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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
19 May 2022 (19.05.2022)

WIPO | PCT

(10) International Publication Number
WO 2022/101405 A3

(51) International Patent Classification:
H04L 9/40 (2022.01) *G06F 21/57* (2013.01)
H04L 1/18 (2006.01) *G07C 9/00* (2020.01)

(21) International Application Number:
PCT/EP2021/081512

(22) International Filing Date:
12 November 2021 (12.11.2021)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
63/113,423 13 November 2020 (13.11.2020) US

(71) Applicant: **ASSA ABLOY AB** [SE/SE]; P.O. Box 70340, 107 23 Stockholm (SE).

(72) Inventors: **SACHDEVA, Kapil**; 5009 Sendero Springs Drive, Round Rock, Texas 78681 (US). **PREVOST, Sylvain, Jacques**; 9808 Faith and Trust Cv, Austin, Texas 78717 (US). **MUKHA, Matvey**; Schmiedgasse 34-6, 8010 Graz (AT).

(74) Agent: **CREATION IP (EUROPE) LIMITED**; Hillington Park Innovation Centre 1 Ainslie Road, Glasgow G52 4RU (GB).

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

(54) Title: SECURE ELEMENT ARRAYS IN INTERNET-OF-THINGS SYSTEMS

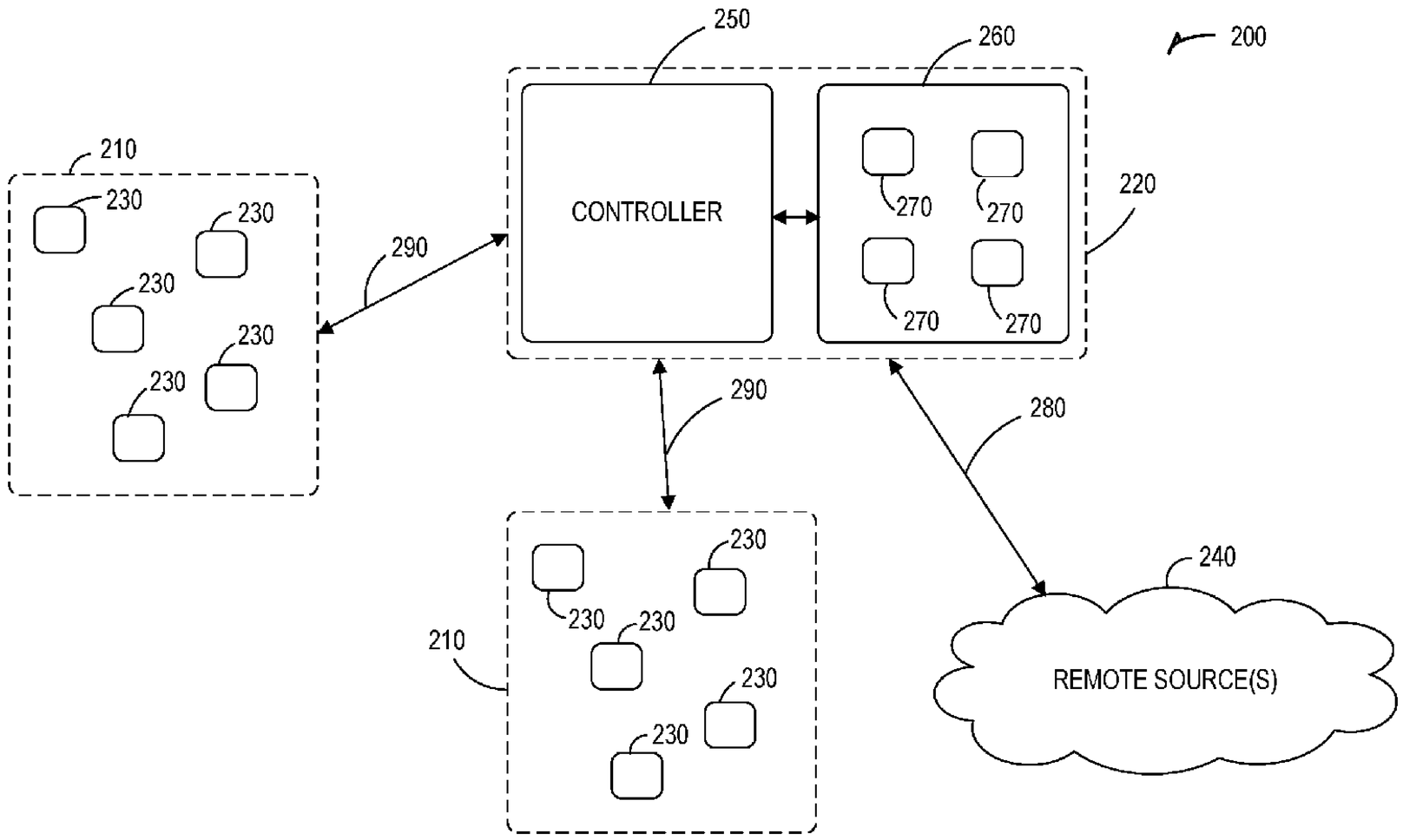

FIG. 2

(57) Abstract: Systems and methods for providing secure execution of functions for edge devices include a plurality of edge devices, a controller, and an array of secure elements. The edge devices are each configured to obtain data for an application of the system. The controller is connected to communicate with the edge devices to receive the data from each of the edge devices. The array of secure elements is connected to the controller, and each secure element executes functions using the data received from the edge devices. The controller associates an identified secure element of the array of secure elements with a respective edge device to execute the functions for data received from the respective edge device, and the controller is connected to communicate a result of the executed functions to the respective edge device.

[Continued on next page]

WO 2022/101405 A3

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

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

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

(88) Date of publication of the international search report:
21 July 2022 (21.07.2022)

SECURE ELEMENT ARRAYS IN INTERNET-OF-THINGS SYSTEMS

PRIORITY APPLICATION

[0001] This application claims priority to U. S. Provisional Patent Application Serial Number 63/113,423, filed November 13, 2020, the disclosure of which is incorporated herein in its entirety/entireties by reference.

TECHNICAL FIELD

[0002] This document pertains generally, but not by way of limitation, to internet of things (IoT) systems, and particularly but not by way of limitation to transparent architecture for IoT systems.

BACKGROUND

[0003] Internet-of-things (IoT) systems often include edge devices that include various sensors or other methods of collecting and communicating data. Some of these edge devices may not have direct network connections or may otherwise be resource constrained. Additionally, there is often a requirement that these edge devices store key material and perform secure processing. However, edge devices in IoT systems often lack processing resources and security capabilities, and the physical locations of the edge devices may also make it inadvisable for the edge device to store key material. Also, these edge devices may implement software that requires regular updates. Due to the limited network connections for some of these edge devices, this may require a technician to visit each individual edge device each time a software update is needed. This can be time and resource consuming in systems with many edge devices.

SUMMARY OF INVENTION

[0004] The present invention provides a system and method for providing secure execution of system functions for edge devices, a non-transitory computer readable medium and a physical access control system as defined in the claims.

BRIEF DESCRIPTION OF THE DRAWINGS

[0005] In the drawings, which are not necessarily drawn to scale, like numerals may describe similar components in different views. Like numerals having different letter suffixes may represent different instances of similar components. Some embodiments are illustrated by way of example, and not limitation, in the figures of the accompanying drawings in which:

[0006] FIG. 1 is a diagram illustrating an example physical access control system.

[0007] FIG. 2 is a block diagram illustrating an example internet-of-things (IoT) architecture that includes an array of secure elements.

[0008] FIG. 3 is a block diagram illustrating an example secure IoT gateway.

[0009] FIG. 4A illustrates an example workflow for authenticating an array of secure elements by a controller.

[0010] FIG. 4B illustrates an example workflow for authenticating a policy and/or controller by a sister board of secure elements.

[0011] FIGS. 5A-5C are diagrams illustrating example transmission frames for a full-duplex protocol for use in a half-duplex system.

[0012] FIG. 6 is a flowchart illustrating an example method of transmitting messaging on a half-duplex communication line using a full-duplex protocol.

[0013] FIG. 7 is a block diagram illustrating an example of a machine upon which one or more embodiments may be implemented.

DETAILED DESCRIPTION

[0014] Systems and methods are disclosed herein for implementing a transparent architecture for internet-of-things (IoT) systems using arrays of secure elements. An example IoT architecture includes a gateway that is equipped with a built-in or connectable array of secure elements. Secure elements include hardware and/or software for performing cryptographic functions or processes— e.g., encryption, decryption, signature generation, signature verification, and/or key generation. Secure elements are contained within an explicitly defined perimeter that establishes the physical bounds of the cryptographic module and that contains any processors and/or other hardware components that store and protect any software and firmware components of the cryptographic module. Secure elements could take the form of (or include) a secure crypto-processor, a smart card, a secure digital (SD) card, a micro SD card, a SIM card, and/or any other cryptographic module.

[0015] The secure element (SE) is a tamper-resistant platform capable of securely hosting applications and their confidential and cryptographic data in accordance with the rules and security requirements set forth by a set of well-identified trusted authorities. The SE can be considered to be a chip that offers a dynamic environment to store data securely, process data securely and perform communication with external entities securely.

