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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
24 October 2013 (24.10.2013)

WIPO | PCT

(10) International Publication Number
WO 2013/158129 A1

(51) International Patent Classification:
G06F 12/00 (2006.01) *G06F 12/14* (2006.01)
G06F 12/08 (2006.01) *G06F 21/24* (2006.01)

(21) International Application Number:
PCT/US2012/037309

(22) International Filing Date:
10 May 2012 (10.05.2012)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
12164501.4 17 April 2012 (17.04.2012) EP

(71) Applicant *(for all designated States except US)*: **ITRON, INC.** [US/US]; 2111 North Miller Road, Liberty Lake, WA 99019 (US).

(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: **PLAINECASSAGNE, Eric** [FR/FR]; c/o Itron, Inc., 2111 North Molter Road, Liberty Lake, WA 99019 (US). **DESCAMPS, Guillaume** [FR/FR]; c/o Itron, Inc., 2111 North Molter Road, Liberty Lake, WA 99019 (US).

(74) Agents: **THOMPSON, David, S.** et al.; Lee & Hayes, PLLC, 601 W. Riverside Ave., Suite 1400, Spokane, WA 99201 (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, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, 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, 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, ML, MR, NE, SN, TD, TG).

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

(54) **Title**: MICROCONTROLLER CONFIGURED FOR EXTERNAL MEMORY DECRYPTON

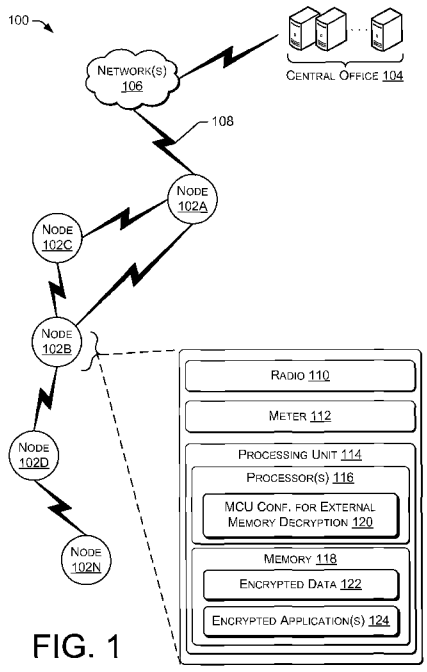

FIG. 1

(57) **Abstract**: In an advanced metering infrastructure environment, software program statements and/or data may be encrypted. A microcontroller unit may include a first cache configured to store a block of encrypted data obtained from an external memory device. A decryption engine may decrypt the block of encrypted data for storage in a second cache. An address alignment module may be configured to receive input from a program counter and to calculate an offset pointer. The offset pointer may indicate a particular word in the block of decrypted data within the second cache for transmission to an instruction register for use by an application program. An address generator may be configured to receive input from the address alignment module and to indicate a block of data in the external memory device to be loaded into the first cache, to thereby replacing the encrypted data sent to the decryption engine.

WO 2013/158129 A1

MICROCONTROLLER CONFIGURED FOR EXTERNAL MEMORY DECRYPTION

CROSS REFERENCE TO RELATED APPLICATIONS

[0001] This application claims priority to European Patent Application No. EP 12164501.4, filed April 17, 2012, and entitled "Microcontroller Configured for External Memory Decryption," which is incorporated herein by reference.

BACKGROUND

[0002] An advanced metering infrastructure (AMI) node may be associated with a device for metering a consumable resource, such as electricity, natural gas or water. Such nodes may be organized as an autonomous routing area (ARA), having a tree structure that may be headed by a root node and/or router node. Within each node, a number of sensors, meters or other devices may provide data to a processing unit, memory and/or an application specific integrated circuit (ASIC). Application(s) operating on the node may process data received as input, and relay it up the network.

[0003] Such data, and the executable code that forms the application(s), may be valuable. For example, data may represent consumption quantities of electricity, natural gas or other utility, and integrity of the data may be important for financial reasons. Additionally, the executable code may include propriety technology. Unfortunately, the nodes in an ARA may be vulnerable to tampering, and the data may be vulnerable to unauthorized viewing or copying. Additionally, the executable code of the application(s) (e.g., program statements) may be vulnerable to copying, decompiling and reverse engineering, which can reveal information on the operation and functionality of individual nodes and/or the entire network.

BRIEF DESCRIPTION OF THE DRAWINGS

[0004] The detailed description is described with reference to the accompanying figures. In the figures, the left-most digit(s) of a reference number identifies the figure in which the reference number first appears. The same numbers are used throughout the drawings to reference like features and components. Moreover, the figures are intended to illustrate general concepts, and not to indicate required and/or necessary elements.

[0005] FIG. 1 is a diagram showing an example network having a plurality of nodes, some detail of one example node, a connection to a further network such as the Internet, and a central office.

