



(51) International Patent Classification:

G06F 21/72 (2013.01) H04L 9/16 (2006.01)
H04L 9/12 (2006.01)

(21) International Application Number:

PCT/US2017/013377

(22) International Filing Date:

13 January 2017 (13.01.2017)

(25) Filing Language:

English

(26) Publication Language:

English

(30) Priority Data:

14/997,113 15 January 2016 (15.01.2016) US

(71) Applicant: **BLOCKCHAIN ASICS LLC** [US/US]; 500 Bryant Street, San Francisco, California 94107 (US).

(72) Inventors: **RODRIGUEZ DE CASTRO, Edward L.**; 2131 Folsom Avenue, San Francisco, California 94110 (US). **PORT, Adrian George**; 2131 Folsom Avenue, San Francisco, California 94110 (US). **REDDY, Sagar V.**; 121 W. Washington Avenue, Suite 214, Sunnyvale, California 94086 (US).

(74) Agents: **PERDOK, Monique M.** et al.; Schwegman, Lundberg & Woessner, P.A., P.O. Box 2938, Minneapolis, Minnesota 55402 (US).

(81) Designated States (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, 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, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

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

(54) Title: CRYPTOGRAPHIC ASIC INCLUDING CIRCUITRY-ENCODED TRANSFORMATION FUNCTION

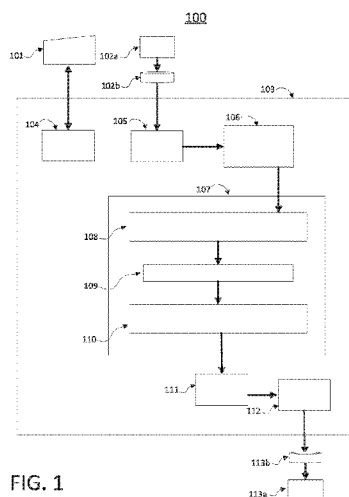


FIG. 1

(57) Abstract: A transform-enabled integrated circuit for use in cryptographic proof-of-work systems is provided. The transform-enabled integrated circuit includes a transformation block embedded among other circuitry components within the cryptographic datapath of the transform-enabled integrated circuit. The transformation block may be configured at a time subsequent to the manufacture of the integrated circuit to embody as circuitry any one of a plurality of mathematical transformation functions, thus enabling a user to systemically modify the results of cryptographic operations performed by the integrated circuit while retaining the high performance and efficiency characteristics of application specific integrated circuits. Embodiments of the technology disclosed herein provides an hereto unachievable level of flexibility in the deployment of application-specific integrated circuits within proof-of-work verification systems, such as private block chain systems, public block chain systems, digital rights management, secure token and other cryptography-related fields.



CRYPTOGRAPHIC ASIC INCLUDING CIRCUITRY-ENCODED TRANSFORMATION FUNCTION

Technical Field

5 The disclosed technology relates generally to the design of integrated electronic circuits, and more particularly, some embodiments relate to the design of cryptographic integrated circuits.

Background

10 Application-specific integrated circuits (ASICs) are integrated circuits designed and built to serve a particular purpose or application. ASICs provide fast computational speed compared with slower, more generalized solutions, such as software solutions running on general-purpose processors or field programmable gate arrays (FPGAs). As the name implies, ASICs are generally
15 designed to perform only one specific application, resulting in a trade-off between flexibility and computational speed. ASICs are increasing in importance in cryptography-related fields, such as proof-of-work systems, digital rights management systems, and other applications generally having stringent speed and efficiency requirements.

20

Brief Description of the Drawings

 The technology disclosed herein, in accordance with one or more various embodiments, is described in detail with reference to the following figures. The drawings are provided for purposes of illustration only and merely depict typical
25 or example embodiments of the disclosed technology. These drawings are provided to facilitate the reader's understanding of the disclosed technology and shall not be considered limiting of the breadth, scope, or applicability thereof. It should be noted that for clarity and ease of illustration these drawings are not necessarily made to scale.

30 Figure 1 illustrates an example cryptographic processing system within which embodiments of the technology disclosed herein may be implemented.

 Figure 2 illustrates an example transform-enabled integrated circuit in accordance with embodiments of the technology disclosed herein and containing

a programmable transform block located at one of the points along the integrated circuits datapath contemplated as part of the technology disclosed herein.

Figure 3 illustrates an example transform-enabled integrated circuit in accordance with embodiments of the technology disclosed herein and containing
5 a programmable transform block located at a different point along the integrated circuits datapath contemplated as part of the technology disclosed herein.

Figure 4 illustrates an example transform-enabled integrated circuit in accordance with embodiments of the technology disclosed herein and containing two programmable transform blocks, both located at points along the integrated
10 circuits datapath contemplated as part of the technology disclosed herein.

Figure 5A illustrates an example programmable transform block configuration prior to coding the transformation function in accordance with embodiments of the technology disclosed herein.

Figure 5B illustrates an example programmable transform block coded
15 according to a binary key in accordance with embodiments of the technology disclosed herein.

Figure 6 illustrates an example function coding process in accordance with embodiments of the technology disclosed herein.

Figure 7 is a block diagram of a basic cryptographic processing system
20 implementing a cryptographic integrated circuit with multiple cryptographic processing cores in accordance with embodiments of the technology disclosed herein.

Figure 8 illustrates an example transform-enabled integrated circuit having multiple cryptographic processing cores in accordance with embodiments
25 of the technology disclosed herein.

Figure 9 illustrates another example transform-enabled integrated circuit having multiple cryptographic processing cores, each cryptographic processing core having a programmable transform block included, in accordance with
embodiments of the technology disclosed herein.

Figure 10 illustrates an example dual-purpose, transform-enabled
30 integrated circuit in accordance with embodiments of the technology disclosed herein.

Figure 11 illustrates an example computing component that may be used in implementing various features of embodiments of the disclosed technology.

Figure 12 illustrates an example a transform-enabled cryptographic result validator containing one transform enabled cryptographic hashing core and one configuration key.

Figure 13 illustrates an example a transform-enabled cryptographic result
5 validator containing two separate transform enabled cryptographic hashing core and two separate configuration keys.

The figures are not intended to be exhaustive or to limit the invention to the precise form disclosed. It should be understood that the invention can be practiced with modification and alteration, and that the disclosed technology be
10 limited only by the claims and the equivalents thereof.

Detailed Description of the Embodiments

According to various embodiments of the disclosed technology, an integrated circuit is provided for use in proof-of-work based cryptographic
15 verification processes, including but not limited to the cryptographic network transaction verification systems often used in the emerging field of blockchain technology. The integrated circuit includes one or more blocks of circuitry implementing a cryptographic function, generally a cryptographic one-way function (OWF) that is hard to reverse, such as a secure hashing algorithm. One
20 or more transformation functions selected by one or more users and embodied as blocks of datapath circuitry are placed on the integrated circuits datapath at a point prior to at least one of the blocks of circuitry implementing a cryptographic function described above. Each block of circuitry implementing the transformation function may be programmed by the user by supplying a
25 configuration key, for example, a key composed of a string of binary digits. Such a key is then embodied as datapath circuitry within the transformation block.

Following such programing, the circuitry within the transformation block will effect a specific programmatic transformation reflecting the key
30 programmed by the user on the data it receives from circuitry prior to it along the integrated circuits datapath, and pass transformed data onto further circuitry along the datapath. Thus, and regardless of the content of the data received by the transformation block, the transformation it applies will directly and consistently affect the final value calculated by circuitry further along the

datapath, including the blocks or blocks of circuitry implementing a cryptographic function described above.

Due to interaction of the transformation with the mathematical properties of the cryptographic algorithms involved, particularly their nature as OWFs that
5 are hard to revert (here, and at other points in this document, terms such as “hard” and “easy” may be understood in the sense of computational complexity theory, particularly polynomial time theory), the combined effect is to produce a systemic transformation of the bits contained in the final value calculated by the circuit that is not easily deciphered, not easily distinguishable from noise, and
10 not easily replicable by a party lacking full prior knowledge of the user’s key or keys, but yet is fully consistent and easily replicable and thus verifiable by a party with prior knowledge of the keys or access to the means to utilize them in calculation even while lacking knowledge of them (such as, a party in possession of an ASIC programmed to embody the keys within its datapath circuitry).

15 The application of the above within the proof-of-work calculations often used field of blockchain technology enables a great expansion in the flexibility of its deployment. Specifically, it enables users to, in effect, create a near unlimited number of orthogonal blockchains. Any such blockchain may be fully distinct and separate from any other, as it is characterized by a unique set of
20 valid solutions to the proof-of-work problem, yet all such blockchains are fully internally coherent as regards the cryptographic references of each block to the preceding, and fully demonstrable to any party with access to an ASIC configured to operate within the cryptographic framework of the given blockchain.

25 Thus, various embodiments of the technology described herein enable the creation of a new generation of ASICs dedicated to the network transaction verification of blockchain operations that are capable of verifying transactions across a near unlimited number of distinct and fully orthogonal blockchains, public or private. Such blockchains may be safely operated in the immediate
30 proximity of each other and across multiple or the same networks.

Additionally, various implementations of the technology described herein are provided describing various methods to embody the transformation function key or keys as datapath circuitry within the integrated circuit in such a manner as

to result in a very limited performance overhead, thus retaining core advantages ASICs have over other types of solutions in terms of speed and efficiency.

Further, various methods are provided to further protect the key or keys embodied as datapath circuitry within the integrated circuit. Such methods
5 include, without limitation: various methods to prevent the discovery of the key or keys via an analysis of the inputs and outputs of the integrated circuit; the use of one or several lock fuses in the connecting circuitry between the configuration interface over which the user may program the transform function and the block or blocks of circuitry embodying the transformation as circuitry; the use of
10 multi-part keys; the use of multiple transform functions; the use of transform functions that may only be accessed during the manufacturing process of the integrated circuit; and others.

Other features and aspects of the disclosed technology will become apparent from the following detailed description, taken in conjunction with the
15 accompanying drawings, which illustrate, by way of example, the features in accordance with embodiments of the disclosed technology.

Proof-of-work verification systems are a class of verification systems predicated on solving complex computational problems, not for any intrinsic value of the answer arrived at, but for the probative value such an answer
20 provides that a significant amount of computational work has been expended in producing such a result. Hence the name. Proof-of-work verification systems have applications across a broad range of modern computational systems, including, without limitation: systems used for the deterrence of denial-of-service (DoS) network attacks; systems used for the prevention of unwanted
25 commercial email (spam); and systems used for other applications.

One such other use of proof-of-work verification in its operations is the cryptographic network transaction verification process underlying the emerging field of blockchain technology. Blockchain technology includes the various underlying technologies enabling decentralized trustless transaction systems,
30 such as those supporting cryptographic currencies, the most widely known of which is bitcoin.

As a means to provide a context within which some implementations of the technology described herein may operate, it is useful to briefly discuss how

blockchain systems operate prior to entering into a description of the each illustration.

A blockchain is a listing of time-sequenced, non-editable records, identifying all past transactions within a system starting from the time of an
5 initial block of transactions (which may be referred to as the genesis block) to the present. Blocks of transactions, containing detailed information pertaining to one or several valid transactions that have occurred within the network but were not included in any block of transactions previously appended to the blockchain are continually appended to the end of the blockchain.

10 Each block of transactions is identified by one or several unique block headers which contain a reference to all transactions included within the block of transactions as well as other pertinent information. As an example, reference to the various transactions incorporated into a block of transactions may be incorporated into a transaction block header by means of a Merkle–Damgård
15 construct, or by other means.

Transaction block headers and the information contained within them are the primary organizing principle of a blockchain. For instance, and among other items, transaction block headers may contain a cryptographically verifiable reference to the transaction block header of the immediately preceding valid
20 block of transactions in the blockchain. Thus such references cryptographically link all neighboring blocks together, each to each. This results in the organization of all blocks into a single uninterrupted chain of blocks, or blockchain, in much the same manner that a set of physical chain links joined each to the preceding form a physical chain.

25 Which blocks of transactions are to be appended to the end of the blockchain may be identified through a proof-of-work system, as described previously. Such a proof of work system may be based on cryptographic OWFs, as also described previously. OWFs are sometimes referred to as cryptographic hashing operations. Within the field of blockchain technology, they may
30 sometimes be referred to as hashing operations.

Within the field of blockchain technology, the proof-of-work verification process is generally carried out on transaction block headers, rather than on the transactions themselves. For this reason, while the proof-of-work process is being carried out, one block of transactions may be referenced by many

transaction block header even though ultimately the overwhelming majority of them will be discarded. Thus it is entirely normal for any group of transactions to be referenced by a large number of transaction block headers, all of which are correctly constructed in accordance to the protocol and all of which contain true
5 and accurate information, yet all of which are subtly different from each other.

Such transaction block headers are sometimes referred to as candidate transaction block headers. It is entirely normal for the number of candidate transaction block headers to be large, particularly if high-performance ASICs are involved. For example, a single high-performance ASIC may assemble, hash,
10 evaluate and discard several hundred billion candidate transaction block headers per second.

Generally, candidate transaction block headers will be subjected to one or several hashing operations, and the resulting hash value will be compared to certain preexisting criteria for validity. In the case of bitcoin, the particular
15 hashing operations used happen to be two subsequent applications of the 256-bit version of the 2nd version of the Secure Hashing Algorithm (SHA), as defined by the National Institute for Standards in Technology (NIST). This is sometimes referred to as SHA-256, and two sequential applications of it are sometimes referred to as double SHA-256. For simplicity, it may sometimes also referred
20 to as double SHA. However, it is worth nothing that this is merely the specific algorithm that happens to be used in the case of bitcoin; other blockchains can and do use other algorithms. Similarly, various embodiments of the technology described herein can use various different algorithms within the field of blockchain technology, while other embodiments are applicable to fields other
25 than blockchain technology.

As is common in other uses of proof-of-work verification systems, the comparison of candidate results with the validity criteria determines further processing. If the candidate transaction block header fails to pass the validity criteria, it is discarded and the system moves on to processing the next candidate
30 transaction block header. If the candidate transaction block header passes the validity criteria then it is considered a valid transaction block header.

When a valid transaction block header is found, a series of other operations are performed. Within the field of blockchain technology, such operations generally culminate in the appending of the valid transaction block

header to the end of the blockchain, along with the block of transactions it
references. Other operations, such as the dispensing of a reward to the party that
first reported the discovery of the valid transaction block header may also occur.
Subsequently, the search for a new valid transaction block header begins anew
5 and the process repeats.

The above generally describes certain aspects of the field of blockchain
technology, particularly some aspects relating to the nature of the blockchain, the
process for appending new blocks of transactions to it, the role and composition
of transaction block headers, the operation of the proof-of-work process, and
10 others. However, there many other aspects of blockchain technology have not
been described. As an example, and without limitation, the above does not
describe what may be characterized as the other half of this process, that is, the
process of generating transactions as opposed to the process of grouping and
processing transactions. Generally, the aspects of blockchain technology
15 described in greater detail are those most germane to the technology described
herein and some of its applications within cryptographic integrated circuits used
in this field, while aspects that are generally carried out by software running on
general-purpose microprocessors or are otherwise less germane to the
technology described herein it are described in lesser detail, or not at all.

20 The governing principles behind the operation of all aspects of a
blockchain are defined by a protocol, which is often referred to as a blockchain
protocol. This includes those aspect described in more detail above.
Conversely, blockchain protocols are designed to meet the particular needs of
the specific application the blockchain is designed to serve, as interpreted by the
25 protocol designer or designers. Such requirements may well vary from case to
case, and thus not all blockchain protocols need be the same. Indeed, some may
vary significantly.

The blockchain protocol that governs the operations of the blockchain
underlying bitcoin is sometimes simply referred to as the bitcoin protocol. The
30 bitcoin protocol is a seminal work in the narrow sense of the term; most if not all
of what we now refer to as blockchain technology is derivate work based on the
bitcoin protocol to one degree or another.

