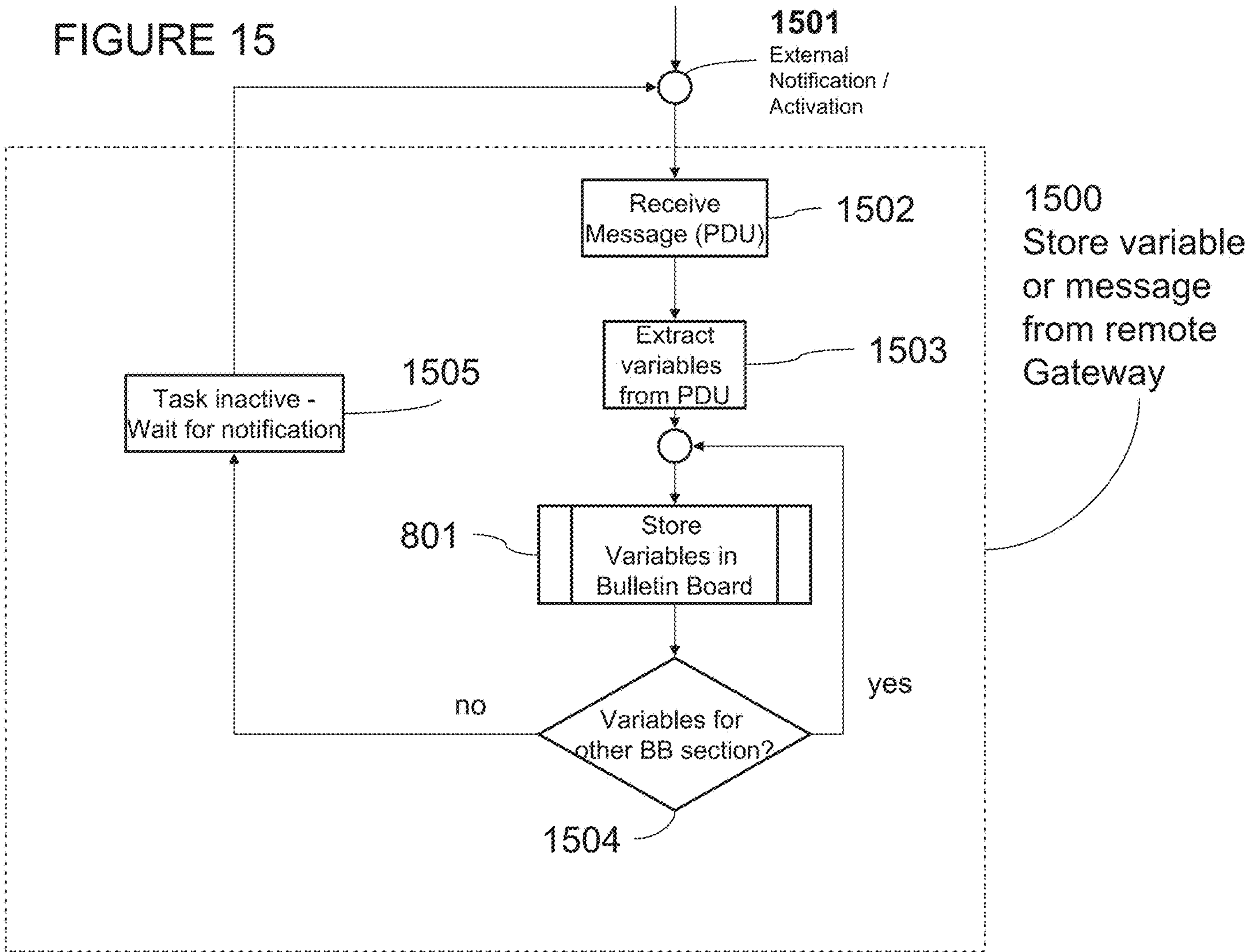

1501
External
Notification /
Activation
Receive
Message (PDU)
1502
Extract
variables
from PDU
1503
1505
Task inactive -
Wait for notification
801
Store
Variables in
Bulletin Board
Variables for
other BB section?
yes
no
1504
1500
Store variable
or message
from remote
Gateway

FIGURE 16

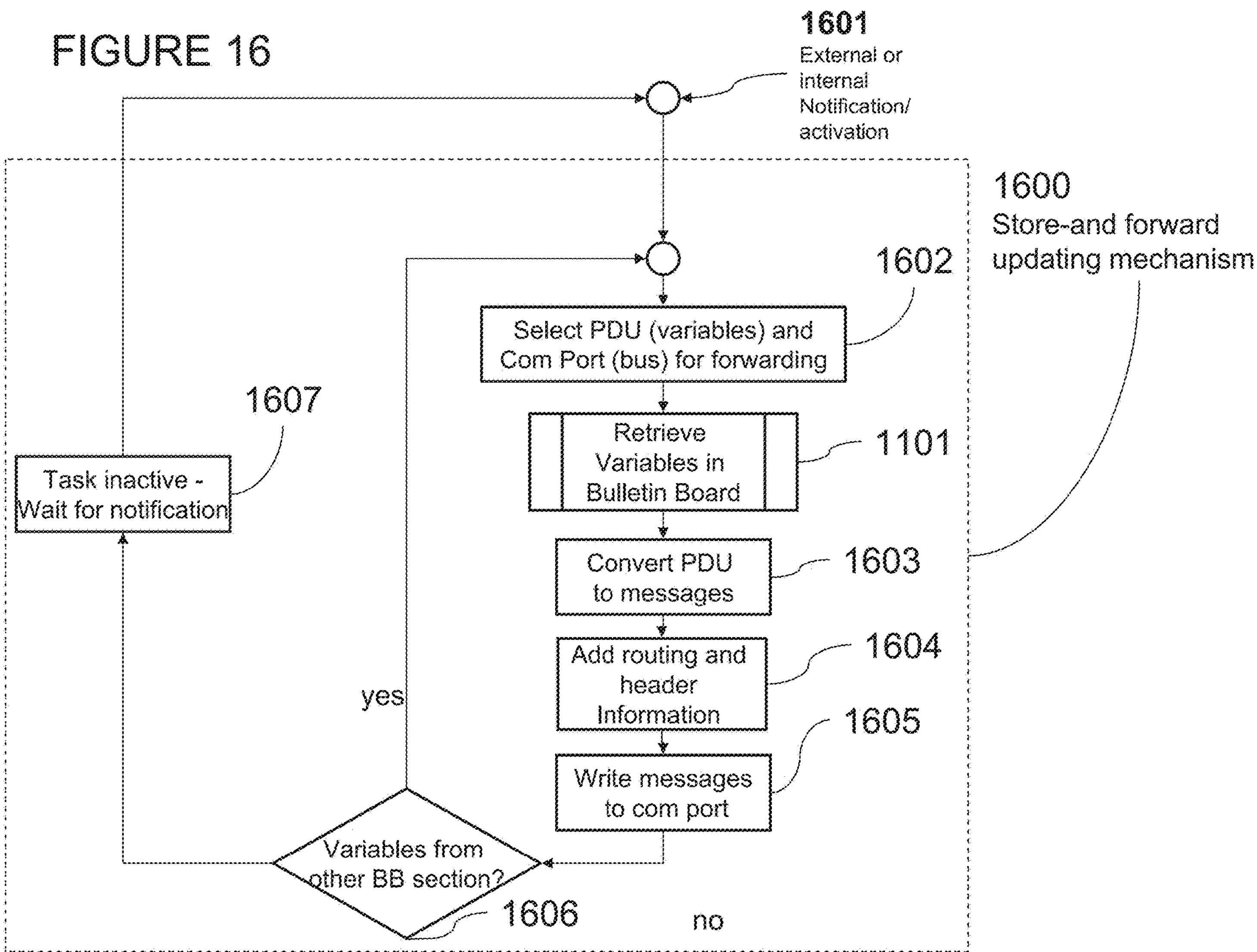
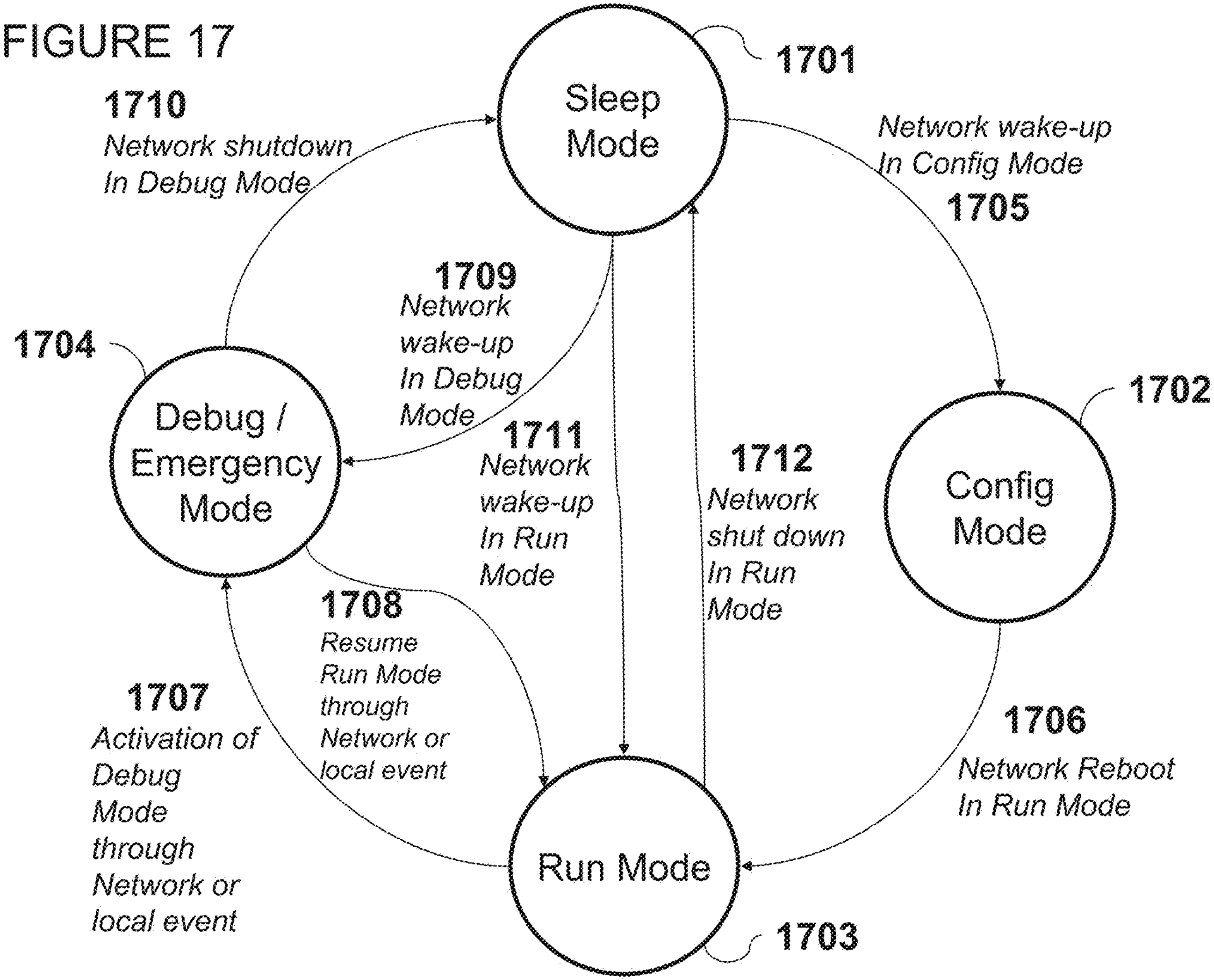

1601
External or internal Notification/ activation
1600
Store-and forward updating mechanism
1602
Select PDU (variables) and Com Port (bus) for forwarding
1101
Retrieve Variables in Bulletin Board
1603
Convert PDU to messages
1604
Add routing and header Information
1605
Write messages to com port
1606
Variables from other BB section?
yes
no
1607
Task inactive - Wait for notification

FIGURE 17


Sleep Mode
1701
Network wake-up In Config Mode
1705
Config Mode
1702
1706
Network Reboot In Run Mode
Run Mode
1703
1707
Activation of Debug Mode through Network or local event
Debug / Emergency Mode
1704
1710
Network shutdown In Debug Mode
1709
Network wake-up In Debug Mode
1708
Resume Run Mode through Network or local event
1711
Network wake-up In Run Mode
1712
Network shut down In Run Mode

SYSTEM, METHOD AND COMPUTER PROGRAM PRODUCT FOR SHARING INFORMATION IN A DISTRIBUTED FRAMEWORK

RELATED APPLICATION(S)

This application is a continuation of U.S. patent application Ser. No. 15/405,110 filed Jan. 12, 2017, which is continuation of U.S. patent application Ser. No. 14/011,705 filed Aug. 27, 2013, which is a continuation of U.S. patent application Ser. No. 13/531,319 filed Jun. 22, 2012, now U.S. Pat. No. 8,566,843, which is a continuation of U.S. patent application Ser. No. 12/182,570 filed Jul. 30, 2008, now U.S. Pat. No. 8,209,705, which is a continuation of U.S. patent application Ser. No. 10/737,690 filed Dec. 15, 2003, now U.S. Pat. No. 7,802,263, which, in turn, claims priority under 35 U.S.C. § 119 based on U.S. Provisional Application No. 60/434,018 filed Dec. 17, 2002, all of which are incorporated herein by reference.

FIELD AND BACKGROUND OF THE INVENTION

The present invention relates to the field of distributed control and monitoring systems that may include certain temporal behavior.

Such technology may optionally apply to electronic vehicle communication and control systems, real-time monitoring systems, industrial automation and control systems, as well as any other desired system.

SUMMARY OF THE INVENTION

A system, method and computer program product are provided for sharing information in a distributed system. After information is received, it is stored on a bulletin board. In use, the information is shared, in real-time, among a plurality of heterogeneous processes.

In one embodiment, both past and present instances of the information may be stored on the bulletin board. As an option, the information may be replicated among a plurality of the bulletin boards. Optionally, first information may be processed utilizing a first bulletin board and stored utilizing a second bulletin board. Still yet, the bulletin boards may be hierarchical.

In another embodiment, the processes may access multiple sections of the bulletin board. Further, the bulletin board may send notifications to the processes based on a state of the information on the bulletin board.

Optionally, the information may include variables. For example, the information may include input variables, output variables, etc. Moreover, the processes may include local processes, remote processes, etc. Still yet, the processes may include event triggered processes and/or time triggered processes. In use, each of the processes may process the information in a manner that is isolated from temporal characteristics associated with the network.

In still another embodiment, the information may be extracted from a message received by a bulletin board manager. Moreover, the information may be converted from a signal received by a bulletin board manager. Even still, the information may be shared in a single task, may be shared according to a schedule, and/or may be shared with an operating system. Optionally, dynamic preemptive scheduling may be provided. Also, the information may be shared across the communication network with only a portion of a

message header that is needed for a specific communication link while other communication links may use a different message header.

As a further option, resources in the network may be protected. Specifically, the resources in the network may be protected utilizing a schedule that allows information sharing utilizing the bulletin board. In another embodiment, the resources in the network may be protected utilizing semaphores.

In even still another embodiment, the information may be shared according to an internal clock, an external clock, etc. During operation, objects may be generated based on a change of state of the information stored in the bulletin board. Such objects may include, but are not limited to flags, events, signals, interrupts, etc. Still yet, the information may be stored in response to interrupts associated with the processes.

In use, the bulletin board may update the processes with information at a first rate that differs from a second rate with which the processes send the information to the bulletin board. Optionally, the bulletin board may be accessed with guaranteed access times, jitter, and bandwidth.

In addition, the bulletin board may be updated irregularly and triggered by internal or external objects including, but not limited to flags, events, signals, interrupts, etc. Event triggers may be provided independent of a link connection between nodes where the processes are carried out. Moreover, failure redundancy may be provided through multiple independent links across diverse physical connections.

As yet another option, the information may have a user-configured constraint associated therewith. Such constraint may include a memory constraint, a real-time constraint, etc. As a further option, the constraint may be configured utilizing a tool.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** is a block diagram of an embodiment of a system of one embodiment;

FIG. **2** is a block diagram generally depicting an embodiment of an ECU as part of the system illustrated in FIG. **1**;

FIG. **3** is a block diagram generally depicting an embodiment of a Gateway device as part of the system illustrated in FIG. **1**;

FIG. **4** is a block diagram of an embodiment of the software architecture assumed for one embodiment.

FIG. **5** is a block diagram of an embodiment of the middleware that contains the methods of one embodiment.

FIG. **6** is a block diagram of an embodiment of the bulletin board that describes the process interaction of one embodiment.

FIG. **7** is a block diagram of an embodiment of the bulletin board that describes the process interaction with multiple external communication buses as part of one embodiment.

FIG. **8** is a flow chart diagram of an embodiment of the variable store from remote I/O method of one embodiment.

FIG. **9** is a flow chart diagram of an embodiment of the variable store from local I/O method of one embodiment.

FIG. **10** is a flow chart diagram of an embodiment of the variable method of one embodiment.

FIG. **11** is a flow chart diagram of an embodiment of the variable retrieve method of one embodiment.

FIG. **12** is a flow chart diagram of an embodiment of the application process using the method of one embodiment.

FIG. **13** is a flow chart diagram of an embodiment of the local I/O update from bulletin board method of one embodiment.

FIG. **14** is a flow chart diagram of an embodiment of the variable replication method of one embodiment.

FIG. **15** is a flow chart diagram of an embodiment of the message store from remote gateway method of one embodiment.

FIG. **16** is a flow chart diagram of an embodiment of the message forward to remote ECU or Gateway method of one embodiment.

FIG. **17** is a state transition diagram of an embodiment of the mode switching method of one embodiment.

DETAILED DESCRIPTION

FIG. **1** is a block diagram generally depicting elements of an embodiment of the present distributed embedded communication and computing system. The system architecture may be situated in automotive electronics or industrial control and monitoring systems. In an automotive environment, the various Electronic Control Units (ECUs, **102**) control complex applications such as engine control, brake control, or diagnostics. They are either connected to sensors and actuators via discrete links or simple standard functions such as sensors and actuators are organized into separate sub networks.

These complex functions such as braking, engine-control, etc. are then grouped into the backbone system functions for the car, such as body control, power train and chassis. The backbone also includes the vehicle's high level functions such as diagnostics, telematics and entertainment systems.

Therefore the system is typically hierarchically organized and includes a variety of gateways (**101**,**104**,**105**), which relay messages up and down through the system layers. Each layer may contain multiple electronic control units (ECU, **102**) that are connected through wired serial multiplexing bus-systems such as Controller Area Network (CAN or ISO11898), Flexray, LIN, J1850, J1708, MOST, IEEE1394, and other similar serial multiplexing buses or through wireless multiplexing systems such as IEEE802.11, IEEE802.15, Bluetooth, Zigbee, or similar other wireless links.

Typically, functions provided by an ECU (**102**) are bound to hard real-time temporal behavior. In the context of the present description, real-time may include any response time that may be measured in milli- or microseconds, and/or is less than 1 second.

The ECU may receive a set of real-time input variables from local sensors (**108**), which are connected via discrete signal lines (**113**), or from networked sensors (**106**), which are connected through a multiplexing bus-system (**112**). The ECU may also share variables with other ECUs (**102**) that are either connected on the same physical multiplexing bus or that it can reach through a gateway (**101**,**103**,**104**).

Then the ECU (**102**) processes the input variables and generates a set of output variables that are either shared with other ECUs (**102**) as described above, or which are output to local actuators (**109**), which are connected via discrete signal lines (**113**), or to networked actuators, which are connected through a multiplexing bus (**112**). ECUs (**102**) typically share information with devices that are connected on the same physical multiplexing system. This method of information sharing is called horizontal information sharing in a hierarchical system. Gateways (**101**,**103**,**104**) link multiple physical multiplexing systems together. In the context of the present description, such information may include data, a signal, and/or anything else capable of being stored and shared.

The highest level in the hierarchical system is the system level. The system level gateway (**101**) may be connected to ECUs on the system level multiplexing bus (**117**), to subsequent gateways (**103**) that also link to subsequent communication buses (**110**), and to external components (**120**) that may contain diagnostics devices (**121**), development tools (**122**), other add-on devices (**123**) or other instances of distributed embedded communication and computing systems (**100**). In addition, the system gateway (**101**) may also be connected to an external gateway (**131**) that may link the system to a remote device (**132**) through wireless or wired wide-area-networks such as the Internet, using standard protocols such as UDP/IP, TCP/IP, RTP, HTTP, SOAP, JAVA, etc. or nonstandard proprietary protocols.

Subsequent to the system level may be several layers of groups and subgroups that are link to the higher levels via gateways (**101**,**103**,**104**,**105**).

During the design-time of the system, not all ECUs may exist. Therefore, the development tool (**122**) may provide a plug-in component or virtual ECU/GW (**115**) that directly links into the wired multiplexing bus or wireless network (**110**) and also allows for separate control functions via a tool-link (**116**).

The block diagram in FIG. **2** depicts the detailed elements within a generic ECU (**200**) that is one embodiment of ECU (**102**). The ECU (**200**) typically contains a micro-processor (**201**), volatile memory (**204**) such as RAM, S-RAM or similar, non-volatile memory (**203**) such as EEPROM, FLASH, etc., a real time clock for internal timing of processes (**205**), a watchdog (**206**) to maintain the health of the system, one or more communication bus controllers (**207**) with associated drivers (**208**), digital I/O (**209**) with line drivers (**210**), and analog I/O (**211**) with associated analog signal conditioning (**212**).

In an alternate embodiment, the ECU (**200**) may also contain a wireless communication controller (**311**) and a RF-Front-end (**312**) as outlined in FIG. **3**. The software (**202**) can either be stored in local non-volatile memory (**203**) or partially downloaded via the communication link (**207**,**208**) and stored in the volatile memory. The software is then executed in the microprocessor (**201**).

The block diagram FIG. **3** depicts the detailed elements within a generic gateway (**300**) that is one embodiment of Gateway (**101**,**103**,**104**,**105**) in FIG. **1**.

FIG. **4** outlines one embodiment of the software architecture in an embedded system. The hardware abstraction layer (**405**) allows the system developer to adapt a standard operating system to a specific hardware as used in an ECU (**200**) or gateway (**300**). The hardware abstraction layer (**405**) adapts the real-time operating system (**403**) and the device drivers (**404**) to a specific hardware implementation.

One embodiment includes the middleware (**402**) that has direct access to the real-time operating system (**403**), the device drivers (**404**) and the hardware abstraction layer (**405**). The middleware isolates the application from input/output functions and allows multiple applications to share common variables locally. In addition, the middleware lets applications share variables with remote applications/processes. In the context of the present description, a process may refer to any hardware and/or software operation, etc.

In one embodiment, the middleware can directly interface with the input/output mechanisms of the hardware without utilizing an operating system (**403**) or hardware abstraction layer (**405**).

Another embodiment of the middleware utilizes a pre-emptive multitasking operating system with explicit control of resources. In an alternate embodiment, the middleware can be built with a static multitasking scheme with implicit resource management or be part of a single task system.

Referring now to FIG. **5**, the middleware (**402**) contains the bulletin board manager (**501**), a local signal communication interface (**503**), a remote message communication interface (**504**), and an application programming interface (**502**). The application interface (**502**) provides methods and data interfaces to a plurality of applications. In one embodiment, the application interface is an object library that can be linked to an application at design time.

The bulletin board manager (**501**) contains an upgrade and configuration manager (**507**), an event manager (**505**), a data access manager (**508**), and a data integrity watchdog (**506**). The upgrade and configuration manager (**507**) is necessary to configure the data structure of the bulletin board and to make executable code available to individual processing nodes. In the context of the present description, the bulletin board may refer to any database that enables users to send and/or read electronic messages, files, and/or other data that are of general interest and/or addressed to no particular person/process.

The access manager provides access control mechanisms for the code update and configuration mode. It also may control access rights for individual applications at execution time in the run mode.

The event manager (**505**) captures input-output events as variables and generates new events, flags, or signals based on operations on state variables in the bulletin board. Such operations may include test of maximum values, the occurrence of logically combined events, the result of an integrity check, or events and signals that are created based on any other logical or arithmetic computation on the state variables that are stored in the bulletin board. The actual processing of data and manipulation of data may be done in the application that uses the middleware (**402**). The data integrity watchdog analyses the stored state variables for its integrity and generates events or flags if any problem occurs.

The local signal communication interface (**503**) interfaces with the local discrete input/output hardware to update the bulletin board with new variables and to update the input/output interfaces with the state variables from the bulletin board. It also converts state variables to input/output signals and input/output signals to state variables that can be stored in the bulletin board. The conversion process may contain scaling of signals as well as offset compensation. Typically this processing helps to convert I/O signals that are measured in Volt to a physical entity and vice versa. The communication with the local discrete input output system can be triggered by events or signals can be sampled time-triggered based on a cyclic global or local time base.

The remote message communication interface (**504**) interfaces to serial multiplexing interfaces (buses) that are connected to the specific processing node (ECU or Gateway). It extracts variables from a plurality of messaging protocols and stores them in the database. It also replicates local bulletin-board state variables to the associated processing nodes by composing the appropriate messages for each communication link. The message transfer can be initiated triggered by a bus event, by a local event, or by a time-triggered mechanism that uses a cyclic local or global time base.

FIG. **6** shows the concept of an extended bulletin board or an embedded real-time database (**601**). In this embodiment the ECU (**102**) or the Gateway (**101**) hosts one or multiple bulletin boards with relational links between the variables in the bulletin boards. The relations are defined by data processing functions that the gateway can operate on bulletin boards to obtain new information that can be stored in yet another bulletin board.

The bulletin board (**601**) may contain but is not limited to events (**607**), real-time variables (**608**), diagnostics data (**609**), configuration parameters (**610**), and firmware (**611**) to upgrade individual components of the executable code or the entire software of a processing node. Each type of information may include one or more sections so that individual processes are not blocked if they access separate sections of data.

The memory of the bulletin board is subdivided into areas that nodes on each external network can read from and write into and other areas that an external network may only read from. The data contained in the bulletin board may be stored in volatile or non-volatile memory. Each data entry may consist of one value or an array of values that also may represent a time series.

In one embodiment, each application process (**603**), local signal communication process (**605**), remote message communication process, and the bulletin manager (**602**) can individually access the bulletin board using operating system functions for resource management that may include semaphores, events, signals, call-back routines, flags, etc. in an alternate embodiment of the system the bulletin-board manager controls all interaction with the bulletin-board and all applications have to pass data to the bulletin-board manager. This approach simplifies the interaction with the bulletin board, but adds delay time and jitter to the state variables.

At design time, various hierarchies of memory management can be applied. In practice it is more efficient to allow each sub network and subsystem to place system variable data into local bulletin boards. This is because many system variables are primarily used only within their subsystem or sub network. By placing local information in a shared memory (local bulletin board), it can be used by multiple processes on this processor node. A group bulletin board allows devices on a sub-network to share information with a minimum of network traffic. A system bulletin board allows access to system-wide variables and information.

FIG. **7** illustrates the logical architecture of the interconnection between three heterogeneous network controllers (**702**, **703**, **704**), the associate Operating System interfaces (**705**), the remote message communication process (**706**), the bulletin board (**608**), and the application process (**606**). The connection to each communication controller is fundamentally implemented at the physical interface (the wire, fiber or electromagnetic wireless interface). Each of the higher level layers (data link, network, etc) in the communication interface (**705**) deals with specific features of the individual communication process. In practice these layers are typically represented in a message by "header" bits that contain information about that layer of the network being used to send the message.

Using this model, each communicated message may be processed at each layer to remove (and use) the associated header information for that level. Once all layers are processed the remaining packet data unit (PDU) represents the datum or core information carried by the overall message. In one embodiment, each communication controller has an associated communication interface and an associated remote message conversion mechanism. For instance com-

munication bus controller **2** (**703**) has an associated communication interface **2** (**709**), and an associated remote message conversion 2 (**710**).

This arrangement allows the remote message process (**706**) to directly access information at the data link layer and interface it with the bulletin board. A network layer is not necessary. The remote message communication process (**706**) has a multi-network access interface (essentially a processing capability that can interpret and apply the header information for a variety of networks) and the bulletin board read/write memory access. Now, the individual processing nodes do not need to know about the existence of multiple networks. Each variable can be accessed from all connected physical networks in their proprietary format. Thus the normalization of the information has only to be handled at the gateway through replication of stored data to multiple attached networks.

Continuing with FIG. **7**, the communication procedure is described. In the given example, an external event (**701**) on communication controller **2** (**703**) triggers the operating system to notify the remote message communication process (**706**) that data is available. The notification may be a flag, a call-back routine, an event, or any other operating signal. The associated remote message conversion method **2** (**710**) extracts the data (e.g. real time variables) from the message PDU and stores the data in the bulletin board (**608**). It may also store the associated event as variable in the bulletin board and signal the bulletin-board event manager that new data is available.

The bulletin event manager then notifies the application process (**606**) with the appropriate mechanism. In addition, the event manager may trigger the sampling of local signals using the local signal communication process (**605**) described in FIG. **6**. Finally the bulletin event manager may trigger the bulletin board manager (**707**) to perform integrity checks or generate additional events based on the change of the state variables.

One embodiment provides a new mechanism for creating an information interconnection between two or more heterogeneous communication networks. In the context of the present description, heterogeneous networks may refer to any different communication networks with at least one aspect that is different.

The approach uses a common, or shared storage system that is connected to all of the system networks through network interfaces. A critically important feature of the bulletin board approach is that the complexity of the bulletin board grows linearly with the number of networks (as opposed to as N(N−1) for the gateway approach), and in one-to-many situations the number of message transformations is half that of the standard networking approach.

In an alternate embodiment of the remote message communication process (**706**) any remote process can access data via a single network interface. This approach requires a network layer in each processing node and therefore adds overhead to communications. To communicate between two heterogeneous networks, this process may then be repeated in reverse by adding back the header information for the various layers of the second network, and eventually putting the message onto the second network's physical link. The remote message communication manager (**706**) then can be simplified to only one message assembly and disassembly mechanism.

FIGS. **8-17** illustrate the method of operation of one embodiment of the present system, and also refer to aspects and elements one embodiment shown in FIGS. **1** through **7**.

FIG. **8** details the remote messaging process (**706**) described in FIG. **7**. Referring now to FIG. **8**, the core process of storing data from remote processes that are communicated through multiplexed communication links, into the bulletin board is described. An external notification or task activation starts the process. Then a message (**802**) is received from the operating system layer.

In an alternate embodiment, the message is directly copied form the input register of the communication controller. Then the process extracts variables from the message. Additional signal adaptation may be necessary. The sub-process **804** stores the variables in the bulletin board. If the process only updates one section of the bulletin board it waits for the next message notification (**806**). If variables in multiple sections need to be updated, the process repeats (**804**).

FIG. **9** shows the data update from local input/output peripherals. The process starts with an internal or external notification or task activation. Typically this process is repeated cyclic triggered by an internal or external real-time clock. When the process is activated, it samples or polls the local input ports that may include analog and digital signals (**902**). Then it converts these signals to real-time variables by using the conversion parameters stored in the bulletin board. The signal conditioning parameters van either be defined at design time or adaptively updated by the application process. Then the process stored the new state variables in the bulletin board using the sub-process (**804**) described above.

FIG. **10** describes the bulletin board store procedure (**804**) in more detail. Before new data can be stored in the bulletin board, the procedure has to request the access right to the common resource, a section of the non-volatile or volatile memory, from the operating system (**1001**). This is called explicit resource management.

If the resource is available, the process gets the resource. If the resource is not available, it may try it again after a waiting period (**1011**) until the resource is available. After a certain time has elapsed (**1009**) beyond a configurable threshold, the temporal behavior of the state variable can't be captured any longer and the middle-ware may send an error notification to the associated process.

After reserving the resource (**1003**), the bulletin board store mechanism (**804**) timestamps the state variable for future temporal reference (**1004**). Then, the bulletin board store procedure (**804**) copies the variables or parameters from its private memory (**1006**) to the shared bulletin-board memory (**601**). Then it releases the bulletin board resource.

In an alternate embodiment, the bulletin board store procedure (**804**) has exclusive access to the bulletin board (**601**) and does not need operations **1002**, **1003**, **1007**, **1009**, **1010**, and **1011** because the resource access is realized through implicit resource management. This can be achieved with either static task scheduling or by allowing only the bulletin board store procedure (**804**) to access the bulletin board (**601**).

FIG. **11** describes the bulletin board retrieve procedure (**1101**) in more detail. Before data can be retrieved from the bulletin board, the procedure has to request the access right to the common resource, a section of the non-volatile or volatile memory, from the operating system (**1102**).

If the resource is available, the process gets the resource. If the resource is not available, it may try it again after a waiting period (**1108**) until the resource is available. After a certain time has elapsed (**1109**) beyond a configurable threshold, the temporal behavior of the state variable can't be captured any longer and the middle-ware may send an error notification to the associated process (**1110**).

After reserving the resource (**1104**), the bulletin board retrieve mechanism (**1101**) copies the variables or parameters from the shared bulletin-board memory (**601**) to its private memory (**1006**). Then, it releases the bulletin board resource.

In an alternate embodiment the bulletin board retrieve procedure (**1101**) has exclusive access to the bulletin board (**601**) and does not need operations **1103**, **1104**, **1106**, **1108**, **1109**, and **1110**. Because the resource access is realized through implicit resource management, this can be achieved with either static task scheduling or by allowing only the bulletin board retrieve procedure (**1101**) to access the bulletin board (**601**).

Referring to FIG. **12**, the application process (**1200**) utilizes the bulletin board retrieve mechanism (**1101**) to access all parameters, events, and real-time variables from the bulletin board. Thus the application process is decoupled from the temporal behavior of the input/output variables and can be triggered by a plurality of events (**1201**).

The application process may retrieve one or multiple sets of variables stored in a plurality of memory sections. Then the application process processes the variables (**1203**) with its method. Because the method is not tied to the location of the input/output variables, the application process can be moved or replicated to a plurality of processing nodes (ECUs or Gateways). After processing the input variables and generating a set of output variables, the application process uses the bulletin board store method (**801**) to update one or a plurality of memory sections in the bulletin board. If the application process is a cyclic procedure, it waits until the next activation occurs (**1205**).

Continuing with FIG. **13**, the update local I/O from bulletin board process (**1300**) utilizes the bulletin board retrieve mechanism (**1101**) to access real-time variables from the bulletin board and convert them to output signals (**1302**) that can be written to the output port (**1303**). The I/O update process may retrieve one or multiple sets of variables stored in a plurality of memory sections. If the I/O update process is a cyclic procedure, it waits until the next activation occurs (**1305**).

FIG. **14** describes the data replication process (**1400**). This process can be triggered by a plurality of notification mechanisms, such as events, alarm signals, internal and external timers, and flags set in the bulletin board. It then selects a subset of variables to be replicated and a communication port (**1402**). Next it retrieves the variables from the bulletin board with mechanism (**1401**) and assembles the messages for the specific communication link (**1403**). The message may include an address or identification number for all bulletin boards and associated processing nodes (ECUs and Gateways).

Finally, it writes the messages to the communication port (**1404**). In an alternate embodiment, it handles the messages to the associated interface procedure of the operating system. Then it repeats the procedure, until all variables are updated on all communication ports. If the data replication process is a cyclic procedure, it waits until the next activation occurs (**1405**).

Referring now to FIG. **15**, the store message from remote processing node (gateway or ECU) process (**1500**) describes how replicated data is stored in the bulletin board. This process can be triggered by a plurality of notification mechanisms, such as internal or remote events, alarm signals, internal and external timers, and flags set in the bulletin board. The communication bus may also issue these notifications. The process (**1500**) then reads a message from the communication port (**1502**), selects a subset of variables to be replicated (**1503**), and stores the variables in the bulletin board with procedure (**801**). In an alternate embodiment, this procedure may also be used to store a packet data unit (PDU) in the bulletin board for later replication on the same or a different communication link.

This store and forward networking mechanism can be implemented without the need for complex networking protocols and is therefore well suited for limited processing power and memory environments. It also works in soft-real time environments when no strict temporal behavior is required. The data store operation (**801**) may be repeated for a plurality of bulletin board sections. If the data replication process is a cyclic procedure, it waits until the next activation occurs (**1505**).

Continuing now with FIG. **16**, the store and forward updating mechanism (**1600**) replicates messages from remote processing nodes to other processing nodes from stored packet data units in the bulletin board. This process can be triggered by a plurality of notification mechanisms (**1601**), such as internal or remote events, alarm signals, internal and external timers, and flags set in the bulletin board. The communication bus may also issue these notifications.

The process (**1600**) then selects a message to be forwarded (**1602**) and the appropriate communication link and retrieves the PDU with the bulletin board retrieve mechanism (**1101**). It then adds the appropriate messages header for the communication link (**1603**) and may add routing information (**1604**). Finally the update process (**1600**) writes the messages to the communication port (**1605**). If the updating process is a cyclic procedure, it waits until the next activation occurs (**1607**).

FIG. **17** describes the various modes that the distributed communications and computing system (**100**) can be operated in. In one embodiment, the system operates in various distinct modes in order to preserve the integrity of the system and still allow for changing the architecture and behavior of the network or the roles of the individual nodes. When the distributed computing and communication system wakes up from the sleep mode (**1701**), it can enter a configuration and upgrade mode (**1702**), an emergency or debug mode (**1704**), or the normal real-time run mode (**1703**). The root node or system gateway in a distributed communication and computing system defines the mode based on the existence of external events, such as an external control command, internal events, a system failure, or failed integrity check.

Referring now to FIG. **1**, the external commands may be generated from a development tool (**122**) or a remote device (**132**) that is connected via a remote gateway (**131**). In an alternate embodiment, each ECU (**102**) or virtual ECU (**115**) can trigger the system to enter a different operating mode.

Continuing with FIG. **17**, in the configuration mode (**1702**), the system software and the information-sharing configuration can be updated via a secure communication link with encrypted commands. Each processing node (ECU or gateway) may have security mechanisms such as a certificate that allows it to identify and authorize another entity (remote gateway, remote ECU, or development tool) to make changes to its bulletin board parameters.

The remote entity may also download a new firmware to the bulletin board. The ECU or gateway can store this new firmware in its non-volatile memory while it backs up the original image on the bulletin board for the case that the new software is not functional. In the update mode, the distributed system can also reconfigure the communication and computing infrastructure based on a new set of parameters that need to be stored in the individual bulletin boards.

In the normal run mode (**1703**), the system operates in the real-time information sharing mode and network configuration and certain parameters can't be changed. That protection allows defining deterministic temporal behavior on all communication links. But any processing node may enter a debug/emergency mode (**1704**) if a failure or other qualifying event occurs.

In the emergency mode, a processor executes an alternate procedure that maintains the temporal behavior on the communication links but may reduce or increase the amount of information shared with other processors. It also lets other processing nodes check on the integrity of sensors and actuators. In the maintenance and upgrade mode, an external system can upgrade executable code images and the bulletin-board configuration via secure communication links.

A system and method are thus provided for sharing information within a distributed embedded communications and computing system and with components outside the embedded system. The information sharing mechanism relies on a bulletin board that may include a small database operating under hard real-time conditions with minimal delays, communication latency, and jitter. The embedded database or bulletin board isolates a real-time application in a Electronic Control Unit (ECU) from various other real time applications and from input output signals in the same module (local information sharing), from event-triggered communications with applications in other modules, and from time-triggered communications with applications in other modules.