[0016] Physical Access Control Systems (PACS) include readers (edge devices) and controllers (intermediary servers/devices). In conventional PACS systems, the readers are smart devices hosting and running software to securely communicate with cards or mobile phones that come within communication range of the reader. Successful PACS systems focus on both user experience and security. Hence, the reader devices must have enough processing power to deal with latency issues as well as both hardware and software-based security primitives. The reader devices have support to authenticate and read data from a wide range of card devices that have different protocols (at both transport and application level) and have different data modalities. This results in a lot of software implemented by the reader devices that must be updated to support new card modalities, bug fixes, and the like. Because reader devices often do not have network connections, the reader device must be physically visited by a technician to perform these updates. Further, this results in secure elements physically located on the reader that store secure software and key material. Some readers are physically located on the external portions of a building, for example, which can raise security concerns for companies, certification entities, or other users of a PACS system that want or need all secure storage of key materially to be physically located within a secure perimeter, such as the perimeter of a building.

[0017] To remedy the above situations, an array of secure elements remote from the reader may be used to execute the security and application software for the reader devices. The array of secure elements may be integrated with, or attached to, the PACS controller and configured to handle multiple parallel requests and connections to the edge devices (readers). The PACS controller may have a network connection to one or more remote devices which may store and provide software or firmware updates for the secure elements. This way, the secure elements can be updated without the need for a technician to visit each individual edge device (reader). The PACS controller may also be physically located within a building or other secure perimeter, eliminating security concerns of having secure elements located on some readers.

[0018] In one example, the array of secure elements may be positioned on a sister board

that can be connected to a host device using one or more busses having bus protocols such as serial peripheral interface (SPI), inter-integrated circuit (I2C), universal serial bus (USB), and the like. Secure elements may be used for several functions. For example, the secure element may act as a cryptographic processor, providing security algorithms and secure storage of sensitive key material. In other examples, the secure elements may be used as an application platform executing application specific software along with the security algorithms and storage of key material. For example, in a PACS system, the secure elements may be used to perform authentication for users that present a credential at a PACS reader. This way, the software functions for performing user authentication are not needed on the reader device itself.

[0019] FIG. 1 depicts an example scenario 100 in which a PACS could be used. As shown in FIG. 1, a wall 102 has disposed therein a door 104. In an example situation, a secured area lies behind the door 104, which has a lockable handle 106 that grants access to the secured area when in an unlocked state and prevents access to the secured area when in a locked state.

[0020] A reader device 108 is positioned proximate to the handle 106 of the door 104. In an example, the handle 106 is locked in the default state. The reader system 108 is operable to selectively place the handle 106 in an unlocked state responsive to being presented with an authorized credential contained in a credential device 112, which can communicate with the reader device 108 via a wireless interface 110. In various examples, the credential device 112 could be a keycard, a fob, a mobile device (e.g., a smart phone), or any other suitable credential device having the communication capabilities and credentials.

[0021] In conventional systems, the application software to authenticate the user may be implemented on the reader device 108 itself. For example, the user presents the credential device 112 which communicates with the reader device 108 to provide a credential to the reader device 108. The reader device 108 then uses the received credential to securely authenticate the user and unlock the handle 106. In other conventional examples, some of this function is included on the reader device 108 and some is included on a remotely located PACS controller. For example, the reader device may perform a secure transaction with the credential device 112 and then transmit data to the PACS controller to authenticate the user. The controller may then communicate to unlock the handle 106.

[0022] Because the reader device 108 performs secure transactions and/or authentication, software or firmware updates may be required for the reader device 108. In some examples, the reader device 108 does not include a network connection and is only connected to the

PACS controller through a single wired connection such as an RS-485 or other connection. This requires a technician to physically travel to the reader device 108 to update software or firmware for each reader device 108 in a PACS system. It is desirable to move some of this functionality away from the reader to provide ease of maintenance while also providing the user with the same or better user experience at the reader device 108.

[0023] To accomplish this, an array of secure elements may be implemented at the PACS controller, each configured to perform secure software execution for respective reader devices 108. These PACS controllers often include one or more network connections, allowing remote updating of software executed by the secure elements, removing the need for technicians to physically travel to each reader device 108. It should be understood that the present disclosure is applicable to numerous types of IoT systems in addition to PACS systems. The system 100 illustrated in FIG. 1 is presented purely by way of example and not limitation.

[0024] FIG. 2 is a diagram illustrating an IoT system 200. System 200 includes edge device clusters 210 and a secure IoT gateway (SIG) 220. The edge device clusters 210 may each include one or more edge devices 230, such as the reader device 108 of FIG. 1. The SIG 220 may communicate with a remote source 240 via a local area network or wide area network 280, such as the Internet. The SIG 220 may include a controller 250 and a sister board 260 that includes secure elements 270. In an example, the controller 250 may be a PACS controller. While illustrated as two clusters 210 each of five edge devices 230, and four secure elements 270, any number of clusters 210 having any number of edge devices 230, and any number of secure elements 270 may be implemented for the system 200. While illustrated as a separate controller 250 and sister board 260, in some examples the secure elements 270 may be integrated with the controller 250. The controller 250 may be connected to communicate with the sister board 260 using any bus protocol such as SPI, I2C, USB, and the like.

[0025] The SIG 220 is connected through a network connection 280 to communicate with the remote source(s) 240 and is connected through individual connections 290 to communicate with respective edge devices 230. For example, the connection 280 may be a local or wide area network connection, such as an Internet connection. The individual connections 290 may be wired or wireless connections such as Ethernet, Wi-Fi, USB, RS-485, or the like. While illustrated as a single connection 290 for each cluster 210, there may be a connection 290 for each individual edge device 230. The remote source 240 may be one

or more servers or other computing devices and may store a firmware file or other software update for secure elements 270. In some examples, the SIG 220 communicates with the remote source 240 to obtain the firmware file or other software update and to update the one or more secure elements 270 implemented by the SIG 220.

[0026] In a PACS system, such as the one illustrated in FIG. 1, the reader device 108 may be an edge device 230 connected to communicate with a controller 250 such as a PACS controller. The connection 290 may be a wired full-duplex connection, wired half-duplex connection such as an RS-485 connection, or any other connection. The secure elements 270 may be configured to execute software to perform functions using application specific hardware, for the reader 108. For example, when a user approaches the reader device 108 and presents a credential using a credential device 112, a secure element 270 may be allocated to the transaction to perform security algorithms and secure storage of sensitive key material, as well as user authentication. In some examples, the reader device 108 may only obtain the credential information from the credential device 112, provide the credential information to the controller, and a secure element 270 performs all of the application specific functions for authenticating the user and unlocking the door handle 106. In other examples, some of the application specific functions may be executed by the reader device 108 and some may be executed by a secure element 270.

[0027] When a user approaches the reader device 108, or when the reader device 108 receives a user credential, the controller 250 or other electronic circuit may select and allocate a secure element 270 for use with the reader device 108. This may be any secure element 270 that is currently available for execution of software for the reader device 108.

[0028] In an example, each edge device 230 may dynamically receive reference to a secure element 270 that the respective edge device 230 is assigned to for a respective session. For example, when an edge device 230 needs a secure element 270, the controller 250 may identify an available secure element 270 that is able to provide the necessary functions for the respective edge device 230. The controller 250 may also be implemented as a “dispatcher”, becoming responsible for dispatching messages or data to a respective secure element 270, shielding the secure element array from respective edge devices 230. This enables a high level of modularity in code development and management, as well as protects against the crash or termination of an edge device 230 or other actor within the system.

[0029] Other devices or circuits may be also implemented within the system 200 to monitor the lifecycle of edge devices 230 or secure elements 270 and implement a policy to

either respawn respective devices or keep the devices terminated and inform a system administrator, for example. In other examples, one or more of the edge devices 230, the controller 250, or secure elements 270 may monitor the life cycle of devices in the system, which can be used in cleaning up or resetting respective states of devices within the system.

[0030] In some cases, it may be desirable to implement applications and key material for edge devices that is not accessible by the remote devices 240 or other entities. For example, in a PACS system, an entity may wish to program the controller 250 or secure elements 270 with specific authentication code that is not accessible by any other entities such as through the remote devices 240. To facilitate this, the secure elements 270 may be configured such that the secure elements 270 are programmable in high level languages. In some examples, even though the secure elements 270 are resource constrained devices, a runtime may be implemented that is capable of running a language runtime for the secure elements 270. Thus, an entity can develop an application and install the application on the secure elements 270.

[0031] In the above scenario, it is desirable to limit who can install these applications on the secure elements 270. In an example, an application to be installed in the secure elements 270 must be signed by the entity and then a higher level or other entity doubly signs the application with corresponding keys. If the application is doubly signed, one or more of the secure elements 270 or a built-in secure element of the controller 250 allows the application to be installed on a respective secure elements 270. This enables entities to independently develop applications and load them in the secure elements 270. In some examples, a virtual firewall may also be implemented by the secure elements 270 to prevent applications installed by the secure elements 270 from interfering with each other.

[0032] FIG. 3 is a block diagram illustrating an example implementation of the SIG 220. The gateway 220 includes control circuitry 300, a processing element 310, and one or more secure elements 320, which may be the secure elements 270 of FIG. 2. In some cases, the gateway 220 includes four secure elements 320. Each secure element 320 of gateway 220 may be configured to perform a same function. In some implementations, secure elements 320 are implemented (both in hardware and software) to provide a higher level of security assurance than typical general-purpose microprocessors. In some implementations, secure elements 320 are implemented as general-purpose microprocessors without providing higher level of security assurance.