Each blockchain is maintained by a blockchain network, which in the
field of blockchain technology is sometimes referred to simply as the network.

In essence the definition of a blockchain network is straightforward; it is the collection of hardware and software that maintains a blockchain by implementing the policies defined by the protocol. In practice, blockchain networks can be complex. They are distributed heterogeneous combinations of a wide variety of devices, permanently or intermittently connected to the network, running different software written at various times by various people, performing different operations, and working together within a trustless system. Some blockchain networks may be extensive, and employ significantly advanced technology, including sophisticated cryptographic ASICs.

Generally, technology for solving the computationally intensive proof-of-work calculations involved in blockchain systems has evolved rapidly in recent years. For example, in the case of bitcoin transactions, the proof-of-work process involved in discovering valid transaction block headers was originally conducted utilizing software applications running on general-purpose processors. However, speed and efficiency are paramount in proof-of-work systems, including those used within the context of blockchain systems. Accordingly, bitcoin mining operations have moved toward specialized hardware solutions, including ASICs. ASICs provide profound increase in speed and efficiency arising from the line-level speed with which calculations may be conducted. As the name implies ASICs are designed and fabricated to perform one specific application, in this case the mathematical operations involved in implementing a specific cryptographic protocol.

The success of the bitcoin network has evidenced the secure nature of blockchain technology. Accordingly, the use of blockchains in other related fields has gained interest over time. However, ASICs are designed to narrowly match specific elements of the protocol within which they are to be implemented, specifically the elements of the protocol describing the particulars of the proof-of-work system defined by the protocol. For the bitcoin network, each ASIC is designed assemble at great speed block headers as described by the bitcoin protocol, subject them to two consecutive instances of the 256-bit version of the Secure Hashing Algorithm (SHA-256) protocol, and finally check the validity of the resulting 256 bit binary hash value by comparing it to a pre-determined validity test, which in the case of the bitcoin protocol is embodied as the difficulty level, as was discussed previously.

This limits the flexibility and opportunity to proliferate blockchain technology outside of the bitcoin network itself, or of networks based or “chained” on it to one degree or another, since high-performance proof-of-work implementations generally require the use of ASICs, and current-generation

5 ASICs are only able to implement one highly specific proof-of-work system. Moreover, due to the nature of the proof-of-work process any private blockchain that relies on the same proof-of-work system used by Bitcoin for its own proof-of-work process can easily be interfered with by third parties using commercially-available equipment designed to solve the proof-of-work system

10 defined by the bitcoin protocol.

Embodiments of the technology disclosed herein are directed toward the design, fabrication and utilization of application specific integrated circuits for cryptography-related applications. More particularly, various embodiments of the technology disclosed herein relate to ASICs having one or several

15 programmable transformation functions embodied as circuitry incorporated into the integrated circuit’s high speed datapath. By encoding transformation function as datapath circuitry, embodiments of the technology disclosed herein enable ASICs to implement any one of a very broad range of proof-of-work systems as selected by the user.

20 This, in turn, enables blockchain systems to be safely operated even in the presence of large numbers of other ASICs specifically designed to solve the proof-of work calculations involved in the cryptographic network transaction verification of blockchain transactions including, but not limited to, Bitcoin mining ASICs that may otherwise threaten the security or integrity of

25 blockchain-based systems not incorporating the technology disclosed herein.

Thus embodiments of the technology described herein enables the creation of a whole new generation of ASICs for specifically designed for blockchain applications that provide a presently unimaginable level of flexibility and security. Further, implementations of the technology described herein

30 further allow for such flexibility to be achieved in return for a near-negligible cost in processing performance.

Before discussing the embodiments of the technology of the present disclosure in detail, it is helpful to discuss the overall workflow surrounding the processing of the proof-of-work operations involved in the network transaction

verification of blockchain transactions by systems utilizing ASICs designed for this application.

Figure 1 illustrates an example of cryptographic network transaction verification system 100 of a general type commonly used for blockchain applications and incorporating an integrated circuit 103 designed for this application within which embodiments of the technology disclosed herein may be implemented.

For ease of discussion, the elements and overall operation of the example cryptographic processing system 100 will be described with respect to the bitcoin protocol, and the network supporting the bitcoin blockchain. Although described in this manner, one of ordinary skill reading the present disclosure would appreciate that the example cryptographic network transaction verification system 100 is applicable to uses other than the bitcoin network.

Examples of other applications can include both private and public blockchains used for applications other than those generally associated with bitcoin. Some examples of applications other than those associated with bitcoin that blockchain technology may be used for include, without exclusion: inter-bank settlement systems; stock exchange systems; commodity exchange systems; digital rights management systems; physical property rights registry systems; intellectual property registry systems; inventory control systems; fulfillment systems; package tracking systems; payroll systems; secure identification systems; use-of-funds tracking and transparency systems; accounting systems generally; payment systems generally; registry systems generally; financial systems generally; and others.

The example cryptographic network transaction verification system 100 includes an integrated circuit 103. In various embodiments, the integrated circuit 103 may include one or more processing components, including a configuration interface 104, a transaction block header framer 105, a buffer 106, a cryptographic hashing core 107, an evaluator 111, and a decision engine 112. In some embodiments one or more components may be combined into a single integrated circuit, as is illustrated in Figure 1, where all components represented within integrated circuit 103 are physically part of the same integrated circuit (that is, are all crafted within the same piece of silicon). In other embodiments some of them may be crafted out of different pieces of silicon and combined

together by means of connecting circuitry. Each of the components are discussed in detail with respect to the example cryptographic processing system 100.

5 A user 101 may configure one or more integrated circuit parameters through the configuration interface 104. Non-limiting examples of integrated circuit parameters that may be configured by the user 101 include: initialization; internal clock speed; or mode of communication with other networked systems; among others. In some embodiments, the user 101 may be a human communicating with the integrated circuit 103 via a terminal. In other
10 embodiments, the user 101 may be an automated process running on a system or network. In various embodiments, the configuration interface 104 may also be used by the user 101 to monitor operation characteristics of the integrated circuit 103. Non-limiting examples of operational characteristics that may be monitored and provided to the user 101 may include: current activities; operating
15 temperature; operating voltage; or fault status; among others.

Each candidate transaction block header that is subsequently processed by integrated circuit 103 is assembled by the transaction block header framer 105 using a combination of data generated by the framer itself and data obtained from the transaction and system data 102b. In various embodiments the
20 transaction and system data 102b is compiled by a prior process, the mining software 102a. In many cases mining software 102a is a piece of software running on a general purpose processor and communicating with the rest of the blockchain network via a connection to the public internet, which it utilizes to compile the transaction and system data 102b.

25 Certain qualities of the overall architecture of blockchain systems, particularly the well-designed compartmentalization between different aspects of the system, provides effective isolation between the operations of the integrated circuit 103 and the particulars regarding the operations of the mining software 102a. As a result, most particulars regarding the operations of the mining
30 software including where it physically resides, what procedures it uses to communicate with the rest of the network, what procedures it uses to compile network data into the transaction and system data 102b, and others are transparent to the operations of the integrated circuit 103. Similarly, whether the network that the mining software 102a connects to happens to be the bitcoin

network or a different network responsible for the maintenance of a different blockchain is also transparent to the integrated circuit 103, and to various embodiments of technology described herein.

The transaction and system data 102b may include, for example: system
5 date and time; protocol version; a Merkle–Damgård construct representing the group of individual transactions that included in the transaction block; a unique identifier of the most recent block to be appended to the blockchain; among others. Upon receiving the transaction and system data 102b, the transaction block header framer 105 further elaborates it in order to generate one or more
10 candidate transaction block headers.

In various embodiments, the transaction block header framer 105 may parse the transaction and system data 102b and add to it certain other elements, such as a sequential counter in order to generate a series of transaction block headers that are all validly formed and all appropriately represent the same set of
15 transactions, but are all slightly different from each other.

In various embodiments, the rate at which the transaction block header framer 105 may produce transaction block headers that all accurately represent a block of transactions as received from the transaction and system data 102b but are all slightly different from each other may be fast, in the range of several
20 billion or several hundred billion individual candidate transaction block headers per second. Depending on the implementation, a buffer 106 may store created candidate transaction block headers awaiting processing by the hashing core 107.

Each integrated circuit 103 may contain one or more hashing cores or other cryptographic processing cores. For simplicity, the example cryptographic
25 processing system 100 is shown including a single hashing core 107. In various embodiments, multiple parallel hashing or other cryptographic processing cores may be implemented, as illustrated in Figure 7. The hashing core 107 includes one or more hashing blocks. In the example illustrated in Figure 1, the hashing core 107 includes two hashing blocks 108, 110. A hashing block may be
30 configured as a set of circuitry that executes the mathematical operations defined by the applicable hashing standard. In various embodiments, the hashing core 107 may embody a common hashing standard, such as for example the Secure Hashing Algorithm (SHA). In some embodiments, the 256-bit version of SHA (SHA-256) may be used. In other embodiments, the hashing core 107 may

include a single hashing block. Various embodiments may include greater than two hashing blocks.

In the illustrated embodiment of Figure 1, each hashing block 108, 110 is designed to execute a sequence of mathematical operations in accordance with the SHA-256 standard. The hashing core 107 processes each candidate transaction block header in accordance with the cryptographic standard implemented. In some embodiments, a first hashing block 108 may accept the candidate transaction block headers generated by the transaction block header framer 105 as an input and subject them to a mathematical operation defined by a standard hashing algorithm. The first hashing block 108 outputs a first hash value 109 for each candidate transaction block header it receives from the transaction block header framer 105. Each first hash value 109 is associated with a given candidate transaction block header, acting as a type of “digital signature.” In various embodiments, the first hash value 109 may be further processed by a second hashing block 110. This is a representation of the double hashing operation, such as that utilized within the bitcoin network. In some embodiments, the second hashing block 110 may implement the same cryptographic operation as the first hashing block 108. In various embodiments, the second hashing block 110 may implement a different cryptographic operation than the first cryptographic hashing block 108.

The output of the hashing core 107 is compared with the pre-existing criteria for identifying a valid block. In various embodiments, the output of the hashing core 107 may be the output from the second hashing block 110. An evaluator 111 of the integrated circuit 103 takes the final hash value output by the hashing core 107 for each candidate transaction block header and checks to see whether the processed output meets the pre-determined validity criteria. In the particular case of the bitcoin network, the validity criteria is expressed by a certain numeric value (often referred to as the difficulty level) that the numeric value of the 256 bit number produced by the final hash output may not exceed if it is to be considered valid. Thus if the numeric value or the final hash exceeds the difficulty level by any amount, the candidate transaction block header fails the validity test, and if it does not exceed the difficulty level it passes the validity test. In blockchain systems other than the one associated with the bitcoin network, some embodiments may employ the same criteria for determining

validity, while other embodiments may employ different criteria for determining validity.

If the evaluator 111 determines that the final hash value does not meet the validity criteria, the decision engine 112 may discard the candidate transaction block header associated with the processed output evaluated by the evaluator 111. If the evaluator 111 determines that the final hash value does meet the validity criteria then the decision engine 112 will report that fact externally via a success report 113b transmitted to a success report recipient 113a. A success report 113b along with other information associated with it, such as the transaction block headers associated with it and other items, is the cryptographic proof-of-work that the system as a whole is designed to produce.

In various embodiments the success report recipient 113b may be the same as the mining software 102a. In other embodiments the success report recipient 113b may be a different process than the mining software 102a. Upon receipt of the success report 113b, the success report recipient 113a may take a number of subsequent actions. In various embodiments, such actions generally relate to the communication of the discovery of a valid transaction block header to the rest of the network, and generally culminate with the appending of a new transaction block to the end of the blockchain and the start of the search for the next valid block of transactions. In the specific case of the bitcoin network, such actions may additionally include other aspects such as the dispensing of a reward to the first party to report the discovery of a new valid transaction block header, and others.

However, and as was the case with the prior description regarding the operations mining software 102a, it is worth noting that exact nature of the success report recipient 113b and the specifics regarding what actions it may or may not take following the receipt of the success report 113a are transparent to various embodiments of the technology described herein. Similarly, whether the success report recipient 113b communicates with the network associated with the maintenance of the blockchain underlying bitcoin or the network associated with the maintenance of a different blockchain is transparent to some embodiments of the technology described herein.

With the basic workflow of blockchain technology described with respect to an example implementation, embodiments of the technology disclosed herein

will be discussed with reference to the example cryptographic processing system 100 of Figure 1. For ease of discussion, the embodiments will be discussed with respect to bitcoin and the blockchain implementation associated with it. As discussed above, one of ordinary skill in the art reading this description, however, would appreciate that the embodiments described herein are applicable to many other related fields.

Figure 2 illustrates an example transform-enabled integrated circuit 203 in accordance with embodiments of the technology disclosed herein. The example transform-enabled integrated circuit 203 is implemented within a cryptographic processing system 200, similar to the system 100 described with respect to Figure 1. The transform-enabled integrated circuit 203 may include a configuration interface 204, a transaction block header framer 205, a buffer 206, a cryptographic processing core 207, an evaluator 211, and a decision engine 212, similar to the components discussed above with respect to the integrated circuit 103 of Figure 1.

As discussed previously, embodiments of the technology disclosed herein places a transformation function embodied as circuitry within the cryptographic datapath of an integrated circuit 203. In the example illustrated in Figure 2, the cryptographic processing core 207 of the transform-enabled integrated circuit 203 includes two hashing blocks 208, 210 and a programmable transformation block 215. Hashing blocks 208, 210 may be implemented to include a hashing or other cryptographic function such as, for example, circuitry to execute a hashing process in accordance with the SHA-256 standard.

In the illustrated example, the programmable transformation block 215 is a block of electronic circuitry specifically designed to be suitable for being integrated directly into the line-speed datapath of a high-performance cryptographic integrated circuit, but yet remain capable of being programmed to perform any one of a vast range of possible mathematical transform operations on input data it receives from circuitry prior to it on the integrated circuits datapath and output the resulting transformed data to circuitry subsequent to it on the integrated circuits datapath at full line speed.

In some embodiments such as that illustrated in Figure 2 the programmable transformation block 215 is integrated into the datapath of the cryptographic processing core 207. The ASIC in this example is configured

such that all components within cryptographic processing core 207 are arranged along a single highspeed datapath, there is a single point of input and a single point of output at either end of the datapath, and there is no external access to signals moving between the components comprising the cryptographic

5 processing core 207.

The integration of the programmable transformation block 215 into the high-speed datapath within cryptographic processing core 207 allows the cryptographic processing core 207 to operate at line speed, and thus the circuit as a whole suffers from very little degradation in performance when compared to a
10 cryptographic processing core not including the programmable transformation block 215. This is achieved by embodying a transformation function as datapath circuitry (i.e., the programmable transformation block 215), in a manner that will be disclosed in greater detail below.

Similarly, the arrangement of circuitry within cryptographic processing
15 core 207 into a unified datapath protects the configuration key 214 (discussed in more detail with respect to Figures 5A, 5B, and 6) from detection. The cryptographic processing core 207 contains a single input point and a single output point at either end, and has no external access to signals moving between the components comprising the cryptographic processing core 207. Therefore,
20 the circuitry within the cryptographic processing core 207 (e.g., the programmable transformation block 215) is protected against disclosure or discovery of the configuration key 214 embodied in the programmable transformation block 215 by comparing the inputs and outputs produced by the different components arranged along the datapath.