One design criteria of the database is that the temporal behavior of communications does not impact the real-time computing task and provides enough information access performance at peak time demand. Typically, distributed embedded systems consist of a static structure that can be analyzed at design time. In addition to the real-time operation, the proposed method for information sharing also provides access to the parameters of the embedded system and allows for software upgrades of certain modules.

The present embodiment addresses the shortcomings of traditional computer networks with following enhancements:

1) The concept of multi-mode storage that links two or more communication networks via a bulletin board. The bulletin board is a multi-mode storage that can be thought of an extension to shared memory that can be accessed by local and remote processes at attached networks. There may be multiple hierarchical layers of bulletin boards depending on the topology of the communication system. The bulletin board increases the network efficiency by reducing the number of transactions needed to access remote variables.

2) The concept of a direct-access bulletin board that does not require a network layer translation of messages on each node of the network. Even though this approach restricts the reach of each node to only adjacent nodes and the next gateway, this still allows cross-network variable sharing though vertical real-time replication of data.

3) The concept of hierarchical bulletin board management that allows restriction of information access to certain levels in a network, but still allows the replication of information to other nodes in the network. This paradigm follows the path of reducing the information amount from the leaves of the network to central control and diagnosis hubs.

4) The concept that a gateway can host an assembly of bulletin boards or embedded database that allows operations on bulletin boards to generate events for associated processes. This extension allows definition of a set of data processing operations that would be done once in a network and would be instantly available for connected nodes. Examples for operations are sensor data state observers, diagnostics, integrity checks, fail-safe mechanisms, etc.

5) The concept that an embedded communication and computing network can run in multiple modes in order to provide for a guaranteed deterministic behavior of the system. This property can be achieved by only allowing change to the configuration and/or the functions (SW code) in a secured configuration and upgrade mode. If the network is booted in the normal operating mode, all processors execute the existing code and only allow data sharing through the bulletin boards. The emergency or debug mode lets the network run in a fail-safe reduced operation mode or in a diagnostic mode that allows inspection of the system, while it is running. For each operating mode, the gateway can store a processing image on the bulletin board. The advantage of this procedure is that only the communication hubs need to deal with secure data transfer and encryption while the peripheral nodes in the network can be relative simple in design.

6) The concept of designing the topology of a distributed computing and communication system independent of the definition of the individual functions that the network performs. Each processing task is only associated with a bulletin board, but isolated from I/O processing.

Of course, these are all optional embodiments/enhancements.

While various embodiments have been described above, it should be understood that they have been presented by the way of example only, and not limitation. Thus, the breadth and scope of a preferred embodiment should be not limited by any of the above described exemplary embodiments, but should be defined only in accordance with the following claims and their equivalents.

What is claimed is:

1. An apparatus, comprising:

an automotive electronic control unit including a non-transitory memory storage comprising instructions, and at least one hardware processor in hardwired communication with the memory storage, wherein the at least one hardware processor executes the instructions to:

identify information associated with a message received utilizing a Flexray network protocol associated with a Flexray network that is a physical network;

in response to the identification of the information, issue a storage resource request in connection with a storage resource of the automotive electronic control unit and determine whether the storage resource is available for storing the information, where the storage resource is configured to store the information that is received utilizing the Flexray network for the purpose of sharing the information utilizing a Controller Area Network that is another physical network;

determine whether a threshold has been reached in association with the storage resource request;

in the event the storage resource is not available and the threshold associated with the storage resource request has not been reached, issue another storage resource request in connection with the storage resource;

in the event the storage resource is not available and the threshold associated with the storage resource request has been reached, send a notification;

in the event the storage resource is available, store the information utilizing the storage resource; and

share the information utilizing a Controller Area Network protocol associated with the Controller Area Network, the automotive electronic control unit remaining in hardwired communication with the Flexray network and the Controller Area Network, and including:

a first interface for interfacing with the Flexray network, the first interface including a first interface-related data link layer component that uses Flexray network-related data link layer header bits and a first interface-related network layer component that uses Flexray network-related network layer header bits; and

a second interface for interfacing with the Controller Area Network, the second interface including a second interface-related data link layer component that uses Controller Area Network-related data link layer header bits and a second interface-related network layer component that uses Controller Area Network-related network layer header bits;

wherein the automotive electronic control unit is configured such that:

the second interface-related network layer component uses the Controller Area Network-related network layer header bits by adding the Controller Area Network-related network layer header bits to a data unit including the information, and then the second interface-related data link layer component uses the Controller Area Network-related data link layer header bits by adding the Controller Area Network-related data link layer header bits to the data unit, before communicating the data unit on a physical link of the Controller Area Network;

the first interface-related data link layer component uses the Flexray network-related data link layer header bits by removing the Flexray network-related data link layer header bits from another data unit, and the first interface-related network layer component uses the Flexray network-related network layer header bits by removing the Flexray network-related network layer header bits from the another data unit, where the information is extracted from the another data unit before the sharing; and

a duration between the information being received at the automotive electronic control unit, and the sharing being completed by arriving at a destination, is less than one millisecond.

2. The apparatus as recited in claim **1**, wherein the automotive electronic control unit is configured such that the information is also shared in less than the one millisecond utilizing, in addition to the Controller Area Network protocol, a Local Interconnect Network protocol associated with a Local Interconnect Network, and the automotive electronic control unit remains in hardwired communication with the Local Interconnect Network via a third interface for interfacing with the Local Interconnect Network, the third interface including a third interface-related data link layer component that uses Local Interconnect Network-related data link layer header bits.

3. The apparatus as recited in claim **1**, wherein the automotive electronic control unit is configured such that the duration between the information being sent to the automotive electronic control unit and the sharing being completed by the information arriving at a destination, is less than one microsecond.

4. The apparatus as recited in claim **1**, wherein the automotive electronic control unit is configured such that the identifying, the issuing the another storage resource request, and the sending the notification collectively occur in less than one microsecond.

5. The apparatus as recited in claim **1**, wherein the automotive electronic control unit is configured such that the identifying, the issuing of the storage resource request, the storing the information, and an initiation of the sharing collectively occur in less than one microsecond.

6. The apparatus as recited in claim **1**, wherein the automotive electronic control unit includes a gateway for concurrently sending the information to the Controller Area Network in addition to at least one other network.

7. The apparatus as recited in claim **1**, wherein the automotive electronic control unit is configured such that the information is shared without multiplexing.

8. The apparatus as recited in claim **1**, wherein the automotive electronic control unit is configured such that the information is shared by the automotive electronic control unit from a first node on the Flexray network directly to a second node on the Controller Area Network.

9. The apparatus as recited in claim **1**, wherein the automotive electronic control unit is configured such that the information is shared by the automotive electronic control unit from a first node on the Flexray network to a second node on the Controller Area Network without involving any other nodes.

10. The apparatus as recited in claim **1**, wherein the automotive electronic control unit is configured such that the storage resource is not a buffer.

11. The apparatus as recited in claim **1**, wherein the automotive electronic control unit includes a gateway for concurrently sending the information to the Controller Area Network in addition to a Local Interconnect Network; the information is shared without multiplexing; and the storage resource is not a buffer.

12. The apparatus as recited in claim **11**, wherein the automotive electronic control unit is configured such that the information is shared by the automotive electronic control unit from a first node on the Flexray network to a second node on the Controller Area Network without involving any other nodes.

13. The apparatus as recited in claim **11**, wherein the automotive electronic control unit is configured such that other information is shared utilizing a Local Interconnect Network protocol associated with the Local Interconnect Network, and the automotive electronic control unit remains in hardwired communication with the Local Interconnect Network via a third interface for interfacing with the Local Interconnect Network, the third interface including a third interface-related data link layer component that uses Local Interconnect Network-related data link layer header bits.

14. The apparatus as recited in claim **13**, wherein the automotive electronic control unit is configured such that a duration between the other information being received at the automotive electronic control unit, and the sharing of the other information being completed by arriving at another destination, is less than one millisecond.

15. An apparatus, comprising:

an automotive electronic control unit including a non-transitory memory storage comprising instructions, and at least one hardware processor in hardwired communication with the memory storage, wherein the at least one hardware processor executes the instructions to:

identify information associated with a message received utilizing a Controller Area Network protocol associated with a Controller Area Network that is a physical network;

in response to the identification of the information, issue a storage resource request in connection with a storage resource of the automotive electronic control unit and determine whether the storage resource is available for storing the information, where the storage resource is configured to store the information that is received utilizing the Controller Area Network for the purpose of sharing the information utilizing a Flexray network that is another physical network;

determine whether a threshold has been reached in association with the storage resource request;

in the event the storage resource is not available and the threshold associated with the storage resource request has not been reached, issue another storage resource request in connection with the storage resource;

in the event the storage resource is not available and the threshold associated with the storage resource request has been reached, send a notification;

in the event the storage resource is available, store the information utilizing the storage resource; and

share the information utilizing a Flexray network protocol associated with the Flexray network;

wherein the automotive electronic control unit is in hardwired communication with the Controller Area Network and the Flexray network and includes:

a first interface in hardwired communication with the Controller Area Network, the first interface including a first interface-related data link layer component that uses Controller Area Network-related data link layer header bits and a first interface-related network layer component that uses Controller Area Network-related network layer header bits; and

a second interface in hardwired communication with the Flexray network, the second interface including a second interface-related data link layer component that uses Flexray network-related data link layer header bits and a second interface-related network layer component that uses Flexray network-related network layer header bits;

wherein the automotive electronic control unit is configured such that:

the second interface-related network layer component uses the Flexray network-related network layer header bits by adding the Flexray network-related network layer header bits to a data unit including the information, and then the second interface-related data link layer component uses the Flexray network-related data link layer header bits by adding the Flexray network-related data link layer header bits to the data unit, before communicating the data unit on a physical link of the Flexray network;

the first interface-related data link layer component uses the Controller Area Network-related data link layer header bits by removing the Controller Area Network-related data link layer header bits from another data unit, and the first interface-related network layer component uses the Controller Area Network-related network layer header bits by removing the Controller Area Network-related network layer header bits from the another data unit, where the information is extracted from the another data unit before the sharing; and

a duration between the information being received at the automotive electronic control unit, and the sharing being initiated at the automotive electronic control unit, is less than one microsecond.

16. The apparatus as recited in claim **15**, wherein the automotive electronic control unit is configured such that the information is shared by the automotive electronic control unit from a first node on the Controller Area Network directly to a second node on the Flexray network.

17. The apparatus as recited in claim **15**, wherein the automotive electronic control unit is configured such that the information is shared by the automotive electronic control unit from a first node on the Controller Area Network to a second node on the Flexray network without involving any other nodes.

18. The apparatus as recited in claim **15**, wherein the automotive electronic control unit includes a gateway for multicasting the information to the Flexray Network in addition to at least one other different network.

19. The apparatus as recited in claim **15**, wherein the automotive electronic control unit includes a gateway that is configured for multicasting the information to the Flexray Network in addition to at least one other different network including a Local Interconnect Network.

20. The apparatus as recited in claim **15**, wherein the automotive electronic control unit is configured such that other information is shared utilizing a Local Interconnect Network protocol associated with a Local Interconnect Network, and the automotive electronic control unit remains in hardwired communication with the Local Interconnect Network via a third interface for interfacing with the Local Interconnect Network, the third interface including a third interface-related data link layer component that uses Local Interconnect Network-related data link layer header bits, where a duration between the other information being received at the automotive electronic control unit and the sharing being initiated at the automotive electronic control unit utilizing the Local Interconnect Network protocol is less than one microsecond.

21. The apparatus as recited in claim **15**, wherein the automotive electronic control unit is configured such that the storage resource takes a form of storage other than a buffer and the information is shared without only multiplexing.

22. The apparatus as recited in claim **15**, wherein the automotive electronic control unit is configured such that a duration between the information being received at the automotive electronic control unit, and the sharing being completed by arriving at a destination, is less than one microsecond.

23. The apparatus as recited in claim **15**, wherein the automotive electronic control unit is configured such that other information is capable of being shared utilizing a Local Interconnect Network protocol associated with a Local Interconnect Network, and the automotive electronic control unit remains in hardwired communication with the Local Interconnect Network via a third interface for interfacing with the Local Interconnect Network, the third interface including a third interface-related data link layer component that uses Local Interconnect Network-related data link layer header bits.

24. The apparatus as recited in claim **23**, wherein the automotive electronic control unit is configured such that another duration between the other information being received at the automotive electronic control unit, and the sharing of the other information being completed by arriving at a destination, is less than one millisecond.

25. The apparatus as recited in claim **24**, wherein the automotive electronic control unit includes a gateway for concurrently sending the information to the Controller Area Network in addition to at least one other network; the information is shared without multiplexing; and the storage resource is not a buffer.

26. A system, comprising:

a vehicle; and

an automotive electronic control unit installed in the vehicle, the automotive electronic control unit including a non-transitory memory storage comprising instructions, and at least one hardware processor in hardwired communication with the memory storage, wherein the at least one hardware processor executes the instructions to:

identify information associated with a message received utilizing a Flexray network protocol associated with a Flexray network that is a physical network;

in response to the identification of the information, issue a storage resource request in connection with a storage resource of the automotive electronic control unit and determine whether the storage resource is available for storing the information, where the storage resource is configured to store the information that is received utilizing the Flexray network for the purpose of sharing the information utilizing a Controller Area Network that is another physical network;

determine whether a threshold has been reached in association with the storage resource request;

in the event the storage resource is not available and the threshold associated with the storage resource request has not been reached, issue another storage resource request in connection with the storage resource;

in the event the storage resource is not available and the threshold associated with the storage resource request has been reached, send a notification;

in the event the storage resource is available, store the information utilizing the storage resource; and

share the information utilizing a Controller Area Network protocol associated with the Controller Area Network, the automotive electronic control unit remaining in hardwired communication with the Flexray network and the Controller Area Network, and including:

a first interface for interfacing with the Flexray network, the first interface including a first interface-related data link layer component that uses Flexray network-related data link layer header bits and a first interface-related network layer component that uses Flexray network-related network layer header bits; and

a second interface for interfacing with the Controller Area Network, the second interface including a second interface-related data link layer component that uses Controller Area Network-related data link layer header bits and a second interface-related network layer component that uses Controller Area Network-related network layer header bits;

wherein the automotive electronic control unit is configured such that:

the second interface-related network layer component uses the Controller Area Network-related network layer header bits by adding the Controller Area Network-related network layer header bits to a data unit including the information, and then the second interface-related data link layer component uses the Controller Area Network-related data link layer header bits by adding the Controller Area Network-related data link layer header bits to the data unit, before communicating the data unit on a physical link of the Controller Area Network;

the first interface-related data link layer component uses the Flexray network-related data link layer header bits by removing the Flexray network-related data link layer header bits from another data unit, and the first interface-related network layer component uses the Flexray network-related network layer header bits by removing the Flexray network-related network layer header bits from the another data unit, where the information is extracted from the another data unit before the sharing;

the storage resource is not a buffer and the information is shared without multiplexing;

a duration between the information being received at the automotive electronic control unit, and the sharing being initiated at the automotive electronic control unit, is less than one millisecond; and

other information is shared utilizing a Local Interconnect Network protocol associated with a Local Interconnect Network, and the automotive electronic control unit remains in hardwired communication with the Local Interconnect Network via a third interface for interfacing with the Local Interconnect Network, the third interface including a third interface-related data link layer component that uses Local Interconnect Network-related data link layer header bits; where a duration between the other information being received at the automotive electronic control unit, and the sharing being initiated at the automotive electronic control unit utilizing the Local Interconnect Network protocol associated with the Local Interconnect Network, is less than one millisecond.

27. The system as recited in claim **26**, wherein the automotive electronic control unit is configured such that a duration between the information being sent to the automotive electronic control unit and the sharing being completed by the information arriving at a destination, is less than one microsecond.

28. The system as recited in claim **27**, wherein the automotive electronic control unit is configured such that a duration between the other information being sent to the automotive electronic control unit and the sharing being completed by the other information arriving at another destination, is less than one microsecond.

29. A system, comprising:

a vehicle; and

an automotive electronic control unit installed in the vehicle, the automotive electronic control unit including a non-transitory memory storage comprising instructions, and at least one hardware processor in hardwired communication with the memory storage, wherein the at least one hardware processor executes the instructions to:

identify information associated with a message received utilizing a Controller Area Network protocol associated with a Controller Area Network that is a physical network;

in response to the identification of the information, issue a storage resource request in connection with a storage resource of the automotive electronic control unit and determine whether the storage resource is available for storing the information, where the storage resource is configured to store the information that is received utilizing the Controller Area Network for the purpose of sharing the information utilizing a Flexray network that is another physical network;

determine whether a threshold has been reached in association with the storage resource request;

in the event the storage resource is not available and the threshold associated with the storage resource request has not been reached, issue another storage resource request in connection with the storage resource;

in the event the storage resource is not available and the threshold associated with the storage resource request has been reached, send a notification;

in the event the storage resource is available, store the information utilizing the storage resource; and

share the information utilizing a Flexray network protocol associated with the Flexray network;

wherein the automotive electronic control unit is in hardwired communication with the Controller Area Network and the Flexray network and includes:

a first interface in hardwired communication with the Controller Area Network, the first interface including a first interface-related data link layer component that uses Controller Area Network-related data link layer header bits and a first interface-related network layer component that uses Controller Area Network-related network layer header bits; and

a second interface in hardwired communication with the Flexray network, the second interface including a second interface-related data link layer component that uses Flexray network-related data link layer header bits and a second interface-related network layer component that uses Flexray network-related network layer header bits;

wherein the automotive electronic control unit is configured such that:

the second interface-related network layer component uses the Flexray network-related network layer header bits by adding the Flexray network-related network layer header bits to a data unit including the information, and then the second interface-related data link layer component uses the Flexray network-related data link layer header bits by adding the Flexray network-related data link layer header bits to the data unit, before communicating the data unit on a physical link of the Flexray network;

the first interface-related data link layer component uses the Controller Area Network-related data link layer header bits by removing the Controller Area Network-related data link layer header bits from another data unit, and the first interface-related network layer component uses the Controller Area Network-related network layer header bits by removing the Controller Area Network-related network layer header bits from the another data unit, where the information is extracted from the another data unit before the sharing;

a duration between the information being received at the automotive electronic control unit, and the sharing being initiated at the automotive electronic control unit, is less than one microsecond; and

the storage resource takes a form of storage other than a buffer and the information is shared using a technique other than multiplexing;

other information is shared utilizing a Local Interconnect Network protocol associated with a Local Interconnect Network, and the automotive electronic control unit remains in hardwired communication with the Local Interconnect Network via a third interface for interfacing with the Local Interconnect Network, the third interface including a third interface-related data link layer component that uses Local Interconnect Network-related data link layer header bits; where a duration between the other information being received at the automotive electronic control unit, and the sharing being initiated at the automotive electronic control unit utilizing the Local Interconnect Network protocol associated with the Local Interconnect Network, is less than one microsecond.

30. The system as recited in claim **29**, wherein the automotive electronic control unit is configured such that the information is capable of being shared by the automotive electronic control unit from a first node on the Controller Area Network directly to a second node on the Flexray network without involving another automotive electronic control unit therebetween.

* * * * *

US010002036B2

(12) **United States Patent**
Fuchs et al.

(10) **Patent No.: US 10,002,036 B2**
(45) **Date of Patent: *Jun. 19, 2018**

(54) **SYSTEM, METHOD AND COMPUTER PROGRAM PRODUCT FOR SHARING INFORMATION IN A DISTRIBUTED FRAMEWORK**

(71) Applicant: **Stragent, LLC**, Longview, TX (US)

(72) Inventors: **Axel Fuchs**, San Jose, CA (US); **Scott Sturges Andrews**, Petaluma, CA (US)

(73) Assignee: **Stragent, LLC**, Longview, TX (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 66 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **15/405,110**

(22) Filed: **Jan. 12, 2017**

(65) **Prior Publication Data**

US 2017/0123869 A1 May 4, 2017

Related U.S. Application Data

(63) Continuation of application No. 14/011,705, filed on Aug. 27, 2013, now Pat. No. 9,575,817, which is a (Continued)

(51) **Int. Cl.**
G06F 9/54 (2006.01)
H04L 12/26 (2006.01)
(Continued)

(52) **U.S. Cl.**
CPC ***G06F 9/546*** (2013.01); ***B60R 16/0231*** (2013.01); ***G06F 9/542*** (2013.01);
(Continued)

(58) **Field of Classification Search**
CPC G06F 9/542; G06F 9/545; G06F 9/546; H04L 43/10; H04L 67/10; B60R 16/0231
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,230,051 A 7/1993 Quan
5,265,250 A 11/1993 Andrade et al.
(Continued)

OTHER PUBLICATIONS

Reinhard Maier, "Event-Triggered Communication on Top of Time-Triggered Architecture," Proceedings of the 21st Digital Avionics Systems Conference, vol. 2, 13.C.5-1-13.C.5-9 (Oct. 27-31, 2002).
(Continued)

Primary Examiner — Charles E Anya

(57) **ABSTRACT**

A system, method and computer program product are provided for receiving information associated with a message, utilizing a first network protocol associated with a first network and causing a determination as to whether a storage resource is available. In use, in the event the storage resource is available, causing storage of the information utilizing the storage resource and sharing the information in less than one second, utilizing at least one message format corresponding to a second network protocol associated with a second network which is different from the first network protocol wherein the system, method and computer program product are associated with an automotive electronic control unit with at least one gateway function that remains in hardwired communication with the first network and the second network, the automotive electronic control unit having a plurality of interfaces.

127 Claims, 17 Drawing Sheets

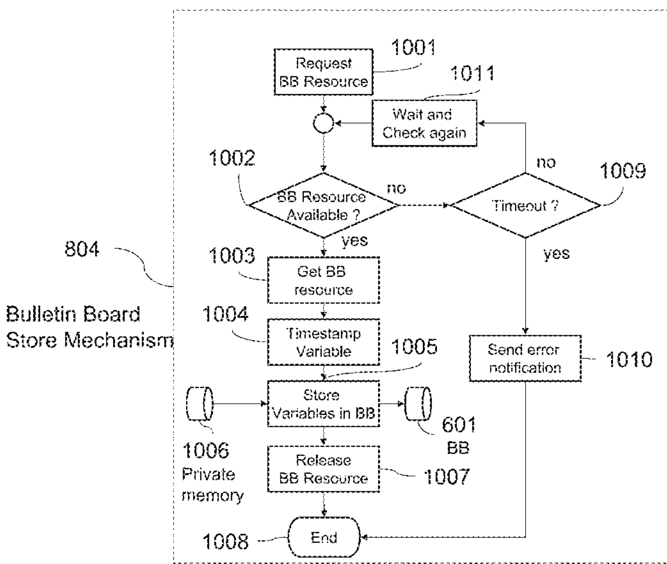


EXHIBIT C

Related U.S. Application Data

continuation of application No. 13/531,319, filed on Jun. 22, 2012, now Pat. No. 8,566,843, which is a continuation of application No. 12/182,570, filed on Jul. 30, 2008, now Pat. No. 8,209,705, which is a continuation of application No. 10/737,690, filed on Dec. 15, 2003, now Pat. No. 7,802,263.

(60) Provisional application No. 60/434,018, filed on Dec. 17, 2002.

(51) **Int. Cl.**
H04L 29/08 (2006.01)
B60R 16/023 (2006.01)

(52) **U.S. Cl.**
CPC ***G06F 9/545*** (2013.01); ***H04L 43/10*** (2013.01); ***H04L 67/10*** (2013.01); *G06F 2209/547* (2013.01)

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,388,189	A	2/1995	Kung	
5,513,324	A	4/1996	Dolin, Jr. et al.	
5,588,002	A	12/1996	Kawanishi et al.	370/462
5,666,485	A	9/1997	Suresh et al.	
5,737,529	A	4/1998	Dolin, Jr. et al.	
5,854,454	A	12/1998	Upender et al.	
5,923,846	A	7/1999	Gage et al.	
5,941,947	A	8/1999	Brown et al.	
5,956,489	A	9/1999	San Andres et al.	709/221
6,034,970	A	3/2000	Levac et al.	370/466
6,128,315	A	10/2000	Takeuchi	
6,141,710	A	10/2000	Miesterfeld	710/110
6,185,466	B1	2/2001	Nicewonger	
6,289,390	B1	9/2001	Kavner	719/310
6,363,427	B1	3/2002	Teibel et al.	
6,378,001	B1	4/2002	Aditham et al.	719/313
6,430,607	B1	8/2002	Kavner	
6,438,632	B1	8/2002	Kikugawa	
6,484,082	B1	11/2002	Millsap et al.	
6,725,348	B1	4/2004	Marier et al.	
6,801,942	B1	10/2004	Dietrich et al.	709/225
6,941,510	B1	9/2005	Ozzie et al.	
7,103,045	B2	9/2006	Lavigne et al.	370/392
7,103,646	B1	9/2006	Suzuki	709/220
7,103,656	B2	9/2006	Lewis et al.	709/223
7,552,440	B1	6/2009	Stewart et al.	719/312
7,950,017	B1	5/2011	Cain et al.	
2001/0018685	A1	8/2001	Saito et al.	
2002/0002586	A1	1/2002	Rafal et al.	
2002/0032856	A1	3/2002	Noguchi et al.	
2002/0047868	A1	4/2002	Miyazawa	
2002/0073243	A1	6/2002	Staiger	719/313
2002/0124007	A1	9/2002	Zhao	
2002/0159460	A1	10/2002	Carrafiello et al.	
2002/0188691	A1	12/2002	Ignatius et al.	
2002/0191543	A1	12/2002	Buskirk et al.	
2002/0198943	A1	12/2002	Zhuang et al.	
2003/0061388	A1	3/2003	Cleghorn et al.	
2003/0074474	A1	4/2003	Roach et al.	
2003/0088568	A1	5/2003	Matsunaga et al.	
2003/0154116	A1	8/2003	Lofton	
2003/0236894	A1	12/2003	Herley	
2004/0003087	A1	1/2004	Chambliss	
2006/0155867	A1	7/2006	Kilian et al.	
2006/0155907	A1	7/2006	Yoshimasa et al.	
2007/0101206	A1	5/2007	Brabant	
2010/0333096	A1	12/2010	Dice et al.	
2011/0047218	A1	2/2011	Nojima et al.	
2011/0047310	A1	2/2011	Bonola	
2012/0029898	A1	2/2012	Carroll et al.	
2013/0111299	A1	5/2013	Hashimoto et al.	

OTHER PUBLICATIONS

William Wong, "Software and Hardware Standards Help, But In-Vehicle Network Growth Will Be Conservative: CAN networks and OSEK/VDX compatible operating systems will drive tomorrow's vehicles," Electronic Design, vol. 49, issue 1, 62 (Jan. 8, 2001).
OSEK/VDX Network Management Concept and Application Programming Interface, Version 2.51 (May 31, 2000).
"An Embedded Software Primer," David E. Simon, 1999.
"Application of In-Vehicle Data Buses for Collision Avoidance Systems," William D. Horne, et al., 1998.
"The Foundation.TM Fieldbus Primer," Fieldbus Inc., Jun. 24, 2001.
"GSA Guide to Specifying Interoperable Building Automation and Control Systems Using ANSI/ASHRAE Standard 135-1995," Steven T. Bushby et al., Nov. 1999.
"Programming in the OSEK/VDX Environment," Joseph Lemieux, 2001.
"OSEK/VDX Binding Specification Version 1.3," OSEK Group, Sep. 17, 2001.
"OSEK/VDX Binding Specification Version 1.4," OSEK Group, Sep. 6, 2002.
"OSEK/VDX Binding Specification Version 1.4.1," OSEK Group, Jan. 21, 2003.
"OSEK/VDX Communication Specification Version 2.2.2," OSEK Group, Dec. 18, 2000.
"OSEK/VDX Communication Specification Version 3.0," OSEK Group, Jul. 26, 2002.
"OSEK/VDX Communication Specification Version 3.0.1," OSEK Group, Jan. 29, 2003.
"OSEK/VDX Communication Specification Version 3.0.2," OSEK Group, Dec. 9, 2003.
"OSEK/VDX Fault-Tolerant Communication Specification Version 1.0," OSEK Group, Jul. 24, 2001.
"OSEK/VDX Network Management Concept and Application Programming Interface Version 2.5.0," OSEK Group, May 31, 1998.
"OSEK/VDX Network Management Concept and Application Programming Interface Version 2.5.1," OSEK Group, May 31, 2000.
"OSEK/VDX Network Management Concept and Application Programming Interface Version 2.5.2," OSEK Group, Jan. 16, 2003.
"OSEK/VDX OSEK Implementation Language Specification Version 2.3," OSEK Group, Sep. 10, 2001.
"OSEK/VDX OSEK Implementation Language Specification Version 2.4," OSEK Group, Dec. 2, 2002.
"OSEK/VDX OSEK Implementation Language Specification Version 2.4.1," OSEK Group, Jan. 23, 2003.
"Goals and Motivation: What is OSEK/VDX," OSEK VDX Portal, 2012.
"OSEK/VDX OSEK Run Time Interface Part A: Language Specification Version 2.1," OSEK Group, Jul. 16, 2001.
"OSEK/VDX OSEK Operating System Specification Version 2.0 revision 1," OSEK Group, Oct. 15, 1997.
"OSEK/VDX OSEK Operating System Specification Version 2.1 revision 1," OSEK Group, Nov. 13, 2000.
"OSEK/VDX OSEK Operating System Specification Version 2.2," OSEK Group, Sep. 10, 2001.
"OSEK/VDX OSEK Operating System Specification Version 2.2. 1," OSEK Group, Jan. 16, 2003.
"OSEK/VDX Time-Triggered Operating System Specification Version 1.0," OSEK Group, Jul. 24, 2001.
Berwanger et al "FlexRay—The Communication System for Advanced Automotive Control Systems" 2001-01-0676, SAE 2001 World Congress, Mar. 5-8, 2001.
Bosch, "CAN Specification Version 2.0" Sep. 1991.
Clarke et al "Formal Verification of Autonomous Systems NASA Intelligent Systems Program" Carnegie Mellon University, Sep. 25, 2001.
Courtois et al "Concurrent Control with "Readers" and "Writers"" Communications of the ACM, vol. 14, No. 10, Oct. 1971.

(56) **References Cited**

OTHER PUBLICATIONS

Fredriksson, Lars-Berno "Controller Area Networks and the Protocol can for Machine Control Systems" Mechatronics vol. 4, No. 2, pp. 159-172, 1994.
Hernandez et al "Algorithms and architectures for real-time control 2000," Proceedings volume from the 6th IFAC Workshop, Palma de Mallorca, Spain, May 15-17, 2000.
Kirrmann et al. "The IEC/IEEE Train Communication Network," IEEE Micro, Mar.-Apr. 2001
Koopman, Philip "Control Area Network (CAN)" 18-540 Distributed Embedded Systems, Presentation Slides, Oct. 4, 2000.
Leen et al, "Digital networks in the automotive vehicle," Computing and Control Engineering 10(6):257-266, Jan. 2000.
IEEE "The Authoritative Dictionary of IEEE Standards Terms" 7th Ed., IEEE Press, 2000.
IEEE "IEEE Standard for Information Technology—Portable Operating System Interface (POSIX®)—Part 1: System Application Program Interface (API)—Amendment d: Additional Realtime Extensions," IEEE Std 1003.1d-1999 (Amendment to IEEE Std 1003.1-1990), Jun. 16, 2000.
Mellor-Crummey et al. "Algorithms for Scalable Synchronization on Shared-Memory Multiprocessors," ACM Transactions on Computer Systems, vol. 9, No. 1, Feb. 1991, pp. 21-65.
Jagannathan et al. "Blackboard Architectures and Applications," vol. 3, Academic Press Inc., 1989, 524 pages.
SAE International, "Surface Vehicle Standard—SAE J1850" Rev May 2001.
Intelligent Transportation Systems Standards, United States Department of Transportation "SAE J2366—Family of Its Data Bus (IDB) Protocol Standards," (online fact sheet written Jul. 12, 2006), available at https://www.standards.its.dot.gov/Factsheets/PrintFactsheet/56.
SAE International, "Surface Vehicle Recommended Practice—SAE J2366-1" Nov. 2001.
SAE International, "Surface Vehicle Recommended Practice—SAE J2366-2" Nov. 2001.
Schill, John "An Overview of the CAN Protocol," Embedded Systems Programming, Sep. 1997, pp. 46-62.
Stewart et al., "Integration of Real-Time Software Modules for Reconfigurable Sensor-Based Control Systems," Proceedings of the 1992 IEEE/RSJ International Conference on Intelligent Robots and Systems, Jul. 7-10, 1992.
Tanenbaum, Andrew S. "Computer Networks," 3rd Ed., Prentice Hall, Inc, 1996, pp. 1-76 (Introduction).
Tanenbaum, Andrew S. "Modern Operating Systems," 2nd Ed., Prentice Hall, Inc., 2001, excerpts from Chapters 2, 7, 8, and 10.
Upender et al., "Embedded Communication Protocol Options," Proceedings of the Fifth Annual Embedded Systems Conference, vol. 2, Miller Freeman, Inc., 1993.
Wargui et al. "Application of Controller Area Network to Mobile Robots," IEEE, Electrotechnical Conference, May 16, 1996, p. 205-207.
Wense et al. "Building Automotive LIN Applications," pp. 279-292, found as part of Krueger et al., "Advanced Microsystems for Automotive Applications 2001," Springer, 2001.
Giusto et al, "Automotive Virtual Integration Platforms: Why's, What's, and How's" Sep. 2002, Proceedings of the 2002 IEEE International Conference on Computer Design: VLSI in Computers and Processors (ICCD'02).
Juan Luis Posadas, et al., "Communications Structure for Sensor Fusion in Distributed Real Time Systems," May 2000, Algorithms and Architectures for Real-Time Control 2000: A Proceedings volume from the 6th IFAC Workshop.
Leen et al. "Expanding Automotive Electronic Systems" Jan. 2002, Computer 35(1).
Posadas et al. "Communications Structure for Sensory Data in Mobile Robots" 2002 , Engineering Applications of Artificial Intelligence, 15:341-350.
SAE, "ITS Data Bus—IDB-C Physical Layer" Nov. 2001, Surface Vehicle Recommended Practice, J2366-1.
Stewart et al. "The Chimera II Real-Time Operating System for Advanced Sensor-Based Control Applications" 1992, IEEE Transactions on Systems, Man, and Cybernetics, vol. 22, No. 6.
U.S. Appl. No. 15/919,201, filed Mar. 12, 2018.
Brad Johanson, Armando Fox, Pat Hanrahan, Terry Winograd, "The Event Heap: An Enabling Infrastructure for Interactive Workspaces," White Paper, Stanford University, Stanford, CA 1999.