[0033] The control circuitry 300 and processing element 310 may be configured to

implement an allocation protocol for assigning a secure element 320 to a respective edge device 230. The control circuitry 300 and processing element 310 may be implemented by the controller 250 and/or on the sister board 260. To facilitate use of the secure elements 320 with the edge devices 230, the edge devices 230, secure elements 320, protocols, and the like may be implemented as actors that manage their own state and only communicate with other actors in the system using in-process messaging. These actors maintain the configuration state of the respective device of the actor and also dynamically receive references to the secure element 320 the actor is assigned for a given session. Similarly, the actors may get attached/registered with an actor capable of communicating using a desired transport and application protocol.

[0034] The control circuitry 300 and/or the processing element 310 may execute software that acts as a dispatcher actor that becomes responsible for dispatching the messages/data to the relevant and appropriate secure element 320 thereby shielding the whole array of secure elements 320 from the actors that represent the edge devices 230, for example. Use of actors and in-process messaging enables a high level of modularity in code development and maintenance, excellent performance due to zero overhead in interaction between components since they are all part of same process, as well as protection from any component malfunctioning. Typically, a fault or bug in a software component of a process leads to crash of the entire process. The actor model enables the system to contain faults within individual actors and thereby shield the process from the fault. This mechanism ends up providing almost 100% uptime and resiliency from ill behaving components/actors in the process.

[0035] The remote source 240 may provide updates for the secure elements 320 through the network connection 280. For example, the firmware of the first secure element 320 may be updated via a security enclave, such as a trusted execution environment, implemented by the processing element 310. In such cases, the security enclave may run applications that make use of crypto support and offer isolation from the general computing environment. In some implementations, the security enclave implemented by the processing element 310 includes symmetric or asymmetric key material that is used by the security enclave to communicate with another device. The cryptographic process and technique used by the security enclave to communicate with devices is different from the cryptographic process implemented by the secure elements of the gateway 220.

[0036] The number of secure elements 320 may be less than the number of edge devices 230 served by the secure elements 320. This may be advantageous when not all edge devices

230 are expected to be active contemporaneously. Further, this facilitates the ability to interleave requests to one secure element. In some examples, one secure element actor may be associated with two edge device actors. Even if the two edge device actors are active at the same time, the two edge device actors may be at different stages of communication. The requests from each edge device actor may be interleaved to the single secure element actor. For example, a transaction may include several command-response pairs between an edge device and a secure element. By the time a secure element returns the response to certain command from the first edge device, the controller could receive the different command from the second edge device. In this situation, interleaving the communication from the two edge devices facilitates efficient usage of a single secure element.

[0037] In examples in which the secure elements 320 are positioned on a sister board physically separable from the controller 250, it is desirable to authenticate the sister board when plugging in or otherwise connecting the sister board to the controller 250. To accomplish this, the controller 250 may include an additional secure element built-in to the controller 250 and having the capability to both authenticate the array of secure elements 320 and verify policy compatibility with a respective controller 250. In another embodiment, rather than including a built-in secure element on the controller 250, the secure enclave provided by microprocessors can be used.

[0038] FIG. 4A illustrates an example workflow for authenticating an array of secure elements 320 by a controller 250. In one example, the enclave or built-in secure element of the controller 250 contains an asymmetric key pair along with a signed root digital certificate. Having a signed digital certificate enables the enclave or built-in secure element in the controller 250 to send a random number to secure array of secure elements. The secure elements 320 in the array then return the signed random numbers along with the certificates that were used by the secure elements 320. Note that each secure element in the array has different certificates and keys, but all of the certificates have the same parent (root) certificate. The enclave or built-in secure element in the controller 250 then verifies the signature on the random number and verifies the certificates of the secure elements 320. This process, which is similar to public key infrastructure (PKI), can be used by the controller 250 to authenticate the secure elements 320. This process may be performed when the sister board is connected to the controller, and then again, or alternatively, at random intervals to protect against vulnerabilities that arise due to man-in-the-middle attacks, for example.

[0039] The above process only verifies the authenticity of the sister board, but it may also

be desirable for the sister board to authenticate the controller 250 and/or the system policies. FIG. 4B illustrates an example workflow for authenticating a policy and/or controller 250 by a sister board of secure elements 320. The policy may be authenticated with the help of the one or more remote devices 240. The enclave or built-in secure element in the controller 250 can authenticate with the remote devices 240 and obtain a signed cryptogram specific to the controller 250 in question. This signed cryptogram is then sent to the array of secure elements 320 in the sister board. Only if the signature is correct then the secure elements 320 will function. In some examples, the secure elements may also include the capability of parsing the policy and only providing a subset of the functionality to the controller 250 based on the parsed policy.

[0040] Some PACS systems are connected using full-duplex connections to communicate between readers and the PACS controllers. However, for some conventional systems, communication between the secure elements 320 and the edge devices 230 use half-duplex connections 290. In these systems, to ensure a desirable user experience, it is desirable to implement a full-duplex communication protocol for legacy half-duplex connections. For example, some conventional PACS systems may include RS-485 connections for the connections 290. In these conventional systems, one of the reader or controller acts as the primary communicator and the other acts as a secondary communicator. To facilitate communication between the edge devices 230 and the secure elements 320, a full-duplex protocol may be implemented for the half-duplex connections such that all devices can act as primary communicators.

[0041] The full-duplex protocol may be designed so that the protocol can be used on generic universal asynchronous receiver-transmitter (UART) hardware present in modern microcontrollers without a need for hardware modifications to existing devices. Together with resolving data collisions, the protocol helps with mitigating data corruption that might occur because of noisy or otherwise poor RS-485 lines. The protocol is intended to be used in combination with higher-level protocols without posing major limitations on them. In an open systems interconnection (OSI) model, the protocol can be implemented at the data-link layer. All data sent by the sender is acknowledged by the receiver. If a proper acknowledgment is not received by the sender in due time, the protocol incorporates a collision resolution algorithm that results in successful data delivery as described with respect to FIG. 6.

[0042] FIGS. 5A-5C are diagrams illustrating example data frame formats for a full-

duplex protocol for use in a half-duplex system. FIG. 5A illustrates a data frame 500 used to communicate data over the half-duplex connections. The data frame 500 is illustrated as including 37 bytes but may include any number of bytes. The data frame 500 includes an options byte (FIG. 5B), a data payload, and a four-byte cyclic redundancy check (CRC). While illustrated as 32 bytes, the data frame 500 may be configured to have a data payload of any size. The CRC may be used to detect errors in transmission of the data in the data frame 500.

[0043] FIG. 5B illustrates an example options field 510 for a data frame 500. The options field 510 includes an error indicator bit, 6 bits that are reserved for future use (RFU), and a toggle bit. The RFU bits may be allocated for any purpose. The error indictor is a single bit that indicates an error in the transmission protocol. If set to 0, the frame transfers data. If set to one, the frame indicates that a critical error has occurred on a device, such as an edge device 230, and the other device, such as the controller 250, should act accordingly. The error indication may be used in addition to error detection in higher level layers than the protocol is implemented. The toggle bit is used when sending data frames and is toggled every consecutive frame. The toggle bit is used to differentiate between two consecutive frames with the same data and retransmission of a single frame.

[0044] FIG. 5C illustrates an example acknowledgement frame 520 provided by a receiver when a data frame is successfully received from a sender. In this example, the acknowledgement frame includes a CRC field that is a copy of the CRC frame of the data frame 500. While illustrated as using the CRC for the acknowledgement frame 520, any data may be used for acknowledgement that allows the sender to verify with reasonable certainty that the data frame 500 was received by the recipient. For example, any bit pattern that both relates the acknowledgment to the data frame 500 and is large enough and random enough that the probability that random noise or a corrupted frame is identified as a valid acknowledgement is very small.

[0045] When implementing the communication protocol, two different roles may be statically assigned to the two devices, role “A” and role “B”. For example, the controller 250 may be assigned the role “A” and the edge devices 230 may be assigned the role “B”. The baud rate, number of stop bits used and bit order may be agreed between the devices in advance. An estimated time unit (ETU) value for the protocol may also be defined. The ETU value may generally be selected to be greater than the time required to transmit a data frame 500 and receive an acknowledgement frame 520 with some added margin. For example, for a

baud rate of 115200 bits-per-second (bps), a value of 5 milliseconds may be used.

[0046] FIG. 6 is a flowchart illustrating a method 600 for transmitting data according to a full-duplex protocol on a half-duplex connection. When sending a data frame 500, a device must wait until the line is idle. Once the line is idle, the data frame is transmitted with the specified number of bytes. When receiving a frame, the specified number of bytes is received and then the device waits for the line to become idle. When idling, both nodes are in reception mode (step 602) and waiting to receive data. Once received, the CRC of the received data frame 500 is checked. If the CRC is incorrect, the node begins receiving another data frame. If the CRC is correct, the node transmits the corresponding acknowledgement and starts receiving another data frame. When a new data frame is received, it is checked if the CRC of the frame is equal to the CRC of the last received frame if there was one. In case they match, the frame is considered a duplicate and is not reported to higher layers. An acknowledgment is still sent for duplicate frames.