25 In the specific case of the example embodiment illustrated in Figure 2, the datapath within cryptographic processing core 207 is arranged in such a manner that the single input point for all data processed by the cryptographic processing core 207 is the first hashing block 208. From there, data such as transaction block headers proceeds along a datapath that takes it through the
30 programmable transformation block 215, and then through the second hashing block 210. The second hashing block 210 is the single output point, and subsequently data leaves cryptographic processing core 207 and proceeds to the evaluator 211. Placing the programmable transformation block 215 before at least one hashing block (in this example, before hashing block 210) protects

discovery of the configuration key 214 embodied in the programmable transformation block 215 via analytical techniques. For example, such placement protects discovery of the configuration key 214 through insertion of data known to produce a certain result upon being subjected to a particular set of cryptographic processes (e.g., double SHA-256 hashing), and then comparing the result produced by the transform-enabled cryptographic processing core 207 to deduce the configuration key 214.

In other embodiments as will be discussed in greater detail below with reference to Figure 3, the programmable transformation block 215 may be placed prior to the first hashing block 208, such that the single input point of the cryptographic processing core 207 is the programmable transformation block 215.

In various embodiments, the programmable transformation block 215 may be programmed at a time subsequent to the manufacturing of the transform-enabled integrated circuit 203 to consistently perform any one of a wide range of possible mathematical transformations on the data flowing through it at line speed due to its location on the datapath of the integrated circuit 203.

In some embodiments, the mechanism by which the programmable transformation block 215 is able to accept and retain such programming may be via a type of non-volatile read-only memory (NVRAM), including but not limited to Flash memory or other types of non-volatile memory. In various embodiments, the means of configuring the programmable transformation block 215 may be via one time programmable (OTP) circuitry components, including but not limited to micro-fuses or other types of OTP circuitry components. Micro-fuses are generally utilized in simple aspects of integrated circuit fabrication, such as writing manufacturing information onto integrated circuits during manufacture, or remedying faulty memory banks identified during testing. For ease of discussion, the technology of the present disclosure will be discussed with reference to a programmable transformation block 215 comprising micro-fuses, which will be discussed in greater detail with respect to Figures 5 and 6.

Various embodiments of the transform-enabled integrated circuit 203 may enable configuring the programmable transformation block 215 by means of a configuration key 214. In various embodiments, the configuration key 214

may be coded into the programmable transformation block 215 by the user 201 acting through the configuration interface 204. In such embodiments, the technology disclosed herein contemplates several different parties potentially taking the role of the user 201 for the purpose of encoding the configuration key 214 into the programmable transformation block 215. Such parties may include, for example: at the factory by the IC manufacturer; by one or several end users; or by some combination of the above.

In some embodiments where the user 201 is a single party, the user 201 may provide a configuration key 104 defining what transformation is to be applied to each bit comprising the data received by the programmable transformation block 215. For example, where the programmable transformation block 215 is designed for a 256-bit system, the user 201 may input a configuration key 214 defining what transformation, if any, the programmable transformation block 215 applies to all 256 bits of data received. Therefore that party will have the freedom to program any one of the 2^{256} mathematical operations that the programmable transformation block 215 is capable of being configured to perform using a 256-bit key.

Similarly, and continuing with the 256 bit example above, in embodiments where the user 201 comprises one or more parties, the configuration of the programmable transformation block 215 may be carried out by each of the one or more users 201 each of whom contributes a part of the configuration key 214. Such a scheme may sometimes be referred to as a multi-key scheme, and the resulting key may sometimes be referred to as a multi-part key. In some embodiments, each of the one or more users 201 may configure what transformation, if any, the programmable transformation block 215 applies to a subset of the bits of data it receives, while in other embodiments each of the one or more users may contribute a partial key that is subsequently processed into a key in a manner that does not allow any one user to determine what transformation, if any, the programmable transformation block 215 is to apply to any one specific bit or bits of data it receives.

In various embodiments of the technology described herein where the programmable transformation block 215 is placed in front of at least one block of circuitry implementing a cryptographic OWF such as the second hashing block 210 implementing a standard cryptographic algorithm in the illustration,

the combined effect of the interaction of the programmable transformation function 215 and certain mathematical properties of the cryptographic algorithms involved, particularly their nature as OWFs that are hard to revert (here the terms “hard” and “easily” may be understood in the sense of computational complexity theory, particularly polynomial time theory) produce certain specific results. Specifically, the combined effect is that even a slight change introduced by the programmable transformation block 215, such as single bit change, will result in a wholesale transformation of the bits contained in the final value calculated by the hashing core 207 that is not easily deciphered, not easily distinguishable from noise, and not easily replicable by a party lacking full prior knowledge of the configuration key 214 and specific aspects of the design of the programmable transformation function 215, but yet is fully consistent and easily replicable and thus verifiable by a party with such knowledge or access to the means to utilize them in calculation even while lacking knowledge of them (such as, a party in possession of the same ASIC, or another ASIC incorporating the technology described herein and programmed to same configuration key 214 within its datapath circuitry).

The effects of the application of the technology described herein outlined in the prior paragraph have uses within the field of blockchain technology that enable a variety of innovative and valuable applications across multiple different fields. Such applications have been briefly aluded to previously and will be further discussed below.

In some embodiments, the programmable transformation block 215 may be configured to enable an end user to program a variety of transformation schemes, such as transposition schemes that transpose the position within the input and output of certain specific bits, while leaving others unchanged. In various embodiments, the programmable transformation block 215 may be configured to perform a direct bit inversion transformation scheme where some bits are inverted while others remain unchanged.

For ease of discussion, the technology of the present disclosure will be discussed with reference to a direct bit inversion transformation scheme where a 256-bit configuration key 204 determines what transformation is to be applied to each one of the 256 bits received by the programmable transformation block 215 by means of the following code: bits received at positions where the 256 bit

configuration key contained a value of "0" are to be treated in one manner (for instance, be left unchanged), while those received at positions where the 256-bit configuration key contains a value of "1" are to be treated in a different manner (for instance, be inverted). Such a transformation is discussed in greater detail in the descriptions relating to Figures 5A, 5B and 6.

Following the production of the final hash value by the second hashing block 210, the evaluation process carried out by the evaluator 211 and the decision engine 212 in Figure 2 is similar to that described with respect to Figure 1. The first step in the process is that the evaluator 211 will determine whether the final hash value produced by a particular transaction block header does or does not meet the validity criteria.

If the final hash value does not meet the validity criteria (as will generally be the case in the vast majority of cases), the final hash value and the transaction block header that produced it will both be discarded. If the final hash value does meet the validity criteria, the final hash value and the transaction block header that produced the final hash value will be passed to the decision engine 212. After receiving the success indication, the decision engine 212 may then issue a success report 213b to the success report recipient 213a. Additionally, and also discussed with respect to Figure 1, the success report recipient 213a may or may not be the same as the mining software 202a.

What is different is that the inclusion of the programmable transformation block 215, its subsequent programming by the user with a certain configuration key 214, and the interaction of the resulting transformed data with the second hashing core 210 may have caused a change in the final hash value calculated by the hashing core 203 for one or more than one transaction block headers generated by the transaction block header framer 205 and passed on the evaluator 211. Further, and as was discussed previously, such changes have certain unique and useful mathematical properties. Given that the task of the evaluator 211 is to compare the hash values received from the hashing core 203 with certain pre-determined validity criteria and on that basis evaluate whether the hash values pass or fail the validity test, it follows that a change in the hash value received by the evaluator 211 may result in a change that the evaluator makes as to the validity or non-validity of the hash value received.

Further given that the operation of the proof-of-work system is predicated upon the discovery of a valid proof of work in the form of a hash value that meets the validity criteria, additionally given that the operation of the cryptographic network transaction verification system is predicated upon the proof-of-work system, additionally given that the construction of the blockchain by means of appending new blocks to the end of the blockchain is predicated upon the cryptographic network transaction verification system, and additionally given that the changes introduced by the transformation function are fully coherent and easily replicable even if they may appear indistinguishable from noise to an external observer, then it follows that, if applied consistently throughout a blockchain network, a change, however slight, in the configuration key 214 would result in a systemic change of the operations of the blockchain system.

Therefore, certain embodiments of the technology disclosed herein allow for the creation of multiple orthogonal blockchains, each one which assigns validity to an entirely different set of transaction block headers according to the configuration key 214 that each user 201 elects to apply to the programmable transformation block 215.

In various embodiments there may be one or several null configuration key 214 values which, if programmed into the programmable transformation block 215 will cause the programmable transformation block 215 to effect no change in the information passing through it, and thus cause the presence of the programmable transformation block 215 to effect no change in the final hash value that is calculated by the cryptographic processing core 207 for each of the transaction header blocks received from the transaction block header framer 205 and passed on to the evaluator 211. In some embodiments, a null configuration key may be represented by a string of zeroes of the same length as the length of the configuration key, which in certain embodiments may be 256 bits in length.

In various embodiments incorporating one or several null keys, the overall effect of programming the programmable transformation function with a null key is such that the set of transaction block header hashes considered valid by the evaluator would not be altered, rendering the output from the cryptographic processing core 207 functionally identical to the output from the hashing core 107 discussed with respect to Figure 1. That is, a transform-

enabled integrated circuit incorporating the technology described herein and programmed with a null key would behave in produce the same results as an integrated circuit not incorporating the technology described herein, and thus be suitable to perform proof-of-work calculations within the context of the same

5 proof-of- work systems, including but not limited to the cryptographic network transaction verification of bitcoin transaction (also referred to as bitcoin mining). This aspect of the technology described herein has commercial implications, as it enables the use of integrated circuits incorporating embodiments of the technology described in the present disclosure to be useful for the purpose of

10 mining bitcoin, in addition to being useful for a wide range of other applications.

In such embodiments, and in other embodiments not incorporating the figure of a null configuration key, the programming of the programmable transformation block 215 with a configuration key 214 different from the null configuration key will cause a change in the final hash value calculated by the

15 cryptographic hashing core 207 for each of the candidate transaction block headers produced by the transaction block header framer 205 and passed on to the evaluator 211. This in turn will cause a change in what subset of all candidate transaction block headers are deemed to be valid by the evaluator 211 and thus further communicated to the decision engine 212, which will then

20 communicated the outside of the integrated circuit 203 in the form of a success report 213b.

Further, and as described above, due to the interaction of the programmable transformation function 215 with certain mathematical functions of OWFs such as most standard modern cryptographic hashing functions, any

25 change (including a single digit) in the input of a cryptographic hashing function causes a wholesale change in the resulting hash value that is not easily distinguishable from noise.

As noted above, in some embodiments the programmable transformation block 215 may be placed prior to the first hashing block, such that the single

30 input point of the cryptographic processing core is the programmable transformation block. One such example of this is illustrated in Figure 3. Referring now to Figure 3, in this example arrangement, the transformation function embodied by the programmable transformation block 315 is applied to the transaction block headers prior to any hashing operation being applied by

hashing blocks 308, 310. Although located in a different location in the example embodiment of Figure 3, the location of programmable transformation block 315 within the datapath still meets the condition that it be placed prior to at least one hashing block (in that case, it is placed in front of the first hashing block 308 and the second hashing block 310). Thus, the embodiment illustrated by Figure 3 is functionally similar to the embodiment illustrated in Figure 2, and provides the same essential qualities, both as regards the properties of the results it produces and as regards the protection it provides against the discovery of configuration key 314 embodied within the programmable transformation block 315 via analytical techniques, such as described above.

In some embodiments, a single integrated circuit may incorporate more than one programmable transformation block 215. Figure 4 illustrates such an example transform-enabled integrated circuit 400. In such cases, embodiments may be implemented to provide more than one user with the ability to program a full configuration key.

Such embodiments implement a process that shares some characteristic with both cascade ciphers and multi-signature schemes but is distinct from both. Specifically, cascade ciphers involve the use sequential application of ciphers, that is, processes that encrypt data in such a manner that at a later time it may be deciphered, that is, be rendered legible again. Various embodiments of the technology described herein as illustrated in Figure 4 incorporate a similar concept of cascading, that is sequentially applying, cryptographic operations but do not involve the use of ciphers of any kind. Rather, they involve the sequential application transformations and OWFs (such as cryptographic hashing processes). OWFs are distinct from ciphers, among other aspects, in that their defining characteristic is that they be undecipherable rather than decipherable. Similarly, various embodiments of the technology described herein as illustrated in Figure 4 in that their end product is a string of characters representing what may be thought of as multiple digital signatures applied to a digital document. However, the defining characteristic of digital signatures of any kind is that they may be validated by a party different from the signer, or one holding the signer's key. That is not the case of various embodiments of the technology described herein, which in that context may be thought of as a signature that may only be verified by the signer. Further, multi-signature schemes are often designed in

such a way that each signature be distinct from the others, and may be individually verified. That again, is not the case for various embodiments of the technology described herein, which are specifically designed in such a manner that neither original signer is able to validate the signature without the presence
5 of the other.

Thus, various embodiments of the technology described herein are distinct from both cascade ciphers, digital multi-signature schemes and other present cryptographic technologies. While lacking a common name due to its novelty, such a system may be described as a multi-key cascading transformed
10 one-way function system.

In some embodiments where the cryptographic hashing algorithm embodied as circuitry in at least the second hashing block 410 may be a proven standard cryptographic hashing algorithm including, without limitation: SHA-2 in its various implementations; Keccak/SHA-3 in its various implementations;
15 Skein in its various implementations; Grøstl in its various implementations; JH in its various implementations; and others, or be a non-standard hashing algorithms that, despite being non-standard, is still a OWF that is hard to reverse (where the term “hard” is understood in the sense of polynomial time theory). In such embodiments, the technology described herein as illustrated in Figure 4
20 enables the implementation of a secure multi-key cascading transformed one-way function system, as described previously.

In some embodiments, both the first and second programmable transformation blocks 418, 415 may be programmed by first and second configuration keys 417, 414 that are 256 bits in length, while in other
25 embodiments either one or both may be of a different length depending on the specifics of the implementation.

In various embodiments of the technology described herein also illustrated by Figure 4, a first party that may be referred to as the primary authority may use the configuration interface 404 to program the first
30 configuration key 417 which may referred to as the primary key into the first transformation block 418, which may be referred to as the primary transformation. Separately a second party that may be referred to as the secondary authority may use the configuration interface 404 to program the second configuration key 414 which may referred to as the secondary key into

the second programmable transformation block 415, which may be referred to as the secondary transformation.

In certain embodiments of the technology described herein also illustrated by Figure 4, one or both the primary authority and the secondary authority may be composed of several different parties. In some such
5 embodiments the primary party as a whole may program the primary key as a multi-part key, while the secondary party as a whole may program the secondary key as a multi-part key. In other embodiments, either or both keys may be jointly configured by both the primary and the secondary parties as multi-part
10 keys. In certain embodiments, the key programmed by the primary party, the secondary party, or both may be null keys. In other embodiments, the first and second programmable transformation blocks 418, 415 may be programmed by a single configuration key inputted by the user 401 through the configuration interface 404.

15 In other embodiments of the technology described herein not illustrated in Figure 4, the process may be generally similar except that the configuration interface used by the primary authority may be different from the configuration interface used by the secondary authority. In some such embodiments, the configuration interface used by the primary authority may be created in such a
20 manner as to be accessible during the manufacturing process of the integrated circuit but be inaccessible in the resulting finished parts, while the configuration interface used by the secondary authority may be structured in such a manner as to be accessible both during the manufacturing process and in the resulting finished parts. In other embodiments of the technology described herein, the
25 relative accessibility or inaccessibility of each configuration interface may be structured differently.

In some embodiments of the technology described herein also not illustrated in Figure 4, the process may be generally similar to that illustrated in Figure 4 except that the design may incorporate more than two hashing rounds,
30 or more than two programmable transformation functions, or both.

Figure 5A illustrates an example programmable transformation block 515 configuration prior to being coded in accordance with embodiments of the technology disclosed herein. The programmable transformation block 515 includes sets of programmable circuitry defining the transformation function

represented by the configuration key 514 (illustrated in Figure 5B). In various embodiments, the programmable transformation block 515 may be configured to enable one of a plurality of transposition operations, whereby one or more bits of input data (e.g., 1a1, 1b1, etc.) is transposed with another bit of the input data to generate a modified output data (e.g., 5a1, 5b1, etc.). In other embodiments, the programmable transformation block 515 may be configured to enable one of a plurality of direct bit inversion, or bit flipping, operations, whereby one or more bits of input data are flipped to generate modified output data.