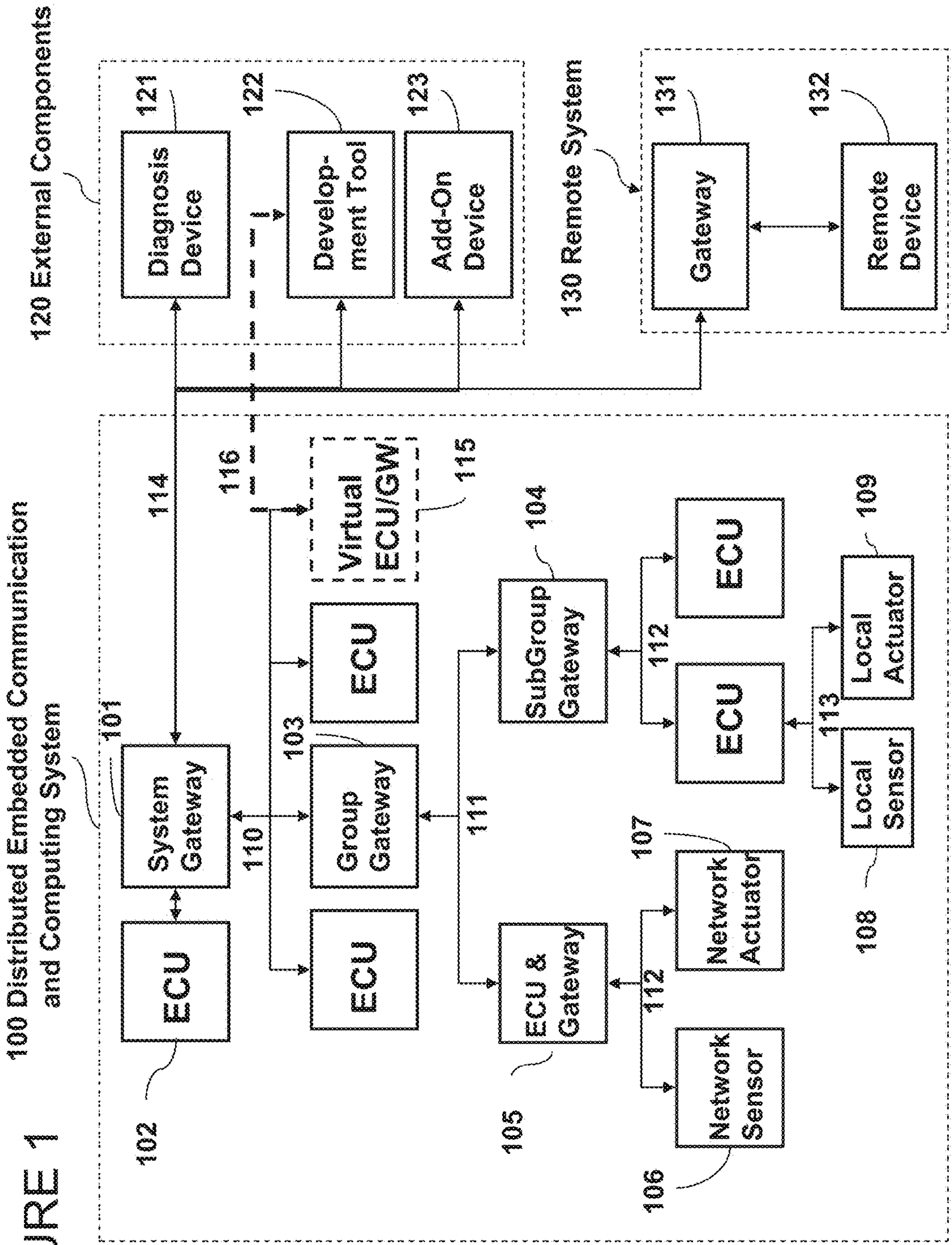

100 Distributed Embedded Communication and Computing System
120 External Components
130 Remote System
ECU
System Gateway
Group Gateway
ECU
ECU
Virtual ECU/GW
ECU & Gateway
SubGroup Gateway
Network Sensor
Network Actuator
ECU
ECU
Local Sensor
Local Actuator
Diagnosis Device
Develop-ment Tool
Add-On Device
Gateway
Remote Device
101
102
103
104
105
106
107
108
109
110
111
112
112
113
114
115
116
121
122
123
131
132

FIGURE 1

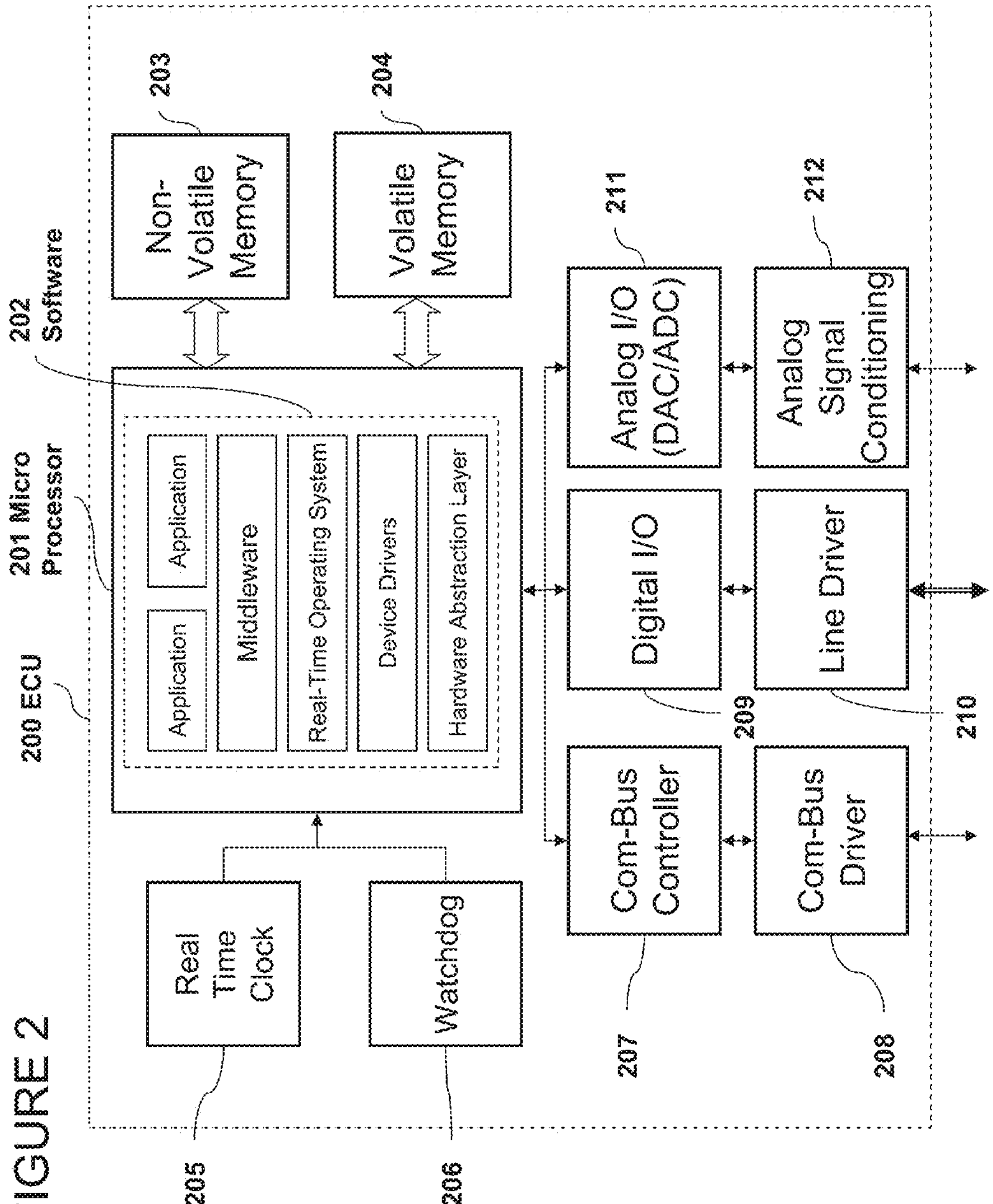

200 ECU
201 Micro Processor
202 Software
203
Non-Volatile Memory
204
Volatile Memory
Application
Application
Middleware
Real-Time Operating System
Device Drivers
Hardware Abstraction Layer
Real Time Clock
Watchdog
205
206
Com-Bus Controller
207
Com-Bus Driver
208
Digital I/O
209
Line Driver
210
Analog I/O (DAC/ADC)
211
Analog Signal Conditioning
212

FIGURE 2

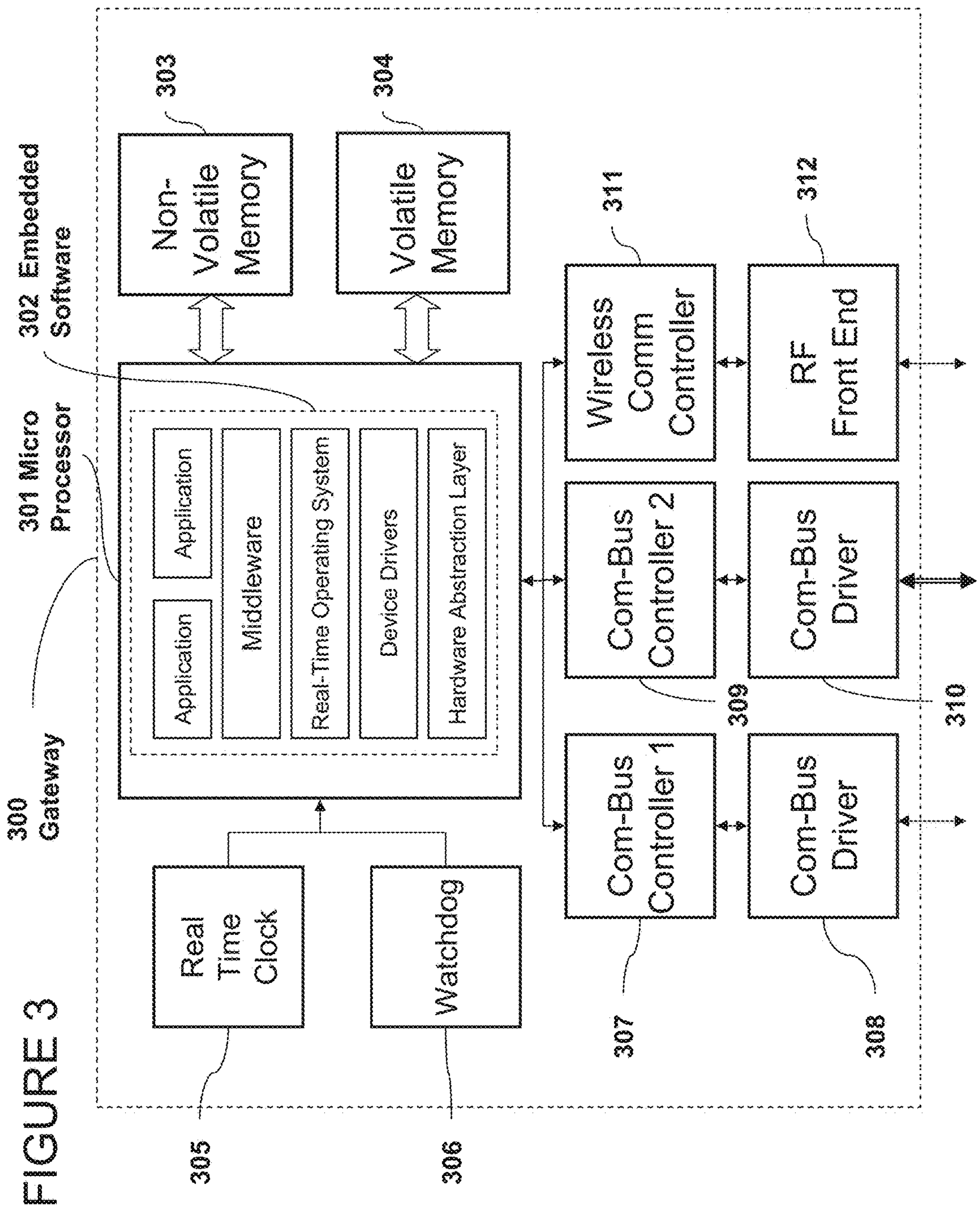

300 Gateway
301 Micro Processor
302 Embedded Software
303
Non-Volatile Memory
304
Volatile Memory
Application
Application
Middleware
Real-Time Operating System
Device Drivers
Hardware Abstraction Layer
Real Time Clock
305
Watchdog
306
Com-Bus Controller 1
307
Com-Bus Driver
308
Com-Bus Controller 2
309
Com-Bus Driver
310
Wireless Comm Controller
311
RF Front End
312

FIGURE 3

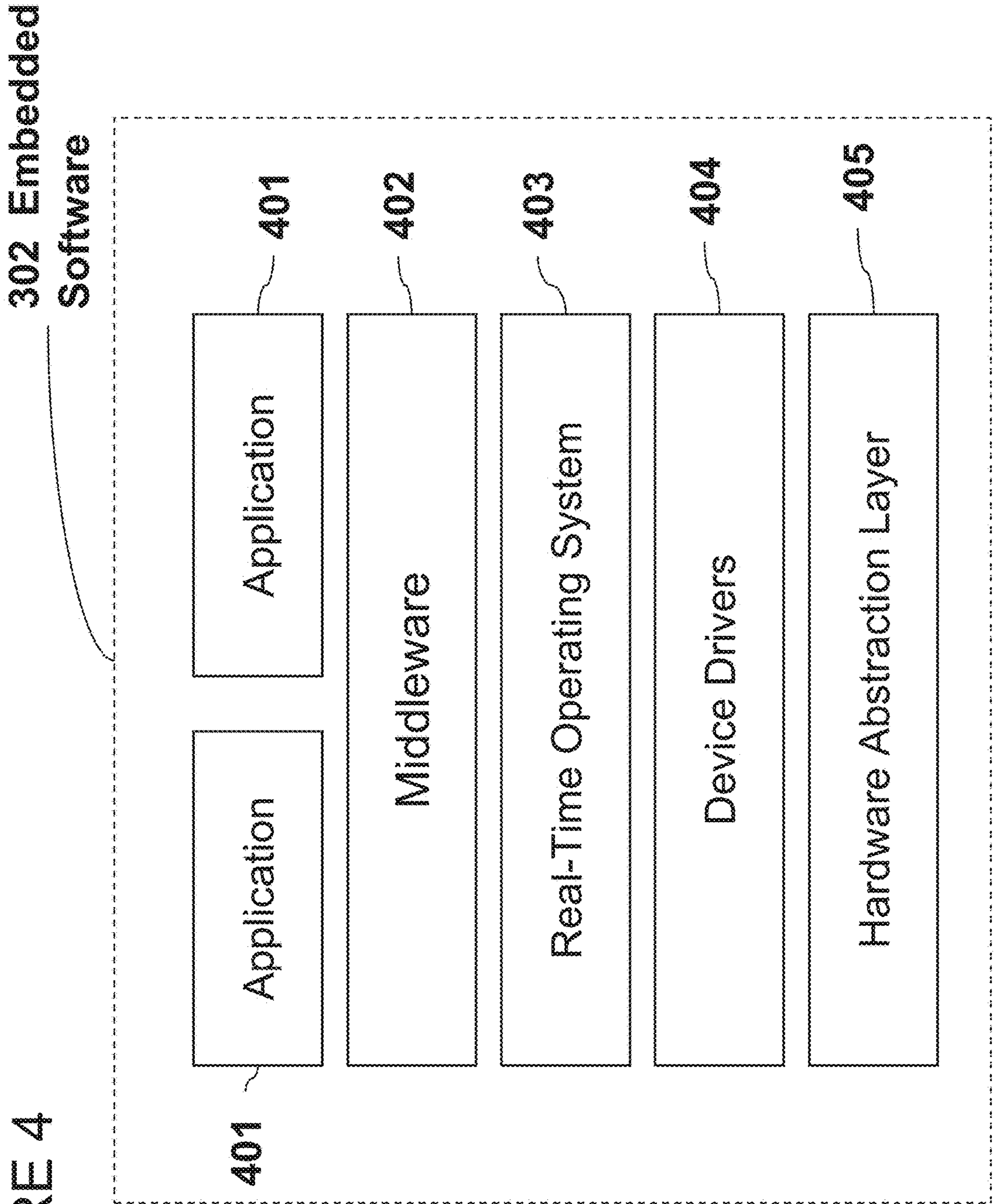

302 Embedded Software
401
Application
401
Application
402
Middleware
403
Real-Time Operating System
404
Device Drivers
405
Hardware Abstraction Layer

FIGURE 4

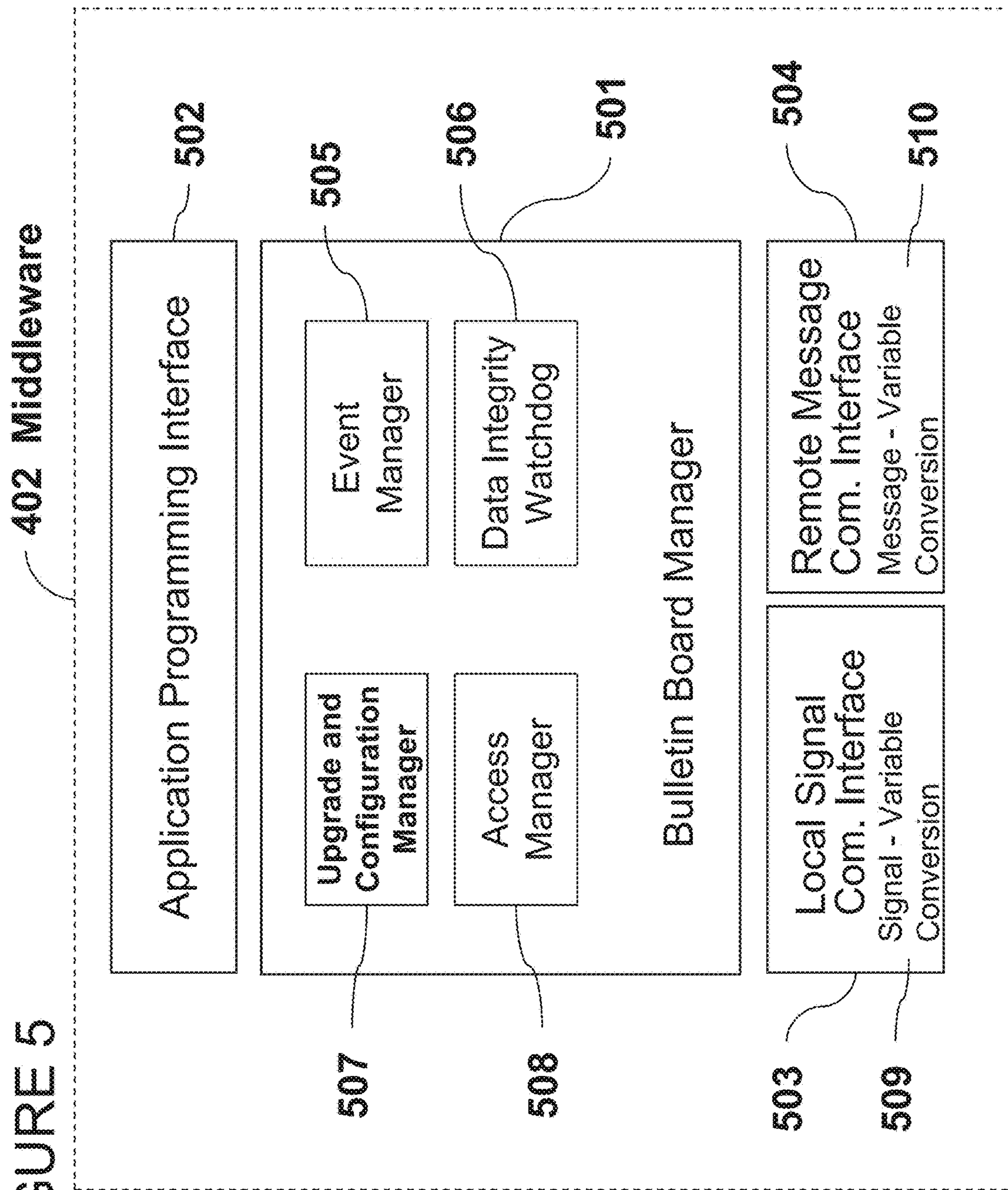

402 Middleware
Application Programming Interface
502
Upgrade and Configuration Manager
507
Event Manager
505
Access Manager
508
Data Integrity Watchdog
506
Bulletin Board Manager
501
Local Signal Com. Interface
Signal - Variable Conversion
503
509
Remote Message Com. Interface
Message - Variable Conversion
504
510

FIGURE 5

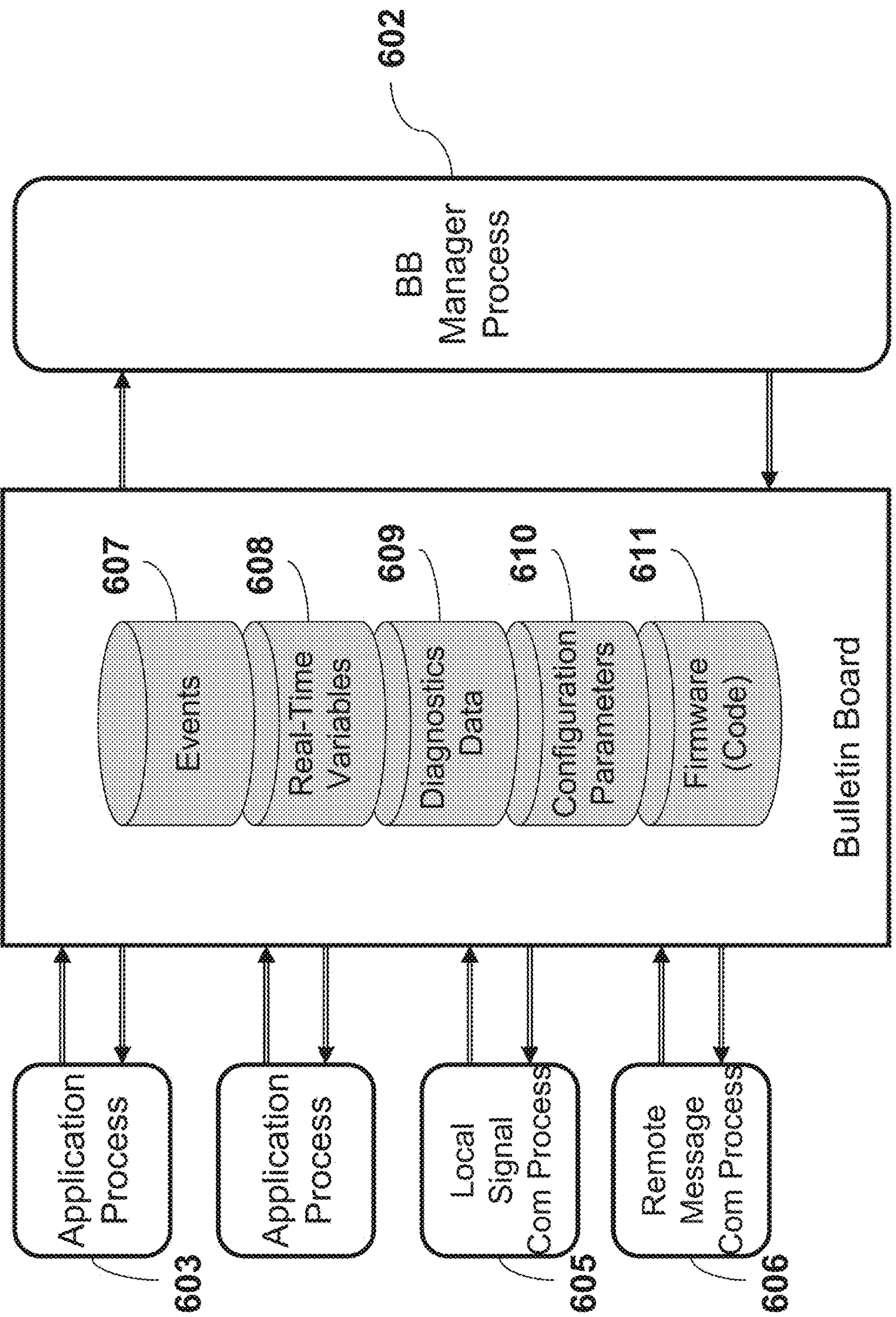

602
BB Manager Process
601
607
608
609
610
611
Events
Real-Time Variables
Diagnostics Data
Configuration Parameters
Firmware (Code)
Bulletin Board
Application Process
Application Process
Local Signal Com Process
Remote Message Com Process
603
605
606

FIGURE 6

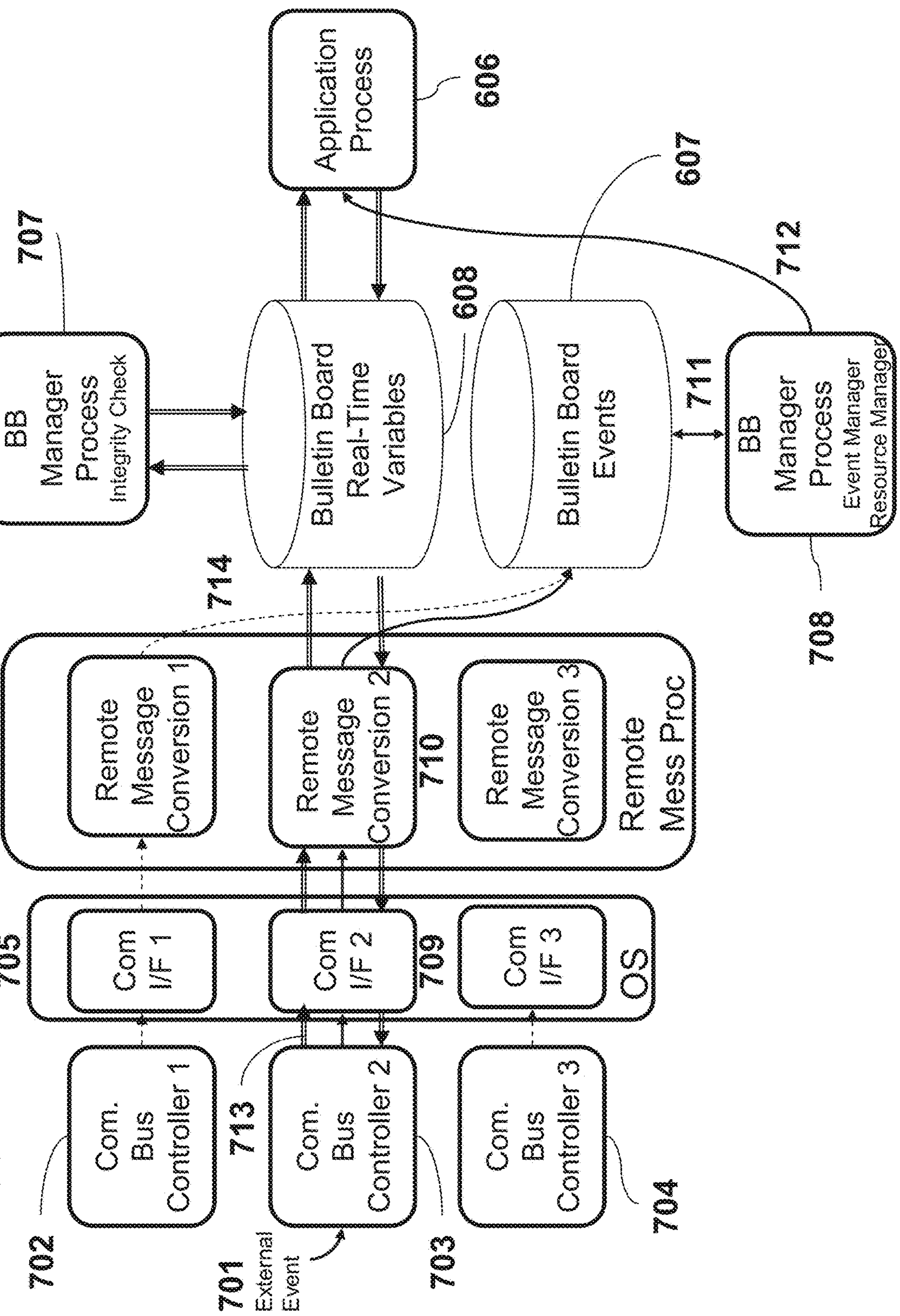

Com. Bus Controller 1
Com. Bus Controller 2
Com. Bus Controller 3
Com I/F 1
Com I/F 2
Com I/F 3
OS
Remote Message Conversion 1
Remote Message Conversion 2
Remote Message Conversion 3
Remote Mess Proc
BB Manager Process Integrity Check
Bulletin Board Real-Time Variables
Bulletin Board Events
BB Manager Process Event Manager Resource Manager
Application Process
701 External Event
702
703
704
705
706
707
708
709
710
711
712
713
714
606
607
608

FIGURE 7

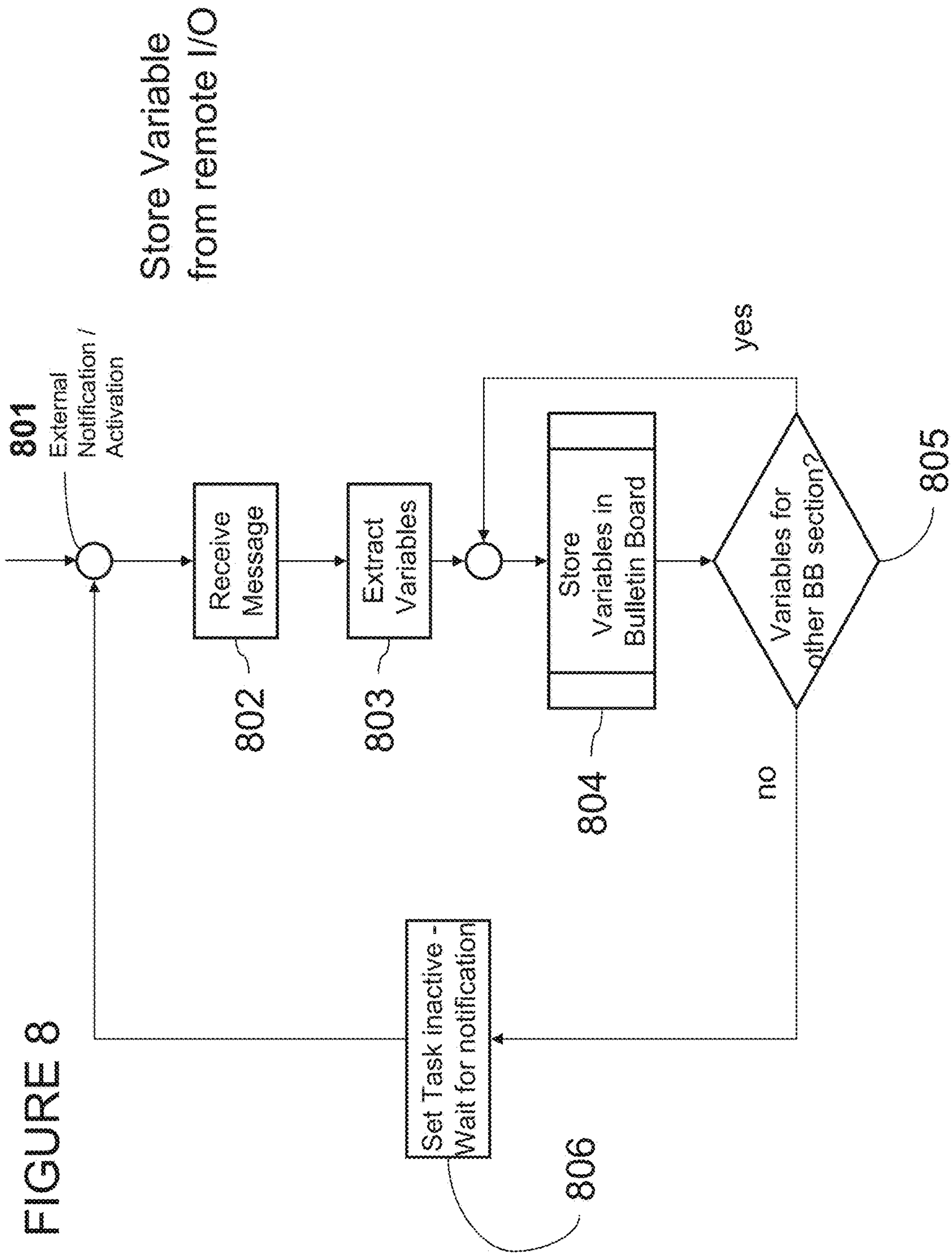

Store Variable
from remote I/O
801
External
Notification /
Activation
Receive
Message
802
Extract
Variables
803
Store
Variables in
Bulletin Board
804
Variables for
other BB section?
805
yes
no
Set Task inactive -
Wait for notification
806

FIGURE 8

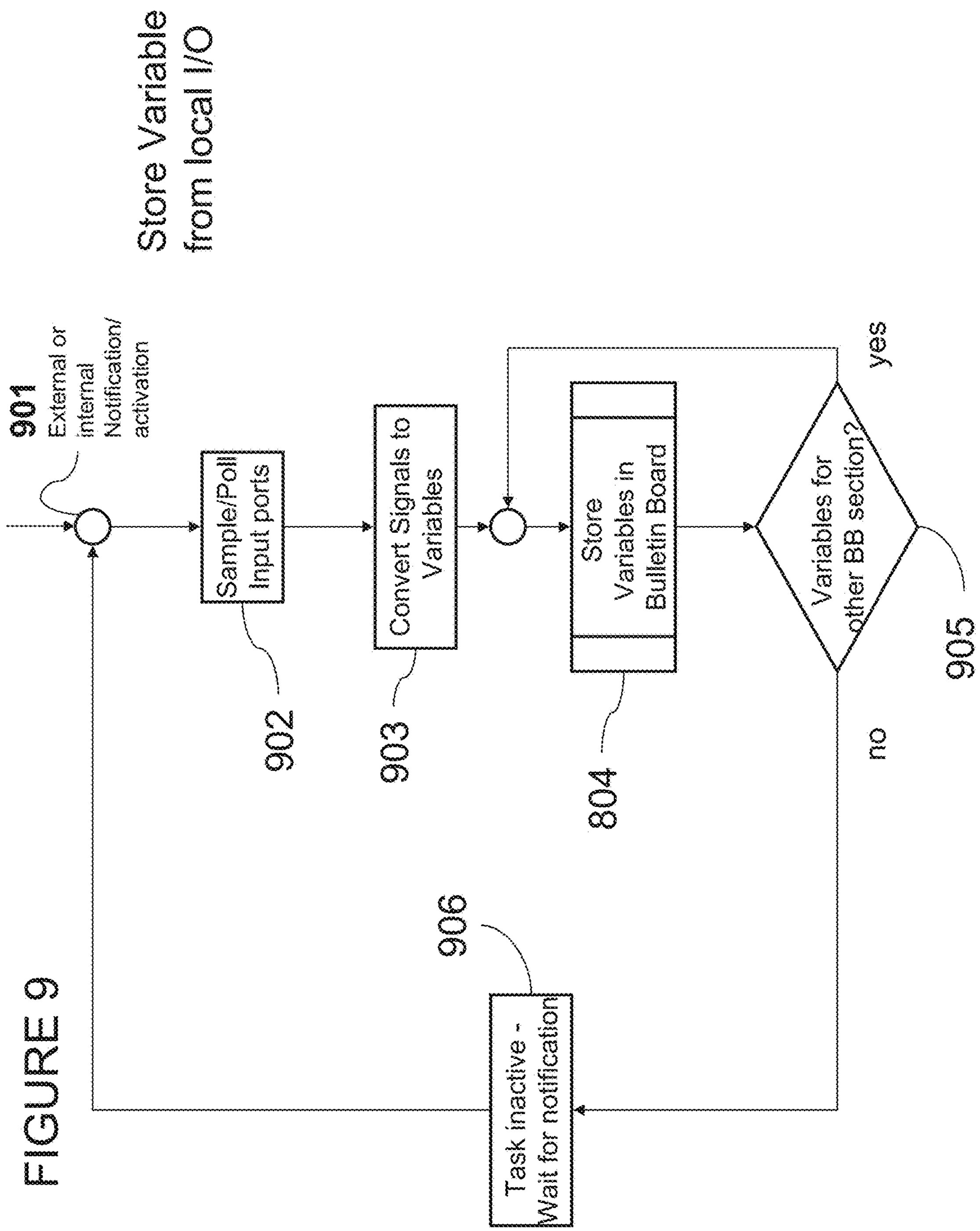

Store Variable
from local I/O
901
External or
internal
Notification/
activation
Sample/Poll
Input ports
902
Convert Signals to
Variables
903
Store
Variables in
Bulletin Board
804
Variables for
other BB section?
905
yes
no
Task inactive -
Wait for notification
906

FIGURE 9

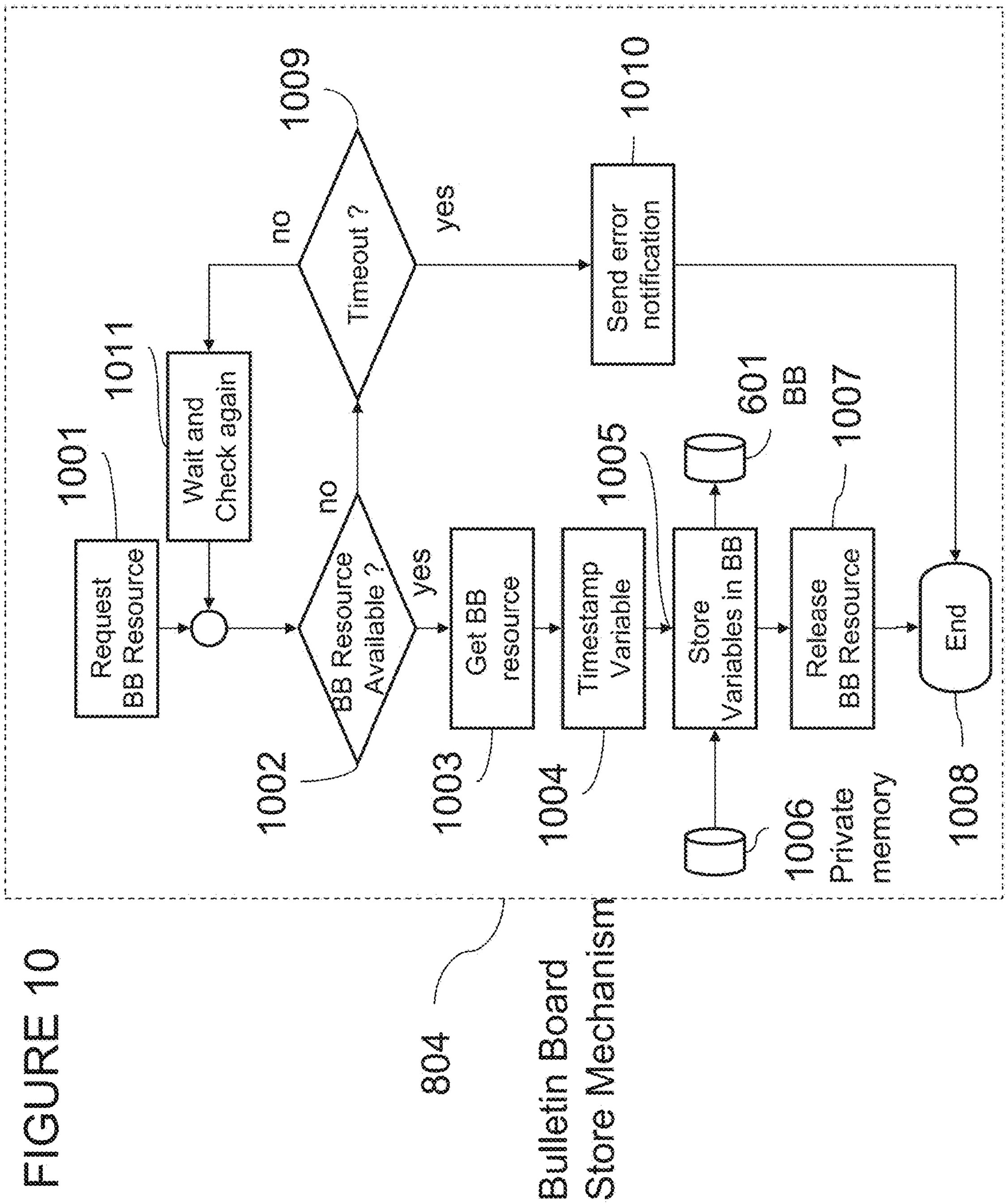

Request BB Resource
1001
Wait and Check again
1011
BB Resource Available ?
1002
no
yes
Timeout ?
1009
no
yes
Send error notification
1010
Get BB resource
1003
Timestamp Variable
1004
Store Variables in BB
1005
601
BB
1006
Private memory
Release BB Resource
1007
End
1008
804
Bulletin Board Store Mechanism

FIGURE 10

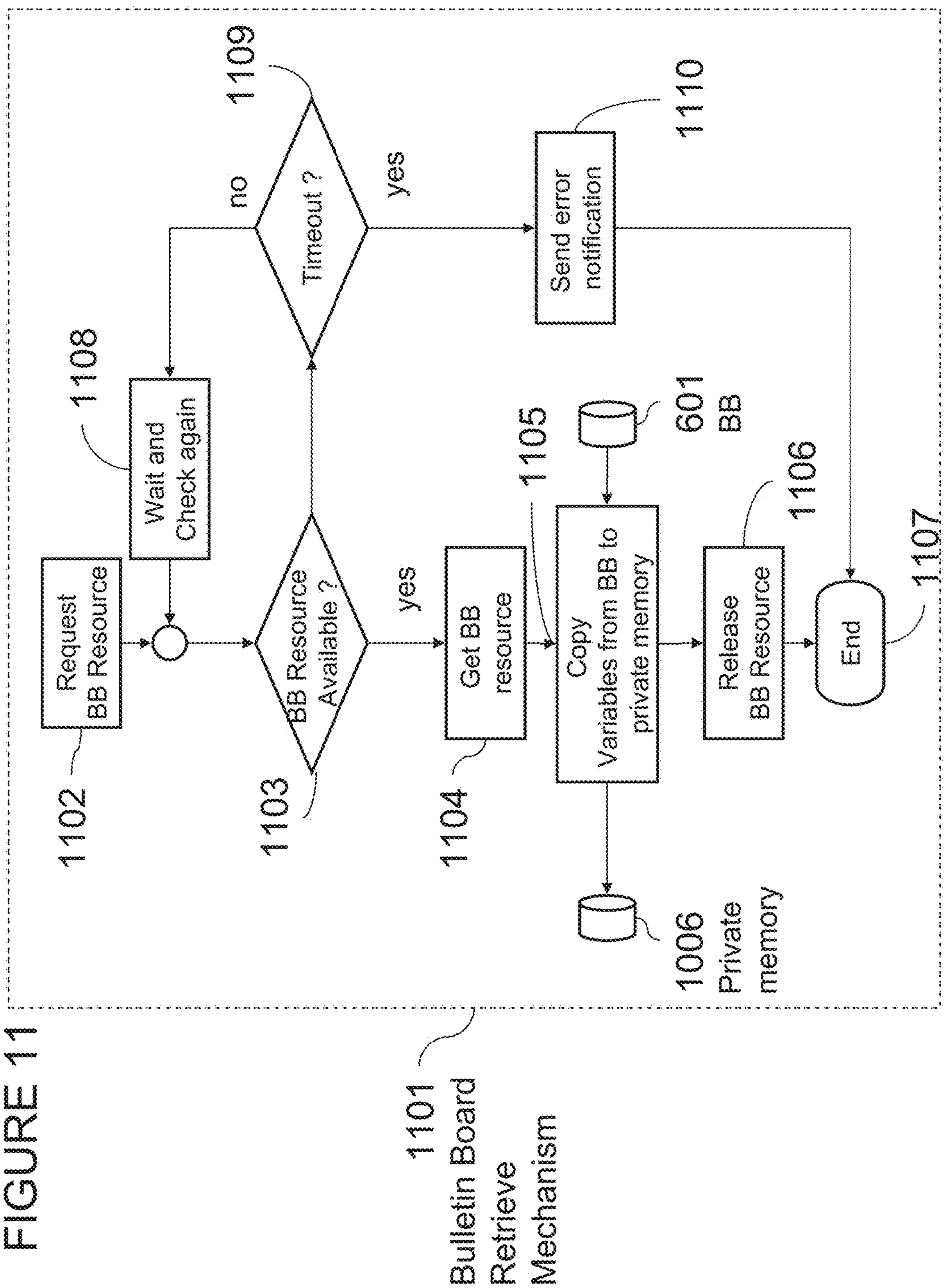

Request BB Resource
1102
Wait and Check again
1108
BB Resource Available ?
1103
no
Timeout ?
1109
yes
Send error notification
1110
yes
Get BB resource
1104
1105
Copy Variables from BB to private memory
601
BB
1006
Private memory
Release BB Resource
1106
End
1107
1101
Bulletin Board Retrieve Mechanism