[0047] When sending a data frame 500, at step 604, the device stops the reception mode. At step 606, the data frame 500 is sent using the specified number of bytes. Following transmission of the data frame 500, the device waits until either an acknowledgement is received (step 608) or an ETU has expired (step 610). If an ETU has expired prior to receiving the acknowledgement, method 600 proceeds to step 614. If the acknowledgement is successfully received, method 600 proceeds to step 612 and checks the CRC of the acknowledgement frame 520. If the CRC does not match that of the transmitted data frame 500, the method 600 proceeds to step 614. If the CRC matches, the data frame 500 was successfully sent and the method returns to step 602 and the device re-enters reception mode. At step 614, the device role is checked. If the device is a role "A" device, the method 600 returns to step 606 and retransmits the data frame 500. If the device is a role "B" device, the device proceeds to step 616 and enters reception mode for two ETUs to minimize collisions on the line and then returns to step 606 to retransmit the data frame 500. Method 600 provides a full-duplex protocol that resolves collisions for use on a half-duplex line.

[0048] FIG. 7 illustrates a block diagram of an example machine 700 upon which any one or more of the techniques (e.g., methodologies) discussed herein may perform. For example, the machine 700 can be any one or more of the edge devices 230, the controller 250, or the secure elements 270. Examples, as described herein, may include, or may operate by, logic or a number of components, or mechanisms in the machine 700. Circuitry (e.g., processing circuitry) is a collection of circuits implemented in tangible entities of the machine 700 that

include hardware (e.g., simple circuits, gates, logic, etc.). Circuitry membership may be flexible over time. Circuitries include members that may, alone or in combination, perform specified operations when operating. In an example, hardware of the circuitry may be immutably designed to carry out a specific operation (e.g., hardwired). In an example, the hardware of the circuitry may include variably connected physical components (e.g., execution units, transistors, simple circuits, etc.) including a machine readable medium physically modified (e.g., magnetically, electrically, moveable placement of invariant massed particles, etc.) to encode instructions of the specific operation. In connecting the physical components, the underlying electrical properties of a hardware constituent are changed, for example, from an insulator to a conductor or vice versa. The instructions enable embedded hardware (e.g., the execution units or a loading mechanism) to create members of the circuitry in hardware via the variable connections to carry out portions of the specific operation when in operation. Accordingly, in an example, the machine readable medium elements are part of the circuitry or are communicatively coupled to the other components of the circuitry when the device is operating. In an example, any of the physical components may be used in more than one member of more than one circuitry. For example, under operation, execution units may be used in a first circuit of a first circuitry at one point in time and reused by a second circuit in the first circuitry, or by a third circuit in a second circuitry at a different time. Additional examples of these components with respect to the machine 700 follow.

[0049] In alternative embodiments, the machine 700 may operate as a standalone device or may be connected (e.g., networked) to other machines. In a networked deployment, the machine 700 may operate in the capacity of a server machine, a client machine, or both in server-client network environments. In an example, the machine 700 may act as a peer machine in peer-to-peer (P2P) (or other distributed) network environment. The machine 700 may be a personal computer (PC), a tablet PC, a set-top box (STB), a personal digital assistant (PDA), a mobile telephone, a web appliance, a network router, switch or bridge, or any machine capable of executing instructions (sequential or otherwise) that specify actions to be taken by that machine. Further, while only a single machine is illustrated, the term "machine" shall also be taken to include any collection of machines that individually or jointly execute a set (or multiple sets) of instructions to perform any one or more of the methodologies discussed herein, such as cloud computing, software as a service (SaaS), other computer cluster configurations.

[0050] The machine (e.g., computer system) 700 may include a hardware processor 702 (e.g., a central processing unit (CPU), a graphics processing unit (GPU), a hardware processor core, or any combination thereof), a main memory 704, a static memory (e.g., memory or storage for firmware, microcode, a basic-input-output (BIOS), unified extensible firmware interface (UEFI), etc.) 706, and mass storage 708 (e.g., hard drive, tape drive, flash storage, or other block devices) some or all of which may communicate with each other via an interlink (e.g., bus) 730. The machine 700 may further include a display unit 710, an alphanumeric input device 712 (e.g., a keyboard), and a user interface (UI) navigation device 714 (e.g., a mouse). In an example, the display unit 710, input device 712 and UI navigation device 714 may be a touch screen display. The machine 700 may additionally include a storage device (e.g., drive unit) 708, a signal generation device 718 (e.g., a speaker), a network interface device 720, and one or more sensors 716, such as a global positioning system (GPS) sensor, compass, accelerometer, or other sensor. The machine 700 may include an output controller 728, such as a serial (e.g., universal serial bus (USB), parallel, or other wired or wireless (e.g., infrared (IR), near field communication (NFC), etc.) connection to communicate or control one or more peripheral devices (e.g., a printer, card reader, etc.).

[0051] Registers of the processor 702, the main memory 704, the static memory 706, or the mass storage 708 may be, or include, a machine readable medium 722 on which is stored one or more sets of data structures or instructions 724 (e.g., software) embodying or utilized by any one or more of the techniques or functions described herein. The instructions 724 may also reside, completely or at least partially, within any of registers of the processor 702, the main memory 704, the static memory 706, or the mass storage 708 during execution thereof by the machine 700. In an example, one or any combination of the hardware processor 702, the main memory 704, the static memory 706, or the mass storage 708 may constitute the machine readable media 722. While the machine readable medium 722 is illustrated as a single medium, the term "machine readable medium" may include a single medium or multiple media (e.g., a centralized or distributed database, and/or associated caches and servers) configured to store the one or more instructions 724.

[0052] The term “machine readable medium” may include any medium that is capable of storing, encoding, or carrying instructions for execution by the machine 700 and that cause the machine 700 to perform any one or more of the techniques of the present disclosure, or that is capable of storing, encoding or carrying data structures used by or associated with such instructions. Non-limiting machine readable medium examples may include solid-state

memories, optical media, magnetic media, and signals (e.g., radio frequency signals, other photon based signals, sound signals, etc.). In an example, a non-transitory machine readable medium comprises a machine readable medium with a plurality of particles having invariant (e.g., rest) mass, and thus are compositions of matter. Accordingly, non-transitory machine-readable media are machine readable media that do not include transitory propagating signals. Specific examples of non-transitory machine readable media may include: non-volatile memory, such as semiconductor memory devices (e.g., Electrically Programmable Read-Only Memory (EPROM), Electrically Erasable Programmable Read-Only Memory (EEPROM)) and flash memory devices; magnetic disks, such as internal hard disks and removable disks; magneto-optical disks; and CD-ROM and DVD-ROM disks.

[0053] The instructions 724 may be further transmitted or received over a communications network 726 using a transmission medium via the network interface device 720 utilizing any one of a number of transfer protocols (e.g., frame relay, internet protocol (IP), transmission control protocol (TCP), user datagram protocol (UDP), hypertext transfer protocol (HTTP), etc.). Example communication networks may include a local area network (LAN), a wide area network (WAN), a packet data network (e.g., the Internet), mobile telephone networks (e.g., cellular networks), Plain Old Telephone (POTS) networks, and wireless data networks (e.g., Institute of Electrical and Electronics Engineers (IEEE) 802.11 family of standards known as Wi-Fi®), IEEE 802.16.4 family of standards, peer-to-peer (P2P) networks, among others. In an example, the network interface device 720 may include one or more physical jacks (e.g., Ethernet, coaxial, or phone jacks) or one or more antennas to connect to the communications network 726. In an example, the network interface device 720 may include a plurality of antennas to wirelessly communicate using at least one of single-input multiple-output (SIMO), multiple-input multiple-output (MIMO), or multiple-input single-output (MISO) techniques. The term “transmission medium” shall be taken to include any intangible medium that is capable of storing, encoding or carrying instructions for execution by the machine 700, and includes digital or analog communications signals or other intangible medium to facilitate communication of such software. A transmission medium is a machine readable medium.

[0054] The above description includes references to the accompanying drawings, which form a part of the detailed description. The drawings show, by way of illustration, specific embodiments in which the invention can be practiced. These embodiments are also referred to herein as “examples.” Such examples can include elements in addition to those shown or

described. However, the present inventors also contemplate examples in which only those elements shown or described are provided. Moreover, the present inventors also contemplate examples using any combination or permutation of those elements shown or described (or one or more aspects thereof), either with respect to a particular example (or one or more aspects thereof), or with respect to other examples (or one or more aspects thereof) shown or described herein.

[0055] In this document, the term "or" is used to refer to a nonexclusive or, such that "A or B" includes "A but not B," "B but not A," and "A and B," unless otherwise indicated. The Abstract is provided to allow the reader to quickly ascertain the nature of the technical disclosure. It is submitted with the understanding that it will not be used to interpret or limit the scope or meaning of the aspects. Also, in the above Detailed Description, various features may be grouped together to streamline the disclosure. This should not be interpreted as intending that an unclaimed disclosed feature is essential to any claim. Rather, inventive subject matter may lie in less than all features of a particular disclosed embodiment. Thus, the following aspects are hereby incorporated into the Detailed Description as examples or embodiments, with each aspect standing on its own as a separate embodiment, and it is contemplated that such embodiments can be combined with each other in various combinations or permutations. The scope of the invention should be determined with reference to the appended aspects, along with the full scope of equivalents to which such aspects are entitled.