Various embodiments utilizing a direct bit inversion transformation scheme as described herein may be implemented to take advantage of the fact that a 256 bit binary configuration key provides a succinct means to enable access to the full key space provided by the programmable transformation function 215. That is, 256 bits is the minimum length necessary to enable the user to specify which one among 256 distinct transformations is to be performed by the programmable transformation function 215 on the first hash value 209. Using a direct bit inversion transformation scheme may also enable the use of a minimal amount of new circuit elements to embody of the transformation function as datapath circuitry. This is important because the fact that the transformation function is embodied as datapath circuitry means that any additional circuitry placed on the datapath will operate at line speed and result in an overall degradation of the performance of the transform-enabled integrated circuit 203 as a whole.

In various embodiments, the configuration of the programmable transformation block 515 may add as little as a single NOT gate to the datapath circuitry of the transform-enabled integrated circuit. By comparison, each hashing block may, depending on the details of their design, contain approximately 4,000 gates. Therefore, the programmable transformation block 515 may add a very modest 128 gates (assuming a 256-bit key) to the cryptographic datapath previously containing approximately 8,000 gates (assuming a double SHA-256 hashing process). Accordingly, the degradation in performance introduced by the programmable transformation block 515 in various embodiments may be very limited, so much so that in practice the difference in performance would be overshadowed by other factors to such a degree that it would be difficult to accurately quantify in isolation.

A direct bit inversion transformation scheme may also provide a straightforward means to disable all effects of the programmable transformation 215 by simply setting all values in the 256 bit configuration key to zero. Such a key may be referred to as a null key. One result of this it simplifies the process of configuring a transform-enabled integrated circuit 203 so that it operates in a manner undistinguishable from that of a comparable integrated circuit not incorporating the programmable transformation function 215. The practical result of this is that integrated circuits incorporating the technology described herein may easily be configured to operate in the same manner as standard bitcoin mining ASICs and be used to mine bitcoins with no particular difficulty, (aside from and in addition to being able to operate in manners that are not replicable by bitcoin mining ASICs not incorporating the technology described herein).

For ease of discussion, Figures 5B and 6 will be discussed with reference to a direct bit inversion configuration.

The coding of the programmable transformation block is illustrated in Figure 5B. The example programmable transformation block 520 is shown being coded in accordance with the configuration key 514. To indicate the configuration key 514 embodied within the programmable transformation block 520, shaded boxes are used to indicate that a micro-fuse has been disabled (i.e., stopping data from flowing through the disabled micro-fuse). For example, in the illustrated example, when micro-fuse 3a2 is disabled, the input bit 1a2 flows through the programmable transformation block 520 unchanged. When micro-fuse 2b2 is disabled, the input bit 1b2 is inverted by the bit flipper 4b2 as it flows through the programmable transformation block 520.

Figure 6 illustrates an example function coding process in accordance with embodiments of the technology disclosed herein. Stage 1 represents a programmable transformation block 15 in an uncoded state, similar to the programmable transformation block 515 of Figure 5A. As illustrated, the programmable transformation block 15 is configured for a six-bit message. In other embodiments, the programmable transformation block 15 may be configured for input strings of any length, such as a length of 256 bits.

At Stage 2, the programmable transformation block 15 is coded, embodying the configuration key 14 as provided by the user. As illustrated, the

configuration key 14 comprises the string "011000." The corresponding micro-fuses are disabled according to the configuration key 14, as illustrated by the shaded boxes within the programmable transformation circuit 15. At Stage 3, a lock fuse 16 may be disabled following coding of the programmable transformation block 15, protecting the configuration key 14 from discovery. A lock fuse 16 may be disposed on the programming circuitry through which the programming transformation block 15 is programmed by the user. By disabling the lock fuse 16, the cryptographic processing core including the programming transformation block 15 returns to having a single input point and single output point, thereby providing the type of protection discussed in detail above with respect to Figure 2.

In various embodiments, multiple cryptographic processing cores may be included in parallel within the same integrated circuit. Figure 7 is a block diagram of a basic cryptographic processing system 700 including multiple cryptographic processing cores. The integrated circuit 703 and corresponding processes and components are similar to those discussed above with respect to the basic system design of Figure 1. Transaction block headers generated by the transaction block header framer 705 may be routed to one of the parallel multiple cryptographic processing cores 707, 708, 709, 710. In this way, multiple transaction block headers may be processed in parallel, increasing the overall speed of the system.

Where multiple cryptographic processing cores are implemented, some embodiments may implement the programmable transformation block to provide the ability to generate unique blockchains as discussed above. Figure 8 illustrates an example transform-enabled integrated circuit 803 having multiple cryptographic processing cores, in accordance with embodiments of the technology disclosed herein. The transform-enabled integrated circuit 803 is similar to the basic integrated circuit 703 discussed with respect to Figure 7. A programmable transformation block 815 is placed within the datapath circuitry of the transform-enabled integrated circuit 803. Although pictured as being outside of the cryptographic processing cores 807, 808, 809, 810, the programmable transformation block 815 is actually located between the two hashing blocks of the cryptographic processing cores 807, 808, 809, 810. For ease of viewing, the programmable transformation block 815 is shown as being

outside to illustrate that the same configuration key 814 may be embodied within each of the cryptographic processing cores 807, 808, 809, 810 in various embodiments. In other embodiments, each cryptographic processing core may have its own programming circuitry, as illustrated in Figure 9. In this way, one
5 or more users 901 may program one or more configuration keys 914a, 914b, 914c, 914d.

In various embodiments, the integrated circuit may include additional, non-cryptographic processes. Figure 10 illustrated an example dual-purpose integrated circuit 1006 implemented within processing system 1000 in
10 accordance with embodiments of the technology disclosed herein. The dual-purpose integrated circuit 1006 includes a cryptographic portion including a cryptographic processing core 1004. The cryptographic processing core 1004 may be configured in a similar manner as the cryptographic processing core 207 discussed above with respect to Figure 2. In Figure 10, the user 1001 may
15 configure the programmable transformation block of the cryptographic processing core 1004 via a configuration key 1003 inputted through the configuration interface 1002.

The general-purpose portion 1007 of the dual-purpose integrated circuit 1006 may include components designed for general purposes, such as a central
20 processing unit or a system-on-chip (SoC) processor designed to perform a number of additional functions (e.g., graphics, audio, communications processing, etc.). Spare memory cells may also be included in the general-purpose portion 1007 in some embodiments to enable swapping out faulty memory components. In some embodiments, the general-purpose portion 1007
25 may have its own interface 1009 through which a user 1008 may configure the components of the general-purpose portion 1007. In addition, in some embodiments the user 1008 may write system information 1010 into permanent memory of the general-purpose portion 1007. In various embodiments, the configuration interface 1002 and the interface 1009 may be a single set of
30 electrical circuitry, and the user 1001 and user 1008 may be the same entity. A bidirectional communication link 1005 may connect the cryptographic processing core 1004 to the general-purpose portion 1007, enabling the CPU or SoC to send tasks to be performed by the cryptographic processing core 1004, and receive results of calculations. The integrated circuit 1006 may interact with

local resources 1011 and external resources 1012. Internal resources 1011 may be, but are not limited to, the user 1001 or 1008, local memory, or software applications running locally with the dual-purpose integrated circuit 1006. External resources 1012 may include external memory or other nodes within the network on which the integrated circuit 1006 is connected.

Occasionally, the dual-purpose integrated circuit 1006 may be connected to other nodes in an untrusted network, such as through an unsecured wireless or wired connection. In some embodiments, the general-purpose portion 1007 may be configured to conduct several processes designed to securely identify an integrated circuit containing a transformation block in accordance with the technology presented herein, over an untrusted network. A string generator 1013 may access system information 1010 written into the integrated circuit 1006 by the user 1008, and combine such system information 1010 with transaction information, such as transaction and system data 102b discussed with respect to Figure 1. The string generator 1013 may then send the data to the message generator 1014.

The message generator 1014 may send the data to the cryptographic processing core 1004 over the bidirectional communication link 1005. In this way, the data may be processed to generate a transform-coded string, which the cryptographic processing core 1004 may then send back to the message generator 1014. The message generator 1014 may then generate a network message 1015. The network message 1015 may include the initial data outputted by the string generator 1013. In various embodiments, additional information may be included in the network message 1015, including but not limited to: system date and time; previously-received external messages; or the transform-coded string; among others. The network message 1015 is then sent to an IO component 1016. In various embodiments, the IO component 1016 may include local communication protocols, external communication protocols, or a combination thereof.

Referring now to Figure 12, an illustration is shown of what may be termed a transform-enabled cryptographic result validator. The illustration shows an implementation of the technology described herein where an integrated circuit 1203 containing a programming and configuration interface 1204, a transform-enabled hashing core 1205 and configuration key 1206. Two users

1201, 1202 access the integrated circuit, with user 1201 accessing the programming and configuration interface 1204 and user 1202 accessing the transform-enabled hashing core.

As can be seen from the diagram, the circuit illustrated in Figure 12 is generally simpler than the circuits illustrated in prior diagrams. In some embodiments, the circuit illustrated in Figure 12 is additionally also a lower power consumption and lower-performance circuit than some of the embodiments illustrated in prior diagrams. In certain embodiments the overall purpose of this circuit is to enable the verification of results produced by some embodiments of the technology disclosed herein, such as some of the embodiments illustrated in prior diagrams. The general mode of operation is that user 1 1201, may use the programming and configuration interface 1204, in order to both configure various parameters of the operation of the integrated circuit such as clock speed, external communications, and others and to program a one or several configuration keys 1204, into the programmable transformation function or functions within the transform-enabled hashing core 1205 where they will be embodied as datapath circuitry. In some implementations certain matters such as: the manner in which the programming is carried out; the mechanism by which the configuration key or keys 1204 are embodied as datapath circuitry; the mechanisms used to protect the key once it is programmed into the programmable transformation function; and other matters, are generally carried out in a manner that is generally similar to manner in which they are carried out in some other implementations illustrated by prior diagrams.

In some implementations, following the programming of the programmable transformation block within the transform-enabled hashing core 1205, user 2 may simply enter some input values which are then directly communicated as input data to the transform-enabled hashing core 1205 subsequently, upon completion of the calculations, the transform-enabled hashing core 1205 will then return the final hash value calculated to user 2. If the implementation of the technology described herein is generally the same in the transform enabled hashing core 1205 as in a different transform-enabled circuit that previously carried out such operations, the input data such as, for example, a transaction block header is the same in both cases, and the configuration key or keys programmed into both are also the same in both cases,

then the final hash value calculated by both integrated circuits will also be the same. Thus the validity of certain items which may, for example, include transaction block headers from the blockchain, other transaction block headers, other candidate transaction block headers, and others may be validated. Further
5 by using a circuit such as some embodiments of the circuit illustrated in Figure 12, such a verification may be carried out without user 2 requiring knowledge of the keys or keys programmed into the programmable transformation block.

In some embodiments, the user may be a person, while in other embodiments the user may be an automated process such as wallet software,
10 mining software or other kinds of automated processes. In certain embodiments user 2 may also have access to the configuration of the various operating aspects of the integrated circuit as described above. In certain embodiments there may be a single interface for the configuration of the various operating aspects of the integrated circuit as a whole, the programming of keys and the obtaining of final
15 transform-enabled hash values calculated on the basis of data supplied by the user, while in other embodiments some or all of those functions may be separate. In certain embodiments, the integrated circuit illustrated by Figure 12 may be part of a larger computing system, such as a mining system, a hardware wallet, a secure token or dongle, or others. In some embodiments, various
20 implementations of the integrated circuit illustrated in Figure 12 may be part of a system incorporating one or more of integrated circuits containing other implementations of the technology described herein, such as some implementation illustrated in previous figures.

In certain embodiments, various implementations of the integrated circuit
25 illustrated in Figure 12 may be physically integrated into the same silicon as other embodiments of the technology described herein. In some such embodiments, the integrated circuit illustrated in Figure 12 may additionally be further connected to other embodiments of the technology described herein. For example, in various such cases the integrated circuit illustrated in Figure 12 may
30 have a shared access to the programmable transformation block as other circuits within the same integrated circuit, in a manner that may be similar to how multiple different hashing cores have access to a single transformation function, as was illustrated in Figures 8 and 9. In various other embodiments, the circuit illustrated in Figure 12 may be physically integrated into the same silicon as

another integrated circuit carrying out a different task, such as a microprocessor, a network processor, a system-on-a-chip, and others. In certain embodiments, the programmable transform block may be embody the configuration key as circuitry by means of one-time-programmable circuit elements such as microfuses, while in certain embodiments re-writeable circuit elements, such as nvRAM may be used and in other embodiments other methods may be used.

Turning now to Figure 13, an illustration is provided showing an integrated circuit 1303 that is overall similar to the integrated circuit in Figure 12 but incorporating two separate transform-enabled hashing cores 1305, 1309 each programmed by a different configuration key 1306, 1307. Additionally, the illustration provided in Figure 13 also shows 3 users 1301, 1302 and 1308 each one of which has access to the programming and configuration interface 1304, the first transform enabled hashing core 1305 and the second transform enabled hashing core 1309.

The method of operation and overall purpose of the integrated circuit illustrated in Figure 13 is similar to the method of operation and overall purpose of the integrated circuit illustrated in Figure 12, but in some embodiments the integrated circuit illustrated in Figure 13 may be used to hold multiple keys rather than only one key. In some embodiments a configuration such as that provided in Figure 13 may be used to maintain two separate keys within a single transform-enabled cryptographic result validator. In other embodiments, more than two configuration keys 1306, 1307, and in some such embodiments more than two transform-enabled cryptographic cores may be contained. As was described in the description accompanying Figure 12 various embodiments may contemplate different variations in terms of which different users have access to which functionalities supported by the integrated circuit, what method is used to program the configuration key into the programmable transform function or functions, the different types of hardware the circuit described may be integrated into, and others.

In this document explicit reference has only been made of certain cryptographic hashing algorithms, specifically the second iteration of the Secure Hashing Algorithm family (SHA-2), and within that particularly the 256-bit version. However, the technology described herein is fully applicable to other cryptographic hashing algorithms, including without limitation: SHA-2 in its

various implementations; Keccak/SHA-3 in its various implementations; Skein in its various implementations; Grøstl in its various implementations; JH in its various implementations; and others.

As used herein, the term set may refer to any collection of elements, whether finite or infinite. The term subset may refer to any collection of elements, wherein the elements are taken from a parent set; a subset may be the entire parent set. The term proper subset refers to a subset containing fewer elements than the parent set. The term sequence may refer to an ordered set or subset. The terms less than, less than or equal to, greater than, and greater than or equal to, may be used herein to describe the relations between various objects or members of ordered sets or sequences; these terms will be understood to refer to any appropriate ordering relation applicable to the objects being ordered.

The term tool can be used to refer to any apparatus configured to perform a recited function. For example, tools can include a collection of one or more components and can also be comprised of hardware, software or a combination thereof. Thus, for example, a tool can be a collection of one or more software components, hardware components, software/hardware components or any combination or permutation thereof. As another example, a tool can be a computing device or other appliance on which software runs or in which hardware is implemented.