FIGURE 11

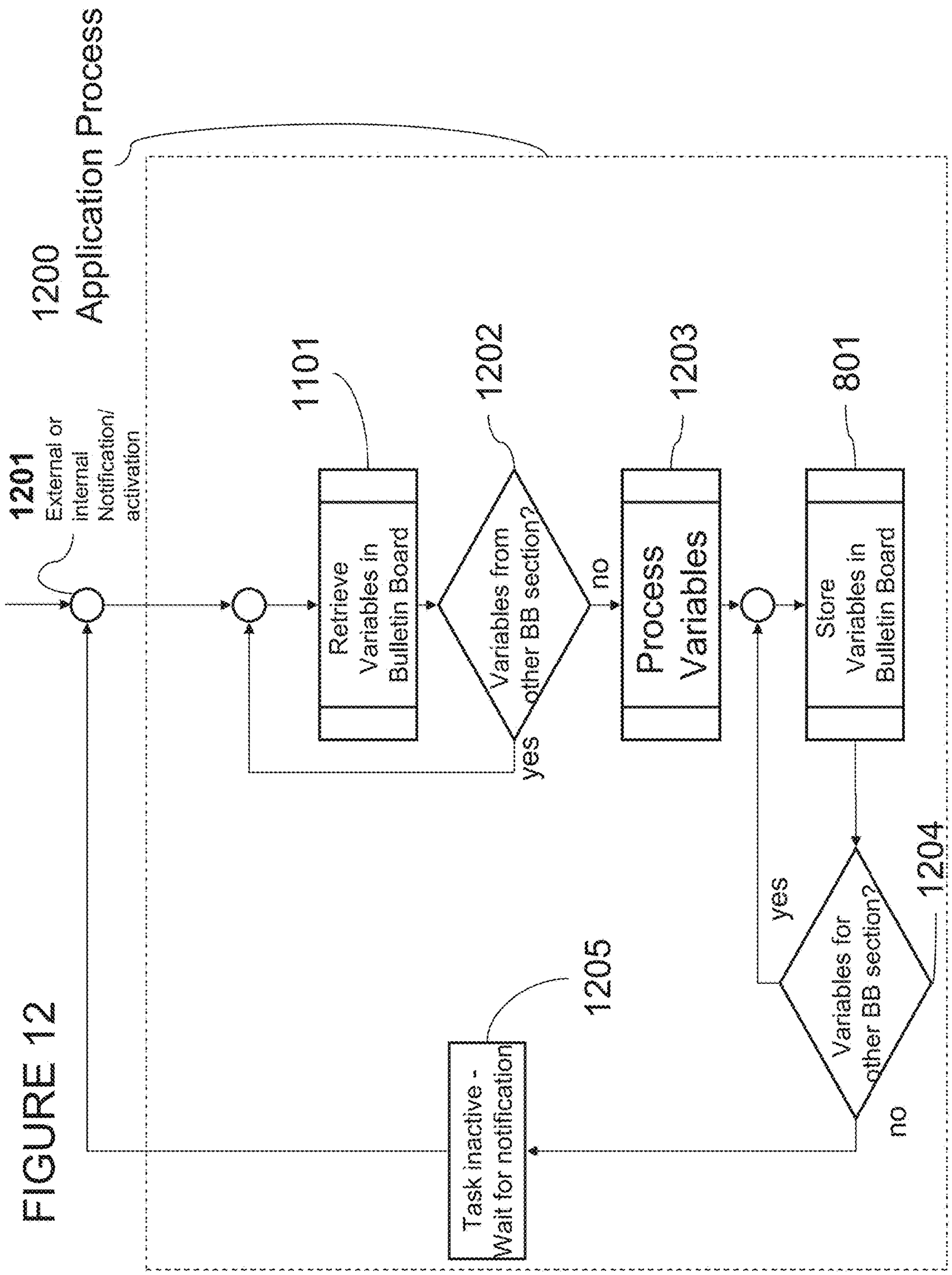

1200
Application Process
1201
External or internal Notification/ activation
1101
Retrieve Variables in Bulletin Board
1202
Variables from other BB section?
yes
no
1203
Process Variables
801
Store Variables in Bulletin Board
1204
Variables for other BB section?
yes
no
1205
Task inactive - Wait for notification

FIGURE 12

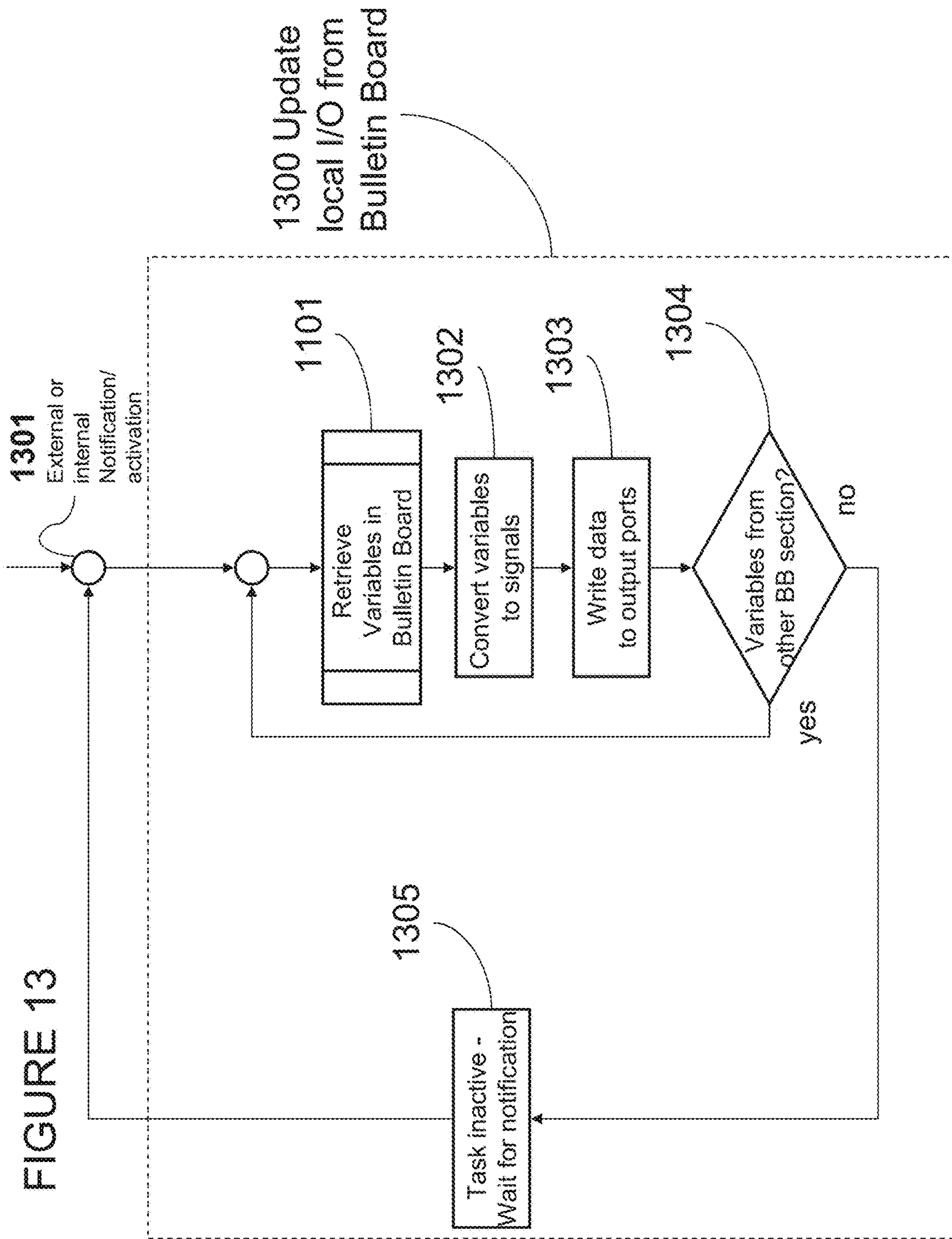

1300 Update local I/O from Bulletin Board
1301 External or internal Notification/ activation
1101
Retrieve Variables in Bulletin Board
1302
Convert variables to signals
1303
Write data to output ports
1304
Variables from other BB section?
yes
no
1305
Task inactive - Wait for notification

FIGURE 13

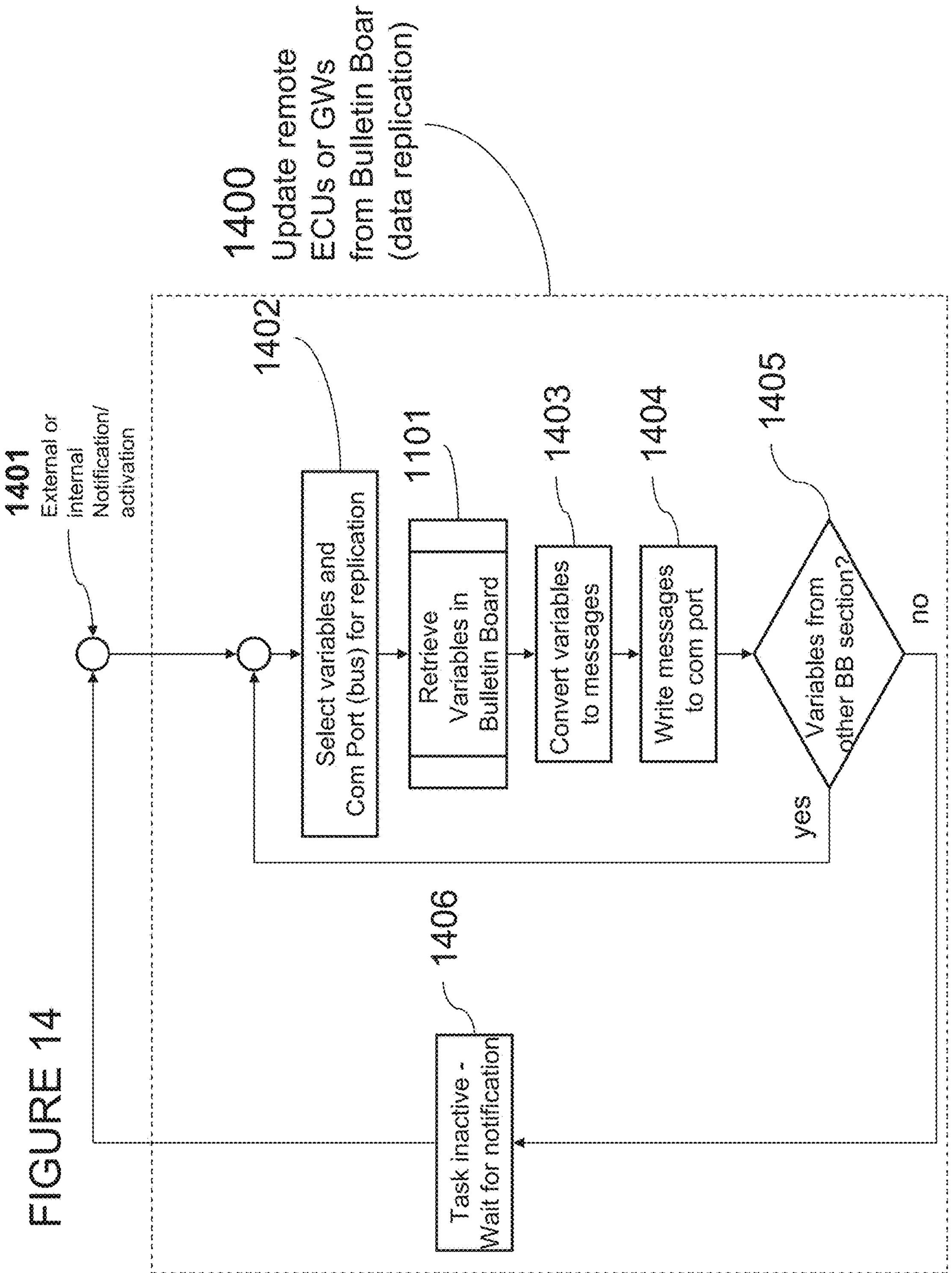

1400
Update remote ECUs or GWs from Bulletin Boar (data replication)
1401
External or internal Notification/ activation
1402
Select variables and Com Port (bus) for replication
1101
Retrieve Variables in Bulletin Board
1403
Convert variables to messages
1404
Write messages to com port
1405
Variables from other BB section?
yes
no
1406
Task inactive - Wait for notification

FIGURE 14

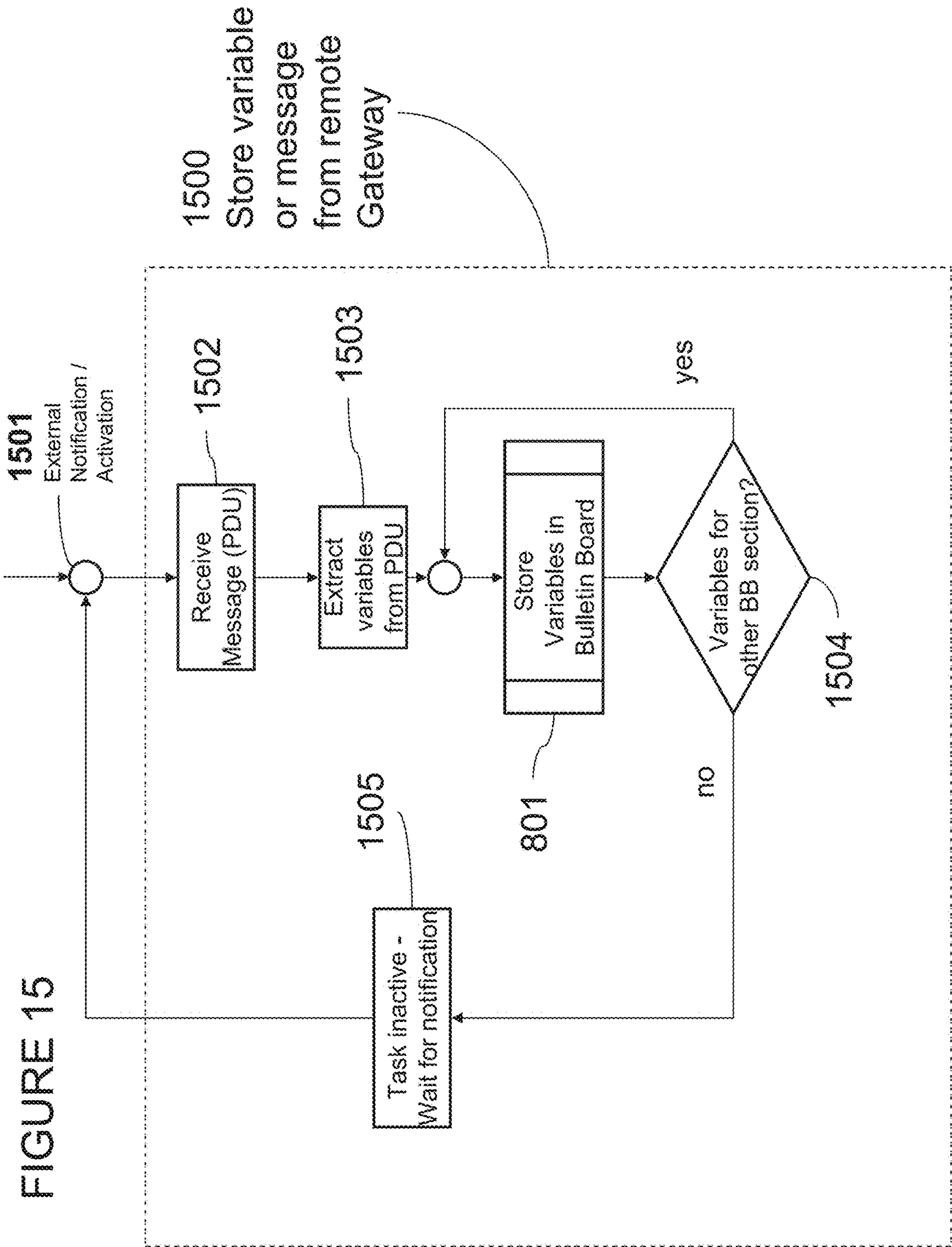

1500
Store variable
or message
from remote
Gateway
1501
External
Notification /
Activation
Receive
Message (PDU)
1502
Extract
variables
from PDU
1503
Store
Variables in
Bulletin Board
801
Variables for
other BB section?
1504
yes
no
Task inactive -
Wait for notification
1505

FIGURE 15

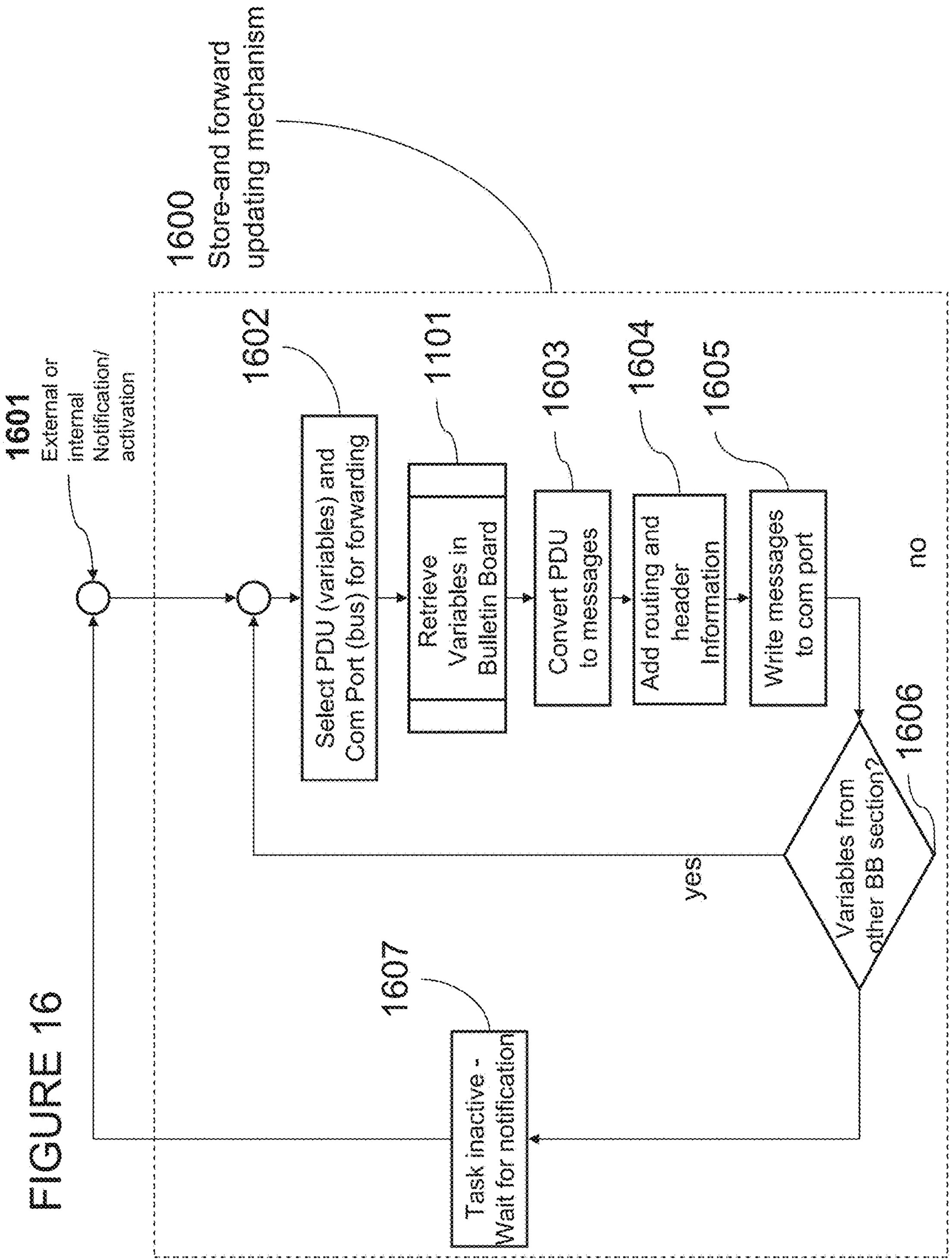

1600
Store-and forward
updating mechanism
1601
External or
internal
Notification/
activation
1602
Select PDU (variables) and
Com Port (bus) for forwarding
1101
Retrieve
Variables in
Bulletin Board
1603
Convert PDU
to messages
1604
Add routing and
header
Information
1605
Write messages
to com port
1606
Variables from
other BB section?
yes
no
1607
Task inactive -
Wait for notification

FIGURE 16

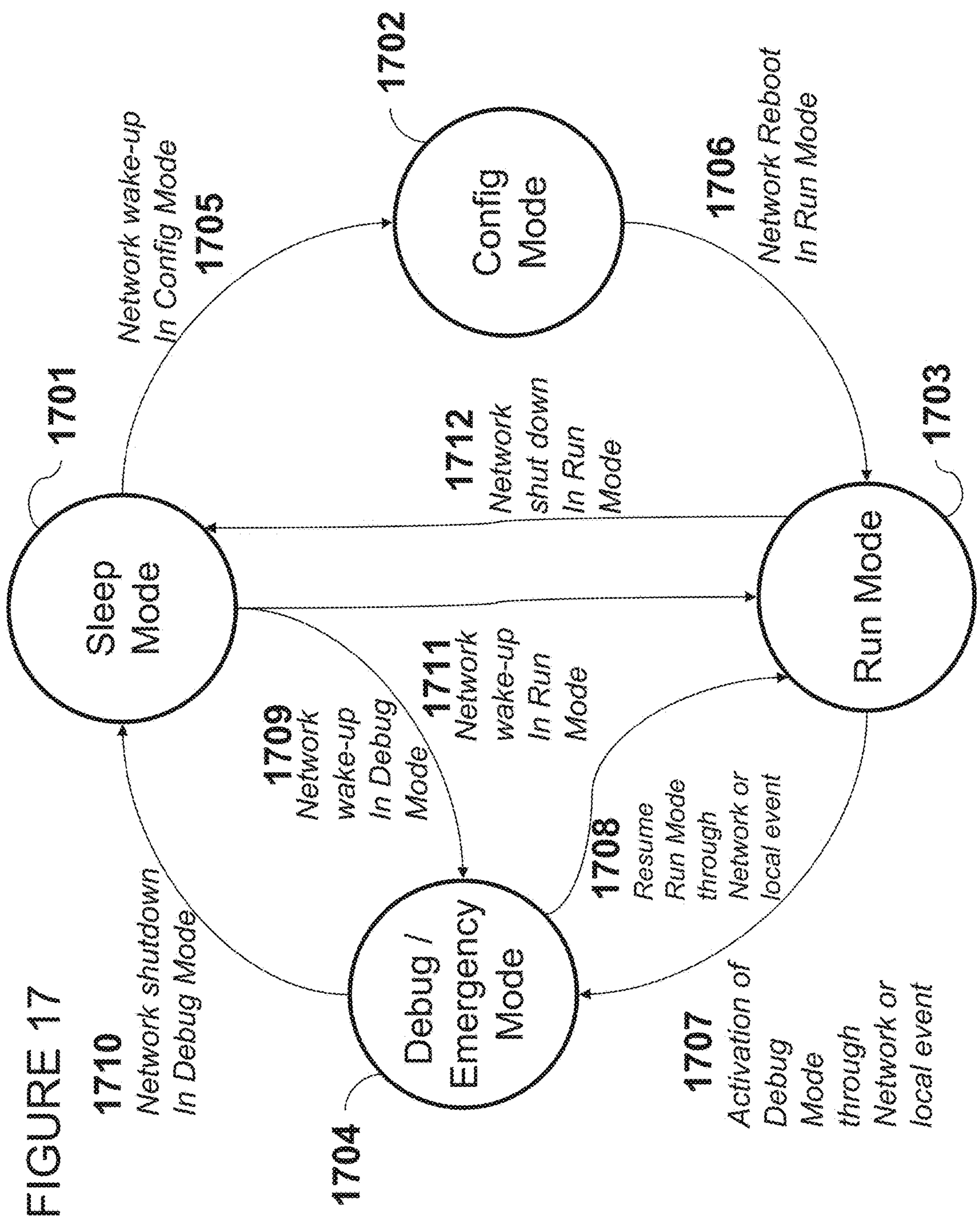

Sleep Mode
1701
Config Mode
1702
Run Mode
1703
Debug / Emergency Mode
1704
1705
Network wake-up In Config Mode
1706
Network Reboot In Run Mode
1707
Activation of Debug Mode through Network or local event
1708
Resume Run Mode through Network or local event
1709
Network wake-up In Debug Mode
1710
Network shutdown In Debug Mode
1711
Network wake-up In Run Mode
1712
Network shut down In Run Mode

FIGURE 17

SYSTEM, METHOD AND COMPUTER PROGRAM PRODUCT FOR SHARING INFORMATION IN A DISTRIBUTED FRAMEWORK

RELATED APPLICATION(S)

This application is a continuation of U.S. patent application Ser. No. 14/011,705 filed Aug. 27, 2013, which is a continuation of U.S. patent application Ser. No. 13/531,319 filed Jun. 22, 2012, now U.S. Pat. No. 8,566,843, which is a continuation of U.S. patent application Ser. No. 12/182,570 filed Jul. 30, 2008, now U.S. Pat. No. 8,209,705, which is a continuation of U.S. patent application Ser. No. 10/737,690 filed Dec. 15, 2003, now U.S. Pat. No. 7,802,263, which, in turn, claims priority under 35 U.S.C. § 119 based on U.S. Provisional Application No. 60/434,018 filed Dec. 17, 2002, all of which are incorporated herein by reference.

FIELD AND BACKGROUND OF THE INVENTION

The present invention relates to the field of distributed control and monitoring systems that may include certain temporal behavior.

Such technology may optionally apply to electronic vehicle communication and control systems, real-time monitoring systems, industrial automation and control systems, as well as any other desired system.

SUMMARY OF THE INVENTION

A system, method and computer program product are provided for sharing information in a distributed system. After information is received, it is stored on a bulletin board. In use, the information is shared, in real-time, among a plurality of heterogeneous processes.

In one embodiment, both past and present instances of the information may be stored on the bulletin board. As an option, the information may be replicated among a plurality of the bulletin boards. Optionally, first information may be processed utilizing a first bulletin board and stored utilizing a second bulletin board. Still yet, the bulletin boards may be hierarchical.

In another embodiment, the processes may access multiple sections of the bulletin board. Further, the bulletin board may send notifications to the processes based on a state of the information on the bulletin board.

Optionally, the information may include variables. For example, the information may include input variables, output variables, etc. Moreover, the processes may include local processes, remote processes, etc. Still yet, the processes may include event triggered processes and/or time triggered processes. In use, each of the processes may process the information in a manner that is isolated from temporal characteristics associated with the network.

In still another embodiment, the information may be extracted from a message received by a bulletin board manager. Moreover, the information may be converted from a signal received by a bulletin board manager. Even still, the information may be shared in a single task, may be shared according to a schedule, and/or may be shared with an operating system. Optionally, dynamic preemptive scheduling may be provided. Also, the information may be shared across the communication network with only a portion of a

message header that is needed for a specific communication link while other communication links may use a different message header.

As a further option, resources in the network may be protected. Specifically, the resources in the network may be protected utilizing a schedule that allows information sharing utilizing the bulletin board. In another embodiment, the resources in the network may be protected utilizing semaphores.

In even still another embodiment, the information may be shared according to an internal clock, an external clock, etc. During operation, objects may be generated based on a change of state of the information stored in the bulletin board. Such objects may include, but are not limited to flags, events, signals, interrupts, etc. Still yet, the information may be stored in response to interrupts associated with the processes.

In use, the bulletin board may update the processes with information at a first rate that differs from a second rate with which the processes send the information to the bulletin board. Optionally, the bulletin board may be accessed with guaranteed access times, jitter, and bandwidth.

In addition, the bulletin board may be updated irregularly and triggered by internal or external objects including, but not limited to flags, events, signals, interrupts, etc. Event triggers may be provided independent of a link connection between nodes where the processes are carried out. Moreover, failure redundancy may be provided through multiple independent links across diverse physical connections.

As yet another option, the information may have a user-configured constraint associated therewith. Such constraint may include a memory constraint, a real-time constraint, etc. As a further option, the constraint may be configured utilizing a tool.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** is a block diagram of an embodiment of a system of one embodiment;

FIG. **2** is a block diagram generally depicting an embodiment of an ECU as part of the system illustrated in FIG. **1**;

FIG. **3** is a block diagram generally depicting an embodiment of a Gateway device as part of the system illustrated in FIG. **1**;

FIG. **4** is a block diagram of an embodiment of the software architecture assumed for one embodiment.

FIG. **5** is a block diagram of an embodiment of the middleware that contains the methods of one embodiment.

FIG. **6** is a block diagram of an embodiment of the bulletin board that describes the process interaction of one embodiment.

FIG. **7** is a block diagram of an embodiment of the bulletin board that describes the process interaction with multiple external communication buses as part of one embodiment.

FIG. **8** is a flow chart diagram of an embodiment of the variable store from remote I/O method of one embodiment.

FIG. **9** is a flow chart diagram of an embodiment of the variable store from local I/O method of one embodiment.

FIG. **10** is a flow chart diagram of an embodiment of the variable method of one embodiment.

FIG. **11** is a flow chart diagram of an embodiment of the variable retrieve method of one embodiment.

FIG. **12** is a flow chart diagram of an embodiment of the application process using the method of one embodiment

FIG. **13** is a flow chart diagram of an embodiment of the local I/O update from bulletin board method of one embodiment.

FIG. **14** is a flow chart diagram of an embodiment of the variable replication method of one embodiment.

FIG. **15** is a flow chart diagram of an embodiment of the message store from remote gateway method of one embodiment.

FIG. **16** is a flow chart diagram of an embodiment of the message forward to remote ECU or Gateway method of one embodiment.

FIG. **17** is a state transition diagram of an embodiment of the mode switching method of one embodiment.

DETAILED DESCRIPTION

FIG. **1** is a block diagram generally depicting elements of an embodiment of the present distributed embedded communication and computing system. The system architecture may be situated in automotive electronics or industrial control and monitoring systems. In an automotive environment, the various Electronic Control Units (ECUs, **102**) control complex applications such as engine control, brake control, or diagnostics. They are either connected to sensors and actuators via discrete links or simple standard functions such as sensors and actuators are organized into separate sub networks.

These complex functions such as braking, engine-control, etc. are then grouped into the backbone system functions for the car, such as body control, power train and chassis. The backbone also includes the vehicle's high level functions such as diagnostics, telematics and entertainment systems.

Therefore the system is typically hierarchically organized and includes a variety of gateways (**101**,**104**,**105**), which relay messages up and down through the system layers. Each layer may contain multiple electronic control units (ECU, **102**) that are connected through wired serial multiplexing bus-systems such as Controller Area Network (CAN or ISO11898), Flexray, LIN, J1850, J1708, MOST, IEEE1394, and other similar serial multiplexing buses or through wireless multiplexing systems such as IEEE802.11, IEEE802.15, Bluetooth, Zigbee, or similar other wireless links.

Typically, functions provided by an ECU (**102**) are bound to hard real-time temporal behavior. In the context of the present description, real-time may include any response time that may be measured in milli- or microseconds, and/or is less than 1 second.

The ECU may receive a set of real-time input variables from local sensors (**108**), which are connected via discrete signal lines (**113**), or from networked sensors (**106**), which are connected through a multiplexing bus-system (**112**). The ECU may also share variables with other ECUs (**102**) that are either connected on the same physical multiplexing bus or that it can reach through a gateway (**101**,**103**,**104**).

Then the ECU (**102**) processes the input variables and generates a set of output variables that are either shared with other ECUs (**102**) as described above, or which are output to local actuators (**109**), which are connected via discrete signal lines (**113**), or to networked actuators, which are connected through a multiplexing bus (**112**). ECUs (**102**) typically share information with devices that are connected on the same physical multiplexing system. This method of information sharing is called horizontal information sharing in a hierarchical system. Gateways (**101**,**103**,**104**) link multiple physical multiplexing systems together. In the context of the present description, such information may include data, a signal, and/or anything else capable of being stored and shared.

The highest level in the hierarchical system is the system level. The system level gateway (**101**) may be connected to ECUs on the system level multiplexing bus (**117**), to subsequent gateways (**103**) that also link to subsequent communication buses (**110**), and to external components (**120**) that may contain diagnostics devices (**121**), development tools (**122**), other add-on devices (**123**) or other instances of distributed embedded communication and computing systems (**100**). In addition, the system gateway (**101**) may also be connected to an external gateway (**131**) that may link the system to a remote device (**132**) through wireless or wired wide-area-networks such as the Internet, using standard protocols such as UDP/IP, TCP/IP, RTP, HTTP, SOAP, JAVA, etc. or nonstandard proprietary protocols.

Subsequent to the system level may be several layers of groups and subgroups that are link to the higher levels via gateways (**101**,**103**,**104**,**105**).

During the design-time of the system, not all ECUs may exist. Therefore, the development tool (**122**) may provide a plug-in component or virtual ECU/GW (**115**) that directly links into the wired multiplexing bus or wireless network (**110**) and also allows for separate control functions via a tool-link (**116**).

The block diagram in FIG. **2** depicts the detailed elements within a generic ECU (**200**) that is one embodiment of ECU (**102**). The ECU (**200**) typically contains a micro-processor (**201**), volatile memory (**204**) such as RAM, S-RAM or similar, non-volatile memory (**203**) such as EEPROM, FLASH, etc., a real time clock for internal timing of processes (**205**), a watchdog (**206**) to maintain the health of the system, one or more communication bus controllers (**207**) with associated drivers (**208**), digital I/O (**209**) with line drivers (**210**), and analog I/O (**211**) with associated analog signal conditioning (**212**).

In an alternate embodiment, the ECU (**200**) may also contain a wireless communication controller (**311**) and a RF-Front-end (**312**) as outlined in FIG. **3**. The software (**202**) can either be stored in local non-volatile memory (**203**) or partially downloaded via the communication link (**207**,**208**) and stored in the volatile memory. The software is then executed in the microprocessor (**201**).

The block diagram FIG. **3** depicts the detailed elements within a generic gateway (**300**) that is one embodiment of Gateway (**101**,**103**,**104**,**105**) in FIG. **1**.

FIG. **4** outlines one embodiment of the software architecture in an embedded system. The hardware abstraction layer (**405**) allows the system developer to adapt a standard operating system to a specific hardware as used in an ECU (**200**) or gateway (**300**). The hardware abstraction layer (**405**) adapts the real-time operating system (**403**) and the device drivers (**404**) to a specific hardware implementation.

One embodiment includes the middleware (**402**) that has direct access to the real-time operating system (**403**), the device drivers (**404**) and the hardware abstraction layer (**405**). The middleware isolates the application from input/output functions and allows multiple applications to share common variables locally. In addition, the middleware lets applications share variables with remote applications/processes. In the context of the present description, a process may refer to any hardware and/or software operation, etc.

In one embodiment, the middleware can directly interface with the input/output mechanisms of the hardware without utilizing an operating system (**403**) or hardware abstraction layer (**405**).

Another embodiment of the middleware utilizes a preemptive multitasking operating system with explicit control of resources. In an alternate embodiment, the middleware can be built with a static multitasking scheme with implicit resource management or be part of a single task system.

Referring now to FIG. **5**, the middleware (**402**) contains the bulletin board manager (**501**), a local signal communication interface (**503**), a remote message communication interface (**504**), and an application programming interface (**502**). The application interface (**502**) provides methods and data interfaces to a plurality of applications. In one embodiment, the application interface is an object library that can be linked to an application at design time.

The bulletin board manager (**501**) contains an upgrade and configuration manager (**507**), an event manager (**505**), a data access manager (**508**), and a data integrity watchdog (**506**). The upgrade and configuration manager (**507**) is necessary to configure the data structure of the bulletin board and to make executable code available to individual processing nodes. In the context of the present description, the bulletin board may refer to any database that enables users to send and/or read electronic messages, files, and/or other data that are of general interest and/or addressed to no particular person/process.

The access manager provides access control mechanisms for the code update and configuration mode. It also may control access rights for individual applications at execution time in the run mode.

The event manager (**505**) captures input-output events as variables and generates new events, flags, or signals based on operations on state variables in the bulletin board. Such operations may include test of maximum values, the occurrence of logically combined events, the result of an integrity check, or events and signals that are created based on any other logical or arithmetic computation on the state variables that are stored in the bulletin board. The actual processing of data and manipulation of data may be done in the application that uses the middleware (**402**). The data integrity watchdog analyses the stored state variables for its integrity and generates events or flags if any problem occurs.

The local signal communication interface (**503**) interfaces with the local discrete input/output hardware to update the bulletin board with new variables and to update the input/output interfaces with the state variables from the bulletin board. It also converts state variables to input/output signals and input/output signals to state variables that can be stored in the bulletin board. The conversion process may contain scaling of signals as well as offset compensation. Typically this processing helps to convert I/O signals that are measured in Volt to a physical entity and vice versa. The communication with the local discrete input output system can be triggered by events or signals can be sampled time-triggered based on a cyclic global or local time base.

The remote message communication interface (**504**) interfaces to serial multiplexing interfaces (buses) that are connected to the specific processing node (ECU or Gateway). It extracts variables from a plurality of messaging protocols and stores them in the database. It also replicates local bulletin-board state variables to the associated processing nodes by composing the appropriate messages for each communication link. The message transfer can be initiated triggered by a bus event, by a local event, or by a time-triggered mechanism that uses a cyclic local or global time base.

FIG. **6** shows the concept of an extended bulletin board or an embedded real-time database (**601**). In this embodiment the ECU (**102**) or the Gateway (**101**) hosts one or multiple bulletin boards with relational links between the variables in the bulletin boards. The relations are defined by data processing functions that the gateway can operate on bulletin boards to obtain new information that can be stored in yet another bulletin board.

The bulletin board (**601**) may contain but is not limited to events (**607**), real-time variables (**608**), diagnostics data (**609**), configuration parameters (**610**), and firmware (**611**) to upgrade individual components of the executable code or the entire software of a processing node. Each type of information may include one or more sections so that individual processes are not blocked if they access separate sections of data.

The memory of the bulletin board is subdivided into areas that nodes on each external network can read from and write into and other areas that an external network may only read from. The data contained in the bulletin board may be stored in volatile or non-volatile memory. Each data entry may consist of one value or an array of values that also may represent a time series.

In one embodiment, each application process (**603**), local signal communication process (**605**), remote message communication process, and the bulletin manager (**602**) can individually access the bulletin board using operating system functions for resource management that may include semaphores, events, signals, call-back routines, flags, etc. in an alternate embodiment of the system the bulletin-board manager controls all interaction with the bulletin-board and all applications have to pass data to the bulletin-board manager. This approach simplifies the interaction with the bulletin board, but adds delay time and jitter to the state variables.