CLAIMS

What is claimed is:

1. A system for providing secure execution of system functions for edge devices, the system comprising:

a controller configured to connect for communication with a plurality of edge devices, each edge device configured to obtain data for an application of the system, the controller further configured to receive the data from each of the plurality of edge devices; and

an array of secure elements connectable to the controller, wherein the array of secure elements is configured to execute functions using the data received from the plurality of edge devices;

wherein the controller is configured to associate an identified secure element of the array of secure elements with a respective edge device of the plurality of edge devices to execute the functions for data received from the respective edge device; and

wherein the controller is configured to communicate a result of the executed functions to the respective edge device.

2. The system of Claim 1, wherein the controller is configured to connect for communication to each of the plurality of edge devices through a connection comprising Ethernet, Wi-Fi, RS-485, or universal serial bus (USB).

3. The system of Claim 1 or 2, wherein the system is a physical access control system, and the plurality of edge devices are readers physically co-located with a physical access device.

4. The system of any preceding claim, wherein a total number of the plurality of edge devices is greater than a total number of the secure elements.

5. The system of Claim 4, wherein the controller is configured to associate at least two of the plurality of edge devices with a single secure element of the array of secure elements, and wherein the single secure element is configured to interleave communications from the at least two of the plurality of edge devices.

6. The system of any preceding claim, wherein the plurality of secure elements are positioned on a sister board, and wherein the sister board physically connects to the controller and communicates with the controller via one or more bus protocols.

7. The system of Claim 6, wherein the controller is configured to verify authenticity of the sister board using at least one certificate signed by each of the secure elements of the array of secure elements.

8. The system of Claim 6 or 7, wherein the sister board is configured to authenticate the controller by verifying a cryptographic signature of a policy cryptogram received from the controller via one or more remote sources that manage the policy.

9. The system of any preceding claim, wherein the secure elements are programmable to perform custom functions, and wherein the secure elements of the array of secure elements are configured to verify authenticity of the custom functions to permit the installation and execution of the custom functions.

10. The system of any preceding claim, further comprising the plurality of edge devices.

11. A method for providing secure execution of functions for a plurality of edge devices in an internet-of-things (IoT) system, the method comprising:

receiving, by a controller of the IoT system, data from a respective edge device of the plurality of edge devices of the IoT system;

associating an identified secure element of an array of secure elements connected to the controller with the respective edge device;

providing the received data to the identified secure element, wherein the secure element executes functions using the data; and

communicating, via the controller, a result of the executed functions of the identified secure element to the respective edge device.

12. The method of Claim 11, wherein the IoT system is a physical access control system, and the plurality of edge devices are readers physically co-located with a physical access device.

13. The method of Claim 11 or 12, wherein the controller is connected to communicate with the respective edge device using an RS-485, Ethernet, Wi-Fi, or universal serial bus (USB) connection.

14. The method of any of Claims 11 to 13, wherein a total number of the plurality of edge devices is greater than a total number of the secure elements, and wherein the controller is configured to associate at least two of the plurality of edge devices with a single secure element of the array of secure elements, and wherein the single secure element is configured to interleave communications from the at least two of the plurality of edge devices.

15. The method of any of Claims 11 to 14, wherein the plurality of secure elements are positioned on a sister board, wherein the sister board physically connects to the controller and communicates with the controller via one or more bus protocols.

16. The method of Claim 15, further comprising verifying authenticity of the sister board, by the controller, using at least one certificate signed by each of the secure elements of the array of secure elements.

17. A non-transitory computer readable medium comprising executable program code, that when executed by one or more processors, causes the one or more processors to:

receive data from a respective edge device of a plurality of edge devices of an IoT system;

associate an identified secure element of an array of secure elements with the respective edge device;

provide the received data to the identified secure element, wherein the secure element executes functions using the data; and

communicate a result of the executed functions of the identified secure element to the respective edge device.

18. The non-transitory computer readable medium of Claim 17, wherein the executable program code further causes the one or more processors to associate at least two of the plurality of edge devices with a single secure element of the array of secure elements,

wherein the single secure element is configured to interleave communications from the at least two of the plurality of edge devices.

19. A physical access control system comprising:

a controller configured to connect to each of a plurality of readers to receive credential information therefrom, each of the plurality of readers positioned to communicate with a credential device to receive respective credential information therefrom for controlling access to a respective secured area using the respective credential information; and

an array of secure elements configured to execute user authentication using the credential information received from the plurality of readers;

wherein the controller is configured associate an identified secure element of the array of secure elements with a respective reader of the plurality of readers to perform user authentication using the credential information received from the respective reader; and

wherein the controller is connected to communicate a result of the user authentication to the respective reader.

20. The physical access control system of Claim 19, further comprising the plurality of readers.

21. The physical access control system of Claim 19 or 20, wherein the controller is configured to connect for communication to each of the plurality of readers through a connection comprising Ethernet, Wi-Fi, RS-485, or universal serial bus (USB).

22. The physical access control system of any of Claims 19 to 21, wherein a total number of the plurality of readers is greater than a total number of the secure elements.

23. The physical access control system of Claim 22, wherein the controller is configured to associate at least two of the plurality of readers with a single secure element of the array of secure elements, and wherein the single secure element is configured to interleave communications from the at least two of the plurality of readers.

24. A method for transmitting data using a collision-resistant full-duplex communication protocol over a half-duplex connection, the method comprising:

ending a data reception mode for a first device transmitting the data;

transmitting, by the first device, a data frame over the half-duplex connection for receipt by a second device;

monitoring, by the first device, for an acknowledgement from the second device, the acknowledgement comprising a cyclic redundancy check value from the transmitted data frame;

determining that the data frame needs to be retransmitted;

identifying a device role of the first device; and

retransmitting the data frame at a time according to the identification of the device role.

25. The method of Claim 24, wherein the half-duplex connection is an RS-485 connection.

26. The method of Claim 24 or 25, wherein the first device is a controller of a physical access control system, and wherein the second device is a reader of the physical access control system.

27. The method of any of Claims 24 to 26, wherein the data frame comprises an options field, a payload field, and a cyclic redundancy check field, and wherein determining that the data frame needs to be retransmitted comprises:

receiving the acknowledgment from the second device;

comparing the acknowledgment to the cyclic redundancy check field of the data frame; and

determining that the data frame needs to be retransmitted if the acknowledgement does not match the cyclic redundancy check field.

28. The method of any of Claims 24 to 26, wherein determining that the data frame needs to be retransmitted comprises failing to receiving, by the first device, the acknowledgement frame from the second device within a specified time period.

29. A non-transitory computer readable medium comprising executable program code, that when executed by one or more processors, causes the one or more processors to:

end a data reception mode of a first device;

transmit a data frame from the first device over a half-duplex connection for receipt by a second device;

monitor for an acknowledgement from the second device, the acknowledgement comprising a cyclic redundancy check value from the transmitted data frame;

determine that the data frame needs to be retransmitted;

identify a device role of the first device; and

retransmit the data frame at a time according to the identification of the device role.

30. The non-transitory computer readable medium of Claim 29, wherein the data frame comprises an options field, a payload field, and a cyclic redundancy check field, and wherein determining that the data frame needs to be retransmitted comprises:

receiving the acknowledgment from the second device;

comparing the acknowledgment to the cyclic redundancy check field of the data frame; and

determining that the data frame needs to be retransmitted if the acknowledgement does not match the cyclic redundancy check field.

31. The non-transitory computer readable medium of Claim 29, wherein determining that the data frame needs to be retransmitted comprises failing to receive the acknowledgement frame from the second device within a specified time period from the second device.