As used herein, the term component might describe a given unit of functionality that can be performed in accordance with one or more embodiments of the technology disclosed herein. As used herein, a component might be implemented utilizing any form of hardware, software, or a combination thereof. For example, one or more processors, controllers, ASICs, PLAs, PALs, CPLDs, FPGAs, logical components, software routines or other mechanisms might be implemented to make up a component. In implementation, the various components described herein might be implemented as discrete components or the functions and features described can be shared in part or in total among one or more components. In other words, as would be apparent to one of ordinary skill in the art after reading this description, the various features and functionality described herein may be implemented in any given application and can be implemented in one or more separate or shared components in various combinations and permutations. Even though various

features or elements of functionality may be individually described or claimed as separate components, one of ordinary skill in the art will understand that these features and functionality can be shared among one or more common software and hardware elements, and such description shall not require or imply that
5 separate hardware or software components are used to implement such features or functionality.

Where components or components of the technology are implemented in whole or in part using software, in one embodiment, these software elements can be implemented to operate with a computing or processing component capable
10 of carrying out the functionality described with respect thereto. One such example computing component is shown in Figure 11. Various embodiments are described in terms of this example-computing component 1100. After reading this description, it will become apparent to a person skilled in the relevant art how to implement the technology using other computing
15 components or architectures.

Referring now to Figure 11, computing component 1100 may represent, for example, computing or processing capabilities found within desktop, laptop and notebook computers; hand-held computing devices (PDA's, smart phones, cell phones, palmtops, etc.); mainframes, supercomputers, workstations or
20 servers; or any other type of special-purpose or general-purpose computing devices as may be desirable or appropriate for a given application or environment. Computing component 1100 might also represent computing capabilities embedded within or otherwise available to a given device. For example, a computing component might be found in other electronic devices
25 such as, for example, digital cameras, navigation systems, cellular telephones, portable computing devices, modems, routers, WAPs, terminals and other electronic devices that might include some form of processing capability.

Computing component 1100 might include, for example, one or more processors, controllers, control components, or other processing devices, such as
30 a processor 1104. Processor 1104 might be implemented using a general-purpose or special-purpose processing engine such as, for example, a microprocessor, controller, or other control logic. In the illustrated example, processor 1104 is connected to a bus 1102, although any communication

medium can be used to facilitate interaction with other components of computing component 1100 or to communicate externally.

Computing component 1100 might also include one or more memory components, simply referred to herein as main memory 1108. For example, preferably random access memory (RAM) or other dynamic memory, might be used for storing information and instructions to be executed by processor 1104. Main memory 1108 might also be used for storing temporary variables or other intermediate information during execution of instructions to be executed by processor 1104. Computing component 1100 might likewise include a read only memory (“ROM”) or other static storage device coupled to bus 1102 for storing static information and instructions for processor 1104.

The computing component 1100 might also include one or more various forms of information storage mechanism 1110, which might include, for example, a media drive 1112 and a storage unit interface 1120. The media drive 1112 might include a drive or other mechanism to support fixed or removable storage media 1114. For example, a hard disk drive, a floppy disk drive, a magnetic tape drive, an optical disk drive, a CD or DVD drive (R or RW), or other removable or fixed media drive might be provided. Accordingly, storage media 1114 might include, for example, a hard disk, a floppy disk, magnetic tape, cartridge, optical disk, a CD or DVD, or other fixed or removable medium that is read by, written to or accessed by media drive 1112. As these examples illustrate, the storage media 1114 can include a computer usable storage medium having stored therein computer software or data.

In alternative embodiments, information storage mechanism 1110 might include other similar instrumentalities for allowing computer programs or other instructions or data to be loaded into computing component 1100. Such instrumentalities might include, for example, a fixed or removable storage unit 1122 and an interface 1120. Examples of such storage units 1122 and interfaces 1120 can include a program cartridge and cartridge interface, a removable memory (for example, a flash memory or other removable memory component) and memory slot, a PCMCIA slot and card, and other fixed or removable storage units 1122 and interfaces 1120 that allow software and data to be transferred from the storage unit 1122 to computing component 1100.

Computing component 1100 might also include a communications interface 1124. Communications interface 1124 might be used to allow software and data to be transferred between computing component 1100 and external devices. Examples of communications interface 1124 might include a modem or
5 softmodem, a network interface (such as an Ethernet, network interface card, WiMedia, IEEE 802.XX or other interface), a communications port (such as for example, a USB port, IR port, RS232 port Bluetooth® interface, or other port), or other communications interface. Software and data transferred via communications interface 1124 might typically be carried on signals, which can
10 be electronic, electromagnetic (which includes optical) or other signals capable of being exchanged by a given communications interface 1124. These signals might be provided to communications interface 1124 via a channel 1128. This channel 1128 might carry signals and might be implemented using a wired or wireless communication medium. Some examples of a channel might include a
15 phone line, a cellular link, an RF link, an optical link, a network interface, a local or wide area network, and other wired or wireless communications channels.

In this document, the terms “computer program medium” and “computer usable medium” are used to generally refer to media such as, for example, memory 1108, storage unit 1120, media 1114, and channel 1128. These and
20 other various forms of computer program media or computer usable media may be involved in carrying one or more sequences of one or more instructions to a processing device for execution. Such instructions embodied on the medium, are generally referred to as “computer program code” or a “computer program product” (which may be grouped in the form of computer programs or other
25 groupings). When executed, such instructions might enable the computing component 1100 to perform features or functions of the disclosed technology as discussed herein.

While various embodiments of the disclosed technology have been described above, it should be understood that they have been presented by way
30 of example only, and not of limitation. Likewise, the various diagrams may depict an example architectural or other configuration for the disclosed technology, which is done to aid in understanding the features and functionality that can be included in the disclosed technology. The disclosed technology is not restricted to the illustrated example architectures or configurations, but the

desired features can be implemented using a variety of alternative architectures and configurations. Indeed, it will be apparent to one of skill in the art how alternative functional, logical or physical partitioning and configurations can be implemented to implement the desired features of the technology disclosed
5 herein. Also, a multitude of different constituent component names other than those depicted herein can be applied to the various partitions. Additionally, with regard to flow diagrams, operational descriptions and method claims, the order in which the steps are presented herein shall not mandate that various embodiments be implemented to perform the recited functionality in the same
10 order unless the context dictates otherwise.

Although the disclosed technology is described above in terms of various exemplary embodiments and implementations, it should be understood that the various features, aspects and functionality described in one or more of the individual embodiments are not limited in their applicability to the particular
15 embodiment with which they are described, but instead can be applied, alone or in various combinations, to one or more of the other embodiments of the disclosed technology, whether or not such embodiments are described and whether or not such features are presented as being a part of a described embodiment. Thus, the breadth and scope of the technology disclosed herein
20 should not be limited by any of the above-described exemplary embodiments.

Terms and phrases used in this document, and variations thereof, unless otherwise expressly stated, should be construed as open ended as opposed to limiting. As examples of the foregoing: the term “including” should be read as meaning “including, without limitation” or the like; the term “example” is used
25 to provide exemplary instances of the item in discussion, not an exhaustive or limiting list thereof; the terms “a” or “an” should be read as meaning “at least one,” “one or more” or the like; and adjectives such as “conventional,” “traditional,” “normal,” “standard,” “known” and terms of similar meaning should not be construed as limiting the item described to a given time period or
30 to an item available as of a given time, but instead should be read to encompass conventional, traditional, normal, or standard technologies that may be available or known now or at any time in the future. Likewise, where this document refers to technologies that would be apparent or known to one of ordinary skill in the

art, such technologies encompass those apparent or known to the skilled artisan now or at any time in the future.

The presence of broadening words and phrases such as “one or more,” “at least,” “but not limited to” or other like phrases in some instances shall not
5 be read to mean that the narrower case is intended or required in instances where such broadening phrases may be absent. The use of the term “component” does not imply that the components or functionality described or claimed as part of the component are all configured in a common package. Indeed, any or all of the various components of a component, whether control logic or other components,
10 can be combined in a single package or separately maintained and can further be distributed in multiple groupings or packages or across multiple locations.

Additionally, the various embodiments set forth herein are described in terms of exemplary block diagrams, flow charts and other illustrations. As will become apparent to one of ordinary skill in the art after reading this document,
15 the illustrated embodiments and their various alternatives can be implemented without confinement to the illustrated examples. For example, block diagrams and their accompanying description should not be construed as mandating a particular architecture or configuration.

20

Claims

1. A cryptographic integrated circuit, comprising:
a transformation block comprising a set of electronic circuitry integrated into a datapath circuitry of the cryptographic integrated circuit, and
programmable at a time subsequent to the manufacture of the integrated circuit to perform a transform operation on data received from datapath circuitry prior to the transformation block;
a programming circuit communicatively coupling the transformation block and a configuration interface;
wherein the integrated circuit is implemented within a cryptographic circuit, and
a user may program the transformation block through the configuration interface such that the transformation block is configured in accordance with a configuration key.
2. The cryptographic integrated circuit of claim 1, wherein the transformation block comprises a configuration key programmable by the user and is configured as datapath circuitry capable of operating at the same speed as other circuitry along the integrated circuits cryptographic datapath.
3. The cryptographic integrated circuit of claim 2, the datapath circuitry further comprising a hashing block configured to apply a hashing algorithm determined in accordance with a cryptographic standard specified by a proof-of-work protocol.
4. The cryptographic integrated circuit of claim 2, further comprising a permanent and indelible mechanism to embody the key entered by the user as physical circuitry on the integrated circuits datapath.
5. The cryptographic integrated circuit of claim 2, further comprising a nonpermanent mechanism to embody the key entered by the user as physical circuitry on the integrated circuits datapath.
6. The cryptographic integrated circuit of claim 2, wherein the transformation block is configured to implement a direct bit inversion scheme.

7. The cryptographic integrated circuit of claim 2, wherein the transformation block may be configured to implement a bit transposition scheme.
8. The cryptographic integrated circuit of claim 2, the integrated circuit designed and fabricated to implement a secure hashing algorithm.
9. The cryptographic integrated circuit of claim 1, the datapath circuitry further comprising a first hashing block configured to apply a first hashing algorithm determined in accordance with a cryptographic standard specified by a proof-of-work protocol, and a second hashing block configured to subsequently apply a second hashing algorithm determined in accordance with a cryptographic standard specified by a proof-of-work protocol.
10. The cryptographic integrated circuit of claim 9, the transformation block being located before the first hashing block such that an output of the transformation block is an input to the first hashing block.
11. The cryptographic integrated circuit of claim 9, the transformation block being located between the first and second hashing blocks such that an output of the first hashing block is an input to the transformation block and an output of the transformation block is an input to the second hashing block.
12. The cryptographic integrated circuit of claim 9, further comprising a second transformation block.
13. The cryptographic integrated circuit of claim 1, the datapath circuitry comprising a plurality of cryptographic cores, each cryptographic core of the plurality of cryptographic cores comprising a first hashing block.
14. The cryptographic integrated circuit of claim 13, each cryptographic core of the plurality of cryptographic cores comprising a transformation block.

15. The cryptographic integrated circuit of claim 1, the datapath circuitry comprising a plurality of cryptographic cores, each cryptographic core of the plurality of cryptographic cores comprising two hashing blocks connected sequentially such that an output from a first hashing block is an input to a second hashing block either directly or after passing through additional circuitry blocks.

16. The cryptographic integrated circuit of claim 15, the first transformation block being located before the first hashing block such that an output of the transformation block is an input to the first hashing block and the second transformation block being located before the second hashing block such that an output from the first hashing core is an input to the second transformation block, and an output from the second transformation block is an input to the second hashing block.

17. The cryptographic integrated circuit of claim 15, both transformation blocks being located before the first hashing block such that the combined output from both transformation blocks is an input to the first hashing block.

18. The cryptographic integrated circuit of claim 15, both transformation blocks being located after the first hashing block such than an output from the first hashing block is an input to the transformation blocks and before the second hashing block such that the combined output from both transformation blocks is an input to the second hashing block.

19. The cryptographic integrated circuit of claim 15, wherein a first transformation block is shared by each cryptographic core of the plurality of cryptographic cores and a second transformation block is shared by each cryptographic core of the plurality of cryptographic cores.

20. The cryptographic integrated circuit of claim 1, further comprising a lock fuse disposed within the programming circuit, wherein the transformation block is inaccessible when the lock fuse is disabled.

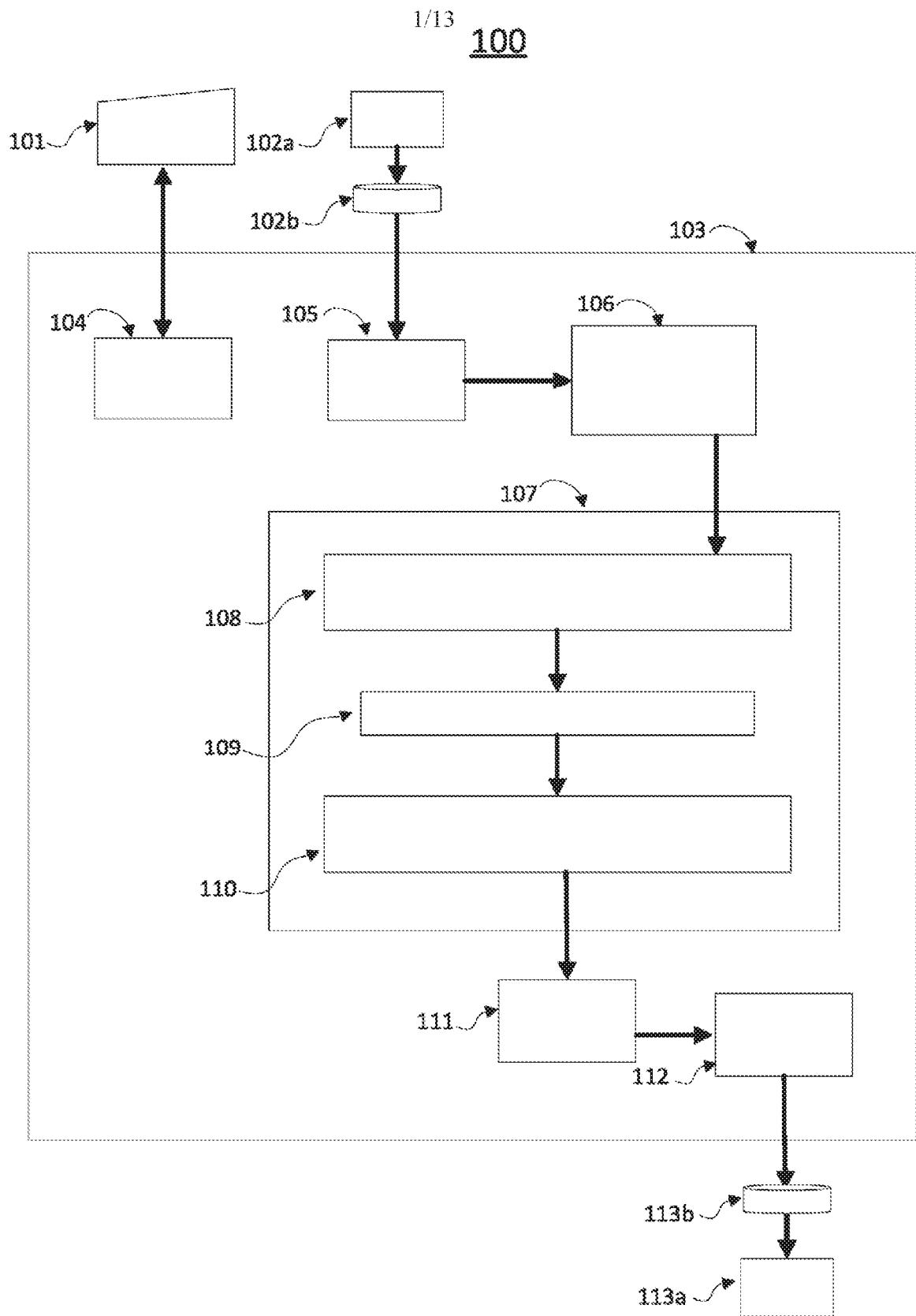
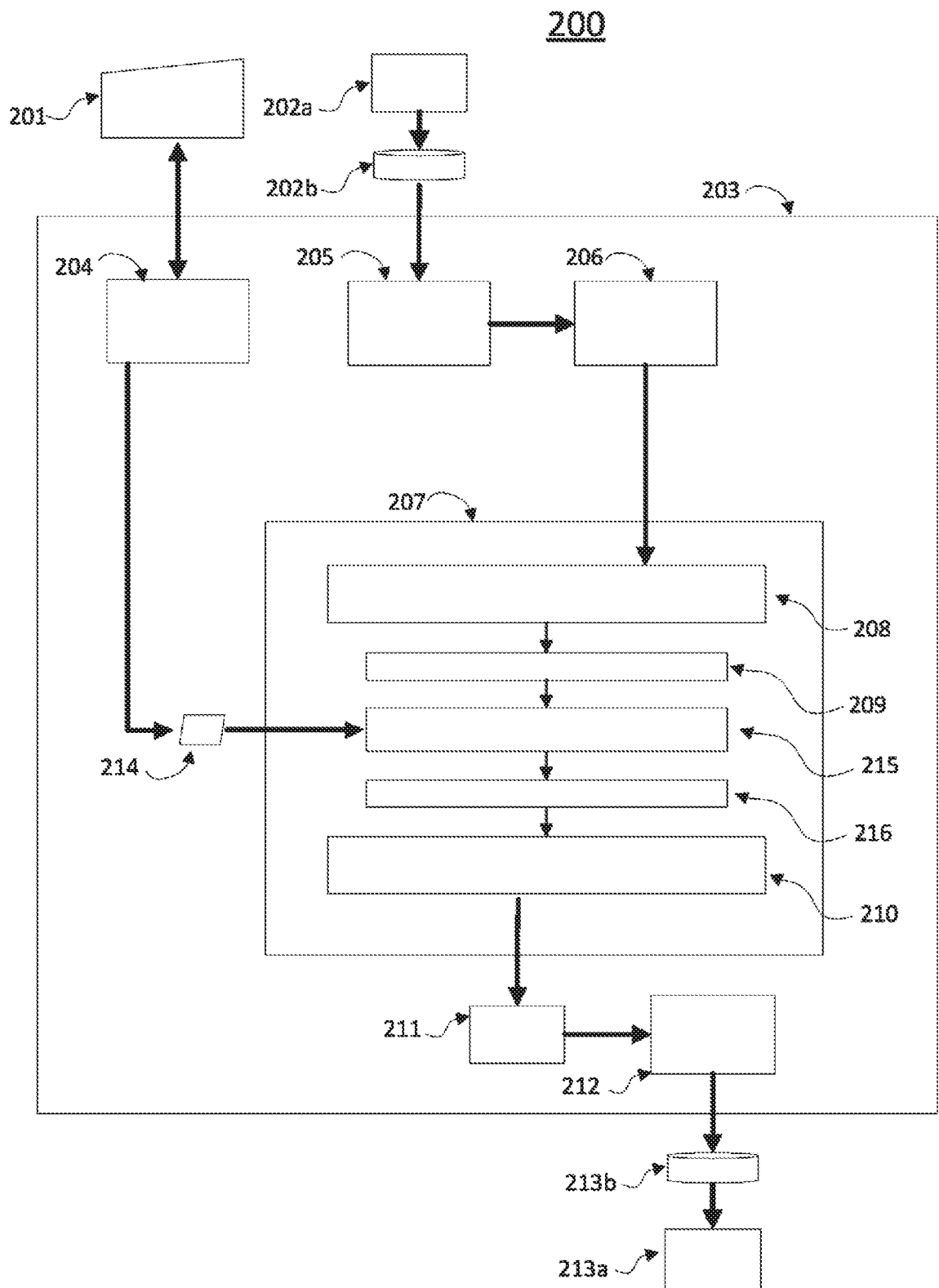