At design time, various hierarchies of memory management can be applied. In practice it is more efficient to allow each sub network and subsystem to place system variable data into local bulletin boards. This is because many system variables are primarily used only within their subsystem or sub network. By placing local information in a shared memory (local bulletin board), it can be used by multiple processes on this processor node. A group bulletin board allows devices on a sub-network to share information with a minimum of network traffic. A system bulletin board allows access to system-wide variables and information.

FIG. **7** illustrates the logical architecture of the interconnection between three heterogeneous network controllers (**702**, **703**, **704**), the associate Operating System interfaces (**705**), the remote message communication process (**706**), the bulletin board (**608**), and the application process (**606**). The connection to each communication controller is fundamentally implemented at the physical interface (the wire, fiber or electromagnetic wireless interface). Each of the higher level layers (data link, network, etc) in the communication interface (**705**) deals with specific features of the individual communication process. In practice these layers are typically represented in a message by "header" bits that contain information about that layer of the network being used to send the message.

Using this model, each communicated message may be processed at each layer to remove (and use) the associated header information for that level. Once all layers are processed the remaining packet data unit (PDU) represents the datum or core information carried by the overall message. In one embodiment, each communication controller has an associated communication interface and an associated remote message conversion mechanism. For instance com-

munication bus controller **2** (**703**) has an associated communication interface **2** (**709**), and an associated remote message conversion **2** (**710**).

This arrangement allows the remote message process (**706**) to directly access information at the data link layer and interface it with the bulletin board. A network layer is not necessary. The remote message communication process (**706**) has a multi-network access interface (essentially a processing capability that can interpret and apply the header information for a variety of networks) and the bulletin board read/write memory access. Now, the individual processing nodes do not need to know about the existence of multiple networks. Each variable can be accessed from all connected physical networks in their proprietary format. Thus the normalization of the information has only to be handled at the gateway through replication of stored data to multiple attached networks.

Continuing with FIG. **7**, the communication procedure is described. In the given example, an external event (**701**) on communication controller **2** (**703**) triggers the operating system to notify the remote message communication process (**706**) that data is available. The notification may be a flag, a call-back routine, an event, or any other operating signal. The associated remote message conversion method **2** (**710**) extracts the data (e.g. real time variables) from the message PDU and stores the data in the bulletin board (**608**). It may also store the associated event as variable in the bulletin board and signal the bulletin-board event manager that new data is available.

The bulletin event manager then notifies the application process (**606**) with the appropriate mechanism. In addition, the event manager may trigger the sampling of local signals using the local signal communication process (**605**) described in FIG. **6**. Finally the bulletin event manager may trigger the bulletin board manager (**707**) to perform integrity checks or generate additional events based on the change of the state variables.

One embodiment provides a new mechanism for creating an information interconnection between two or more heterogeneous communication networks. In the context of the present description, heterogeneous networks may refer to any different communication networks with at least one aspect that is different.

The approach uses a common, or shared storage system that is connected to all of the system networks through network interfaces. A critically important feature of the bulletin board approach is that the complexity of the bulletin board grows linearly with the number of networks (as opposed to as N(N−1) for the gateway approach), and in one-to-many situations the number of message transformations is half that of the standard networking approach.

In an alternate embodiment of the remote message communication process (**706**) any remote process can access data via a single network interface. This approach requires a network layer in each processing node and therefore adds overhead to communications. To communicate between two heterogeneous networks, this process may then be repeated in reverse by adding back the header information for the various layers of the second network, and eventually putting the message onto the second network's physical link. The remote message communication manager (**706**) then can be simplified to only one message assembly and disassembly mechanism.

FIGS. **8**-**17** illustrate the method of operation of one embodiment of the present system, and also refer to aspects and elements one embodiment shown in FIGS. **1** through **7**.

FIG. **8** details the remote messaging process (**706**) described in FIG. **7**. Referring now to FIG. **8**, the core process of storing data from remote processes that are communicated through multiplexed communication links, into the bulletin board is described. An external notification or task activation starts the process. Then a message (**802**) is received from the operating system layer.

In an alternate embodiment, the message is directly copied form the input register of the communication controller. Then the process extracts variables from the message. Additional signal adaptation may be necessary. The sub-process **804** stores the variables in the bulletin board. If the process only updates one section of the bulletin board it waits for the next message notification (**806**). If variables in multiple sections need to be updated, the process repeats (**804**).

FIG. **9** shows the data update from local input/output peripherals. The process starts with an internal or external notification or task activation. Typically this process is repeated cyclic triggered by an internal or external real-time clock. When the process is activated, it samples or polls the local input ports that may include analog and digital signals (**902**). Then it converts these signals to real-time variables by using the conversion parameters stored in the bulletin board. The signal conditioning parameters van either be defined at design time or adaptively updated by the application process. Then the process stored the new state variables in the bulletin board using the sub-process (**804**) described above.

FIG. **10** describes the bulletin board store procedure (**804**) in more detail. Before new data can be stored in the bulletin board, the procedure has to request the access right to the common resource, a section of the non-volatile or volatile memory, from the operating system (**1001**). This is called explicit resource management.

If the resource is available, the process gets the resource. If the resource is not available, it may try it again after a waiting period (**1011**) until the resource is available. After a certain time has elapsed (**1009**) beyond a configurable threshold, the temporal behavior of the state variable can't be captured any longer and the middle-ware may send an error notification to the associated process.

After reserving the resource (**1003**), the bulletin board store mechanism (**804**) timestamps the state variable for future temporal reference (**1004**). Then, the bulletin board store procedure (**804**) copies the variables or parameters from its private memory (**1006**) to the shared bulletin-board memory (**601**). Then it releases the bulletin board resource.

In an alternate embodiment, the bulletin board store procedure (**804**) has exclusive access to the bulletin board (**601**) and does not need operations **1002**, **1003**, **1007**, **1009**, **1010**, and **1011** because the resource access is realized through implicit resource management. This can be achieved with either static task scheduling or by allowing only the bulletin board store procedure (**804**) to access the bulletin board (**601**).

FIG. **11** describes the bulletin board retrieve procedure (**1101**) in more detail. Before data can be retrieved from the bulletin board, the procedure has to request the access right to the common resource, a section of the non-volatile or volatile memory, from the operating system (**1102**).

If the resource is available, the process gets the resource. If the resource is not available, it may try it again after a waiting period (**1108**) until the resource is available. After a certain time has elapsed (**1109**) beyond a configurable threshold, the temporal behavior of the state variable can't be captured any longer and the middle-ware may send an error notification to the associated process (**1110**).

After reserving the resource (**1104**), the bulletin board retrieve mechanism (**1101**) copies the variables or parameters from the shared bulletin-board memory (**601**) to its private memory (**1006**). Then, it releases the bulletin board resource. In an alternate embodiment the bulletin board retrieve procedure (**1101**) has exclusive access to the bulletin board (**601**) and does not need operations **1103**, **1104**, **1106**, **1108**, **1109**, and **1110**. Because the resource access is realized through implicit resource management, this can be achieved with either static task scheduling or by allowing only the bulletin board retrieve procedure (**1101**) to access the bulletin board (**601**).

Referring to FIG. **12**, the application process (**1200**) utilizes the bulletin board retrieve mechanism (**1101**) to access all parameters, events, and real-time variables from the bulletin board. Thus the application process is decoupled from the temporal behavior of the input/output variables and can be triggered by a plurality of events (**1201**).

The application process may retrieve one or multiple sets of variables stored in a plurality of memory sections. Then the application process processes the variables (**1203**) with its method. Because the method is not tied to the location of the input/output variables, the application process can be moved or replicated to a plurality of processing nodes (ECUs or Gateways). After processing the input variables and generating a set of output variables, the application process uses the bulletin board store method (**801**) to update one or a plurality of memory sections in the bulletin board. If the application process is a cyclic procedure, it waits until the next activation occurs (**1205**).

Continuing with FIG. **13**, the update local I/O from bulletin board process (**1300**) utilizes the bulletin board retrieve mechanism (**1101**) to access real-time variables from the bulletin board and convert them to output signals (**1302**) that can be written to the output port (**1303**). The I/O update process may retrieve one or multiple sets of variables stored in a plurality of memory sections. If the I/O update process is a cyclic procedure, it waits until the next activation occurs (**1305**).

FIG. **14** describes the data replication process (**1400**). This process can be triggered by a plurality of notification mechanisms, such as events, alarm signals, internal and external timers, and flags set in the bulletin board. It then selects a subset of variables to be replicated and a communication port (**1402**). Next it retrieves the variables from the bulletin board with mechanism (**1401**) and assembles the messages for the specific communication link (**1403**). The message may include an address or identification number for all bulletin boards and associated processing nodes (ECUs and Gateways).

Finally, it writes the messages to the communication port (**1404**). In an alternate embodiment, it handles the messages to the associated interface procedure of the operating system. Then it repeats the procedure, until all variables are updated on all communication ports. If the data replication process is a cyclic procedure, it waits until the next activation occurs (**1405**).

Referring now to FIG. **15**, the store message from remote processing node (gateway or ECU) process (**1500**) describes how replicated data is stored in the bulletin board. This process can be triggered by a plurality of notification mechanisms, such as internal or remote events, alarm signals, internal and external timers, and flags set in the bulletin board. The communication bus may also issue these notifications. The process (**1500**) then reads a message from the communication port (**1502**), selects a subset of variables to be replicated (**1503**), and stores the variables in the bulletin board with procedure (**801**). In an alternate embodiment, this procedure may also be used to store a packet data unit (PDU) in the bulletin board for later replication on the same or a different communication link.

This store and forward networking mechanism can be implemented without the need for complex networking protocols and is therefore well suited for limited processing power and memory environments. It also works in soft-real time environments when no strict temporal behavior is required. The data store operation (**801**) may be repeated for a plurality of bulletin board sections. If the data replication process is a cyclic procedure, it waits until the next activation occurs (**1505**).

Continuing now with FIG. **16**, the store and forward updating mechanism (**1600**) replicates messages from remote processing nodes to other processing nodes from stored packet data units in the bulletin board. This process can be triggered by a plurality of notification mechanisms (**1601**), such as internal or remote events, alarm signals, internal and external timers, and flags set in the bulletin board. The communication bus may also issue these notifications.

The process (**1600**) then selects a message to be forwarded (**1602**) and the appropriate communication link and retrieves the PDU with the bulletin board retrieve mechanism (**1101**). It then adds the appropriate messages header for the communication link (**1603**) and may add routing information (**1604**). Finally the update process (**1600**) writes the messages to the communication port (**1605**). If the updating process is a cyclic procedure, it waits until the next activation occurs (**1607**).

FIG. **17** describes the various modes that the distributed communications and computing system (**100**) can be operated in. In one embodiment, the system operates in various distinct modes in order to preserve the integrity of the system and still allow for changing the architecture and behavior of the network or the roles of the individual nodes. When the distributed computing and communication system wakes up from the sleep mode (**1701**), it can enter a configuration and upgrade mode (**1702**), an emergency or debug mode (**1704**), or the normal real-time run mode (**1703**). The root node or system gateway in a distributed communication and computing system defines the mode based on the existence of external events, such as an external control command, internal events, a system failure, or failed integrity check.

Referring now to FIG. **1**, the external commands may be generated from a development tool (**122**) or a remote device (**132**) that is connected via a remote gateway (**131**). In an alternate embodiment, each ECU (**102**) or virtual ECU (**115**) can trigger the system to enter a different operating mode.

Continuing with FIG. **17**, in the configuration mode (**1702**), the system software and the information-sharing configuration can be updated via a secure communication link with encrypted commands Each processing node (ECU or gateway) may have security mechanisms such as a certificate that allows it to identify and authorize another entity (remote gateway, remote ECU, or development tool) to make changes to its bulletin board parameters.

The remote entity may also download a new firmware to the bulletin board. The ECU or gateway can store this new firmware in its non-volatile memory while it backs up the original image on the bulletin board for the case that the new software is not functional. In the update mode, the distributed system can also reconfigure the communication and computing infrastructure based on a new set of parameters that need to be stored in the individual bulletin boards.

In the normal run mode (**1703**), the system operates in the real-time information sharing mode and network configuration and certain parameters can't be changed. That protection allows defining deterministic temporal behavior on all communication links. But any processing node may enter a debug/emergency mode (**1704**) if a failure or other qualifying event occurs.

In the emergency mode, a processor executes an alternate procedure that maintains the temporal behavior on the communication links but may reduce or increase the amount of information shared with other processors. It also lets other processing nodes check on the integrity of sensors and actuators. In the maintenance and upgrade mode, an external system can upgrade executable code images and the bulletin-board configuration via secure communication links.

A system and method are thus provided for sharing information within a distributed embedded communications and computing system and with components outside the embedded system. The information sharing mechanism relies on a bulletin board that may include a small database operating under hard real-time conditions with minimal delays, communication latency, and jitter. The embedded database or bulletin board isolates a real-time application in a Electronic Control Unit (ECU) from various other real time applications and from input output signals in the same module (local information sharing), from event-triggered communications with applications in other modules, and from time-triggered communications with applications in other modules.

One design criteria of the database is that the temporal behavior of communications does not impact the real-time computing task and provides enough information access performance at peak time demand Typically, distributed embedded systems consist of a static structure that can be analyzed at design time. In addition to the real-time operation, the proposed method for information sharing also provides access to the parameters of the embedded system and allows for software upgrades of certain modules.

The present embodiment addresses the shortcomings of traditional computer networks with following enhancements:

1) The concept of multi-mode storage that links two or more communication networks via a bulletin board. The bulletin board is a multi-mode storage that can be thought of an extension to shared memory that can be accessed by local and remote processes at attached networks. There may be multiple hierarchical layers of bulletin boards depending on the topology of the communication system. The bulletin board increases the network efficiency by reducing the number of transactions needed to access remote variables.

2) The concept of a direct-access bulletin board that does not require a network layer translation of messages on each node of the network. Even though this approach restricts the reach of each node to only adjacent nodes and the next gateway, this still allows cross-network variable sharing though vertical real-time replication of data.

3) The concept of hierarchical bulletin board management that allows restriction of information access to certain levels in a network, but still allows the replication of information to other nodes in the network. This paradigm follows the path of reducing the information amount from the leaves of the network to central control and diagnosis hubs.

4) The concept that a gateway can host an assembly of bulletin boards or embedded database that allows operations on bulletin boards to generate events for associated processes. This extension allows definition of a set of data processing operations that would be done once in a network and would be instantly available for connected nodes. Examples for operations are sensor data state observers, diagnostics, integrity checks, fail-safe mechanisms, etc.

5) The concept that an embedded communication and computing network can run in multiple modes in order to provide for a guaranteed deterministic behavior of the system. This property can be achieved by only allowing change to the configuration and/or the functions (SW code) in a secured configuration and upgrade mode. If the network is booted in the normal operating mode, all processors execute the existing code and only allow data sharing through the bulletin boards. The emergency or debug mode lets the network run in a fail-safe reduced operation mode or in a diagnostic mode that allows inspection of the system, while it is running. For each operating mode, the gateway can store a processing image on the bulletin board. The advantage of this procedure is that only the communication hubs need to deal with secure data transfer and encryption while the peripheral nodes in the network can be relative simple in design.

6) The concept of designing the topology of a distributed computing and communication system independent of the definition of the individual functions that the network performs. Each processing task is only associated with a bulletin board, but isolated from I/O processing.

Of course, these are all optional embodiments/enhancements.

While various embodiments have been described above, it should be understood that they have been presented by the way of example only, and not limitation. Thus, the breadth and scope of a preferred embodiment should be not limited by any of the above described exemplary embodiments, but should be defined only in accordance with the following claims and their equivalents.

What is claimed is:

1. An apparatus, comprising:

an automotive electronic control unit including a non-transitory memory storage comprising instructions, and at least one hardware processor in hardwired communication with the memory storage, wherein the at least one hardware processor executes the instructions to:

identify information associated with a message received utilizing a Flexray network protocol associated with a Flexray network;

issue a storage resource request in connection with a storage resource of the automotive electronic control unit and determine whether the storage resource is available for storing the information;

determine whether a threshold has been reached in association with the storage resource request;

in the event the storage resource is not available and the threshold associated with the storage resource request has not been reached, issue another storage resource request in connection with the storage resource;

in the event the storage resource is not available and the threshold associated with the storage resource request has been reached, send a notification;

in the event the storage resource is available, store the information utilizing the storage resource; and

share the information in less than one millisecond utilizing a Controller Area Network protocol associated with a Controller Area Network, the automotive electronic control unit remaining in hardwired communication with the Flexray network and the Controller Area Network, and including:

a first interface for interfacing with the Flexray network, the first interface including a first interface-related data link layer component that uses Flexray network-related data link layer header bits and a first interface-related network layer component that uses Flexray network-related network layer header bits; and

a second interface for interfacing with the Controller Area Network, the second interface including a second interface-related data link layer component that uses Controller Area Network-related data link layer header bits and a second interface-related network layer component that uses Controller Area Network-related network layer header bits.

2. The apparatus as recited in claim **1**, wherein the information is also shared in less than the one millisecond utilizing, in addition to the Controller Area Network protocol, a Local Interconnect Network protocol associated with a Local Interconnect Network, and the automotive electronic control unit remains in hardwired communication with the Local Interconnect Network via a third interface for interfacing with the Local Interconnect Network, the third interface including a third interface-related data link layer component that uses Local Interconnect Network-related data link layer header bits.

3. The apparatus as recited in claim **1**, wherein the identifying, the issuing the another storage resource request, and the sending the notification collectively occur in less than one microsecond.

4. The apparatus as recited in claim **1**, wherein the identifying, the issuing of the storage resource request, the storing the information, and an initiation of the sharing collectively occur in less than one microsecond.

5. The apparatus as recited in claim **4**, wherein the second interface-related network layer component uses the Controller Area Network-related network layer header bits by adding the Controller Area Network-related network layer header bits to a data unit including the information, and then the second interface-related data link layer component uses the Controller Area Network-related data link layer header bits by adding the Controller Area Network-related data link layer header bits to the data unit, before communicating the data unit on a physical link of the Controller Area Network.

6. The apparatus as recited in claim **5**, wherein the first interface-related data link layer component uses the Flexray network-related data link layer header bits by removing the Flexray network-related data link layer header bits from another data unit, and the first interface-related network layer component uses the Flexray network-related network layer header bits by removing the Flexray network-related network layer header bits from the another data unit, where the information is extracted from the another data unit before the sharing.

7. The apparatus as recited in claim **1**, wherein a duration between the information being sent to the automotive electronic control unit and the sharing being completed by the information arriving at a destination, is less than the one millisecond.

8. The apparatus as recited in claim **1**, wherein a duration between the information being received at the automotive electronic control unit, and the sharing being completed by arriving at a destination, is less than the one millisecond.

9. The apparatus as recited in claim **1**, wherein the storage resource request and the another storage resource request each includes a request for an access.

10. The apparatus as recited in claim **1**, wherein the storage resource is configured to store the information that is received utilizing the Flexray network that is a physical network, for the purpose of sharing the information utilizing the Controller Area Network that is another physical network.

11. The apparatus as recited in claim **1**, wherein the hardware processor and the instructions are for releasing the storage resource after the storing.

12. The apparatus as recited in claim **1**, wherein the apparatus is configured such that the information is replicated among a plurality of the storage resources.

13. The apparatus as recited in claim **1**, wherein the apparatus is configured such that both past and present instances of the information are stored on the storage resource.

14. The apparatus as recited in claim **1**, wherein a plurality of the storage resources are included which are organized in a hierarchical manner.

15. The apparatus as recited in claim **1**, wherein the apparatus is configured such that different processes are capable of accessing multiple sections of the storage resource.

16. The apparatus as recited in claim **15**, wherein the different processes include local processes.

17. The apparatus as recited in claim **15**, wherein the different processes include remote processes.

18. The apparatus as recited in claim **15**, wherein the different processes include both event triggered processes and time triggered processes.

19. The apparatus as recited in claim **1**, wherein the apparatus is configured such that a storage resource manager sends notifications to different processes based on a state of the information stored utilizing the storage resource.

20. The apparatus as recited in claim **1**, wherein the apparatus is configured such that the information is shared without multiplexing.

21. The apparatus as recited in claim **1**, wherein the apparatus is configured such that at least one of a plurality of different processes process the information in a manner that is isolated from temporal characteristics associated with at least one of the Controller Area Network or the Flexray Network.

22. The apparatus as recited in claim **1**, wherein the apparatus is configured such that the information is shared in a single task.

23. The apparatus as recited in claim **1**, wherein the apparatus is configured such that the information is shared according to a schedule.

24. The apparatus as recited in claim **1**, wherein the apparatus is configured such that the information is shared with an operating system.

25. The apparatus as recited in claim **1**, wherein the apparatus is configured such that dynamic preemptive scheduling is provided.

26. The apparatus as recited in claim **1**, wherein the apparatus is configured such that the storage resource is protected utilizing a schedule that allows information sharing utilizing the storage resource.

27. The apparatus as recited in claim **1**, wherein the apparatus is configured such that the storage resource is protected utilizing semaphores.

28. The apparatus as recited in claim **1**, wherein the apparatus is configured such that the information is shared according to an internal clock.

29. The apparatus as recited in claim **1**, wherein the apparatus is configured such that the information is shared according to an external clock.

30. The apparatus as recited in claim **1**, wherein the apparatus is configured such that objects are generated based on a change of state of the information stored utilizing the storage resource.

31. The apparatus as recited in claim **30**, wherein the objects include at least one of flags, events, signals, or interrupts.

32. The apparatus as recited in claim **1**, wherein the apparatus is configured such that the information is stored in response to interrupts associated with different processes.

33. The apparatus as recited in claim **1**, wherein the storage resource is replicated through a distributed system.

34. The apparatus as recited in claim **1**, wherein the apparatus is configured such that the information is shared by the automotive electronic control unit with only a portion of a network communication protocol header.

35. The apparatus as recited in claim **1**, wherein the apparatus is configured such that the storage resource updates at least one process with the information at a first rate that differs from a second rate with which at least one other process sends the information to the storage resource.

36. The apparatus as recited in claim **1**, wherein the apparatus is configured such that event triggers are provided independent of a link connection between nodes where different processes are performed.

37. The apparatus as recited in claim **1**, wherein the apparatus is configured such that failure redundancy is provided across multiple independent links across diverse physical connections.

38. The apparatus as recited in claim **1**, wherein the apparatus is configured such that the storage resource is accessed with guaranteed access times.

39. The apparatus as recited in claim **1**, wherein the apparatus is configured such that the storage resource is accessed with guaranteed jitter.

40. The apparatus as recited in claim **1**, wherein the apparatus is configured such that the storage resource is accessed with guaranteed bandwidth.

41. The apparatus as recited in claim **1**, wherein the information has a user-configured constraint associated therewith.

42. The apparatus as recited in claim **41**, wherein the constraint includes a memory constraint.

43. The apparatus as recited in claim **1**, wherein the apparatus is configured such that the information is processed utilizing a first storage resource and stored utilizing a second storage resource.

44. The apparatus as recited in claim **1**, wherein the apparatus is configured such that instances of the information are time-stamped when stored utilizing the storage resource for temporal reference.

45. The apparatus as recited in claim **1**, wherein the apparatus is configured such that the storage resource updates different processes based on an event.

46. The apparatus as recited in claim **1**, wherein the apparatus is part of an embedded system.

47. The apparatus as recited in claim **1**, wherein the instructions include code.

48. The apparatus as recited in claim **1**, wherein the apparatus includes middleware.

49. The apparatus as recited in claim **1**, wherein the automotive electronic control unit includes a gateway for multicasting the information to the Flexray Network in addition to at least one other different network.

50. The apparatus as recited in claim **1**, wherein the automotive electronic control unit includes a gateway for concurrently sending the information to the Flexray Network in addition to at least one other network.

51. The apparatus as recited in claim **1**, wherein the storage resource includes a multi-mode storage.

52. The apparatus as recited in claim **1**, wherein the storage resource includes a section of the memory storage.

53. The apparatus as recited in claim **1**, wherein the storage resource is a component of firmware.

54. The apparatus as recited in claim **1**, wherein the automotive electronic control unit includes a gateway.

55. The apparatus as recited in claim **1**, wherein the automotive electronic control unit includes a gateway for sending the information to the Flexray Network in addition to at least one other network.

56. The apparatus as recited in claim **1**, wherein the automotive electronic control unit includes a gateway for sending the information to multiple Flexray Networks.

57. The apparatus as recited in claim **1**, wherein the storage resource includes non-volatile memory and volatile memory.

58. The apparatus as recited in claim **1**, wherein the storage resource stores a data entry.

59. The apparatus as recited in claim **58**, wherein the data entry includes an array of values.

60. The apparatus as recited in claim **58**, wherein the data entry includes an array of values that represents a time series.

61. The apparatus as recited in claim **1**, wherein at least one of the determinations is carried out as part of explicit resource management.

62. The apparatus as recited in claim **1**, wherein at least one of the determinations is carried out as part of implicit resource management.

63. The apparatus as recited in claim **1**, wherein the storage resource includes at least two different types of memory.

64. The apparatus as recited in claim **1**, wherein at least one aspect of the storage resource grows linearly with a number of heterogeneous networks to which the apparatus is coupled.

65. The apparatus as recited in claim **1**, wherein the information is associated with at least one of engine control or brake control.

66. The apparatus as recited in claim **1**, wherein the information is associated with diagnostics.

67. The apparatus as recited in claim **1**, wherein the apparatus is part of a system including a plurality of the automotive electronic control units which are each capable of interfacing at least one of an actuator or a sensor.

68. The apparatus as recited in claim **67**, wherein the apparatus further includes at least one gateway capable of interfacing the plurality of electronic control units.

69. The apparatus as recited in claim **68**, wherein the gateway includes a group gateway.

70. The apparatus as recited in claim **68**, wherein the gateway includes a subgroup gateway.

71. The apparatus as recited in claim **1**, wherein the automotive electronic control unit is a virtual automotive electronic control unit.

72. The apparatus as recited in claim **1**, wherein the apparatus is part of a system that is capable of interfacing a diagnostic tool and a design tool.

73. The apparatus as recited in claim **1**, wherein the apparatus is part of a system including a plurality of layers including an application layer, a middleware layer, a real-time operating system layer, a device driver layer, and a hardware abstraction layer.

74. The apparatus as recited in claim **1**, wherein the apparatus is part of a system including a plurality of layers including at least two of an application layer, a middleware layer, a real-time operating apparatus layer, a device driver layer, and a hardware abstraction layer.

75. The apparatus as recited in claim **1**, wherein the apparatus is part of a system including a plurality of layers including at least three of an application layer, a middleware layer, a real-time operating system layer, a device driver layer, and a hardware abstraction layer.

76. The apparatus as recited in claim **1**, wherein the apparatus is part of a system including a plurality of layers including at least four of an application layer, a middleware layer, a real-time operating system layer, a device driver layer, and a hardware abstraction layer.

77. The apparatus as recited in claim **1**, wherein the storage resource is configured such that temporal behavior of communications does not impact a real-time computing task.

78. The apparatus as recited in claim **1**, wherein the storage resource is configured for providing software upgrades for each of a plurality of modules.

79. The apparatus as recited in claim **78**, wherein the apparatus is configured such that the software upgrades are downloaded via a communication link.

80. The apparatus as recited in claim **1**, wherein the storage resource is configured such that network efficiency is increased by reducing a number of transactions needed to access remote variables.

81. The apparatus as recited in claim **1**, wherein the storage resource is a direct-access storage resource.

82. The apparatus as recited in claim **1**, wherein the apparatus is configured so as to avoid a network layer translation of the message.

83. The apparatus as recited in claim **1**, wherein the storage resource is configured so as to allow cross-network variable sharing though vertical real-time replication of data.

84. The apparatus as recited in claim **1**, wherein the storage resource is configured to restrict information access to certain levels in a particular network.

85. The apparatus as recited in claim **1**, wherein the storage resource is configured to restrict information access to certain network levels, while allowing replication of the information to nodes.

86. The apparatus as recited in claim **1**, wherein the apparatus is configured such that multiple modes of operation are enabled in order to provide for a guaranteed deterministic behavior.

87. The apparatus as recited in claim **1**, wherein the apparatus is configured such that multiple modes of operation are enabled, wherein at least one of the modes includes a secured configuration or upgrade mode that allows changes to configuration and/or functions.

88. The apparatus as recited in claim **1**, wherein the apparatus is configured such that multiple modes of operation are enabled, wherein at least one of the modes includes a diagnostic mode.

89. The apparatus as recited in claim **1**, wherein the apparatus is configured such that a topology of a distributed computing and communication system is capable of being designed independent of a definition of individual functions that a particular network performs.

90. The apparatus as recited in claim **1**, and further comprising: a hardware abstraction layer that allows a system developer to adapt a real-time operating system and a plurality of device drivers to a specific hardware implementation of the automotive electronic control unit.

91. The apparatus as recited in claim **90**, and further comprising: middleware that has access to the real-time operating system, the device drivers, and the hardware abstraction layer.

92. The apparatus as recited in claim **91**, wherein the middleware isolates multiple application from input or output functions.

93. The apparatus as recited in claim **92**, wherein the middleware allows the multiple applications to share common variables locally.

94. The apparatus as recited in claim **91**, wherein the middleware interfaces with input or output mechanisms without utilizing the real-time operating system nor the hardware abstraction layer.

95. The apparatus as recited in claim **91**, wherein the middleware is adapted for implementing different types of control of the storage resource.

96. The apparatus as recited in claim **95**, wherein the different types of control of the storage resource include implicit control and explicit control.

97. The apparatus as recited in claim **95**, wherein the different types of control of the storage resource include a control associated with the identifying, the issuing of the storage resource request, the storing of the information, and the sharing.

98. The apparatus as recited in claim **1**, and further comprising: a data integrity watchdog that analyses stored state variables for integrity and generates events or flags if a problem occurs.

99. The apparatus as recited in claim **1**, wherein the information includes state variables that are converted before storage utilizing the storage resource.

100. The apparatus as recited in claim **99**, wherein the state variables are converted for offset compensation in connection with the state variables.

101. The apparatus as recited in claim **1**, wherein the storage resource takes a form of storage other than a buffer.

102. An apparatus, comprising:

an automotive electronic control unit including a non-transitory memory storage comprising instructions, and at least one hardware processor in hardwired communication with the memory storage, wherein the at least one hardware processor executes the instructions to:

identify information associated with a message received utilizing a Controller Area Network protocol associated with a Controller Area Network;

issue a storage resource request in connection with a storage resource of the automotive electronic control unit and determine whether the storage resource is available for storing the information;

determine whether a threshold has been reached in association with the storage resource request;

in the event the storage resource is not available and the threshold associated with the storage resource request has not been reached, issue another storage resource request in connection with the storage resource;

in the event the storage resource is not available and the threshold associated with the storage resource request has been reached, send a notification;

in the event the storage resource is available, store the information utilizing the storage resource; and share the information in less than one millisecond utilizing a Flexray network protocol associated with a Flexray network;

wherein the automotive electronic control unit is in hardwired communication with the Controller Area Network and the Flexray network and includes:

a first interface in hardwired communication with the Controller Area Network, the first interface including a first interface-related data link layer component that uses Controller Area Network-related data link layer header bits and a first interface-related network layer component that uses Controller Area Network-related network layer header bits; and

a second interface in hardwired communication with the Flexray network, the second interface including a second interface-related data link layer component that uses Flexray network-related data link layer header bits and a second interface-related network layer component that uses Flexray network-related network layer header bits.

103. The apparatus as recited in claim **102**, wherein the information is also shared in less than the one millisecond utilizing, in addition to the Flexray network protocol, a Local Interconnect Network protocol associated with a Local Interconnect Network, and the automotive electronic control unit remains in hardwired communication with the Local Interconnect Network via a third interface for interfacing with the Local Interconnect Network, the third interface including a third interface-related data link layer component that uses Local Interconnect Network-related data link layer header bits.

104. The apparatus as recited in claim **102**, wherein the identifying, the issuing the another storage resource request, and the sending the notification collectively occur in less than one microsecond.

105. The apparatus as recited in claim **102**, wherein the identifying, the issuing of the storage resource request, the storing the information, and an initiation of the sharing collectively occur in less than one microsecond.

106. The apparatus as recited in claim **102**, wherein the second interface-related network layer component uses the Flexray network-related network layer header bits by adding the Flexray network-related network layer header bits to a data unit including the information, and then the second interface-related data link layer component uses the Flexray network-related data link layer header bits by adding the Flexray network-related data link layer header bits to the data unit, before communicating the data unit on a physical link of the Flexray network.

107. The apparatus as recited in claim **106**, wherein the first interface-related data link layer component uses the Controller Area Network-related data link layer header bits by removing the Controller Area Network-related data link layer header bits from another data unit, and the first interface-related network layer component uses the Controller Area Network-related network layer header bits by removing the Controller Area Network-related network layer header bits from the another data unit, where the information is extracted from the another data unit before the sharing.

108. The apparatus as recited in claim **107**, wherein a duration between the information being sent to the automotive electronic control unit and the sharing being completed by the information arriving at a destination, is less than the one millisecond.

109. The apparatus as recited in claim **108**, and further comprising: a hardware abstraction layer that allows a system developer to adapt a real-time operating system and a plurality of device drivers to a specific hardware implementation of the automotive electronic control unit.

110. The apparatus as recited in claim **109**, and further comprising: middleware that has access to the real-time operating system, the device drivers, and the hardware abstraction layer.

111. The apparatus as recited in claim **110**, wherein the middleware isolates multiple application from input or output functions.

112. The apparatus as recited in claim **111**, wherein the middleware allows the multiple applications to share common variables locally.

113. The apparatus as recited in claim **110**, wherein the middleware interfaces with input or output mechanisms without utilizing the real-time operating system nor the hardware abstraction layer.

114. The apparatus as recited in claim **110**, wherein the middleware is adapted for implementing different types of control of the storage resource.

115. The apparatus as recited in claim **114**, wherein the different types of control of the storage resource include implicit control and explicit control.

116. The apparatus as recited in claim **114**, wherein the different types of control of the storage resource include a control associated with the identifying, the issuing of the storage resource request, the storing the information, and the sharing.

117. The apparatus as recited in claim **108**, and further comprising: a data integrity watchdog that analyses stored state variables for integrity and generates events or flags if a problem occurs.

118. The apparatus as recited in claim **108**, wherein the information includes state variables that are converted before storage utilizing the storage resource.

119. The apparatus as recited in claim **118**, wherein the state variables are converted for scaling the state variables.

120. The apparatus as recited in claim **118**, wherein the state variables are converted for offset compensation in connection with the state variables.

121. The apparatus as recited in claim **107**, wherein a duration between the information being received at the automotive electronic control unit, and the sharing being completed by arriving at a destination, is less than the one millisecond.

122. The apparatus as recited in claim **121**, wherein the first interface-related data link layer component and the second interface-related data link layer component are drivers.

123. The apparatus as recited in claim **122**, wherein the first interface-related data link layer component and the second interface-related data link layer component are part of a hardware abstraction layer.

124. The apparatus as recited in claim **123**, wherein the hardware processor and the instructions are for releasing the storage resource after the storing.

125. The apparatus as recited in claim **121**, wherein the apparatus is configured such that the information is shared without multiplexing.

126. The apparatus as recited in claim **102**, where the storage resource is configured to store the information that is received utilizing the Controller Area Network protocol associated with the Controller Area Network that is a first physical network, for the purpose of sharing the information utilizing the Flexray network protocol associated with the Flexray network that is another physical network.

127. The apparatus as recited in claim **102**, wherein the storage resource takes a form of storage other than a buffer.

* * * * *

US009705765B2

(12) **United States Patent**
Fuchs et al.

(10) **Patent No.: US 9,705,765 B2**
(45) **Date of Patent: *Jul. 11, 2017**

(54) **SYSTEM, METHOD AND COMPUTER PROGRAM PRODUCT FOR SHARING INFORMATION IN A DISTRIBUTED FRAMEWORK**

(71) Applicant: **Stragent, LLC**, Longview, TX (US)

(72) Inventors: **Axel Fuchs**, San Jose, CA (US); **Scott Sturges Andrews**, Petaluma, CA (US)

(73) Assignee: **Stragent, LLC**, Longview, TX (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **15/405,088**

(22) Filed: **Jan. 12, 2017**

(65) **Prior Publication Data**

US 2017/0126524 A1 May 4, 2017

Related U.S. Application Data

(63) Continuation of application No. 14/011,705, filed on Aug. 27, 2013, now Pat. No. 9,575,817, which is a continuation of application No. 13/531,319, filed on Jun. 22, 2012, now Pat. No. 8,566,843, which is a continuation of application No. 12/182,570, filed on Jul. 30, 2008, now Pat. No. 8,209,705, which is a continuation of application No. 10/737,690, filed on Dec. 15, 2003, now Pat. No. 7,802,263.

(60) Provisional application No. 60/434,018, filed on Dec. 17, 2002.

(51) **Int. Cl.**

H04L 12/26	(2006.01)
H04L 29/08	(2006.01)
H04L 29/06	(2006.01)

(52) **U.S. Cl.**

CPC ***H04L 43/08*** (2013.01); ***H04L 43/04*** (2013.01); ***H04L 43/06*** (2013.01); ***H04L 67/02*** (2013.01); ***H04L 67/12*** (2013.01); *H04L 65/102* (2013.01)

(58) **Field of Classification Search**

CPC H04L 43/04; H04L 43/06; H04L 43/08; H04L 67/02; H04L 65/102

See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,230,051	A	7/1993	Quan	
5,265,250	A	11/1993	Andrade et al.	
5,388,189	A	2/1995	Kung	
5,513,324	A	4/1996	Dolin, Jr. et al.	
5,588,002	A	12/1996	Kawanishi et al.	.... 370/462
5,666,485	A	9/1997	Suresh et al.	

(Continued)

OTHER PUBLICATIONS

Reinhard Maier, "Event-Triggered Communication on Top of Time-Triggered Architecture," Proceedings of the 21st Digital Avionics Systems Conference, vol. 2, 13.C.5-1-13.C.5-9 (Oct. 27-31, 2002).

(Continued)

Primary Examiner — Charles E Anya

(57) **ABSTRACT**

A system, method and computer program product are provided for receiving information associated with a message, issuing a storage resource request in connection with a storage resource and determining whether the storage resource is available. In use, the information is capable of being shared in less than one second, utilizing an automotive electronic control unit which includes a plurality of interfaces.