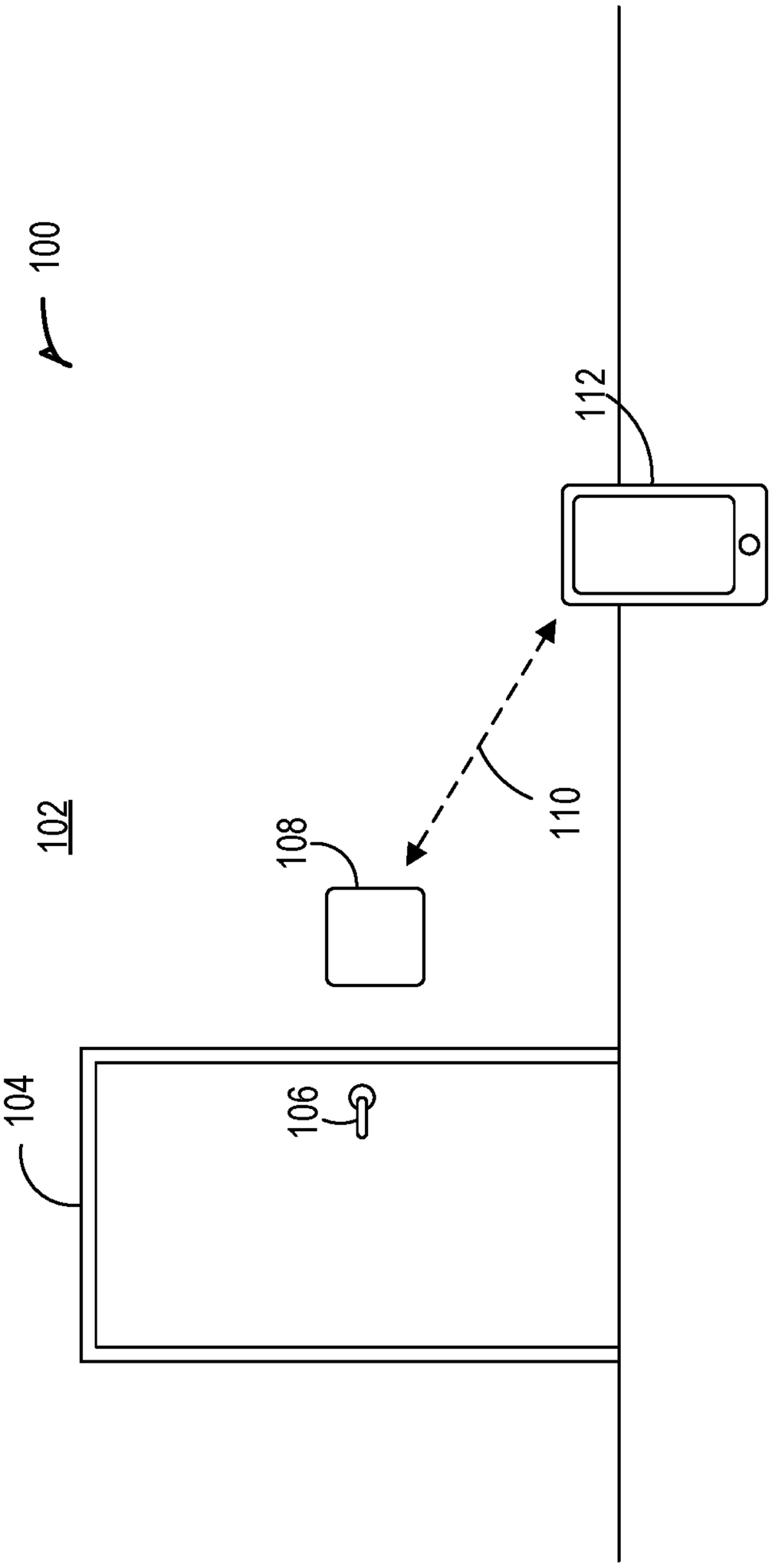

FIG. 1

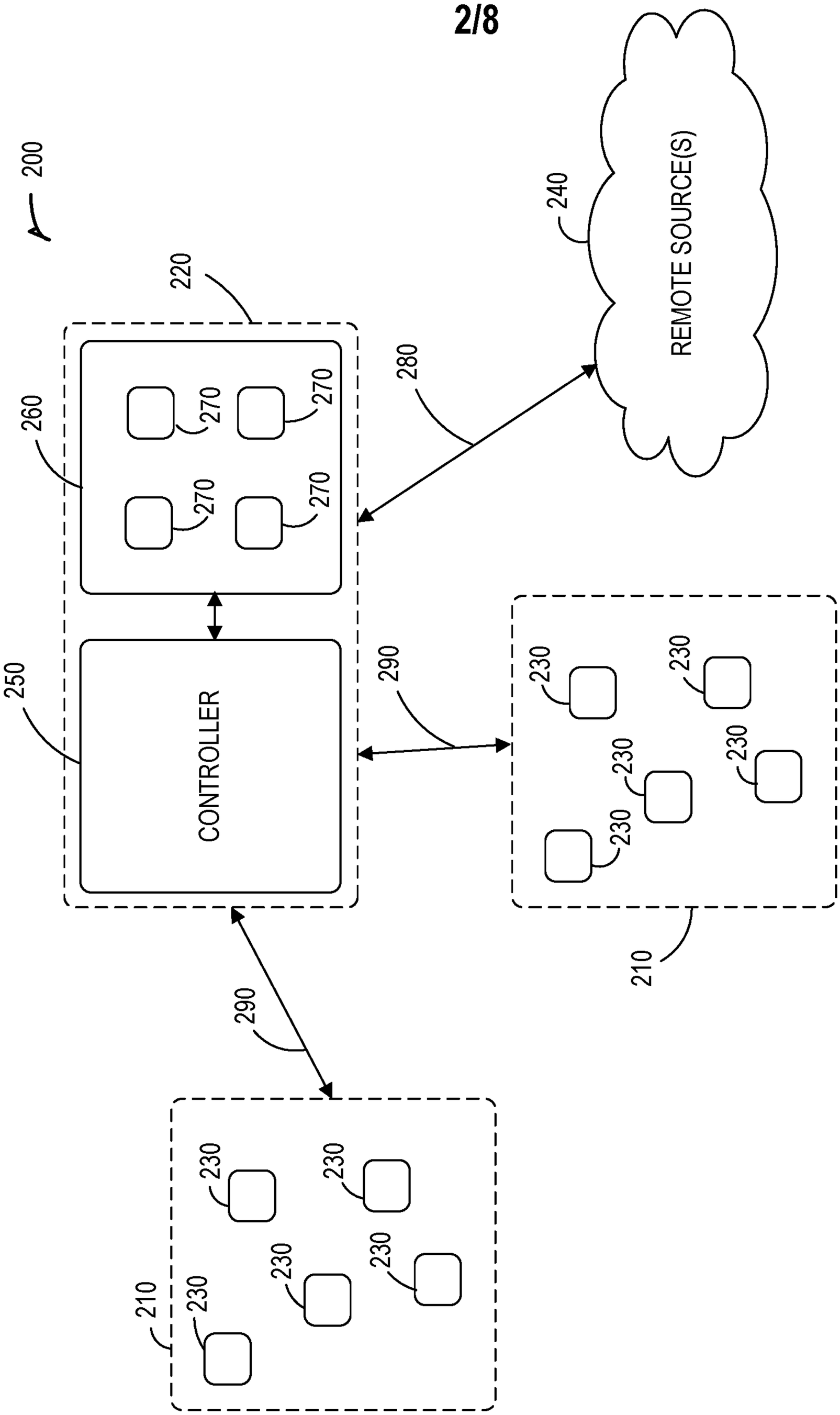
200
220
260
270
270
270
270
250
CONTROLLER
280
240
REMOTE SOURCE(S)
290
230
230
230
230
230
210
290
230
230
230
210

FIG. 2

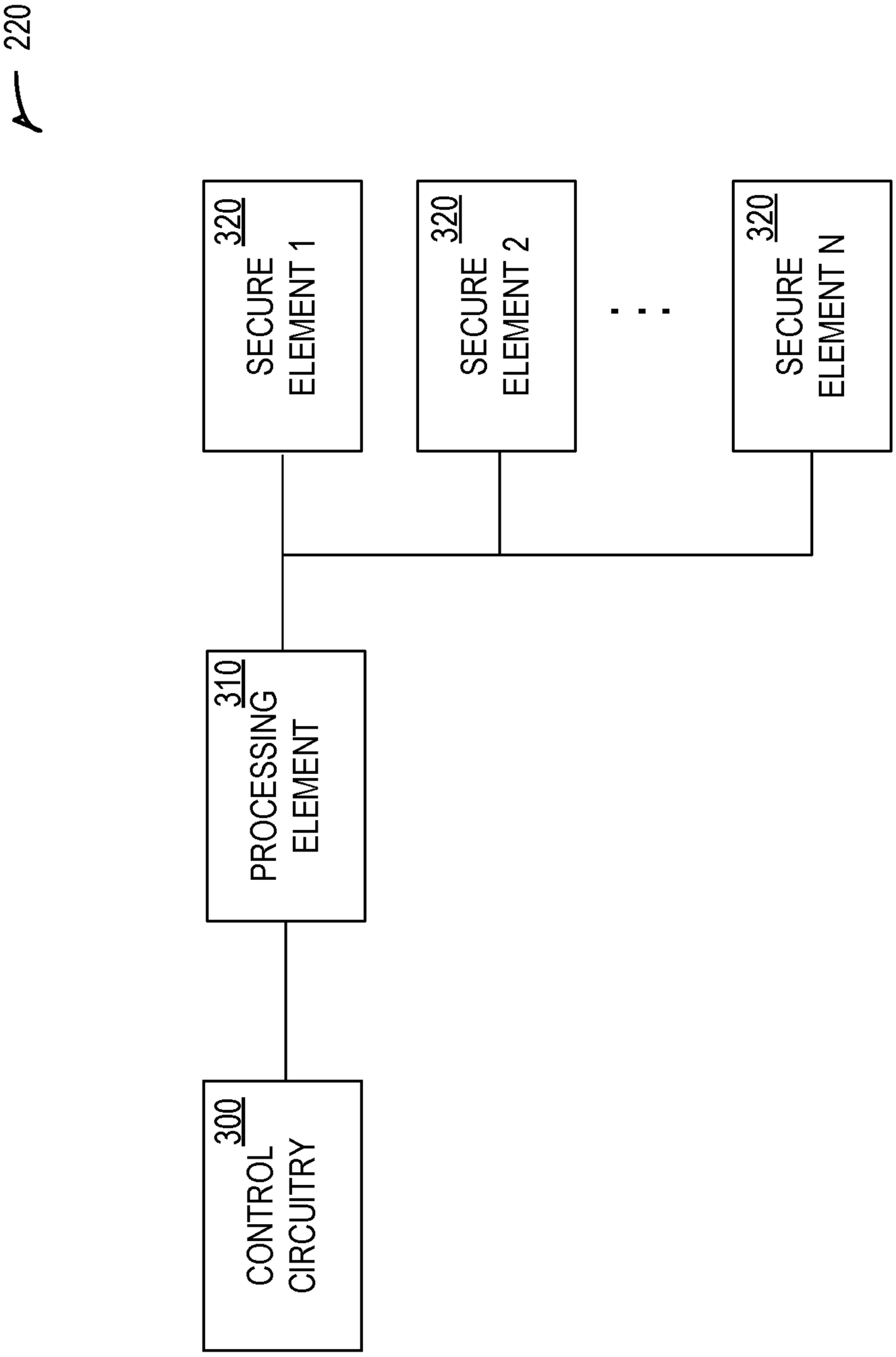
220
320
SECURE ELEMENT 1
320
SECURE ELEMENT 2
320
SECURE ELEMENT N
310
PROCESSING ELEMENT
300
CONTROL CIRCUITRY