[0006] FIG. 2 is a schematic diagram of an example microcontroller unit configured for decryption of data contained in an external memory device.

[0007] FIG. 3 is a flow diagram illustrating an example method by which a microcontroller configured for external memory decryption may be operated.

DETAILED DESCRIPTION

Overview

[0008] Advanced encryption standard (AES) and other encryption techniques represent significant advances in data security. For example, blocks of 128 bits may be encrypted using a symmetric key, resulting in strong encryption. Using such encryption, data may be secured, even in environments in which it is not possible to secure and control the memory devices that contain the data.

[0009] In an advanced metering infrastructure (AMI), nodes may contain data related to the consumption of electricity, natural gas, water and/or other consumable resources. The nodes may include memory devices which control records of the consumption. Such records have considerable financial value,

since customers are billed based on the amounts consumed. Accordingly, encryption is useful to prevent tampering.

[0010] However, existing microcontrollers are configured to execute code and to use data that is not encrypted. In cases where encrypted executable code and/or encrypted data are used, considerable overhead is introduced, which slows down program execution. Also, the decrypted code and data is put at risk during the periods that it remains decrypted. The below discussion includes several representative examples that provide techniques for configuring a microcontroller for external memory encryption.

[0011] An example illustrating some of the techniques discussed herein—not to be considered a full or comprehensive discussion—may assist the reader. In an AMI environment, a meter may be configured to generate encrypted data representing consumption of a resource and to record the encrypted data in a memory device located external to a microcontroller unit. The meter may utilize an application (e.g., a software program) that includes executable statements that are stored in an encrypted form on the memory. Within the microcontroller unit (and/or microprocessor, processor, etc.) a first cache may be configured to store a block of encrypted data obtained from the memory device. The data may represent any encrypted information, such as consumption records or encrypted program statements. Also within the microcontroller unit, a decryption engine may be configured to obtain and decrypt the block of encrypted data from the first cache. A second cache, also located within the microcontroller unit, may be configured to receive the block of decrypted data from the decryption engine. An address alignment module may be configured to receive input from a program counter and to calculate an offset pointer. The offset pointer may indicate a particular word in the block of decrypted data within the second cache for transmission to an instruction register for use by an application program. An

address generator may be configured to receive input from the address alignment module and to indicate a block of memory in the memory device to be loaded into the first cache, to thereby replacing the encrypted data sent to the decryption engine.

[0012] The discussion herein includes several sections. The discussion, drawings and each section are intended to be examples of techniques and/or structures, but are not intended to indicate elements which must be used and/or performed. More particularly, this entire description is intended to illustrate components and techniques that may be utilized in a microcontroller configured for external memory decryption, but not components or techniques which are necessarily required. The discussion begins with a section entitled "Example Network," which describes one environment that may implement the techniques described herein. This section depicts and describes an example of an ARA that may include the techniques for use with encrypted data on a memory device that is external to a microcontroller and/or processor. Next, a section entitled "Example Microcontroller Unit" illustrates and describes an example of a microcontroller unit, and also shows its relationship to an external memory device. This section provides example structures that perform representative functionality, including decrypting data obtained from the external memory device. A third section, entitled "Example Methods," discusses aspects of methods operational in devices including processors, memory devices, application specific integrated circuits (ASICs), etc. In particular, the example methods may be applied to any of the techniques discussed herein. Next, a section entitled "Example Methods of External Memory Decryption" illustrates and describes aspects that may be used to read, decrypt and manage encrypted data on a memory device that is external to the microcontroller. Finally, the discussion ends with a brief conclusion.

[0013] This brief introduction, including section titles and corresponding summaries, is provided for the reader's convenience and is not intended to describe and/or limit the scope of the claims or any section of this disclosure.

Example Network

[0014] FIG. 1 is a diagram showing an example network 100, which may be an ARA or other network. FIG. 1 shows a plurality of nodes 102, including a root or router node 102A, which may be connected to a central office 104 by a network 106 such as the Internet. Within the network, nodes may communicate by means of radio RF links 108, power line communication (i.e., a signal superimposed over current flowing in an electrical power transmission grid) or other medium. The example network 100 is provided as a specific instance to illustrate more general concepts of the use and management of encrypted data in a network environment, and not to indicate required and/or necessary elements. In particular, the example network 100 will be used to illustrate examples of nodes utilizing microcontrollers and external memory devices. In particular, the microcontrollers may execute software code associated with applications operating on the nodes that may utilize encrypted data and/or encrypted program statements.