31 Claims, 17 Drawing Sheets

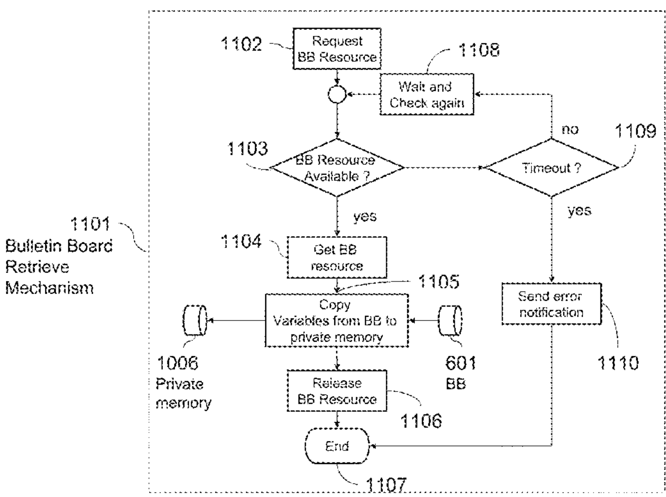


EXHIBIT D

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,737,529	A	4/1998	Dolin, Jr. et al.	
5,923,846	A	7/1999	Gage et al.	
5,941,947	A	8/1999	Brown et al.	
5,956,489	A	9/1999	San Andres et al.	709/221
6,034,970	A	3/2000	Levac et al.	370/466
6,128,315	A	10/2000	Takeuchi	
6,141,710	A	10/2000	Miesterfeld	710/100
6,185,466	B1	2/2001	Nicewonger	
6,289,390	B1	9/2001	Kavner	719/310
6,363,427	B1	3/2002	Teibel et al.	
6,378,001	B1	4/2002	Aditham et al.	719/313
6,430,607	B1	8/2002	Kavner	
6,438,632	B1	8/2002	Kikugawa	
6,484,082	B1	11/2002	Millsap et al.	
6,725,348	B1	4/2004	Marier et al.	
6,801,942	B1	10/2004	Dietrich et al.	709/225
6,941,510	B1	9/2005	Ozzie et al.	
7,103,045	B2	9/2006	Lavigne et al.	370/392
7,103,646	B1	9/2006	Suzuki	709/220
7,103,656	B2	9/2006	Lewis et al.	709/223
7,552,440	B1	6/2009	Stewart et al.	719/312
7,950,017	B1	5/2011	Cain et al.	
2001/0018685	A1	8/2001	Saito et al.	
2002/0002586	A1	1/2002	Rafal et al.	
2002/0032856	A1	3/2002	Noguchi et al.	
2002/0047868	A1	4/2002	Miyazawa	
2002/0073243	A1	6/2002	Staiger	719/313
2002/0159460	A1	10/2002	Carrafiello et al.	
2002/0188691	A1	12/2002	Ignatius et al.	
2002/0191543	A1	12/2002	Buskirk et al.	
2002/0198943	A1	12/2002	Zhuang et al.	
2003/0061388	A1	3/2003	Cleghorn et al.	
2003/0074474	A1	4/2003	Roach et al.	
2003/0088568	A1	5/2003	Matsunaga et al.	
2003/0154116	A1	8/2003	Lofton	
2003/0236894	A1	12/2003	Herley	
2004/0003087	A1	1/2004	Chambliss	
2006/0155867	A1	7/2006	Kilian et al.	
2006/0155907	A1*	7/2006	Yoshida	G06F 13/24 710/260
2007/0101206	A1*	5/2007	Brabant	G06F 11/0757 714/55
2010/0333096	A1	12/2010	Dice et al.	
2011/0047218	A1	2/2011	Nojima et al.	
2011/0047310	A1	2/2011	Bonola	
2012/0029898	A1	2/2012	Carroll et al.	
2013/0111299	A1	5/2013	Hashimoto et al.	

OTHER PUBLICATIONS

William Wong, "Software and Hardware Standards Help, But In-Vehicle Network Growth Will Be Conservative: CAN networks and OSEK/VDX compatible operating systems will drive tomorrow's vehicles," Electronic Design, vol. 49, issue 1, 62 (Jan. 8, 2001).

OSEK/VDX Network Management Concept and Application Programming Interface, Version 2.51 (May 31, 2000).

Brad Johanson, Armando Fox, Pat Hanrahan, Terry Winograd, "The Event Heap: An Enabling Infrastructure for Interactive Workspaces," White Paper, Stanford University, Stanford, CA 1999.

Berwanger et al "FlexRay—The Communication System for Advanced Automotive Control Systems" 2001-01-0676, SAE 2001 World Congress, Mar. 5-8, 2001.

Bosch, "CAN Specification Version 2.0" Sep. 1991.

Clarke et al "Formal Verification of Autonomous Systems NASA Intelligent Systems Program" Carnegie Mellon University, Sep. 25, 2001.

Courtois et al "Concurrent Control with "Readers" and "Writers"" Communications of the ACM, vol. 14, No. 10, Oct. 1971.

Fredriksson, Lars-Berno "Controller Area Networks and the Protocol can for Machine Control Systems" Mechatronics vol. 4, No. 2, pp. 159-172, 1994.

Hernandez et al "Algorithms and architectures for real-time control 2000," Proceedings volume from the 6th IFAC Workshop, Palma de Mallorca, Spain, May 15-17, 2000.

Kirrmann et al. "The IEC/IEEE Train Communication Network," IEEE Micro, Mar.-Apr. 2001.

Koopman, Philip "Control Area Network (CAN)" 18-540 Distributed Embedded Systems, Presentation Slides, Oct. 4, 2000.

Leen et al, "Digital networks in the automotive vehicle," Computing and Control Engineering 10(6):257-266, Jan. 2000.

IEEE "The Authoritative Dictionary of IEEE Standards Terms" 7th Ed., IEEE Press, 2000.

IEEE "IEEE Standard for Information Technology—Portable Operating System Interface (POSIX®)—Part 1: System Application Program Interface (API)—Amendment d: Additional Realtime Extensions," IEEE Std 1003.1d-1999 (Amendment to IEEE Std 1003.1-1990), Jun. 16, 2000.

Mellor-Crummey et al. "Algorithms for Scalable Synchronization on Shared-Memory Multiprocessors," ACM Transactions on Computer Systems, vol. 9, No. 1, Feb. 1991, pp. 21-65.

Jagannathan et al. "Blackboard Architectures and Applications," vol. 3, Academic Press Inc., 1989, 524 pages.

SAE International, "Surface Vehicle Standard—SAE J1850" Rev May 2001.

Intelligent Transportation Systems Standards, United States Department of Transportation "SAE J2366—Family of ITS Data Bus (IDB) Protocol Standards," (online fact sheet written Jul. 12, 2006), available at https://www.standards.its.dot.gov/Factsheets/PrintFactsheet/56.

SAE International, "Surface Vehicle Recommended Practice—SAE J2366-1" Nov. 2001.

SAE International, "Surface Vehicle Recommended Practice—SAE J2366-2" Nov. 2001.

Schill, John "An Overview of the CAN Protocol," Embedded Systems Programming, Sep. 1997, pp. 46-62.

Stewart et al., "Integration of Real-Time Software Modules for Reconfigurable Sensor-Based Control Systems," Proceedings of the 1992 IEEE/RSJ International Conference on Intelligent Robots and Systems, Jul. 7-10, 1992.

Tanenbaum, Andrew S. "Computer Networks," 3rd Ed., Prentice Hall, Inc, 1996, pp. 1-76 (Introduction).

Tanenbaum, Andrew S. "Modern Operating Systems," 2nd Ed., Prentice Hall, Inc., 2001, excerpts from Chapters 2, 7, 8, and 10.

Upender et al., "Embedded Communication Protocol Options," Proceedings of the Fifth Annual Embedded Systems Conference, vol. 2, Miller Freeman, Inc., 1993.

Wargui et al. "Application of Controller Area Network to Mobile Robots," IEEE, Electrotechnical Conference, May 16, 1996, p. 205-207.

Wense et al. "Building Automotive LIN Applications," pp. 279-292, found as part of Krueger et al., "Advanced Microsystems for Automotive Applications 2001," Springer, 2001.

"An Embedded Software Primer," David E. Simon, 1999.

"Application of In-Vehicle Data Buses for Collision Avoidance Systems," William D. Horne, et al., 1998.

"The Foundation.TM Fieldbus Primer," Fieldbus Inc., Jun. 24, 2001.

"GSA Guide to Specifying Interoperable Building Automation and Control Systems Using ANSI/ASHRAE Standard 135-1995," Steven T. Bushby et al., Nov. 1999.

"Programming in the OSEK/VDX Environment," Joseph Lemieux, 2001.

"OSEK/VDX Binding Specification Version 1.3," OSEK Group, Sep. 17, 2001.

"OSEK/VDX Binding Specification Version 1.4," OSEK Group, Sep. 6, 2002.

"OSEK/VDX Binding Specification Version 1.4.1," OSEK Group, Jan. 21, 2003.

"OSEK/VDX Communication Specification Version 2.2.2," OSEK Group, Dec. 18, 2000.

"OSEK/VDX Communication Specification Version 3.0," OSEK Group, Jul. 26, 2002.

"OSEK/VDX Communication Specification Version 3.0.1," OSEK Group, Jan. 29, 2003.

(56) **References Cited**

OTHER PUBLICATIONS

"OSEK/VDX Communication Specification Version 3.0.2," OSEK Group, Dec. 9, 2003.
"OSEK/VDX Fault-Tolerant Communication Specification Version 1.0," OSEK Group, Jul. 24, 2001.
"OSEK/VDX Network Management Concept and Application Programming Interface Version 2.5.0," OSEK Group, May 31, 1998.
"OSEK/VDX Network Management Concept and Application Programming Interface Version 2.5.1," OSEK Group, May 31, 2000.
"OSEK/VDX Network Management Concept and Application Programming Interface Version 2.5.2," OSEK Group, Jan. 16, 2003.
"OSEK/VDX OSEK Implementation Language Specification Version 2.3," OSEK Group, Sep. 10, 2001.
"OSEK/VDX OSEK Implementation Language Specification Version 2.4," OSEK Group, Dec. 2, 2002.
"OSEK/VDX OSEK Implementation Language Specification Version 2.4.1," OSEK Group, Jan. 23, 2003.
"Goals and Motivation: What is OSEK/VDX," OSEK VDX Portal, 2012.
"OSEK/VDX OSEK Run Time Interface Part A: Language Specification Version 2.1," OSEK Group, Jul. 16, 2001.
"OSEK/VDX OSEK Operating System Specification Version 2.0 revision 1," OSEK Group, Oct. 15, 1997.
"OSEK/VDX OSEK Operating System Specification Version 2.1 revision 1," OSEK Group, Nov. 13, 2000.
"OSEK/VDX OSEK Operating System Specification Version 2.2," OSEK Group, Sep. 10, 2001.
"OSEK/VDX OSEK Operating System Specification Version 2.2.1," OSEK Group, Jan. 16, 2003.
"OSEK/VDX Time-Triggered Operating System Specification Version 1.0," OSEK Group, Jul. 24, 2001.

* cited by examiner

FIGURE 1

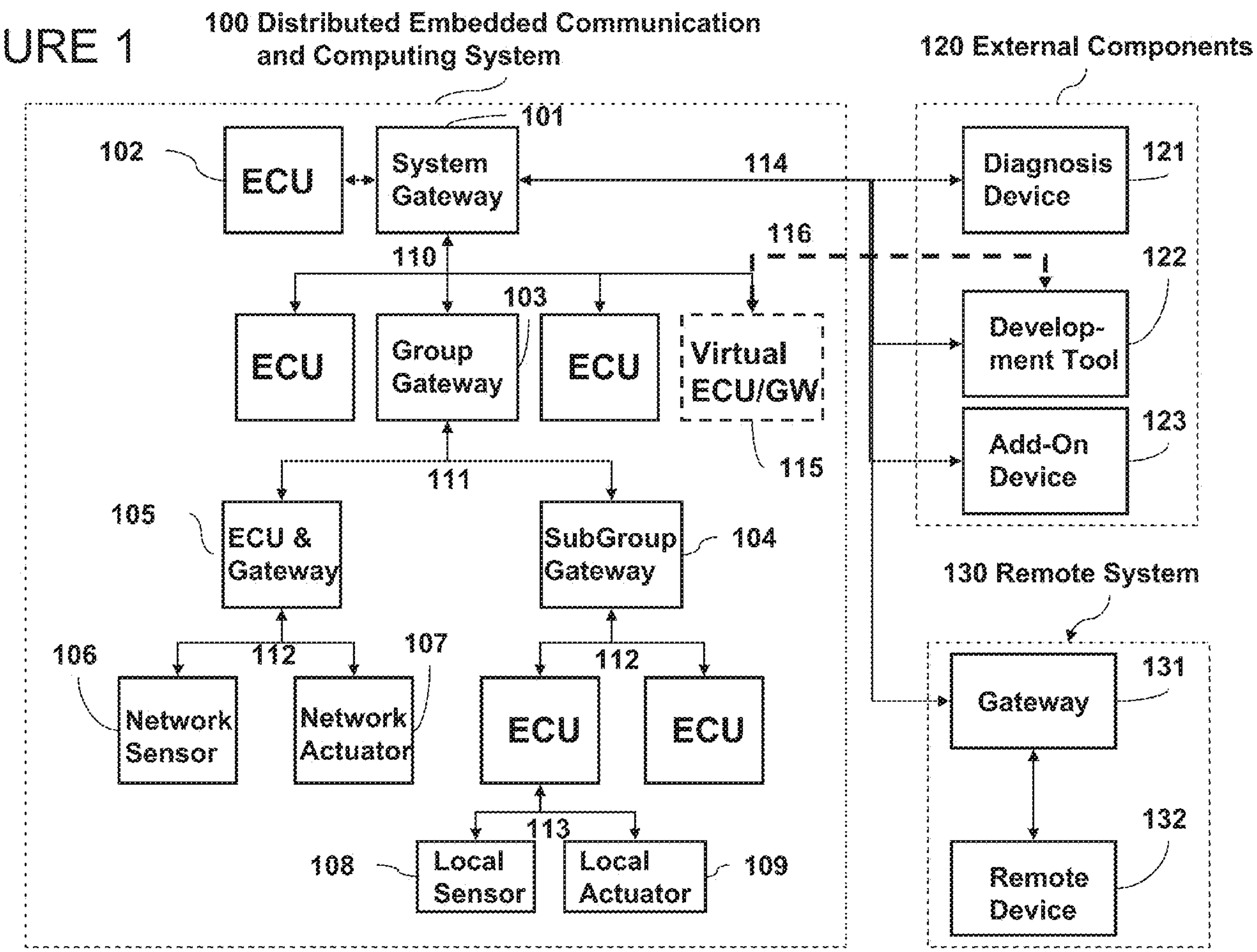

100 Distributed Embedded Communication and Computing System
120 External Components
101
102
ECU
System Gateway
114
Diagnosis Device
121
116
110
122
103
ECU
Group Gateway
ECU
Virtual ECU/GW
Develop-ment Tool
123
Add-On Device
115
111
105
ECU & Gateway
SubGroup Gateway
104
130 Remote System
106
112
107
112
Network Sensor
Network Actuator
ECU
ECU
Gateway
131
132
113
108
Local Sensor
Local Actuator
109
Remote Device

FIGURE 2

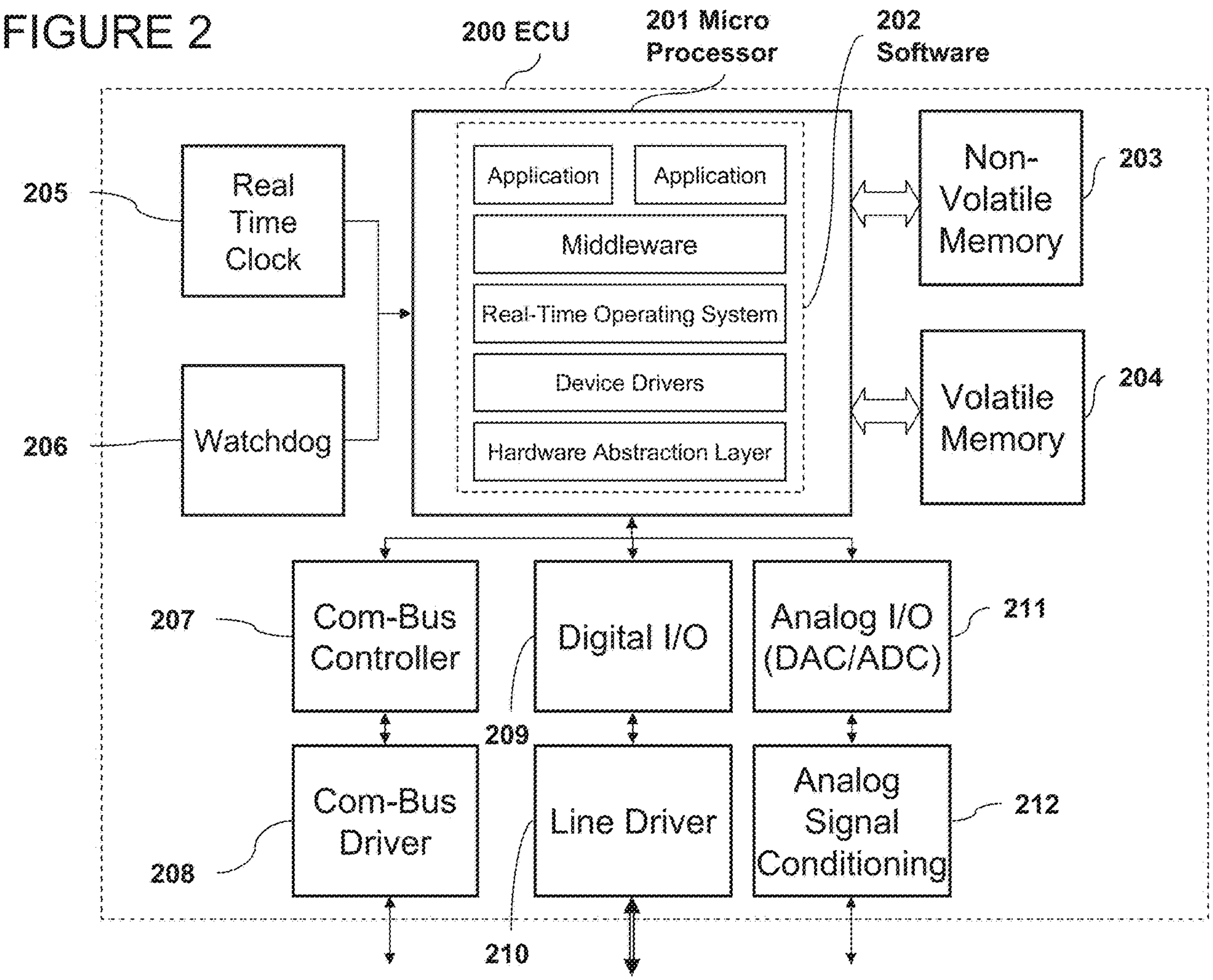

200 ECU
201 Micro Processor
202 Software
Application
Application
Middleware
Real-Time Operating System
Device Drivers
Hardware Abstraction Layer
Real Time Clock
205
Watchdog
206
Non-Volatile Memory
203
Volatile Memory
204
Com-Bus Controller
207
Digital I/O
209
Analog I/O (DAC/ADC)
211
Com-Bus Driver
208
Line Driver
210
Analog Signal Conditioning
212

FIGURE 3

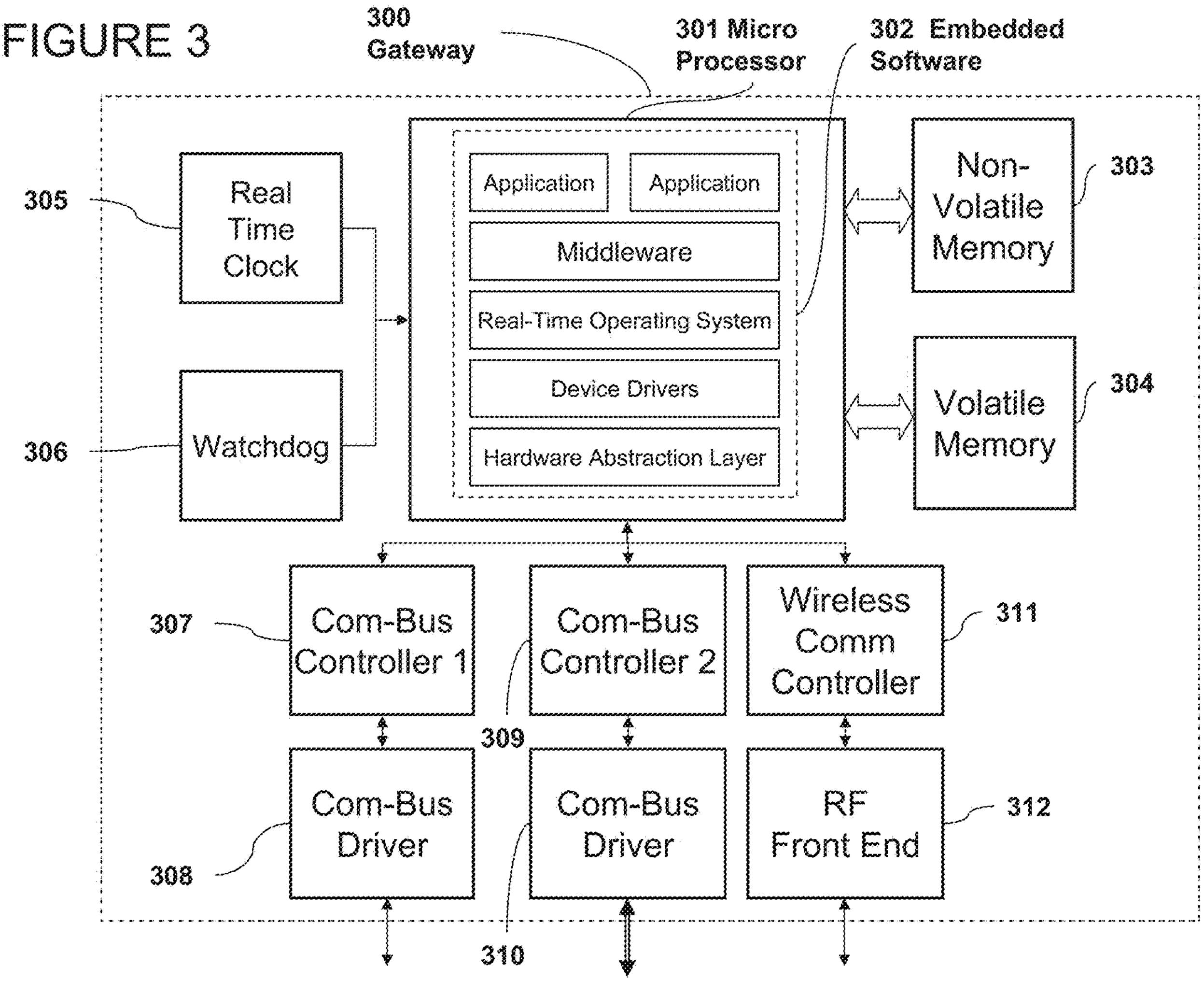

300
Gateway
301 Micro
Processor
302 Embedded
Software
Application
Application
Middleware
Real-Time Operating System
Device Drivers
Hardware Abstraction Layer
Real
Time
Clock
305
Watchdog
306
Non-
Volatile
Memory
303
Volatile
Memory
304
Com-Bus
Controller 1
307
Com-Bus
Controller 2
309
Wireless
Comm
Controller
311
Com-Bus
Driver
308
Com-Bus
Driver
310
RF
Front End
312

FIGURE 4

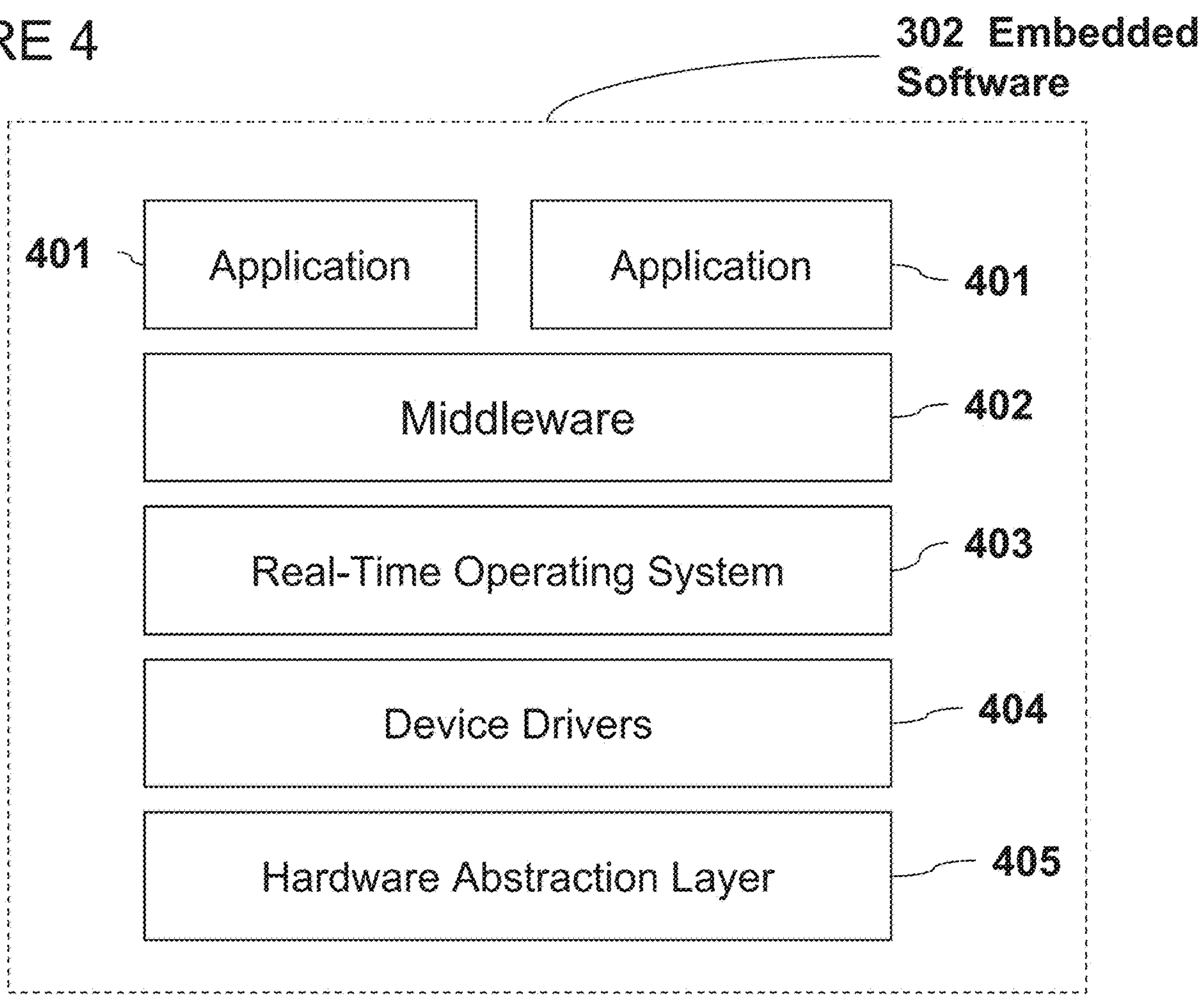

302 Embedded Software
401
Application
Application
401
Middleware
402
Real-Time Operating System
403
Device Drivers
404
Hardware Abstraction Layer
405

FIGURE 5

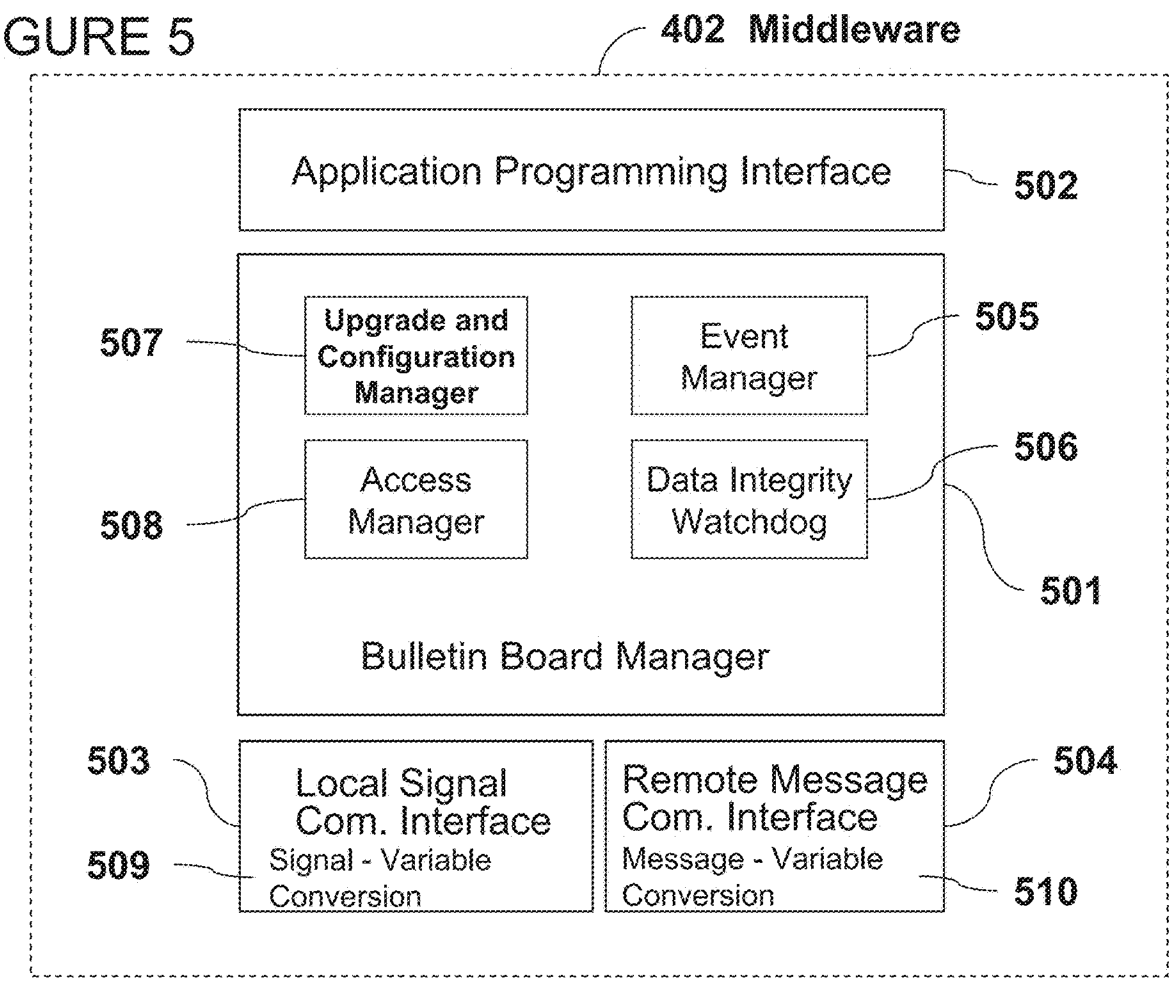

402 Middleware
Application Programming Interface
502
507
Upgrade and Configuration Manager
Event Manager
505
506
Access Manager
Data Integrity Watchdog
508
501
Bulletin Board Manager
503
Local Signal Com. Interface
Signal - Variable Conversion
509
Remote Message Com. Interface
Message - Variable Conversion
504
510

FIGURE 6

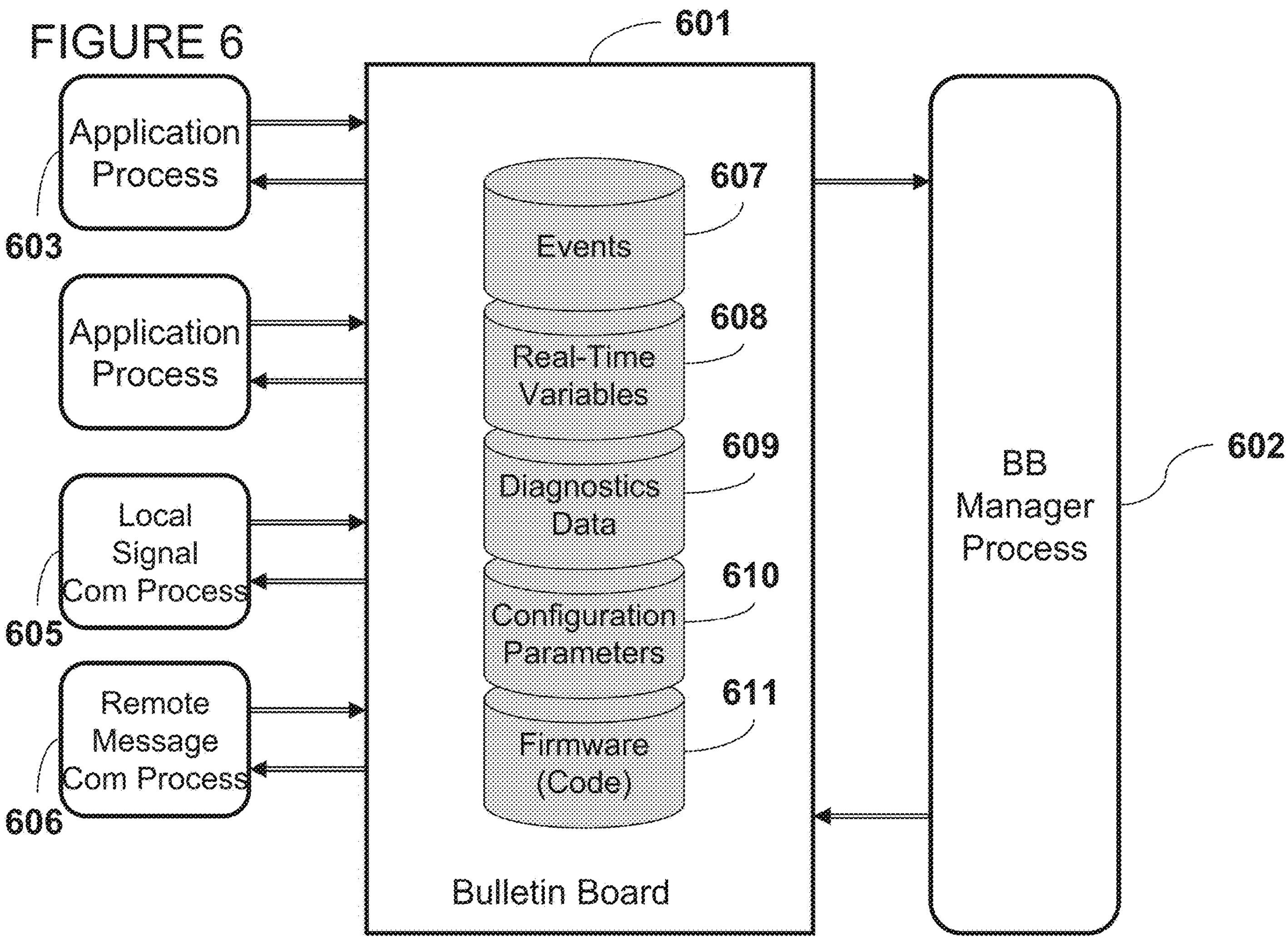

601
Application Process
603
Application Process
Local Signal Com Process
605
Remote Message Com Process
606
Events
607
Real-Time Variables
608
Diagnostics Data
609
Configuration Parameters
610
Firmware (Code)
611
Bulletin Board
BB Manager Process
602

FIGURE 7

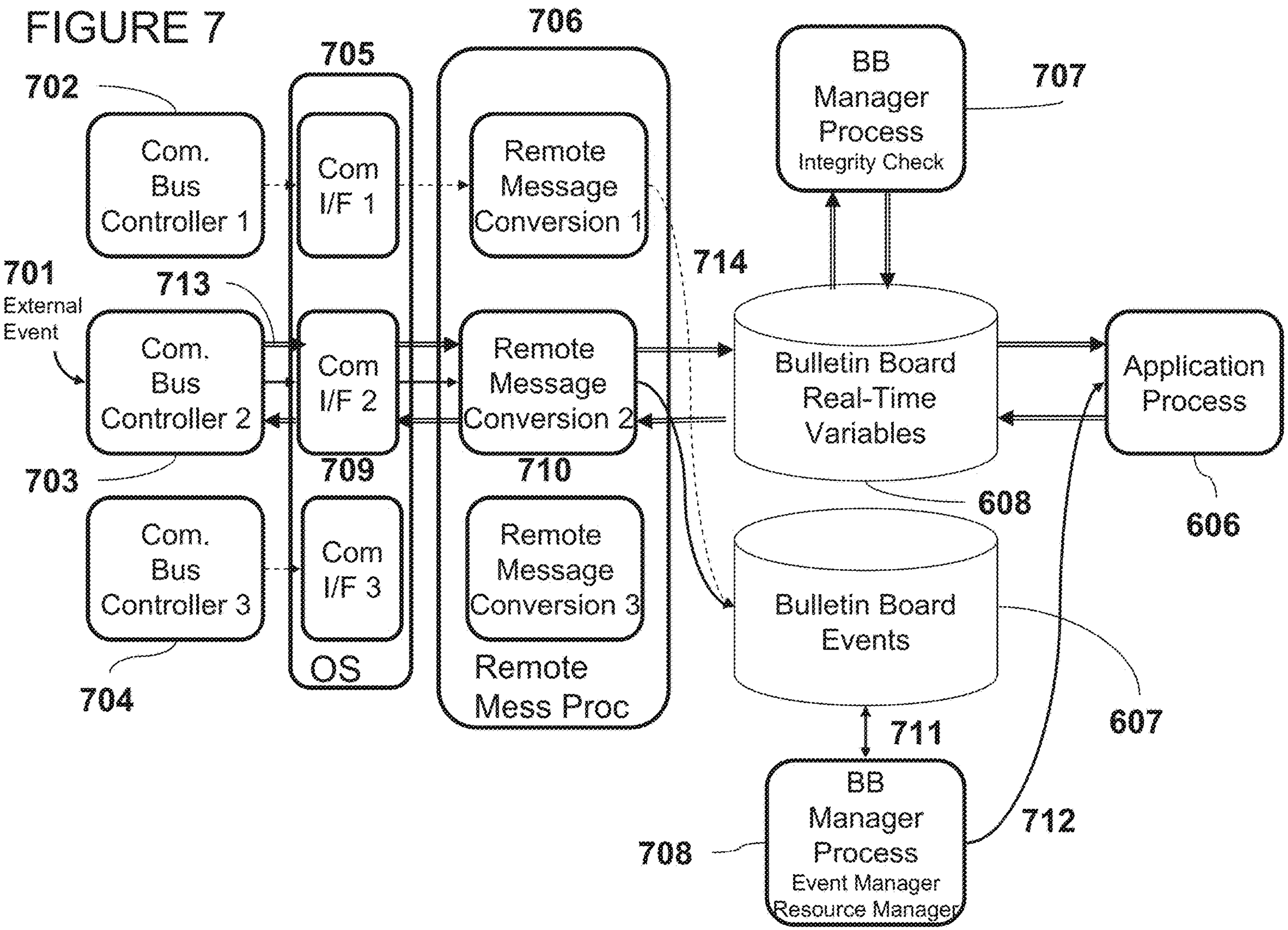

702
Com. Bus Controller 1
701
External Event
713
Com. Bus Controller 2
703
Com. Bus Controller 3
704
705
Com I/F 1
Com I/F 2
709
Com I/F 3
OS
706
Remote Message Conversion 1
Remote Message Conversion 2
710
Remote Message Conversion 3
Remote Mess Proc
714
BB Manager Process
Integrity Check
707
Bulletin Board Real-Time Variables
608
Bulletin Board Events
607
711
Application Process
606
708
BB Manager Process
Event Manager
Resource Manager
712