FIG. 1

2/13

**FIG. 2**

3/13

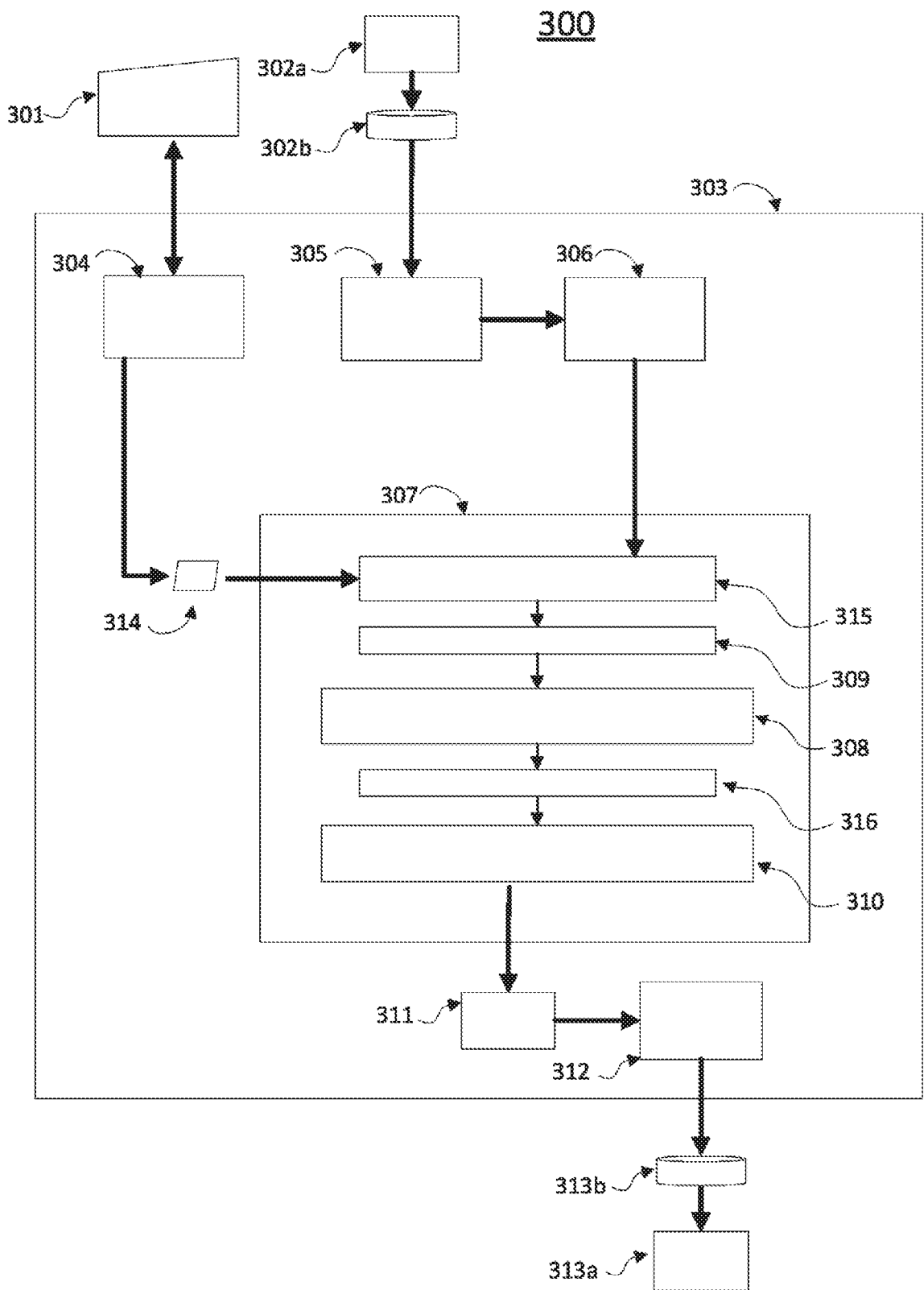


FIG. 3

4/13

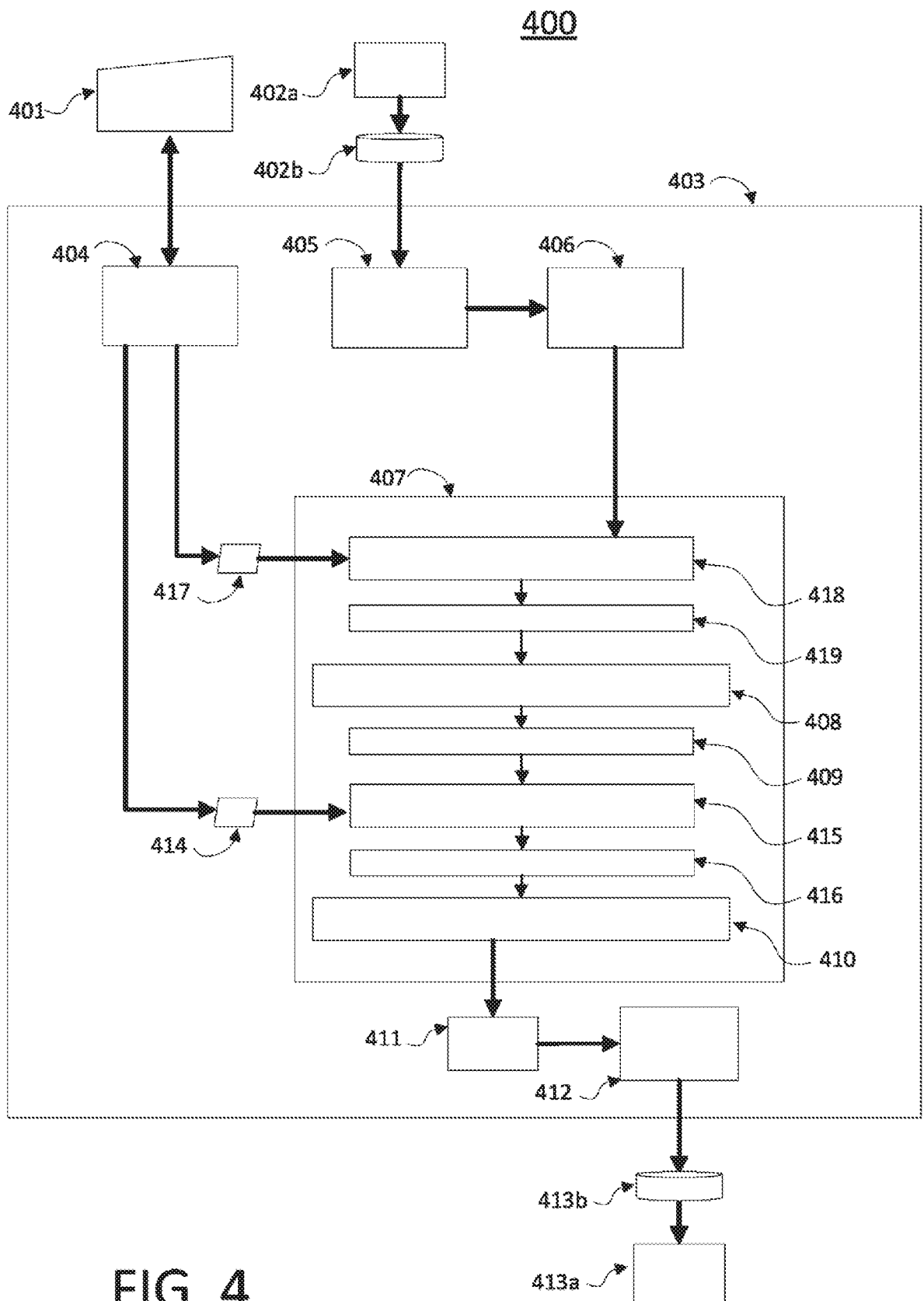


FIG. 4

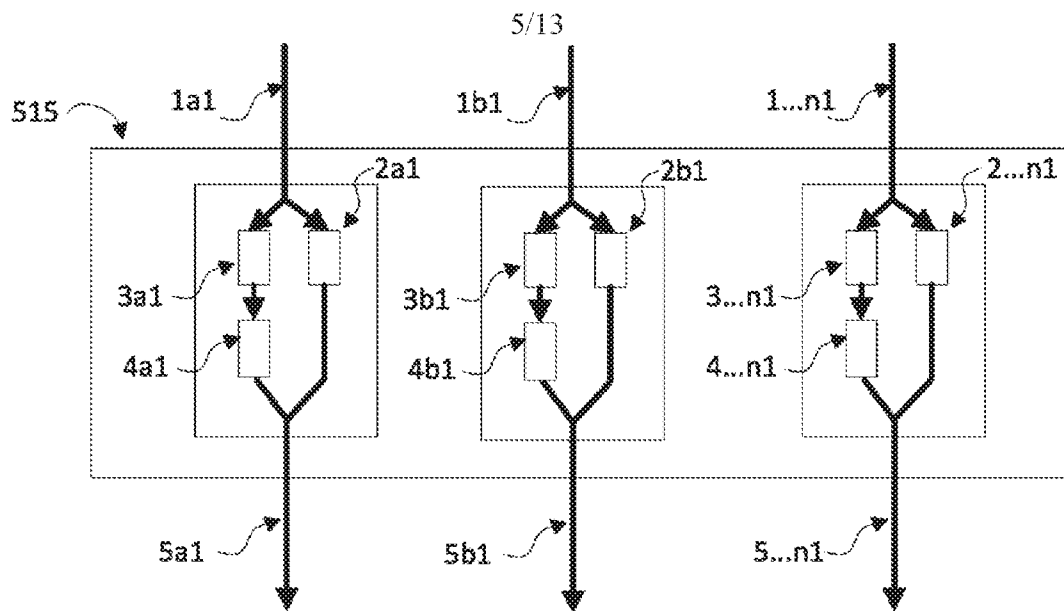


FIG. 5A

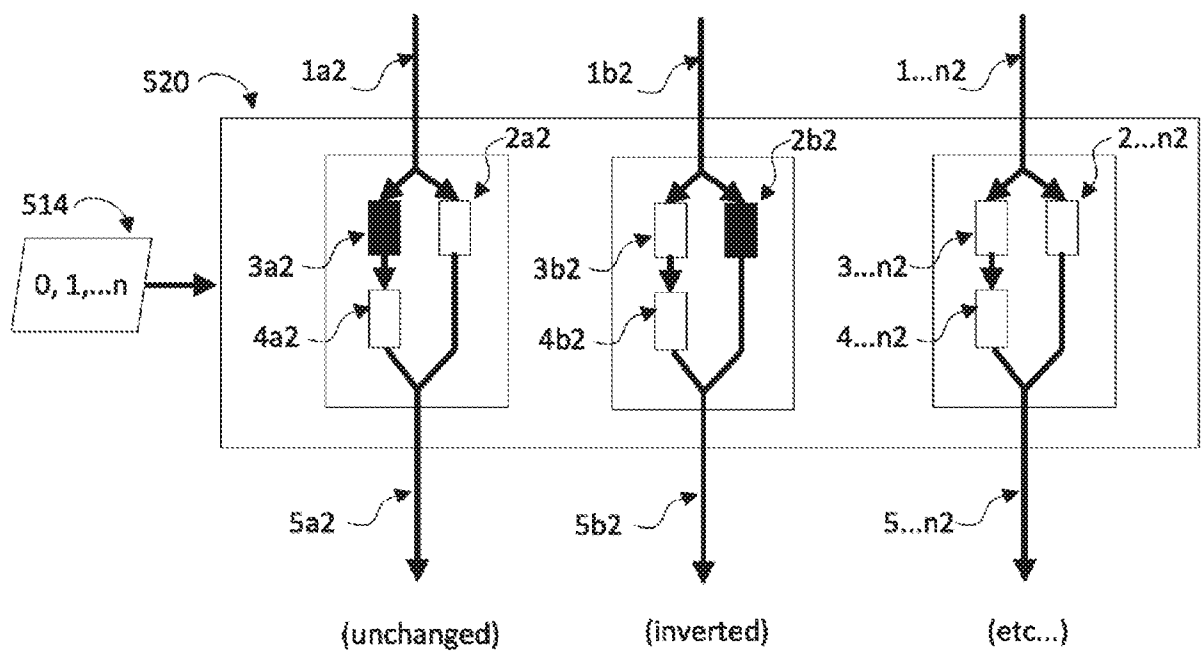


FIG. 5B

6/13

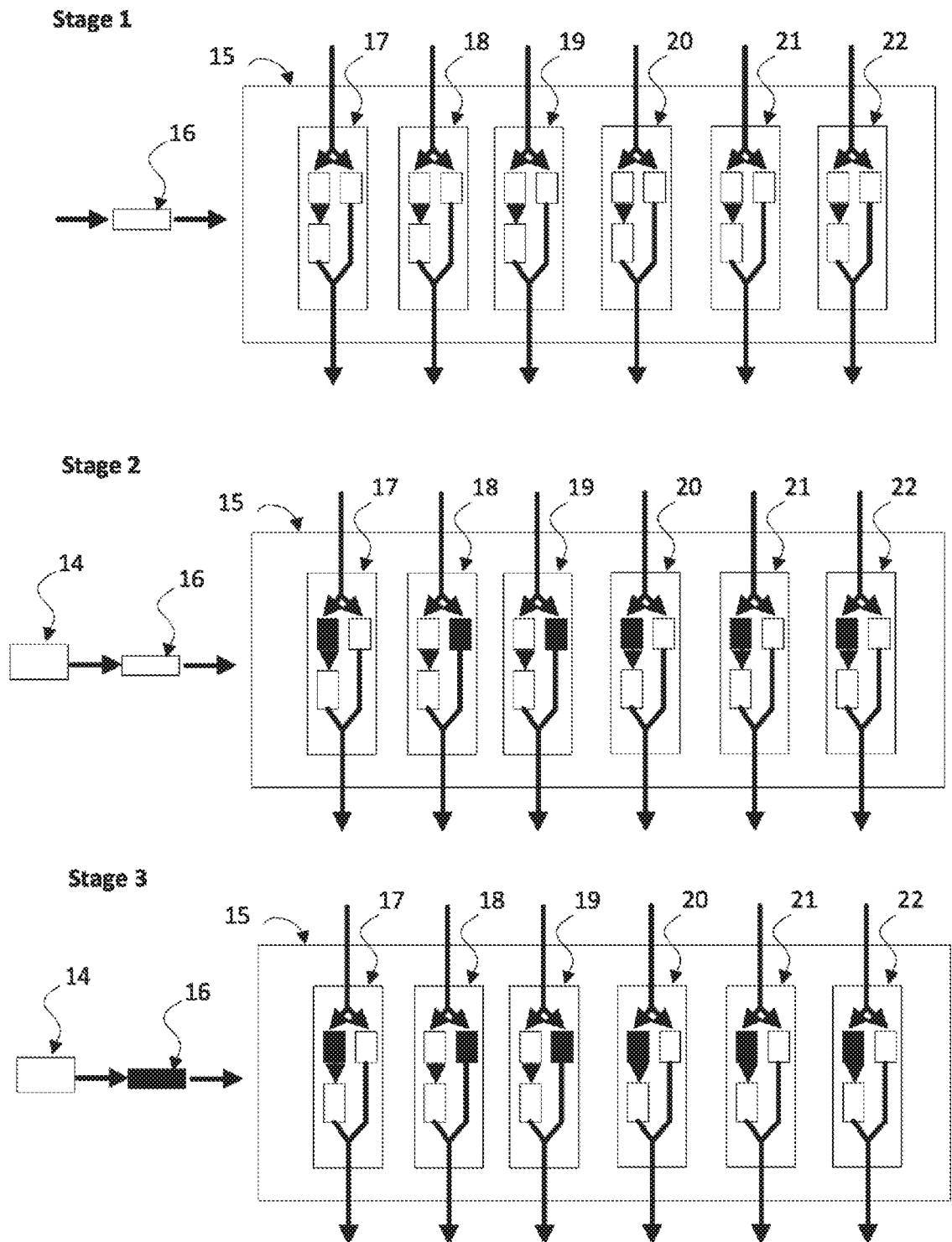


FIG. 6

7/13

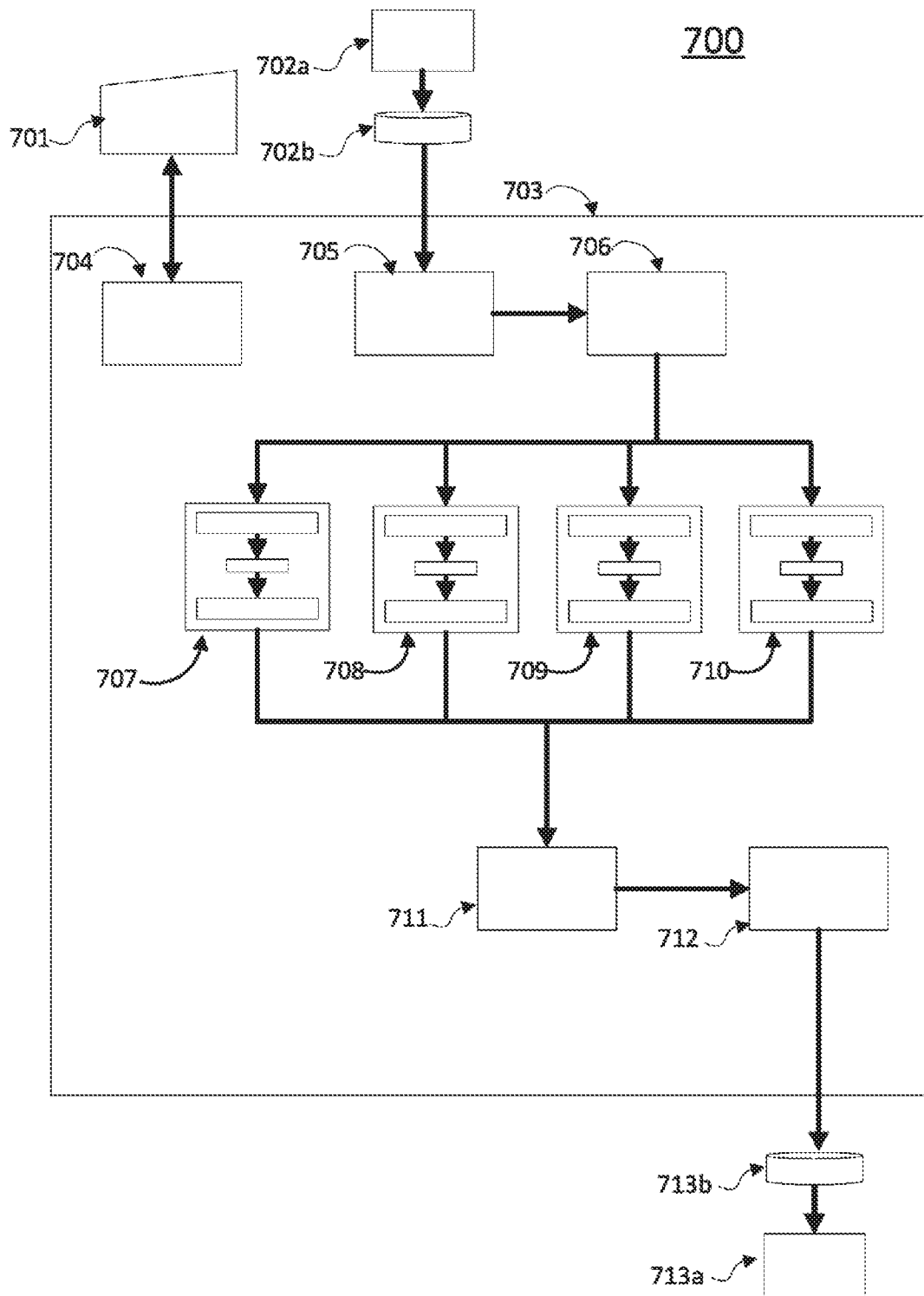


FIG. 7

8/13

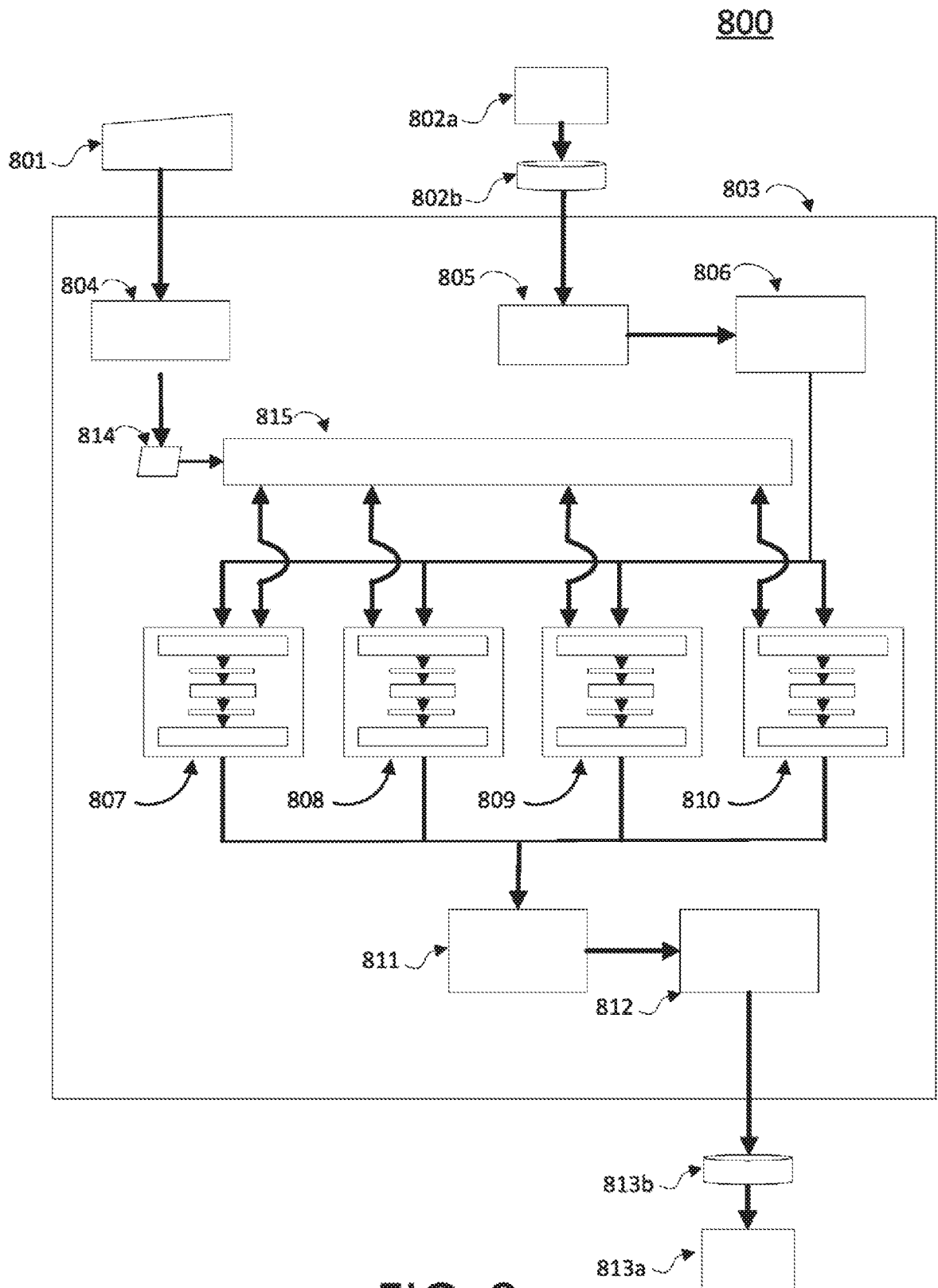


FIG. 8

9/13

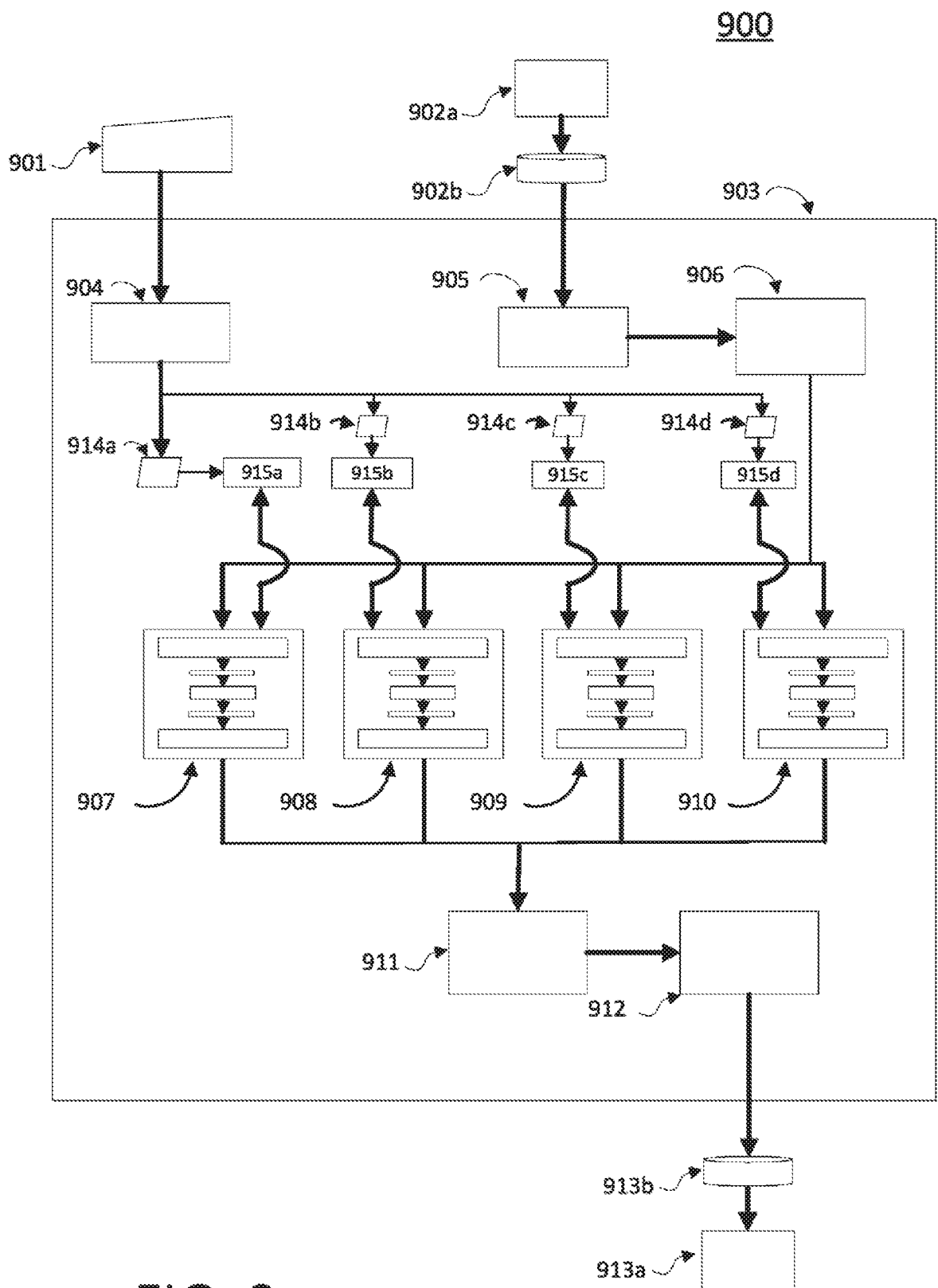


FIG. 9

10/13

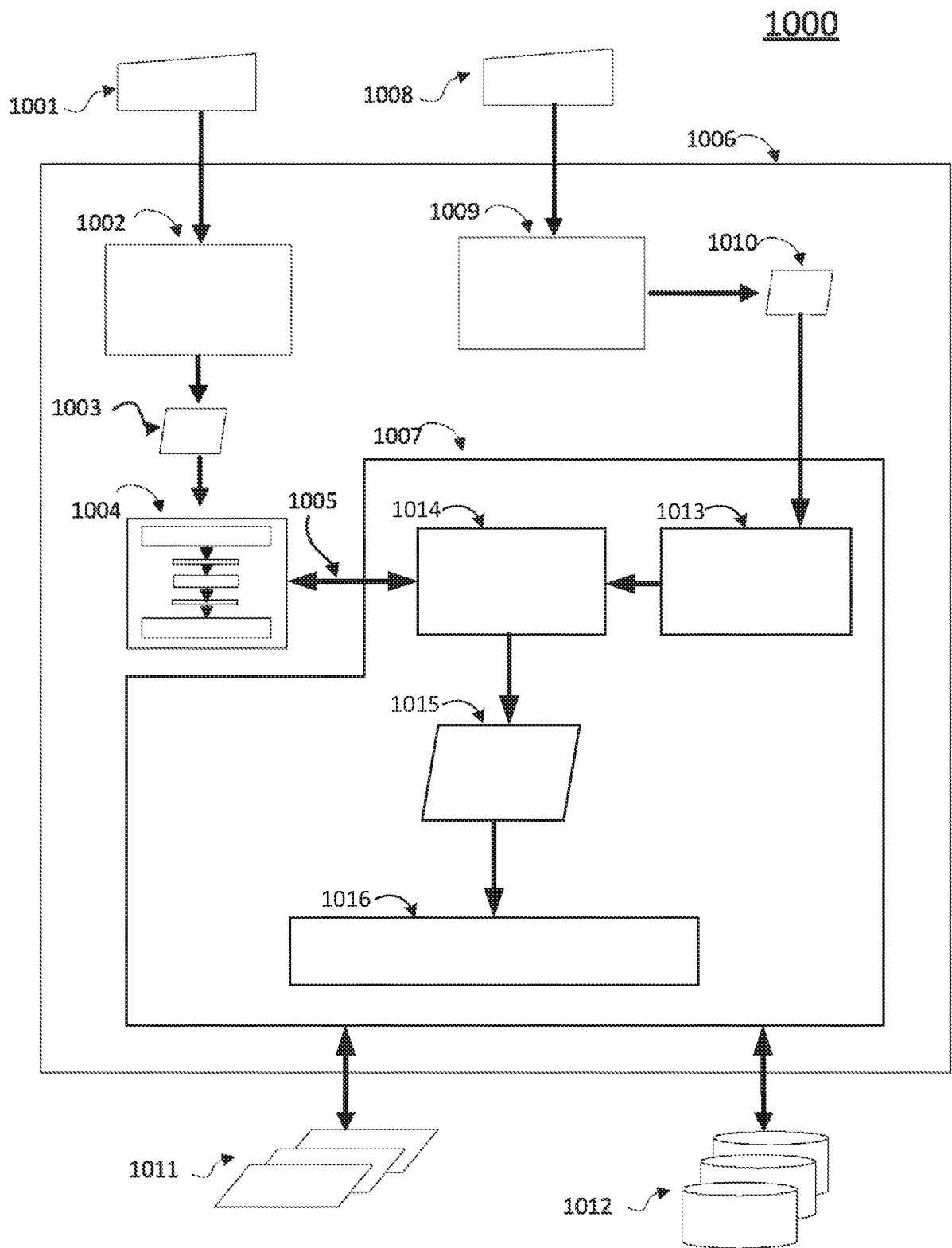


FIG. 10

11/13

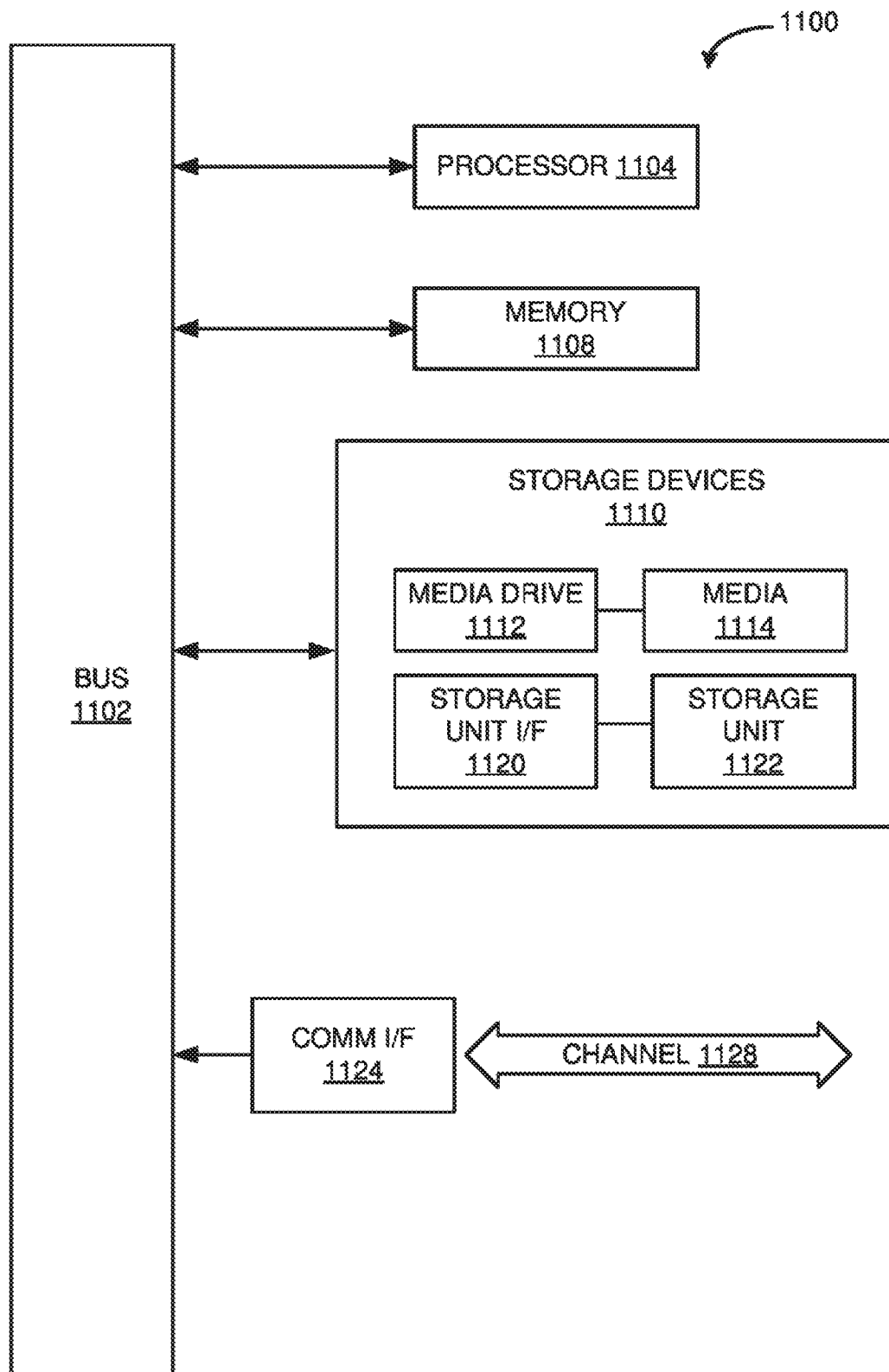


FIG. 11

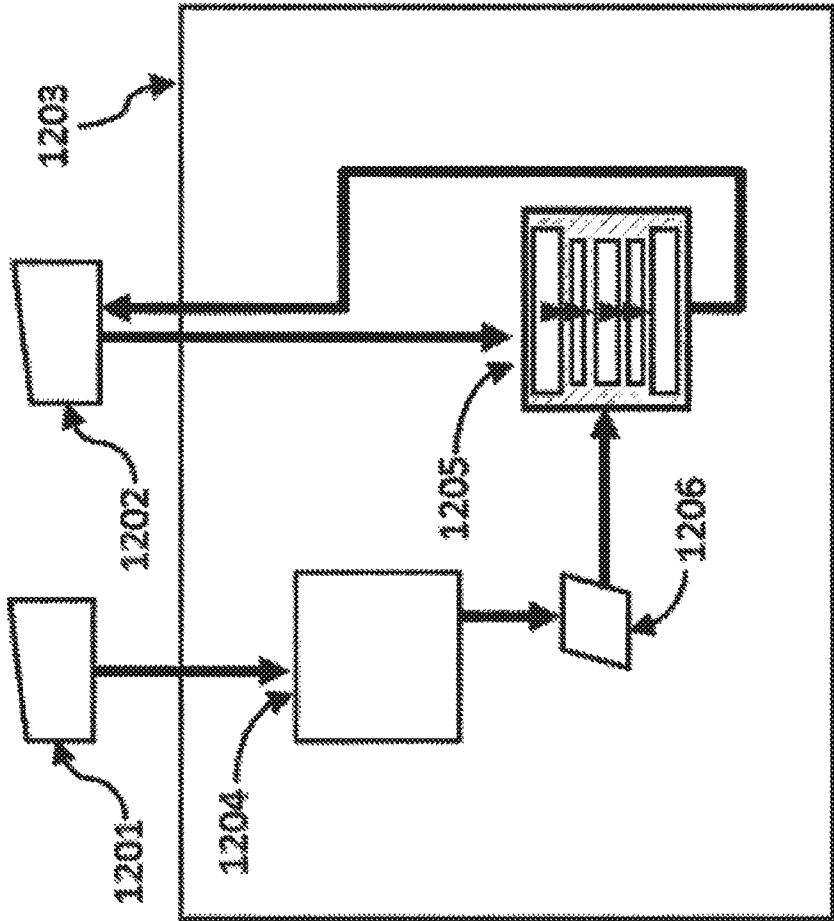


FIG. 12

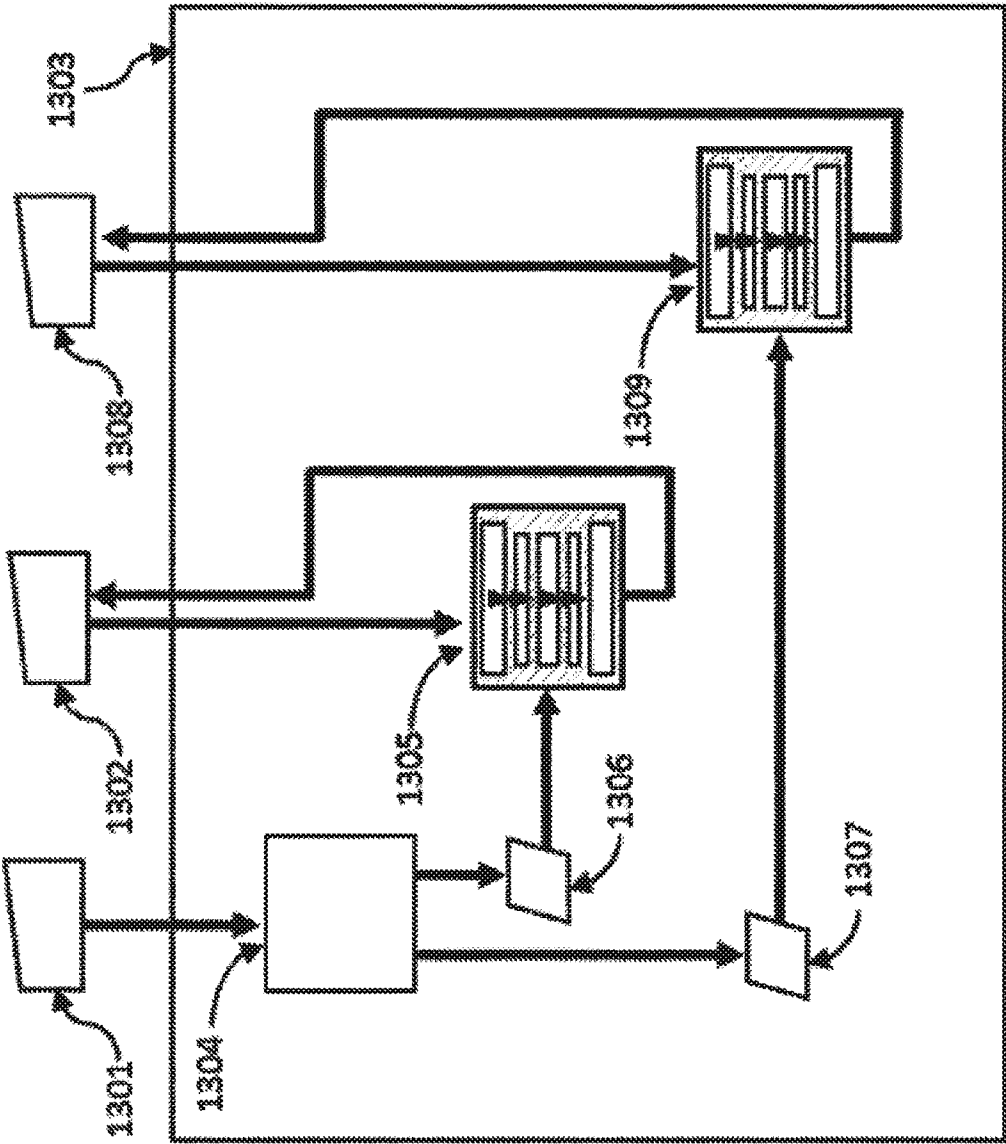


FIG. 13

INTERNATIONAL SEARCH REPORT

International application No.

PCT/US 17/13377

A. CLASSIFICATION OF SUBJECT MATTER
 IPC(8) - G06F 21/72; H04L 9/12; H04L 9/16 (2017.01)