FIG. 3

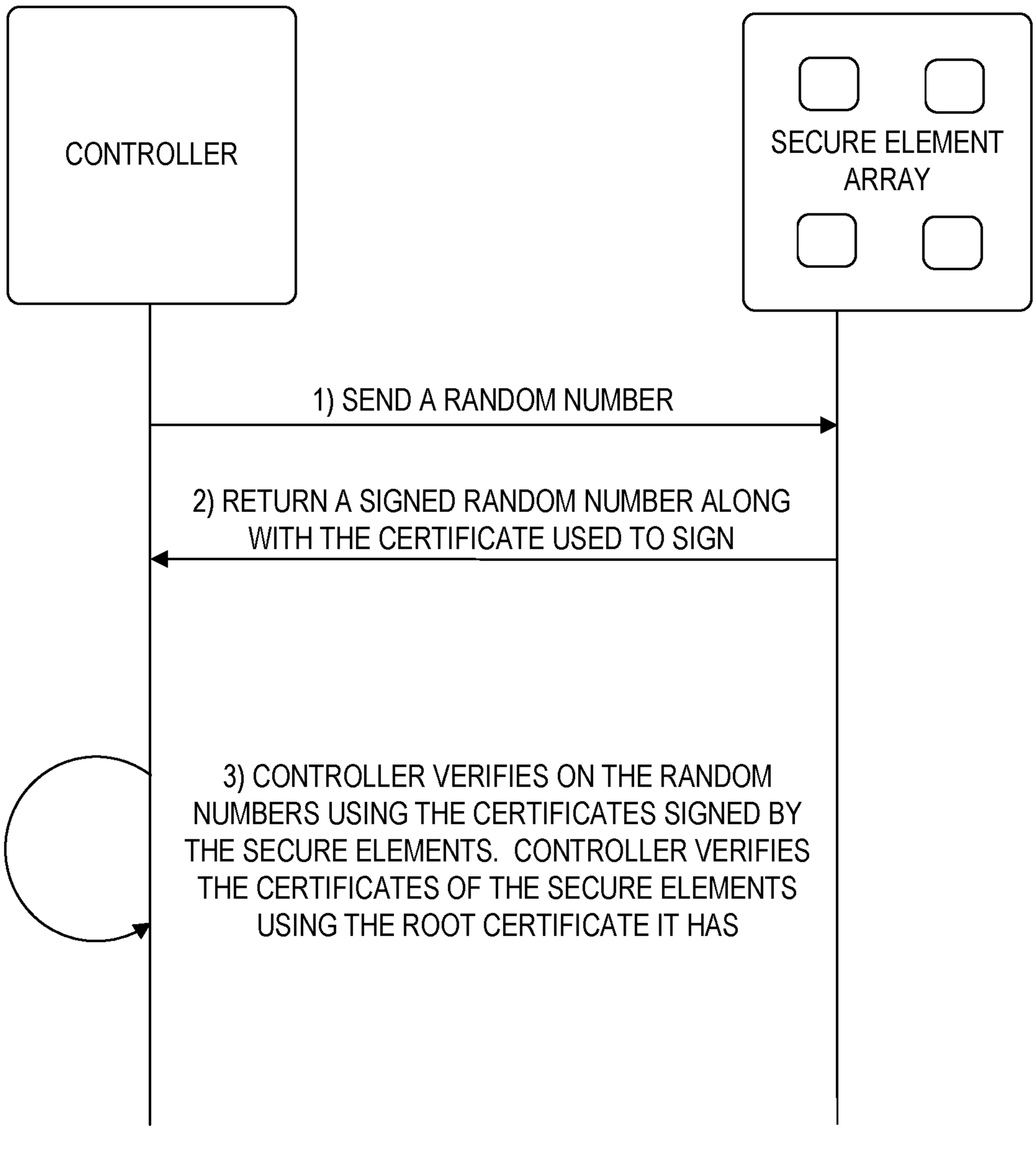
CONTROLLER
SECURE ELEMENT ARRAY
1) SEND A RANDOM NUMBER
2) RETURN A SIGNED RANDOM NUMBER ALONG WITH THE CERTIFICATE USED TO SIGN
3) CONTROLLER VERIFIES ON THE RANDOM NUMBERS USING THE CERTIFICATES SIGNED BY THE SECURE ELEMENTS. CONTROLLER VERIFIES THE CERTIFICATES OF THE SECURE ELEMENTS USING THE ROOT CERTIFICATE IT HAS

FIG. 4A

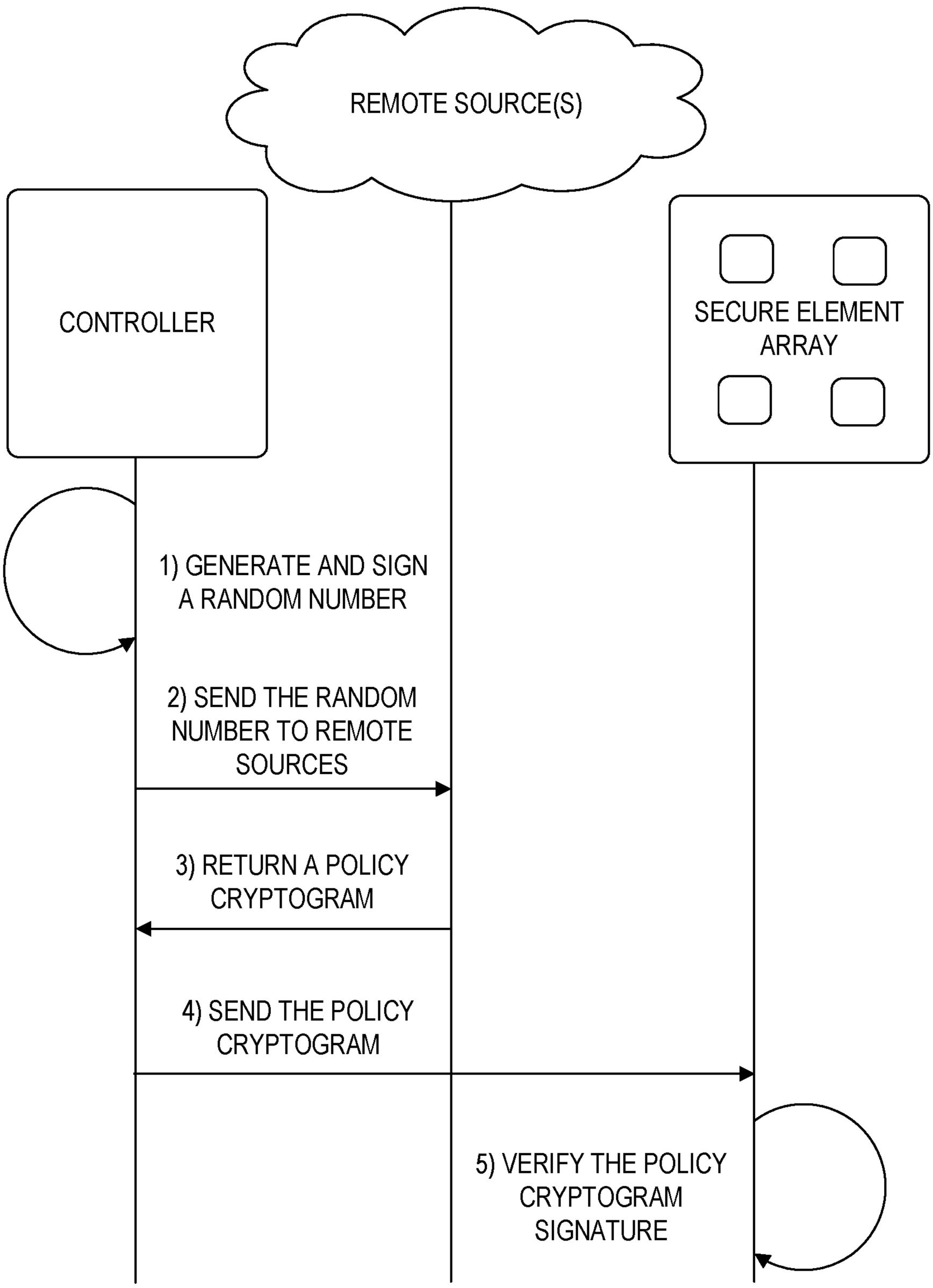
REMOTE SOURCE(S)
CONTROLLER
SECURE ELEMENT ARRAY
1) GENERATE AND SIGN A RANDOM NUMBER
2) SEND THE RANDOM NUMBER TO REMOTE SOURCES
3) RETURN A POLICY CRYPTOGRAM
4) SEND THE POLICY CRYPTOGRAM
5) VERIFY THE POLICY CRYPTOGRAM SIGNATURE