FIGURE 8

Store Variable from remote I/O

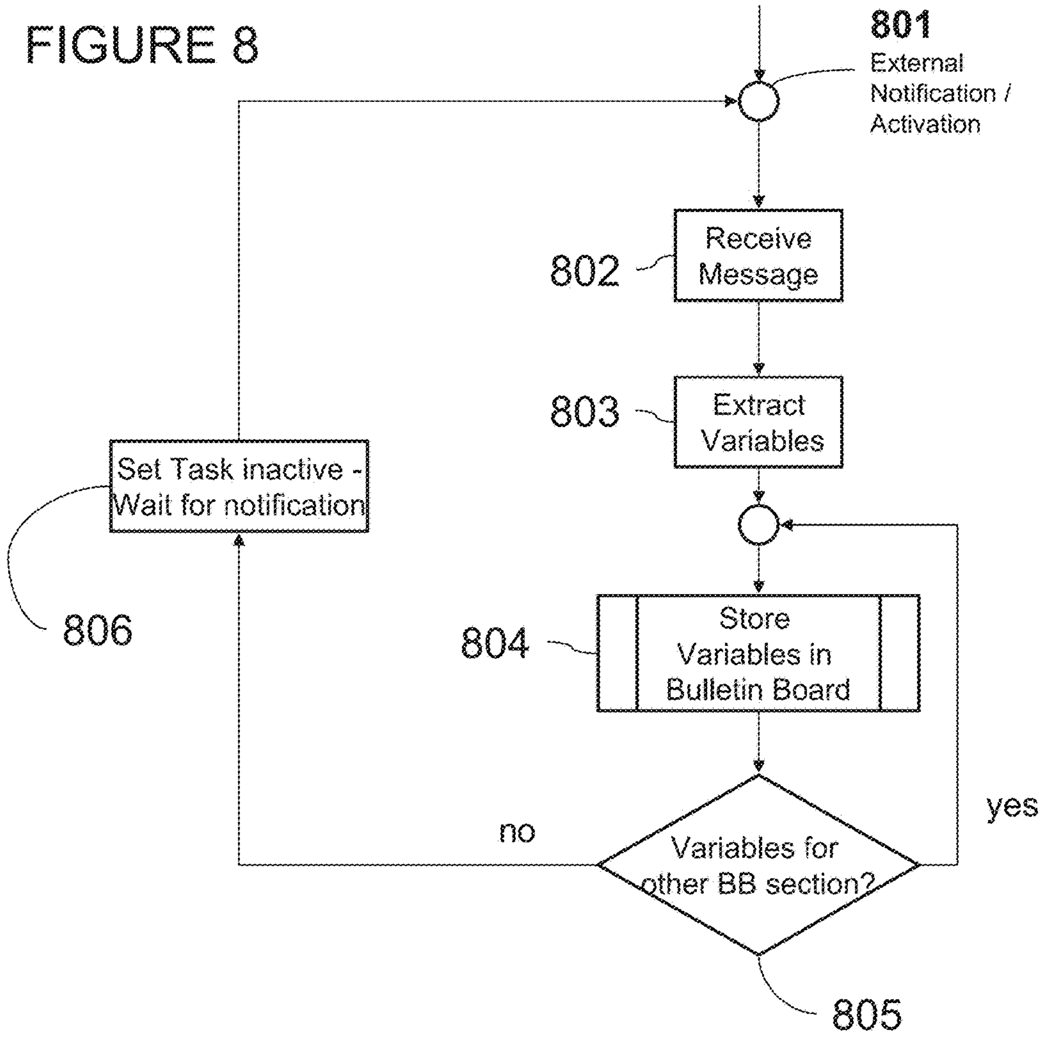


FIGURE 9

Store Variable from local I/O

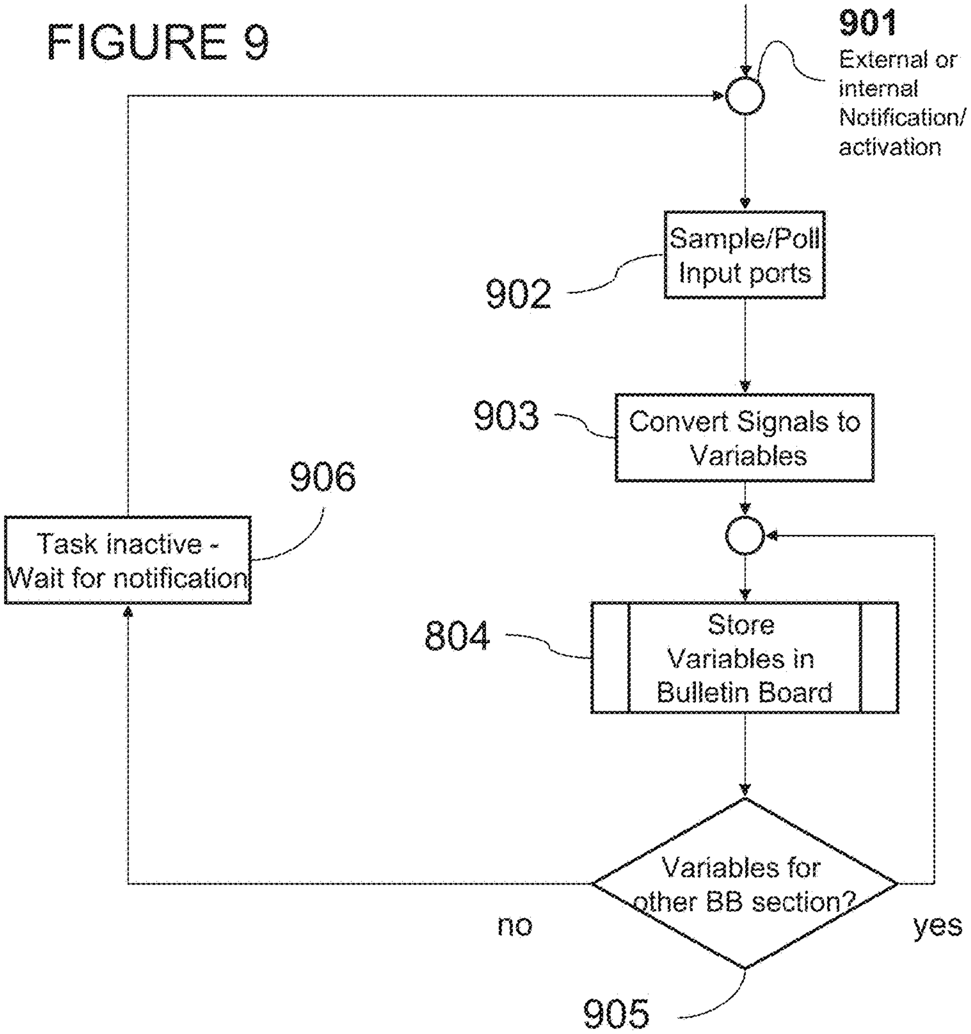
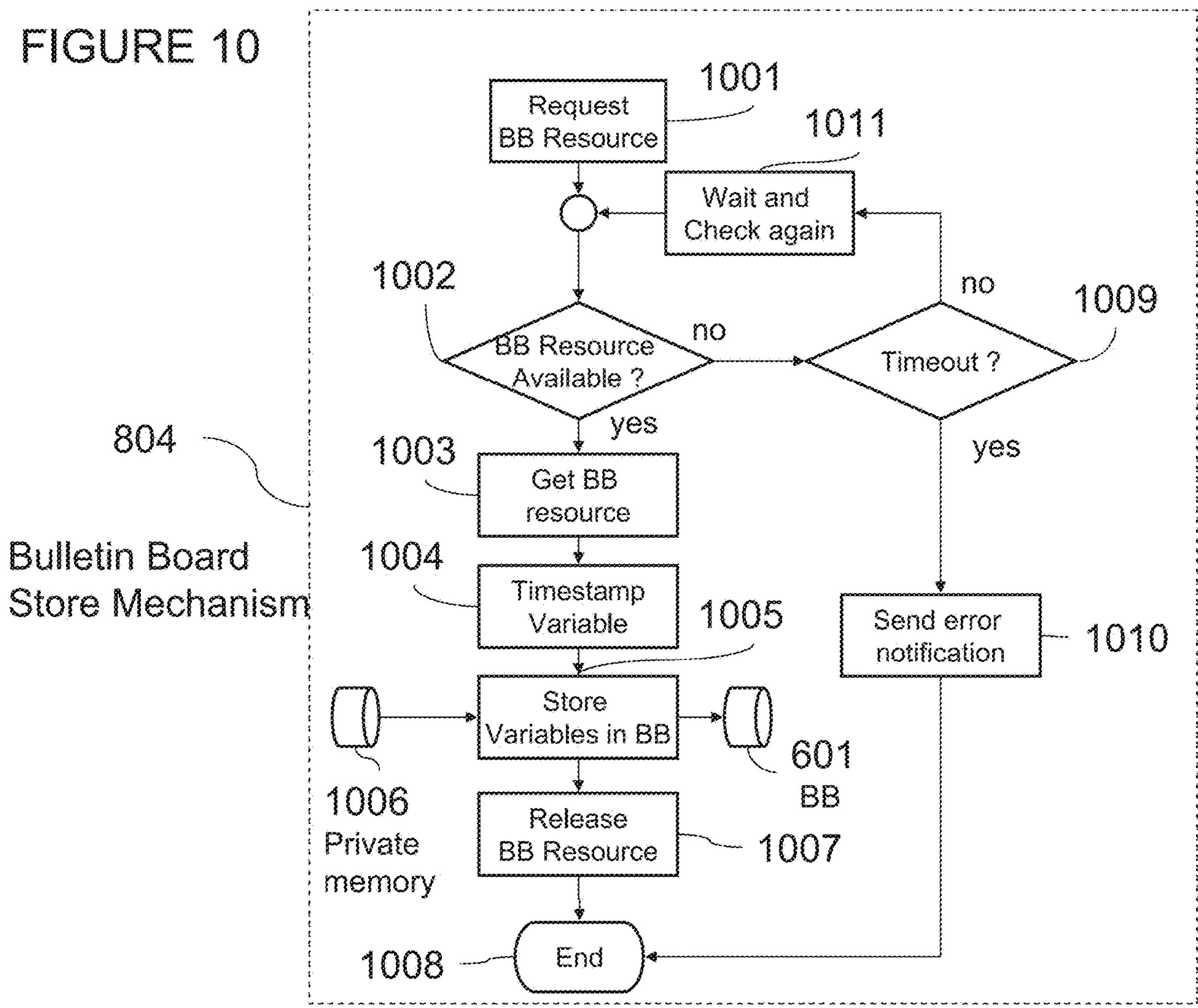


FIGURE 10



804
Bulletin Board
Store Mechanism
Request
BB Resource
1001
1011
Wait and
Check again
1002
BB Resource
Available ?
no
no
1009
Timeout ?
yes
yes
1003
Get BB
resource
1004
Timestamp
Variable
1005
Send error
notification
1010
Store
Variables in BB
601
BB
1006
Private
memory
Release
BB Resource
1007
1008
End

FIGURE 11

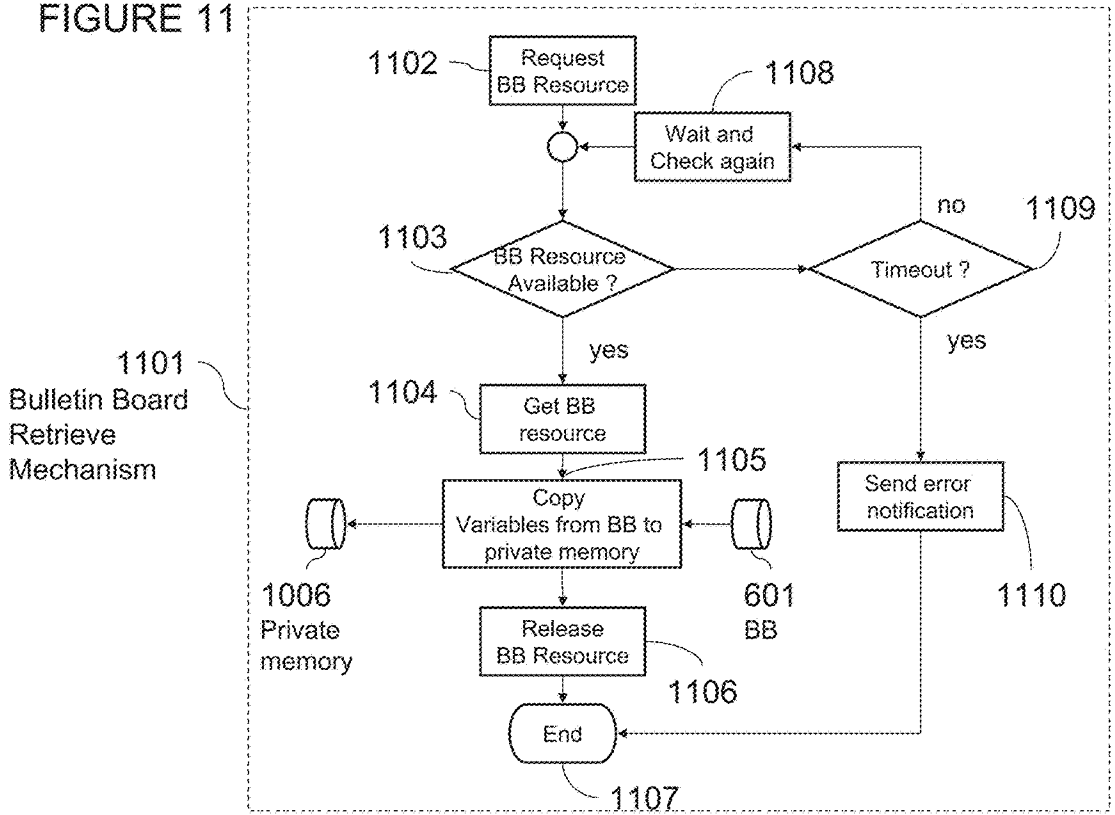

1102
Request BB Resource
1108
Wait and Check again
1103
BB Resource Available ?
no
1109
Timeout ?
yes
yes
1101
Bulletin Board Retrieve Mechanism
1104
Get BB resource
1105
Copy Variables from BB to private memory
Send error notification
1006
Private memory
601
BB
1110
Release BB Resource
1106
End
1107

FIGURE 12

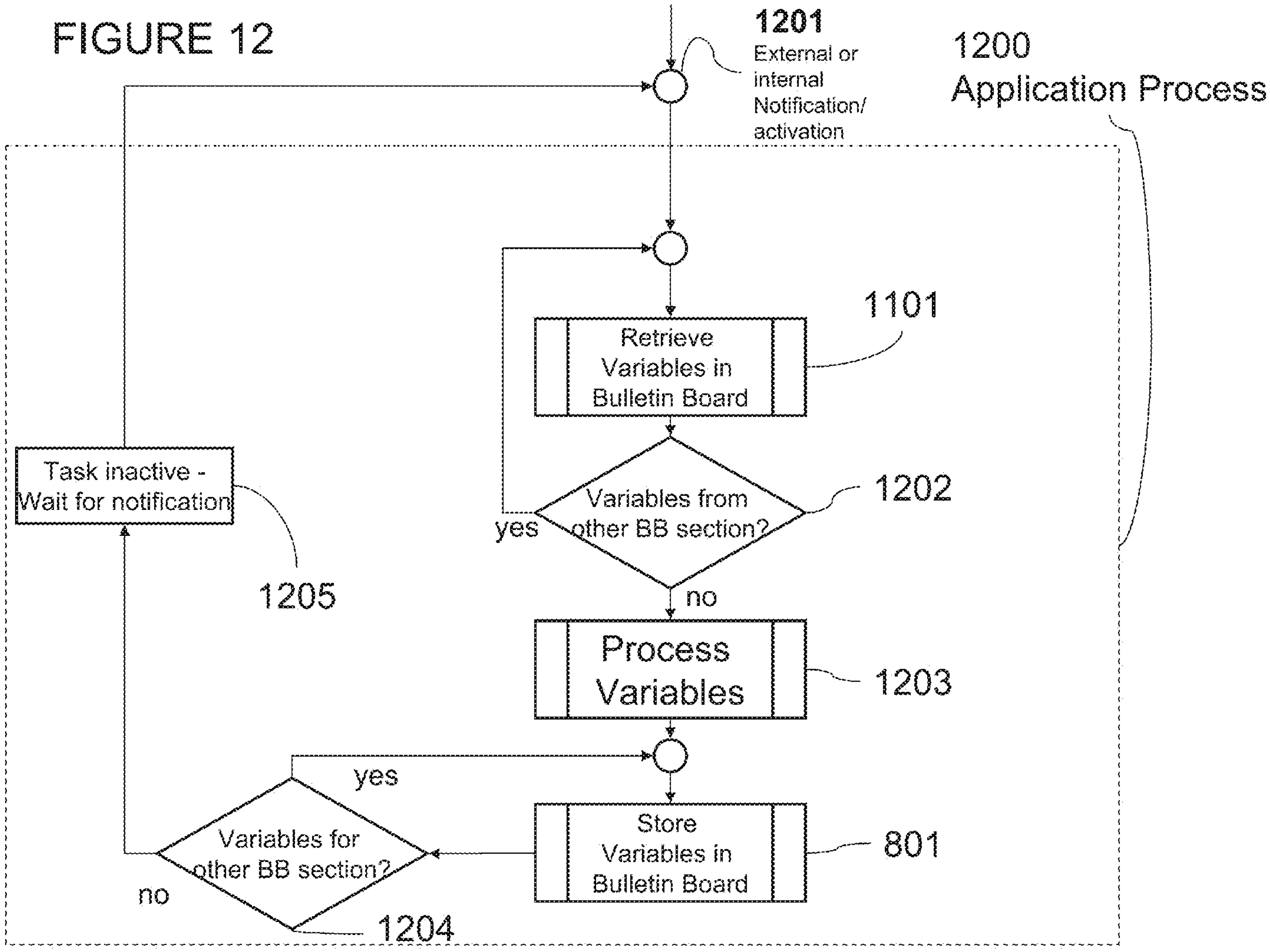

1201
External or internal Notification/ activation
1200
Application Process
1101
Retrieve Variables in Bulletin Board
Variables from other BB section?
yes
no
1202
Process Variables
1203
Task inactive - Wait for notification
1205
yes
Variables for other BB section?
no
1204
Store Variables in Bulletin Board
801

FIGURE 13

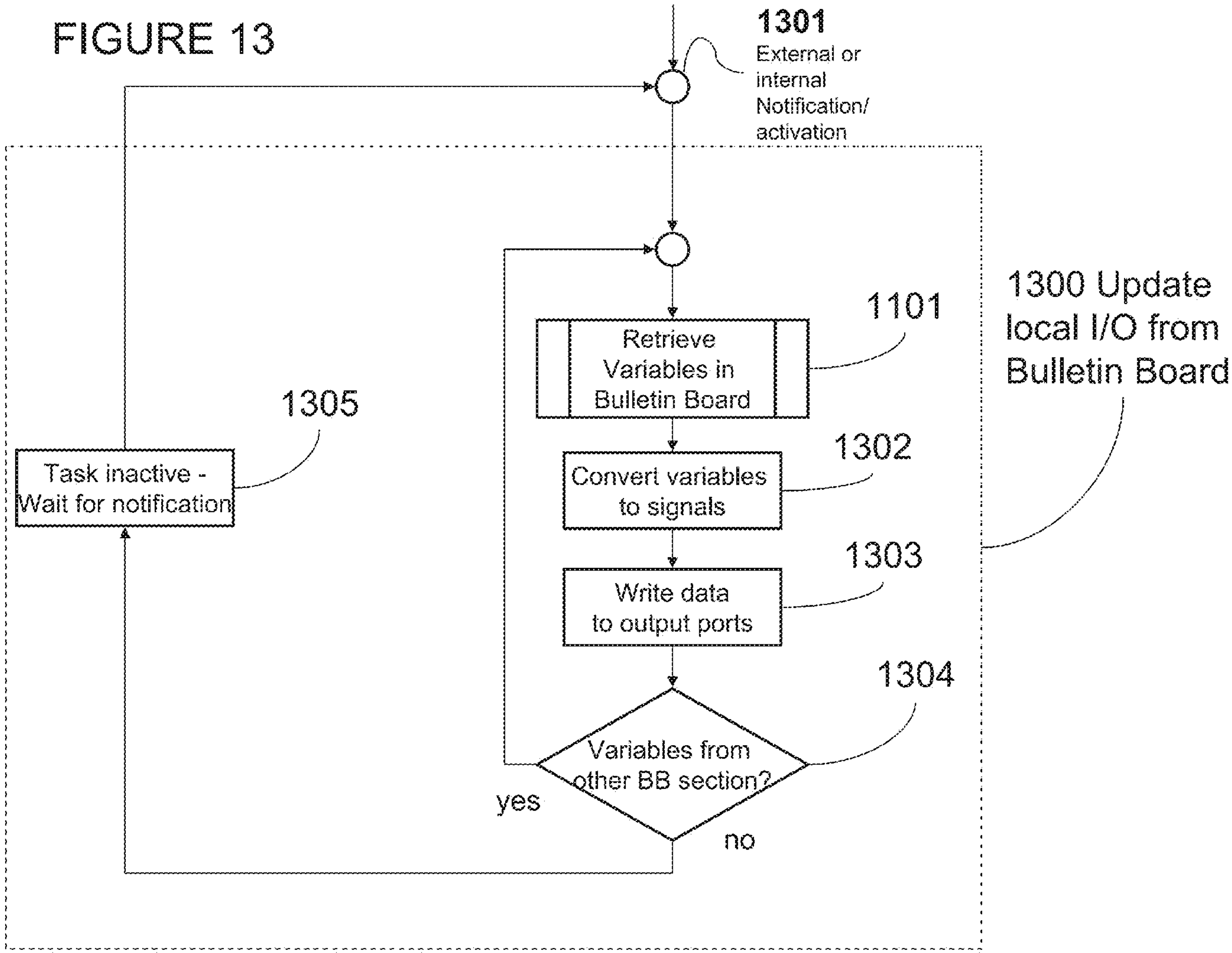
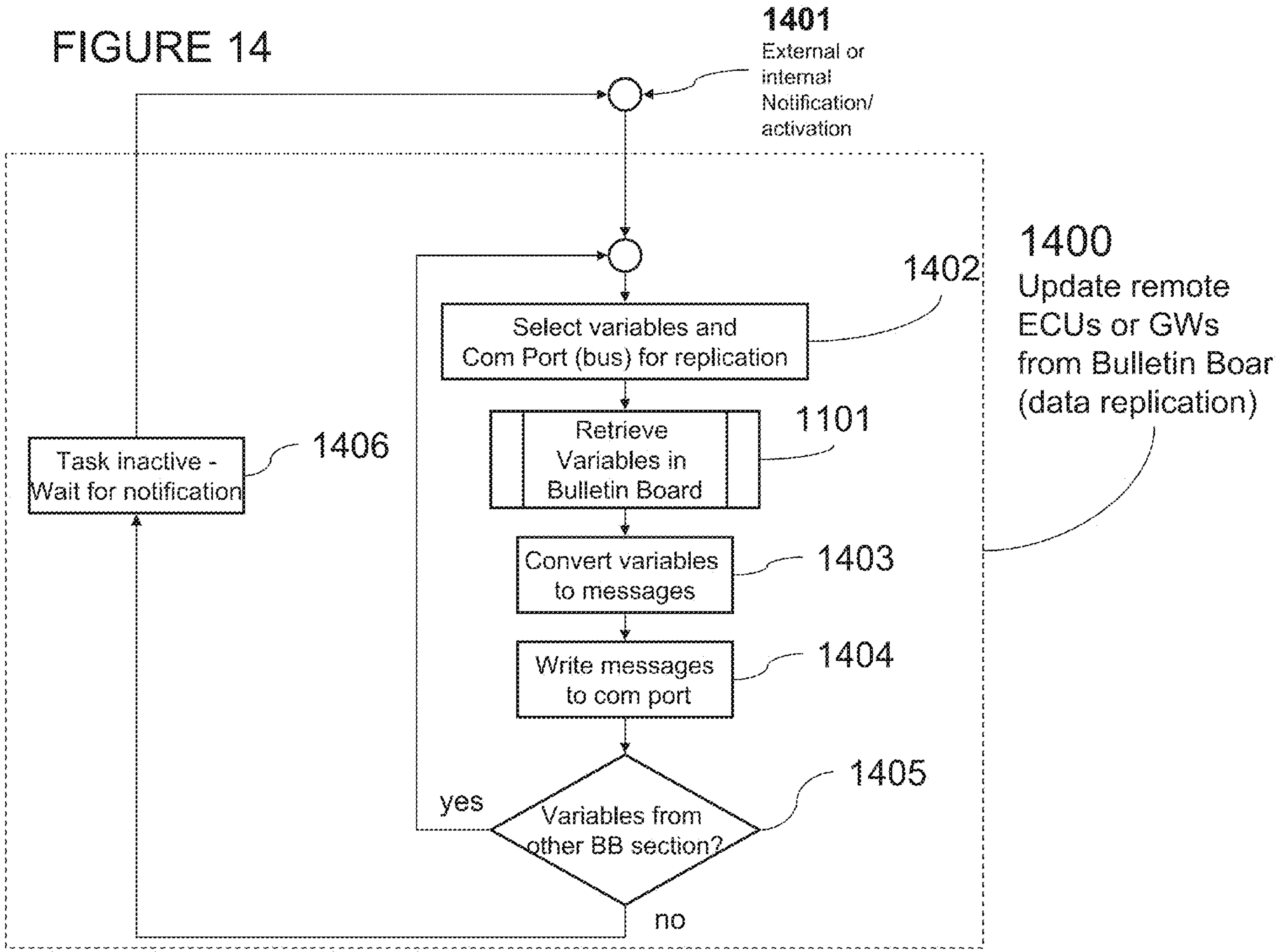

1301
External or
internal
Notification/
activation
1300 Update
local I/O from
Bulletin Board
1101
Retrieve
Variables in
Bulletin Board
1302
Convert variables
to signals
1303
Write data
to output ports
1304
Variables from
other BB section?
yes
no
1305
Task inactive -
Wait for notification

FIGURE 14



1401
External or internal Notification/ activation
1400
Update remote ECUs or GWs from Bulletin Boar (data replication)
1402
Select variables and Com Port (bus) for replication
1101
Retrieve Variables in Bulletin Board
1403
Convert variables to messages
1404
Write messages to com port
1405
Variables from other BB section?
yes
no
1406
Task inactive - Wait for notification

FIGURE 15

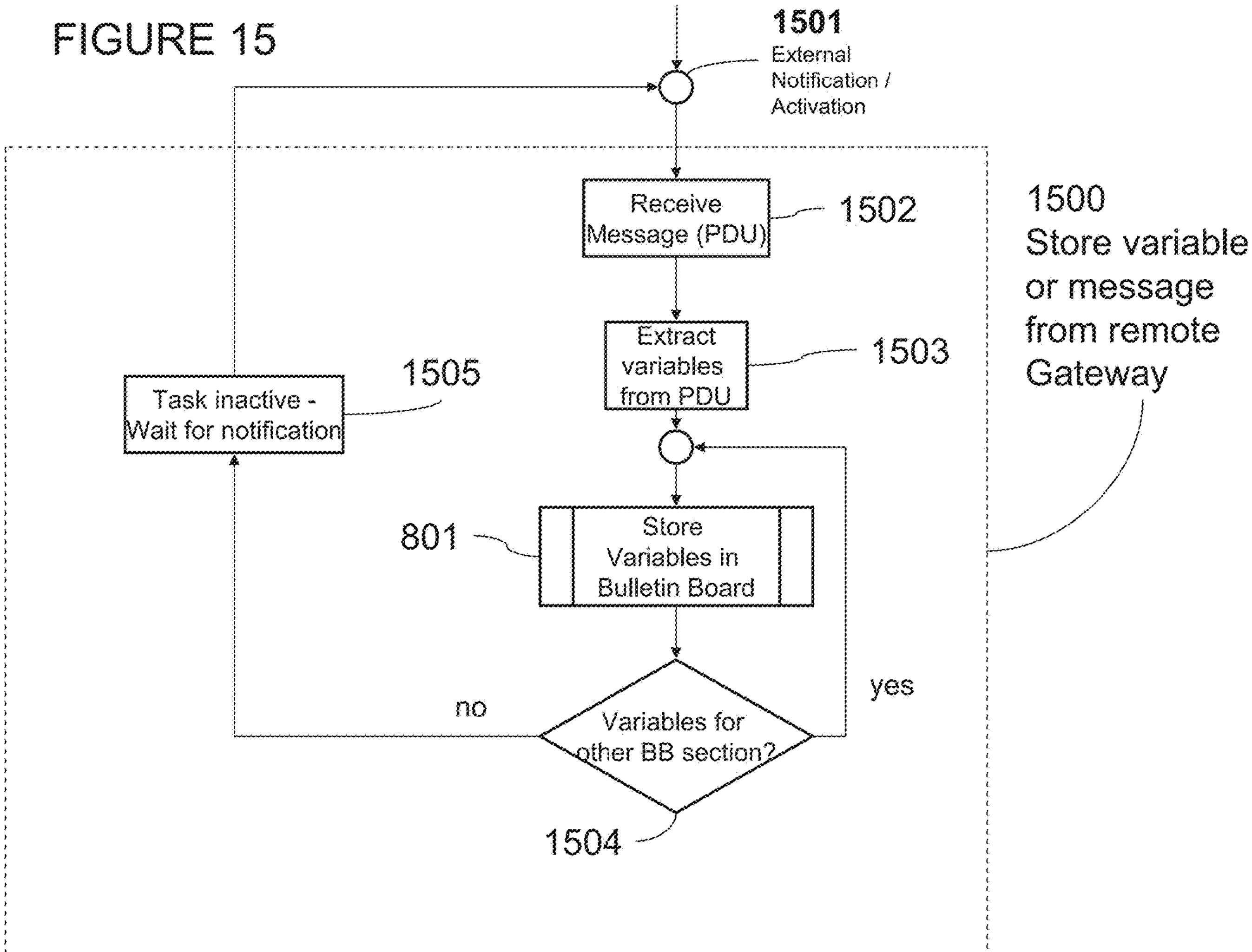

1501
External Notification / Activation
Receive Message (PDU)
1502
Extract variables from PDU
1503
1500
Store variable or message from remote Gateway
Task inactive - Wait for notification
1505
801
Store Variables in Bulletin Board
no
yes
Variables for other BB section?
1504

FIGURE 16

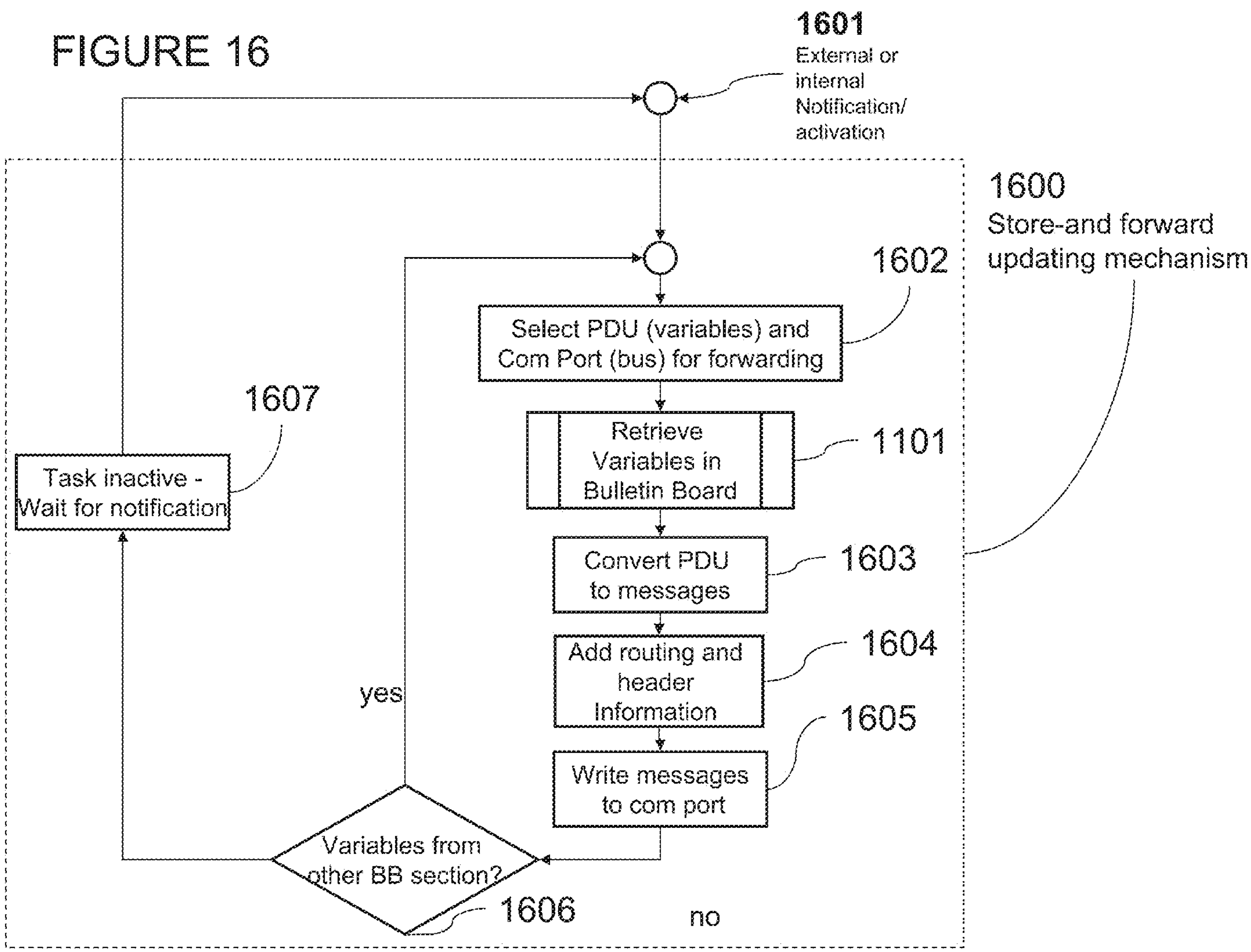
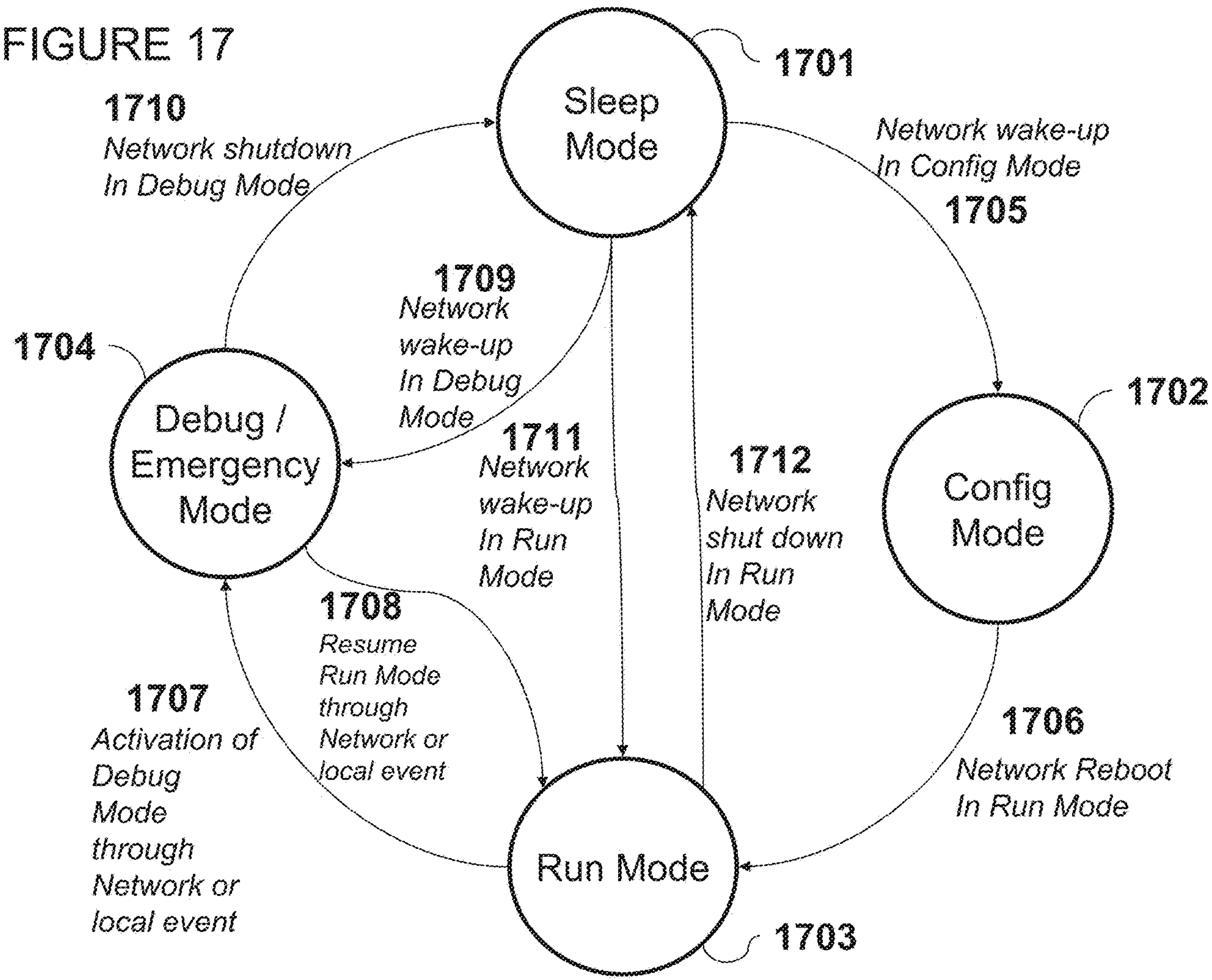

1601
External or internal Notification/ activation
1600
Store-and forward updating mechanism
1602
Select PDU (variables) and Com Port (bus) for forwarding
1101
Retrieve Variables in Bulletin Board
1607
Task inactive - Wait for notification
1603
Convert PDU to messages
1604
Add routing and header Information
1605
Write messages to com port
yes
Variables from other BB section?
1606
no

FIGURE 17


Sleep Mode
1701
Network wake-up In Config Mode
1705
Config Mode
1702
1710
Network shutdown In Debug Mode
1709
Network wake-up In Debug Mode
1704
Debug / Emergency Mode
1711
Network wake-up In Run Mode
1712
Network shut down In Run Mode
1708
Resume Run Mode through Network or local event
1707
Activation of Debug Mode through Network or local event
1706
Network Reboot In Run Mode
Run Mode
1703

SYSTEM, METHOD AND COMPUTER PROGRAM PRODUCT FOR SHARING INFORMATION IN A DISTRIBUTED FRAMEWORK

RELATED APPLICATION(S)

This application is a continuation of U.S. patent application Ser. No. 14/011,705 filed Aug. 27, 2013, which is a continuation of U.S. patent application Ser. No. 13/531,319 filed Jun. 22, 2012, now U.S. Pat. No. 8,566,843, which is a continuation of U.S. patent application Ser. No. 12/182,570 filed Jul. 30, 2008, now U.S. Pat. No. 8,209,705, which is a continuation of U.S. patent application Ser. No. 10/737,690 filed Dec. 15, 2003, now U.S. Pat. No. 7,802,263, which, in turn, claims priority under 35 U.S.C. §119 based on U.S. Provisional Application No. 60/434,018 filed Dec. 17, 2002, all of which are incorporated herein by reference.

FIELD AND BACKGROUND OF THE INVENTION

The present invention relates to the field of distributed control and monitoring systems that may include certain temporal behavior.

Such technology may optionally apply to electronic vehicle communication and control systems, real-time monitoring systems, industrial automation and control systems, as well as any other desired system.

SUMMARY OF THE INVENTION

A system, method and computer program product are provided for sharing information in a distributed system. After information is received, it is stored on a bulletin board. In use, the information is shared, in real-time, among a plurality of heterogeneous processes.