 CPC - G06F 21/72; H04L 9/12; H04L 9/16

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

See Search History Document

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

See Search History Document

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

See Search History Document

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X --- Y --- A	US 8,347,096 B2 (HOORNAERT et al.) 01 January 2013 (01.01.2013), Fig 1, abstract, col 7, ln 15-col 8, ln 9, col 9, ln 1-30	1, 2, 6, 8 ----- 3-5, 7, 13-15, 19, 20 ----- 9-12, 16-18
Y --- A	US 8,132,005 B2 (TARKKALA et al.) 06 March 2012 (06.03.2012), Fig 1, abstract, col 1, ln 7-15, col 8, ln 50-col 9, ln 18	3 ----- 9-12
Y	US 6,581,162 B1 (ANGELO et al.) 17 June 2003 (17.06.2003), abstract, col 10, ln 20-44	4, 5
Y	US 5,442,705 A (MIYANO) 15 August 1995 (15.08.1995), Fig 1, abstract, col 2, ln 54-col 3, ln 15	7
Y --- A	US 7,191,341 B2 (PAASKE et al.) 13 March 2007 (13.03.2007), Fig 2, abstract, col 4, ln 17-19, col 4, ln 49-col 6, ln 7	13-15, 19 ----- 9-12, 16-18
Y	US 8,402,241 B2 (MIRANDA et al.) 19 March 2013 (19.03.2013), Fig 1, abstract, col 2, ln 18-53	20
Y --- A	US 7,177,421 B2 (BUER et al.) 13 February 2007 (13.02.2007), Fig 1, abstract, col 1, ln 17-23, col 2, ln 56-col 3, ln 5	15, 19 ----- 9-12, 16-18

☐ Further documents are listed in the continuation of Box C.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search

14 March 2017

Date of mailing of the international search report

18 APR 2017

Name and mailing address of the ISA/US

Mail Stop PCT, Attn: ISA/US, Commissioner for Patents

P.O. Box 1450, Alexandria, Virginia 22313-1450

Facsimile No. 571-273-8300

Authorized officer:

Lee W. Young

PCT Helpdesk: 571-272-4300

PCT OSP: 571-272-7774