FIG. 4B

500

OPTIONS	DATA	CRC
1 BYTE	32 BYTES	4 BYTES

FIG. 5A

510

BIT 7	BITS 6...1	BIT 0
ERROR INDICATOR	RFU	TOGGLING BIT

FIG. 5B

520

CRC

FIG. 5C

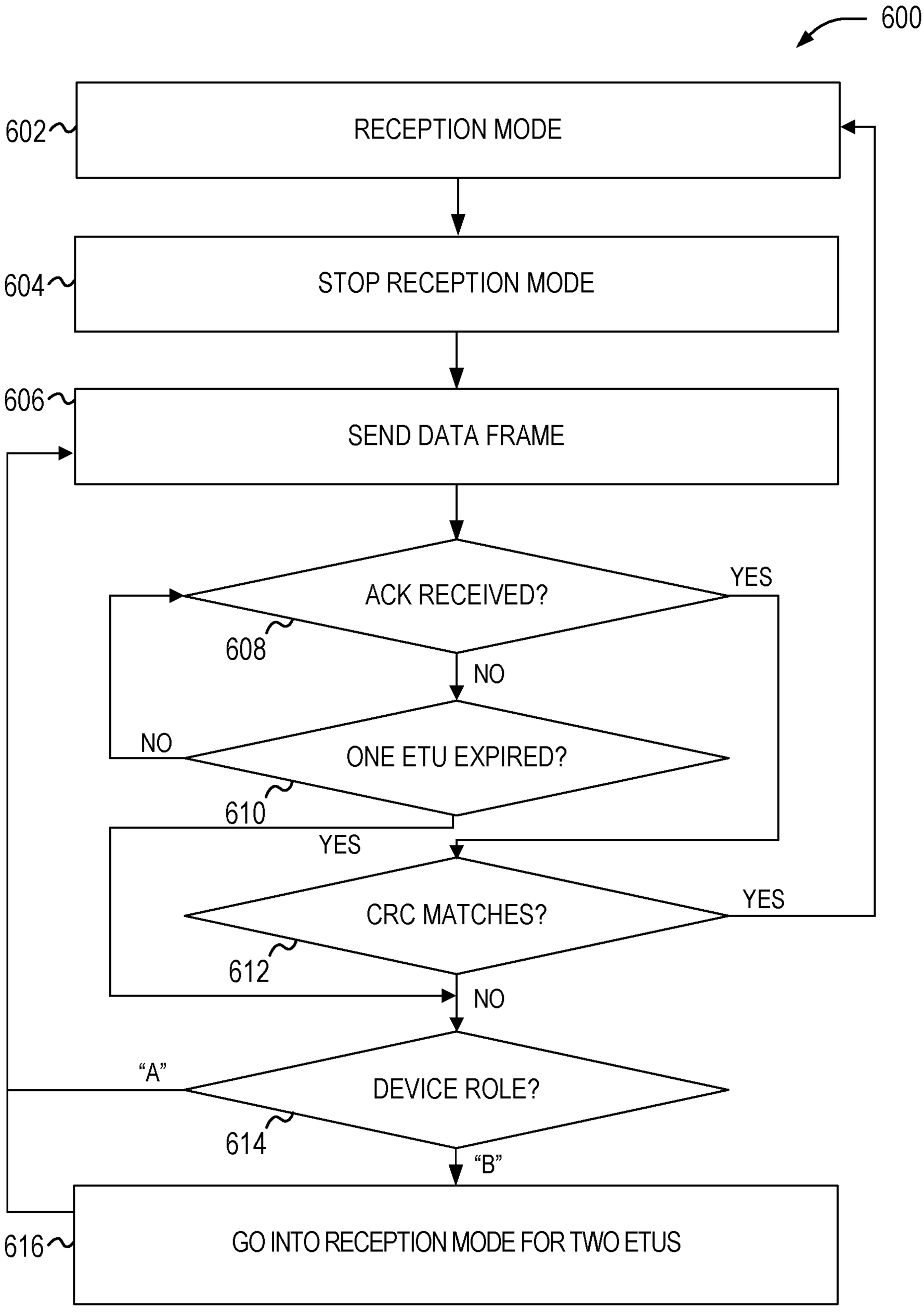
600
602
RECEPTION MODE
604
STOP RECEPTION MODE
606
SEND DATA FRAME
608
ACK RECEIVED?
YES
NO
610
ONE ETU EXPIRED?
NO
YES
612
CRC MATCHES?
YES
NO
614
DEVICE ROLE?
"A"
"B"
616
GO INTO RECEPTION MODE FOR TWO ETUS

FIG. 6

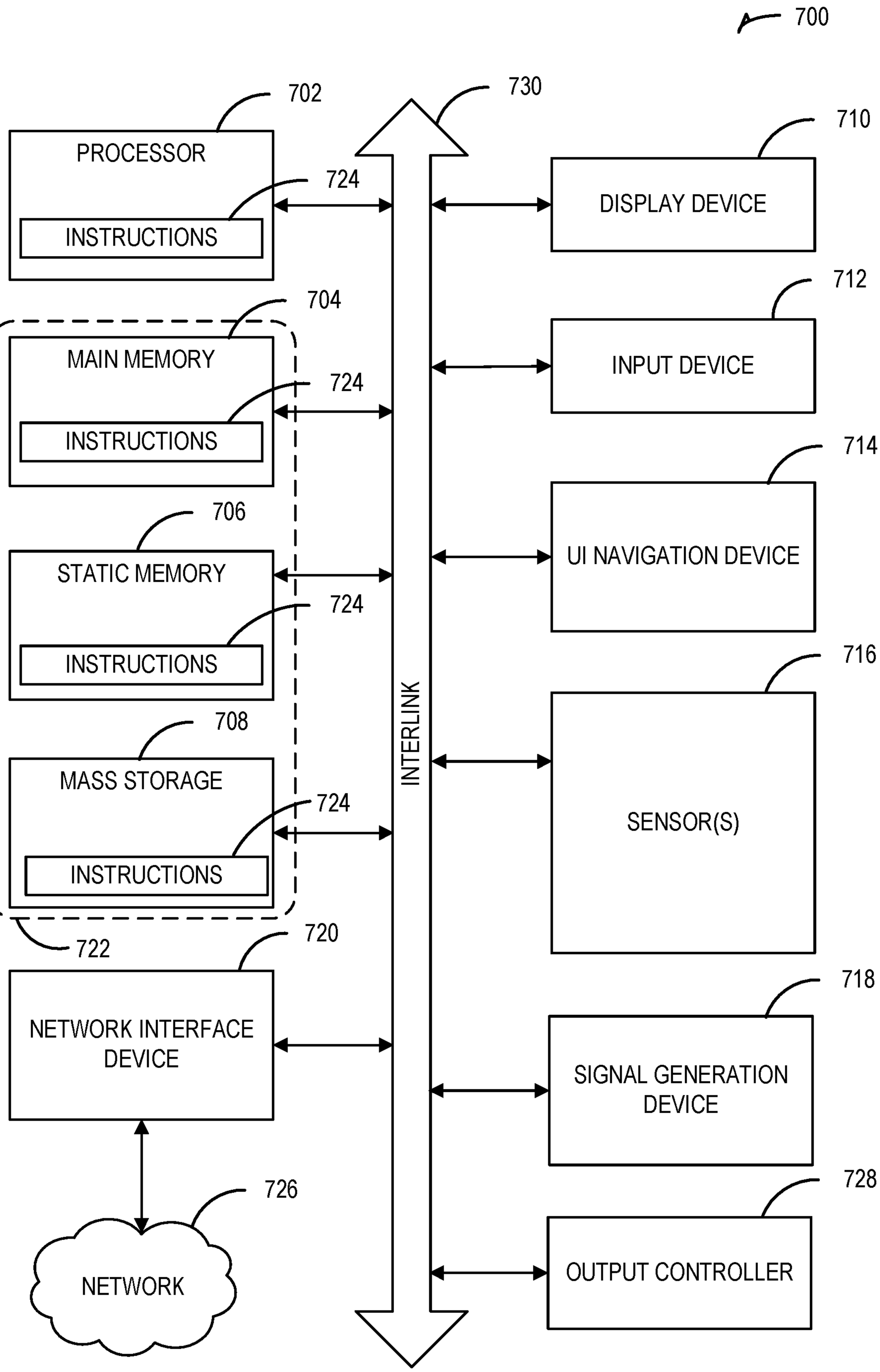
700
702
PROCESSOR
724
INSTRUCTIONS
730
710
DISPLAY DEVICE
704
MAIN MEMORY
724
INSTRUCTIONS
712
INPUT DEVICE
706
STATIC MEMORY
724
INSTRUCTIONS
714
UI NAVIGATION DEVICE
708
MASS STORAGE
724
INSTRUCTIONS
INTERLINK
716
SENSOR(S)
722
720
NETWORK INTERFACE DEVICE
718
SIGNAL GENERATION DEVICE
726
NETWORK
728
OUTPUT CONTROLLER

FIG. 7