In one embodiment, both past and present instances of the information may be stored on the bulletin board. As an option, the information may be replicated among a plurality of the bulletin boards. Optionally, first information may be processed utilizing a first bulletin board and stored utilizing a second bulletin board. Still yet, the bulletin boards may be hierarchical.

In another embodiment, the processes may access multiple sections of the bulletin board. Further, the bulletin board may send notifications to the processes based on a state of the information on the bulletin board.

Optionally, the information may include variables. For example, the information may include input variables, output variables, etc. Moreover, the processes may include local processes, remote processes, etc. Still yet, the processes may include event triggered processes and/or time triggered processes. In use, each of the processes may process the information in a manner that is isolated from temporal characteristics associated with the network.

In still another embodiment, the information may be extracted from a message received by a bulletin board manager. Moreover, the information may be converted from a signal received by a bulletin board manager. Even still, the information may be shared in a single task, may be shared according to a schedule, and/or may be shared with an operating system. Optionally, dynamic preemptive scheduling may be provided. Also, the information may be shared across the communication network with only a portion of a message header that is needed for a specific communication link while other communication links may use a different message header.

As a further option, resources in the network may be protected. Specifically, the resources in the network may be protected utilizing a schedule that allows information sharing utilizing the bulletin board. In another embodiment, the resources in the network may be protected utilizing semaphores.

In even still another embodiment, the information may be shared according to an internal clock, an external clock, etc. During operation, objects may be generated based on a change of state of the information stored in the bulletin board. Such objects may include, but are not limited to flags, events, signals, interrupts, etc. Still yet, the information may be stored in response to interrupts associated with the processes.

In use, the bulletin board may update the processes with information at a first rate that differs from a second rate with which the processes send the information to the bulletin board. Optionally, the bulletin board may be accessed with guaranteed access times, jitter, and bandwidth.

In addition, the bulletin board may be updated irregularly and triggered by internal or external objects including, but not limited to flags, events, signals, interrupts, etc. Event triggers may be provided independent of a link connection between nodes where the processes are carried out. Moreover, failure redundancy may be provided through multiple independent links across diverse physical connections.

As yet another option, the information may have a user-configured constraint associated therewith. Such constraint may include a memory constraint, a real-time constraint, etc. As a further option, the constraint may be configured utilizing a tool.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** is a block diagram of an embodiment of a system of one embodiment;

FIG. **2** is a block diagram generally depicting an embodiment of an ECU as part of the system illustrated in FIG. **1**;

FIG. **3** is a block diagram generally depicting an embodiment of a Gateway device as part of the system illustrated in FIG. **1**;

FIG. **4** is a block diagram of an embodiment of the software architecture assumed for one embodiment.

FIG. **5** is a block diagram of an embodiment of the middleware that contains the methods of one embodiment.

FIG. **6** is a block diagram of an embodiment of the bulletin board that describes the process interaction of one embodiment.

FIG. **7** is a block diagram of an embodiment of the bulletin board that describes the process interaction with multiple external communication buses as part of one embodiment.

FIG. **8** is a flow chart diagram of an embodiment of the variable store from remote I/O method of one embodiment.

FIG. **9** is a flow chart diagram of an embodiment of the variable store from local I/O method of one embodiment.

FIG. **10** is a flow chart diagram of an embodiment of the variable method of one embodiment.

FIG. **11** is a flow chart diagram of an embodiment of the variable retrieve method of one embodiment.

FIG. **12** is a flow chart diagram of an embodiment of the application process using the method of one embodiment

FIG. **13** is a flow chart diagram of an embodiment of the local I/O update from bulletin board method of one embodiment.

FIG. **14** is a flow chart diagram of an embodiment of the variable replication method of one embodiment.

FIG. **15** is a flow chart diagram of an embodiment of the message store from remote gateway method of one embodiment.

FIG. **16** is a flow chart diagram of an embodiment of the message forward to remote ECU or Gateway method of one embodiment.

FIG. **17** is a state transition diagram of an embodiment of the mode switching method of one embodiment.

DETAILED DESCRIPTION

FIG. **1** is a block diagram generally depicting elements of an embodiment of the present distributed embedded communication and computing system. The system architecture may be situated in automotive electronics or industrial control and monitoring systems. In an automotive environment, the various Electronic Control Units (ECUs, **102**) control complex applications such as engine control, brake control, or diagnostics. They are either connected to sensors and actuators via discrete links or simple standard functions such as sensors and actuators are organized into separate sub networks.

These complex functions such as braking, engine-control, etc. are then grouped into the backbone system functions for the car, such as body control, power train and chassis. The backbone also includes the vehicle's high level functions such as diagnostics, telematics and entertainment systems.

Therefore the system is typically hierarchically organized and includes a variety of gateways (**101**,**104**,**105**), which relay messages up and down through the system layers. Each layer may contain multiple electronic control units (ECU, **102**) that are connected through wired serial multiplexing bus-systems such as Controller Area Network (CAN or ISO11898), Flexray, LIN, J1850, J1708, MOST, IEEE1394, and other similar serial multiplexing buses or through wireless multiplexing systems such as IEEE802.11, IEEE802.15, Bluetooth, Zigbee, or similar other wireless links.

Typically, functions provided by an ECU (**102**) are bound to hard real-time temporal behavior. In the context of the present description, real-time may include any response time that may be measured in milli- or microseconds, and/or is less than 1 second.

The ECU may receive a set of real-time input variables from local sensors (**108**), which are connected via discrete signal lines (**113**), or from networked sensors (**106**), which are connected through a multiplexing bus-system (**112**). The ECU may also share variables with other ECUs (**102**) that are either connected on the same physical multiplexing bus or that it can reach through a gateway (**101**,**103**,**104**).

Then the ECU (**102**) processes the input variables and generates a set of output variables that are either shared with other ECUs (**102**) as described above, or which are output to local actuators (**109**), which are connected via discrete signal lines (**113**), or to networked actuators, which are connected through a multiplexing bus (**112**). ECUs (**102**) typically share information with devices that are connected on the same physical multiplexing system. This method of information sharing is called horizontal information sharing in a hierarchical system. Gateways (**101**,**103**,**104**) link multiple physical multiplexing systems together. In the context of the present description, such information may include data, a signal, and/or anything else capable of being stored and shared.

The highest level in the hierarchical system is the system level. The system level gateway (**101**) may be connected to ECUs on the system level multiplexing bus (**117**), to subsequent gateways (**103**) that also link to subsequent communication buses (**110**), and to external components (**120**) that may contain diagnostics devices (**121**), development tools (**122**), other add-on devices (**123**) or other instances of distributed embedded communication and computing systems (**100**). In addition, the system gateway (**101**) may also be connected to an external gateway (**131**) that may link the system to a remote device (**132**) through wireless or wired wide-area-networks such as the Internet, using standard protocols such as UDP/IP, TCP/IP, RTP, HTTP, SOAP, JAVA, etc. or nonstandard proprietary protocols.

Subsequent to the system level may be several layers of groups and subgroups that are link to the higher levels via gateways (**101**,**103**,**104**,**105**).

During the design-time of the system, not all ECUs may exist. Therefore, the development tool (**122**) may provide a plug-in component or virtual ECU/GW (**115**) that directly links into the wired multiplexing bus or wireless network (**110**) and also allows for separate control functions via a tool-link (**116**).

The block diagram in FIG. **2** depicts the detailed elements within a generic ECU (**200**) that is one embodiment of ECU (**102**). The ECU (**200**) typically contains a micro-processor (**201**), volatile memory (**204**) such as RAM, S-RAM or similar, non-volatile memory (**203**) such as EEPROM, FLASH, etc., a real time clock for internal timing of processes (**205**), a watchdog (**206**) to maintain the health of the system, one or more communication bus controllers (**207**) with associated drivers (**208**), digital I/O (**209**) with line drivers (**210**), and analog I/O (**211**) with associated analog signal conditioning (**212**).

In an alternate embodiment, the ECU (**200**) may also contain a wireless communication controller (**311**) and a RF-Front-end (**312**) as outlined in FIG. **3**. The software (**202**) can either be stored in local non-volatile memory (**203**) or partially downloaded via the communication link (**207**,**208**) and stored in the volatile memory. The software is then executed in the microprocessor (**201**).

The block diagram FIG. **3** depicts the detailed elements within a generic gateway (**300**) that is one embodiment of Gateway (**101**,**103**,**104**,**105**) in FIG. **1**.

FIG. **4** outlines one embodiment of the software architecture in an embedded system. The hardware abstraction layer (**405**) allows the system developer to adapt a standard operating system to a specific hardware as used in an ECU (**200**) or gateway (**300**). The hardware abstraction layer (**405**) adapts the real-time operating system (**403**) and the device drivers (**404**) to a specific hardware implementation.

One embodiment includes the middleware (**402**) that has direct access to the real-time operating system (**403**), the device drivers (**404**) and the hardware abstraction layer (**405**). The middleware isolates the application from input/output functions and allows multiple applications to share common variables locally. In addition, the middleware lets applications share variables with remote applications/processes. In the context of the present description, a process may refer to any hardware and/or software operation, etc.

In one embodiment, the middleware can directly interface with the input/output mechanisms of the hardware without utilizing an operating system (**403**) or hardware abstraction layer (**405**).

Another embodiment of the middleware utilizes a pre-emptive multitasking operating system with explicit control of resources. In an alternate embodiment, the middleware can be built with a static multitasking scheme with implicit resource management or be part of a single task system.

Referring now to FIG. **5**, the middleware (**402**) contains the bulletin board manager (**501**), a local signal communication interface (**503**), a remote message communication interface (**504**), and an application programming interface (**502**). The application interface (**502**) provides methods and data interfaces to a plurality of applications. In one embodiment, the application interface is an object library that can be linked to an application at design time.

The bulletin board manager (**501**) contains an upgrade and configuration manager (**507**), an event manager (**505**), a data access manager (**508**), and a data integrity watchdog (**506**). The upgrade and configuration manager (**507**) is necessary to configure the data structure of the bulletin board and to make executable code available to individual processing nodes. In the context of the present description, the bulletin board may refer to any database that enables users to send and/or read electronic messages, files, and/or other data that are of general interest and/or addressed to no particular person/process.

The access manager provides access control mechanisms for the code update and configuration mode. It also may control access rights for individual applications at execution time in the run mode.

The event manager (**505**) captures input-output events as variables and generates new events, flags, or signals based on operations on state variables in the bulletin board. Such operations may include test of maximum values, the occurrence of logically combined events, the result of an integrity check, or events and signals that are created based on any other logical or arithmetic computation on the state variables that are stored in the bulletin board. The actual processing of data and manipulation of data may be done in the application that uses the middleware (**402**). The data integrity watchdog analyses the stored state variables for its integrity and generates events or flags if any problem occurs.

The local signal communication interface (**503**) interfaces with the local discrete input/output hardware to update the bulletin board with new variables and to update the input/output interfaces with the state variables from the bulletin board. It also converts state variables to input/output signals and input/output signals to state variables that can be stored in the bulletin board. The conversion process may contain scaling of signals as well as offset compensation. Typically this processing helps to convert I/O signals that are measured in Volt to a physical entity and vice versa. The communication with the local discrete input output system can be triggered by events or signals can be sampled time-triggered based on a cyclic global or local time base.

The remote message communication interface (**504**) interfaces to serial multiplexing interfaces (buses) that are connected to the specific processing node (ECU or Gateway). It extracts variables from a plurality of messaging protocols and stores them in the database. It also replicates local bulletin-board state variables to the associated processing nodes by composing the appropriate messages for each communication link. The message transfer can be initiated triggered by a bus event, by a local event, or by a time-triggered mechanism that uses a cyclic local or global time base.

FIG. **6** shows the concept of an extended bulletin board or an embedded real-time database (**601**). In this embodiment the ECU (**102**) or the Gateway (**101**) hosts one or multiple bulletin boards with relational links between the variables in the bulletin boards. The relations are defined by data processing functions that the gateway can operate on bulletin boards to obtain new information that can be stored in yet another bulletin board.

The bulletin board (**601**) may contain but is not limited to events (**607**), real-time variables (**608**), diagnostics data (**609**), configuration parameters (**610**), and firmware (**611**) to upgrade individual components of the executable code or the entire software of a processing node. Each type of information may include one or more sections so that individual processes are not blocked if they access separate sections of data.

The memory of the bulletin board is subdivided into areas that nodes on each external network can read from and write into and other areas that an external network may only read from. The data contained in the bulletin board may be stored in volatile or non-volatile memory. Each data entry may consist of one value or an array of values that also may represent a time series.

In one embodiment, each application process (**603**), local signal communication process (**605**), remote message communication process, and the bulletin manager (**602**) can individually access the bulletin board using operating system functions for resource management that may include semaphores, events, signals, call-back routines, flags, etc. in an alternate embodiment of the system the bulletin-board manager controls all interaction with the bulletin-board and all applications have to pass data to the bulletin-board manager. This approach simplifies the interaction with the bulletin board, but adds delay time and jitter to the state variables.

At design time, various hierarchies of memory management can be applied. In practice it is more efficient to allow each sub network and subsystem to place system variable data into local bulletin boards. This is because many system variables are primarily used only within their subsystem or sub network. By placing local information in a shared memory (local bulletin board), it can be used by multiple processes on this processor node. A group bulletin board allows devices on a sub-network to share information with a minimum of network traffic. A system bulletin board allows access to system-wide variables and information.

FIG. **7** illustrates the logical architecture of the interconnection between three heterogeneous network controllers (**702**, **703**, **704**), the associate Operating System interfaces (**705**), the remote message communication process (**706**), the bulletin board (**608**), and the application process (**606**). The connection to each communication controller is fundamentally implemented at the physical interface (the wire, fiber or electromagnetic wireless interface). Each of the higher level layers (data link, network, etc) in the communication interface (**705**) deals with specific features of the individual communication process. In practice these layers are typically represented in a message by "header" bits that contain information about that layer of the network being used to send the message.

Using this model, each communicated message may be processed at each layer to remove (and use) the associated header information for that level. Once all layers are processed the remaining packet data unit (PDU) represents the datum or core information carried by the overall message. In one embodiment, each communication controller has an associated communication interface and an associated remote message conversion mechanism. For instance com-

munication bus controller **2** (**703**) has an associated communication interface **2** (**709**), and an associated remote message conversion **2** (**710**).

This arrangement allows the remote message process (**706**) to directly access information at the data link layer and interface it with the bulletin board. A network layer is not necessary. The remote message communication process (**706**) has a multi-network access interface (essentially a processing capability that can interpret and apply the header information for a variety of networks) and the bulletin board read/write memory access. Now, the individual processing nodes do not need to know about the existence of multiple networks. Each variable can be accessed from all connected physical networks in their proprietary format. Thus the normalization of the information has only to be handled at the gateway through replication of stored data to multiple attached networks.

Continuing with FIG. **7**, the communication procedure is described. In the given example, an external event (**701**) on communication controller **2** (**703**) triggers the operating system to notify the remote message communication process (**706**) that data is available. The notification may be a flag, a call-back routine, an event, or any other operating signal. The associated remote message conversion method **2** (**710**) extracts the data (e.g. real time variables) from the message PDU and stores the data in the bulletin board (**608**). It may also store the associated event as variable in the bulletin board and signal the bulletin-board event manager that new data is available.

The bulletin event manager then notifies the application process (**606**) with the appropriate mechanism. In addition, the event manager may trigger the sampling of local signals using the local signal communication process (**605**) described in FIG. **6**. Finally the bulletin event manager may trigger the bulletin board manager (**707**) to perform integrity checks or generate additional events based on the change of the state variables.

One embodiment provides a new mechanism for creating an information interconnection between two or more heterogeneous communication networks. In the context of the present description, heterogeneous networks may refer to any different communication networks with at least one aspect that is different.

The approach uses a common, or shared storage system that is connected to all of the system networks through network interfaces. A critically important feature of the bulletin board approach is that the complexity of the bulletin board grows linearly with the number of networks (as opposed to as N(N−1) for the gateway approach), and in one-to-many situations the number of message transformations is half that of the standard networking approach.

In an alternate embodiment of the remote message communication process (**706**) any remote process can access data via a single network interface. This approach requires a network layer in each processing node and therefore adds overhead to communications. To communicate between two heterogeneous networks, this process may then be repeated in reverse by adding back the header information for the various layers of the second network, and eventually putting the message onto the second network's physical link. The remote message communication manager (**706**) then can be simplified to only one message assembly and disassembly mechanism.

FIGS. **8**-**17** illustrate the method of operation of one embodiment of the present system, and also refer to aspects and elements one embodiment shown in FIGS. **1** through **7**.

FIG. **8** details the remote messaging process (**706**) described in FIG. **7**. Referring now to FIG. **8**, the core process of storing data from remote processes that are communicated through multiplexed communication links, into the bulletin board is described. An external notification or task activation starts the process. Then a message (**802**) is received from the operating system layer.

In an alternate embodiment, the message is directly copied form the input register of the communication controller. Then the process extracts variables from the message. Additional signal adaptation may be necessary. The sub-process **804** stores the variables in the bulletin board. If the process only updates one section of the bulletin board it waits for the next message notification (**806**). If variables in multiple sections need to be updated, the process repeats (**804**).

FIG. **9** shows the data update from local input/output peripherals. The process starts with an internal or external notification or task activation. Typically this process is repeated cyclic triggered by an internal or external real-time clock. When the process is activated, it samples or polls the local input ports that may include analog and digital signals (**902**). Then it converts these signals to real-time variables by using the conversion parameters stored in the bulletin board. The signal conditioning parameters van either be defined at design time or adaptively updated by the application process. Then the process stored the new state variables in the bulletin board using the sub-process (**804**) described above.

FIG. **10** describes the bulletin board store procedure (**804**) in more detail. Before new data can be stored in the bulletin board, the procedure has to request the access right to the common resource, a section of the non-volatile or volatile memory, from the operating system (**1001**). This is called explicit resource management.

If the resource is available, the process gets the resource. If the resource is not available, it may try it again after a waiting period (**1011**) until the resource is available. After a certain time has elapsed (**1009**) beyond a configurable threshold, the temporal behavior of the state variable can't be captured any longer and the middle-ware may send an error notification to the associated process.

After reserving the resource (**1003**), the bulletin board store mechanism (**804**) timestamps the state variable for future temporal reference (**1004**). Then, the bulletin board store procedure (**804**) copies the variables or parameters from its private memory (**1006**) to the shared bulletin-board memory (**601**). Then it releases the bulletin board resource.

In an alternate embodiment, the bulletin board store procedure (**804**) has exclusive access to the bulletin board (**601**) and does not need operations **1002**, **1003**, **1007**, **1009**, **1010**, and **1011** because the resource access is realized through implicit resource management. This can be achieved with either static task scheduling or by allowing only the bulletin board store procedure (**804**) to access the bulletin board (**601**).

FIG. **11** describes the bulletin board retrieve procedure (**1101**) in more detail. Before data can be retrieved from the bulletin board, the procedure has to request the access right to the common resource, a section of the non-volatile or volatile memory, from the operating system (**1102**).

If the resource is available, the process gets the resource. If the resource is not available, it may try it again after a waiting period (**1108**) until the resource is available. After a certain time has elapsed (**1109**) beyond a configurable threshold, the temporal behavior of the state variable can't be captured any longer and the middle-ware may send an error notification to the associated process (**1110**).

After reserving the resource (**1104**), the bulletin board retrieve mechanism (**1101**) copies the variables or parameters from the shared bulletin-board memory (**601**) to its private memory (**1006**). Then, it releases the bulletin board resource.

In an alternate embodiment the bulletin board retrieve procedure (**1101**) has exclusive access to the bulletin board (**601**) and does not need operations **1103**, **1104**, **1106**, **1108**, **1109**, and **1110**. Because the resource access is realized through implicit resource management, this can be achieved with either static task scheduling or by allowing only the bulletin board retrieve procedure (**1101**) to access the bulletin board (**601**).

Referring to FIG. **12**, the application process (**1200**) utilizes the bulletin board retrieve mechanism (**1101**) to access all parameters, events, and real-time variables from the bulletin board. Thus the application process is decoupled from the temporal behavior of the input/output variables and can be triggered by a plurality of events (**1201**).

The application process may retrieve one or multiple sets of variables stored in a plurality of memory sections. Then the application process processes the variables (**1203**) with its method. Because the method is not tied to the location of the input/output variables, the application process can be moved or replicated to a plurality of processing nodes (ECUs or Gateways). After processing the input variables and generating a set of output variables, the application process uses the bulletin board store method (**801**) to update one or a plurality of memory sections in the bulletin board. If the application process is a cyclic procedure, it waits until the next activation occurs (**1205**).

Continuing with FIG. **13**, the update local I/O from bulletin board process (**1300**) utilizes the bulletin board retrieve mechanism (**1101**) to access real-time variables from the bulletin board and convert them to output signals (**1302**) that can be written to the output port (**1303**). The I/O update process may retrieve one or multiple sets of variables stored in a plurality of memory sections. If the I/O update process is a cyclic procedure, it waits until the next activation occurs (**1305**).

FIG. **14** describes the data replication process (**1400**). This process can be triggered by a plurality of notification mechanisms, such as events, alarm signals, internal and external timers, and flags set in the bulletin board. It then selects a subset of variables to be replicated and a communication port (**1402**). Next it retrieves the variables from the bulletin board with mechanism (**1401**) and assembles the messages for the specific communication link (**1403**). The message may include an address or identification number for all bulletin boards and associated processing nodes (ECUs and Gateways).

Finally, it writes the messages to the communication port (**1404**). In an alternate embodiment, it handles the messages to the associated interface procedure of the operating system. Then it repeats the procedure, until all variables are updated on all communication ports. If the data replication process is a cyclic procedure, it waits until the next activation occurs (**1405**).

Referring now to FIG. **15**, the store message from remote processing node (gateway or ECU) process (**1500**) describes how replicated data is stored in the bulletin board. This process can be triggered by a plurality of notification mechanisms, such as internal or remote events, alarm signals, internal and external timers, and flags set in the bulletin board. The communication bus may also issue these notifications. The process (**1500**) then reads a message from the communication port (**1502**), selects a subset of variables to

be replicated (**1503**), and stores the variables in the bulletin board with procedure (**801**). In an alternate embodiment, this procedure may also be used to store a packet data unit (PDU) in the bulletin board for later replication on the same or a different communication link.

This store and forward networking mechanism can be implemented without the need for complex networking protocols and is therefore well suited for limited processing power and memory environments. It also works in soft-real time environments when no strict temporal behavior is required. The data store operation (**801**) may be repeated for a plurality of bulletin board sections. If the data replication process is a cyclic procedure, it waits until the next activation occurs (**1505**).

Continuing now with FIG. **16**, the store and forward updating mechanism (**1600**) replicates messages from remote processing nodes to other processing nodes from stored packet data units in the bulletin board. This process can be triggered by a plurality of notification mechanisms (**1601**), such as internal or remote events, alarm signals, internal and external timers, and flags set in the bulletin board. The communication bus may also issue these notifications.

The process (**1600**) then selects a message to be forwarded (**1602**) and the appropriate communication link and retrieves the PDU with the bulletin board retrieve mechanism (**1101**). It then adds the appropriate messages header for the communication link (**1603**) and may add routing information (**1604**). Finally the update process (**1600**) writes the messages to the communication port (**1605**). If the updating process is a cyclic procedure, it waits until the next activation occurs (**1607**).

FIG. **17** describes the various modes that the distributed communications and computing system (**100**) can be operated in. In one embodiment, the system operates in various distinct modes in order to preserve the integrity of the system and still allow for changing the architecture and behavior of the network or the roles of the individual nodes. When the distributed computing and communication system wakes up from the sleep mode (**1701**), it can enter a configuration and upgrade mode (**1702**), an emergency or debug mode (**1704**), or the normal real-time run mode (**1703**). The root node or system gateway in a distributed communication and computing system defines the mode based on the existence of external events, such as an external control command, internal events, a system failure, or failed integrity check.

Referring now to FIG. **1**, the external commands may be generated from a development tool (**122**) or a remote device (**132**) that is connected via a remote gateway (**131**). In an alternate embodiment, each ECU (**102**) or virtual ECU (**115**) can trigger the system to enter a different operating mode.

Continuing with FIG. **17**, in the configuration mode (**1702**), the system software and the information-sharing configuration can be updated via a secure communication link with encrypted commands Each processing node (ECU or gateway) may have security mechanisms such as a certificate that allows it to identify and authorize another entity (remote gateway, remote ECU, or development tool) to make changes to its bulletin board parameters.

The remote entity may also download a new firmware to the bulletin board. The ECU or gateway can store this new firmware in its non-volatile memory while it backs up the original image on the bulletin board for the case that the new software is not functional. In the update mode, the distributed system can also reconfigure the communication and computing infrastructure based on a new set of parameters that need to be stored in the individual bulletin boards.

In the normal run mode (**1703**), the system operates in the real-time information sharing mode and network configuration and certain parameters can't be changed. That protection allows defining deterministic temporal behavior on all communication links. But any processing node may enter a debug/emergency mode (**1704**) if a failure or other qualifying event occurs.

In the emergency mode, a processor executes an alternate procedure that maintains the temporal behavior on the communication links but may reduce or increase the amount of information shared with other processors. It also lets other processing nodes check on the integrity of sensors and actuators. In the maintenance and upgrade mode, an external system can upgrade executable code images and the bulletin-board configuration via secure communication links.

A system and method are thus provided for sharing information within a distributed embedded communications and computing system and with components outside the embedded system. The information sharing mechanism relies on a bulletin board that may include a small database operating under hard real-time conditions with minimal delays, communication latency, and jitter. The embedded database or bulletin board isolates a real-time application in a Electronic Control Unit (ECU) from various other real time applications and from input output signals in the same module (local information sharing), from event-triggered communications with applications in other modules, and from time-triggered communications with applications in other modules.

One design criteria of the database is that the temporal behavior of communications does not impact the real-time computing task and provides enough information access performance at peak time demand Typically, distributed embedded systems consist of a static structure that can be analyzed at design time. In addition to the real-time operation, the proposed method for information sharing also provides access to the parameters of the embedded system and allows for software upgrades of certain modules.

The present embodiment addresses the shortcomings of traditional computer networks with following enhancements:

1) The concept of multi-mode storage that links two or more communication networks via a bulletin board. The bulletin board is a multi-mode storage that can be thought of an extension to shared memory that can be accessed by local and remote processes at attached networks. There may be multiple hierarchical layers of bulletin boards depending on the topology of the communication system. The bulletin board increases the network efficiency by reducing the number of transactions needed to access remote variables.

2) The concept of a direct-access bulletin board that does not require a network layer translation of messages on each node of the network. Even though this approach restricts the reach of each node to only adjacent nodes and the next gateway, this still allows cross-network variable sharing though vertical real-time replication of data.

3) The concept of hierarchical bulletin board management that allows restriction of information access to certain levels in a network, but still allows the replication of information to other nodes in the network. This paradigm follows the path of reducing the information amount from the leaves of the network to central control and diagnosis hubs.

4) The concept that a gateway can host an assembly of bulletin boards or embedded database that allows operations on bulletin boards to generate events for associated processes. This extension allows definition of a set of data processing operations that would be done once in a network and would be instantly available for connected nodes. Examples for operations are sensor data state observers, diagnostics, integrity checks, fail-safe mechanisms, etc.

5) The concept that an embedded communication and computing network can run in multiple modes in order to provide for a guaranteed deterministic behavior of the system. This property can be achieved by only allowing change to the configuration and/or the functions (SW code) in a secured configuration and upgrade mode. If the network is booted in the normal operating mode, all processors execute the existing code and only allow data sharing through the bulletin boards. The emergency or debug mode lets the network run in a fail-safe reduced operation mode or in a diagnostic mode that allows inspection of the system, while it is running. For each operating mode, the gateway can store a processing image on the bulletin board. The advantage of this procedure is that only the communication hubs need to deal with secure data transfer and encryption while the peripheral nodes in the network can be relative simple in design.

6) The concept of designing the topology of a distributed computing and communication system independent of the definition of the individual functions that the network performs. Each processing task is only associated with a bulletin board, but isolated from I/O processing.

Of course, these are all optional embodiments/enhancements.

While various embodiments have been described above, it should be understood that they have been presented by the way of example only, and not limitation. Thus, the breadth and scope of a preferred embodiment should be not limited by any of the above described exemplary embodiments, but should be defined only in accordance with the following claims and their equivalents.

What is claimed is:

1. An apparatus, comprising:

an automotive electronic control unit comprising a hardware and instructions for:

identifying information associated with a message received utilizing a Controller Area Network protocol associated with a Controller Area Network;

issuing a storage resource request in connection with a storage resource of the automotive electronic control unit and determining whether the storage resource is available for storing the information;

determining whether a threshold has been reached in association with the storage resource request;

in the event the storage resource is not available and the threshold associated with the storage resource request has not been reached, issuing another storage resource request in connection with the storage resource;

in the event the storage resource is not available and the threshold associated with the storage resource request has been reached, sending a notification;

in the event the storage resource is available, storing the information utilizing the storage resource; and

sharing the information in less than one second utilizing a Flexray network protocol associated with a Flexray network, wherein the automotive electronic control unit remains in hardwired communication with the Controller Area Network and the Flexray network and includes:

a first interface for interfacing with the Controller Area Network, the first interface including a first interface-related data link layer component for using

Controller Area Network-related data link layer header bits and a first interface-related network layer component for using Controller Area Network-related network layer header bits; and

a second interface for interfacing with the Flexray network, the second interface including a second interface-related data link layer component for using Flexray network-related data link layer header bits and a second interface-related network layer component for using Flexray network-related network layer header bits.

2. The apparatus as recited in claim **1**, wherein the information is also shared in less than the one second utilizing, in addition to the Flexray network protocol, a Local Interconnect Network protocol associated with a Local Interconnect Network, and the automotive electronic control unit remains in hardwired communication with the Local Interconnect Network via a third interface for interfacing with the Local Interconnect Network, the third interface including a third interface-related data link layer component for using Local Interconnect Network-related data link layer header bits.

3. The apparatus as recited in claim **1**, wherein the identifying, the issuing of the another storage resource request, and the sending of the notification collectively occur in less than the one second.

4. The apparatus as recited in claim **1**, wherein the identifying, the issuing of the storage resource request, the storing the information, and the sharing collectively occur in less than the one second.

5. The apparatus as recited in claim **4**, wherein the second interface-related network layer component uses the Flexray network-related network layer header bits by adding the Flexray network-related network layer header bits to a data unit including the information, and then the second interface-related data link layer component uses the Flexray network-related data link layer header bits by adding the Flexray network-related data link layer header bits to the data unit, before communicating the data unit on a physical link of the Flexray network.

6. The apparatus as recited in claim **5**, wherein the first interface-related data link layer component uses the Controller Area Network-related data link layer header bits by removing the Controller Area Network-related data link layer header bits from another data unit, and the first interface-related network layer component uses the Controller Area Network-related network layer header bits by removing the Controller Area Network-related network layer header bits from the another data unit, where the information is extracted from the another data unit before the sharing.

7. The apparatus as recited in claim **1**, wherein a duration between the information being sent to the automotive electronic control unit and the sharing being completed is less than a second.

8. The apparatus as recited in claim **1**, wherein a duration between the information being received and the sharing is less than a millisecond.

9. The apparatus as recited in claim **1**, wherein the first interface-related data link layer component and the second interface-related data link layer component are drivers.

10. The apparatus as recited in claim **1**, wherein the first interface-related data link layer component and the second interface-related data link layer component are part of a hardware abstraction layer.

11. The apparatus as recited in claim **1**, wherein the hardware and the instructions are for releasing the storage resource after the storing.

12. An apparatus, comprising:

an automotive electronic control unit comprising a hardware and instructions for:

receiving information associated with a message received utilizing a Controller Area Network protocol associated with a Controller Area Network;

determining whether a storage resource is available;

if the storage resource is not available, ascertaining whether a threshold has been reached and re-trying an access in connection with the storage resource if the threshold has not been reached;

if the threshold has been reached, sending an error notification;

if the storage resource is available, storing the information utilizing the storage resource; and

sharing the information utilizing a Flexray network protocol associated with a Flexray network;

wherein the receiving, the determining, the storing, and the sharing all occur in less than one second; the automotive electronic control unit remains in hardwired communication with the Controller Area Network and the Flexray network; and the automotive electronic control unit includes:

a first interface for interfacing with the Controller Area Network, the first interface including a first interface-related data link layer component for using Controller Area Network-related data link layer header bits and a first interface-related network layer component for using Controller Area Network-related network layer header bits; and

a second interface for interfacing with the Flexray network, the second interface including a second interface-related data link layer component for using Flexray network-related data link layer header bits and a second interface-related network layer component for using Flexray network-related network layer header bits;

wherein the second interface-related network layer component uses the Flexray network-related network layer header bits by adding the Flexray network-related network layer header bits to a data unit including the information, and then the second interface-related data link layer component uses the Flexray network-related data link layer header bits by adding the Flexray network-related data link layer header bits to the data unit, before communicating the data unit on a physical link of the Flexray network;

wherein the first interface-related data link layer component uses the Controller Area Network-related data link layer header bits by removing the Controller Area Network-related data link layer header bits from another data unit, and the first interface-related network layer component uses the Controller Area Network-related network layer header bits by removing the Controller Area Network-related network layer header bits from the another data unit, where the information is extracted from the another data unit before the sharing.

13. An apparatus, comprising:

an automotive electronic control unit comprising a hardware and instructions for:

identifying information associated with a message received utilizing a Flexray network protocol associated with a Flexray network;

issuing a storage resource request in connection with a storage resource of the automotive electronic control unit and determining whether the storage resource is available for storing the information;

determining whether a threshold has been reached in association with the storage resource request;

in the event the storage resource is not available and the threshold associated with the storage resource request has not been reached, issuing another storage resource request in connection with the storage resource;

in the event the storage resource is not available and the threshold associated with the storage resource request has been reached, sending a notification;

in the event the storage resource is available, storing the information utilizing the storage resource; and

sharing the information in less than one second utilizing a Controller Area Network protocol associated with a Controller Area Network, the automotive electronic control unit remaining in hardwired communication with the Flexray network and the Controller Area Network, and including:

a first interface for interfacing with the Flexray network, the first interface including a first interface-related data link layer component for using Flexray network-related data link layer header bits and a first interface-related network layer component for using Flexray network-related network layer header bits; and

a second interface for interfacing with the Controller Area Network, the second interface including a second interface-related data link layer component for using Controller Area Network-related data link layer header bits and a second interface-related network layer component for using Controller Area Network-related network layer header bits.

14. The apparatus as recited in claim **13**, wherein the information is also shared in less than the one second utilizing, in addition to the Controller Area Network protocol, a Local Interconnect Network protocol associated with a Local Interconnect Network, and the automotive electronic control unit remains in hardwired communication with the Local Interconnect Network via a third interface for interfacing with the Local Interconnect Network, the third interface including a third interface-related data link layer component for using Local Interconnect Network-related data link layer header bits.

15. The apparatus as recited in claim **13**, wherein the identifying, the issuing of the another storage resource request, and the sending of the notification collectively occur in less than the one second.

16. The apparatus as recited in claim **13**, wherein the identifying, the issuing of the storage resource request, the storing the information, and the sharing collectively occur in less than the one second.

17. The apparatus as recited in claim **16**, wherein the second interface-related network layer component uses the Controller Area Network-related network layer header bits by adding the Controller Area Network-related network layer header bits to a data unit including the information, and then the second interface-related data link layer component uses the Controller Area Network-related data link layer header bits by adding the Controller Area Network-related data link layer header bits to the data unit, before communicating the data unit on a physical link of the Controller Area Network.

18. The apparatus as recited in claim **17**, wherein the first interface-related data link layer component uses the Flexray network-related data link layer header bits by removing the Flexray network-related data link layer header bits from another data unit, and the first interface-related network layer component uses the Flexray network-related network layer header bits by removing the Flexray network-related network layer header bits from the another data unit, where the information is extracted from the another data unit before the sharing.

19. The apparatus as recited in claim **13**, wherein a duration between the information being sent to the automotive electronic control unit and the sharing being completed is less than a second.

20. The apparatus as recited in claim **13**, wherein a duration between the information being received and the sharing is less than a millisecond.

21. The apparatus as recited in claim **13**, wherein the storage resource request and the issuing of the another storage resource request each includes a request for an access.

22. The apparatus as recited in claim **13**, wherein the notification is sent by middleware.

23. The apparatus as recited in claim **13**, wherein the hardware and the instructions are for releasing the storage resource after the storing.

24. An apparatus, comprising:

an automotive electronic control unit comprising a hardware and instructions for:

receiving information associated with a message received utilizing a Flexray network protocol associated with a Flexray network;

determining whether a storage resource is available;

if the storage resource is not available, ascertaining whether a threshold has been reached and re-trying an access in connection with the storage resource if the threshold has not been reached;

if the threshold has been reached, sending a notification;

if the storage resource is available, storing the information utilizing the storage resource;

sharing the information utilizing a Controller Area Network protocol associated with a Controller Area Network;

wherein the receiving, the determining, the storing, and the sharing collectively occur in less than one second; the automotive electronic control unit remains in hardwired communication with the Flexray network and the Controller Area Network; and the automotive electronic control unit includes:

a first interface for interfacing with the Flexray network, the first interface including a first interface-related data link layer component for using Flexray network-related data link layer header bits and a first interface-related network layer component for using Flexray network-related network layer header bits; and

a second interface for interfacing with the Controller Area Network, the second interface including a second interface-related data link layer component for using Controller Area Network-related data link layer header bits and a second interface-related network layer component for using Controller Area Network-related network layer header bits;

wherein the second interface-related network layer component uses the Controller Area Network-related network layer header bits by adding the Controller Area Network-related network layer header bits to a data unit including the information, and the second interface-related data link layer component uses the Controller Area Network-related data link layer header bits by adding the Controller Area Network-related data link layer header bits to the data unit, before communicating the data unit on a physical link of the Controller Area Network;

wherein the first interface-related data link layer component uses the Flexray network-related data link layer header bits by removing the Flexray network-related data link layer header bits from another data unit, and the first interface-related network layer component uses the Flexray network-related network layer header bits by removing the Flexray network-related network layer header bits from the another data unit, where the information is extracted from the another data unit before the sharing.

25. The apparatus as recited in claim **24**, wherein the threshold is time-related.

26. The apparatus as recited in claim **24**, wherein the receiving, the determining, the ascertaining, the storing, and the sharing collectively occur in less than one millisecond.

27. The apparatus as recited in claim **24**, wherein the information is sent from a sensor origin on the Flexray Network and reaches an actuator destination on the Controller Area Network in less than one second.

28. The apparatus as recited in claim **24**, wherein the information is sent from an origin on the Flexray Network, is shared, and reaches a destination on the Controller Area Network in less than one millisecond.

29. The apparatus as recited in claim **24**, wherein the first interface-related data link layer component and the second interface-related data link layer component are part of a hardware abstraction layer.

30. The apparatus as recited in claim **24**, wherein the first interface-related data link layer component and the second interface-related data link layer component are drivers.

31. The apparatus as recited in claim **24**, wherein the information is also shared in less than the one second utilizing, in addition to the Controller Area Network protocol, a Local Interconnect Network protocol associated with a Local Interconnect Network, and the automotive electronic control unit remains in hardwired communication with the Local Interconnect Network via a third interface for interfacing with the Local Interconnect Network, the third interface including a third interface-related data link layer component for using Local Interconnect Network-related data link layer header bits.

* * * * *