[0015] The network architecture 100 may be configured as a smart grid, such as an AMI network including a number of utility meters having wireless communication capabilities. The utility meters may measure consumption of electricity, natural gas, water or other consumable resources. The central office 104 may be implemented by one or more computing devices, such as servers, personal computers, laptop computers, etc. The one or more computing devices may be equipped with one or more processor(s) communicatively coupled to memory. In some examples, the central office 104 includes a centralized meter

data management system which performs processing, analysis, storage, and/or management of data received from one or more of the nodes 102. For instance, the central office 104 may process, analyze, store, and/or manage data obtained from a smart utility meter, sensor, control device, router, regulator, server, relay, switch, valve, and/or other nodes. Although the example of FIG. 1 illustrates the central office 104 in a single location, in some examples the central office may be distributed amongst multiple locations and/or may be eliminated entirely (e.g., in the case of a highly decentralized distributed computing platform).

[0016] The network(s) 106 may comprise a wireless or a wired network, or a combination thereof, such as the Internet. The network 106 may be a collection of discrete and/or interconnected networks, which may function as a single large network.

[0017] The network architecture 100 may include a plurality of nodes 102A, 102B, 102C, 102D, … 102N (collectively referred to as nodes 102) communicatively coupled to each other via direct communication RF signals, power line communication (PLC) links, or other types of transmissions. Some of the nodes may be associated with utility meters. In this example, N represents an example number of nodes in an ARA, which may be configured as a wide area network (WAN), metropolitan area network (MAN), local area network (LAN), neighborhood area network (NAN), personal area network (PAN), a combination of the foregoing, or the like.

[0018] The node 102A may be considered to be a "root," "root node," "field area router," or similar, and may be configured to connect the ARA to servers within the central office 104 by way of a back-haul network, such as the network 106. The nodes 102 may communicate by means of signals 108, which facilitate both upstream and downstream transfer of data, information, reports, queries and packets, etc.

[0019] Detail of the node 102B includes a radio 110, configured for communication by means of RF signals 108. The radio 110 may comprise a RF transceiver configured to transmit and/or receive RF signals via one or more of a plurality of channels/frequencies. In one example of a wireless implementation, the node 102 may include a single radio 110 configured to send and receive data on multiple different channels, such as a control channel and multiple data channels on each communication link 108. The radio 110 may also be configured to implement a plurality of different modulation techniques, data rates, protocols, signal strengths, and/or power levels. Additionally, the radio may be configured to sequentially tune a plurality of different frequencies, each for a short period of time, in a "frequency hopping" scheme. In other implementations, each of the nodes may be configured for wired communication. By way of example and not limitation, wired communications may include power line communications (PCL) or other wired communication network technologies, such as Ethernet. The architecture of the network 100 may represent a heterogeneous network of nodes, in that the nodes 102 may include different types of nodes (e.g., smart meters, cellular relays, sensors, etc.), different generations or models of nodes, and/or nodes that otherwise are capable transmitting on different channels and using different modulation techniques, data rates, protocols, signal strengths, and/or power levels.

[0020] A meter 112 may be an electric meter, a natural gas meter, a water meter or other meter. The meter 112 may operate in cooperation with one or more applications, and may create data in encrypted or non-encrypted states. In one example, a processing unit or microcontroller may provide a control and data-exchange interface with the meter 112. The meter may provide encrypted data to the application, or the application may encrypt data received from the meter. The

encrypted data may be stored in an external memory (i.e., a memory "external" to, or not part of, the microcontroller on which the software application operates).

[0021] A processing unit 114 may include one or more processors 116 communicatively coupled to memory 118. The processor(s) 116 may execute, and the memory 118 may contain, various software statements, software modules, procedures, managers, algorithms, etc. Such functional blocks may be configured in software and/or firmware, and may be executable by the processor(s) 116. In alternative embodiments, any or all of the processor(s) 116, memory 118 and/or software operable on the processor(s) and memory may be implemented in whole or in part by hardware. Examples of hardware include a microcontroller or other digital device, such as an application specific integrated circuit (ASIC) or other device configured to execute the described functions.

[0022] In one example, one, several or all of the processor(s) 116 may be microcontroller unit(s) (MCU) 120 configured for external memory decryption. The MCU 120 may be configured to read encrypted data 122 or encrypted software program statements of one or more encrypted applications 124. In the example, the MCU 120 decrypts the data 122 and/or program statements of the application(s) 124 in real time and/or "on the fly." Thus, the MCU 120 is able to execute decrypted program statements and/or utilize decrypted data. In one example, the program statements may be associated with a software application that may interface with the meter 112. In such an example, blocks of data representing encrypted versions of the program statements are decrypted, and the decrypted statements executed, by the MCU 120 configured for external memory decryption.

[0023] The memory 118, while shown as a monolithic entity, may also be configured as a plurality of similarly and/or differently configured devices, such as read-only memory, writable memory, persistent or non-persistent memory, etc.

The memory 118 may be configured to store one or more software and/or firmware modules, which are executable by the processor(s) 116 to implement various functions. The memory 118 may comprise computer-readable media and may take the form of volatile memory, such as random access memory (RAM) and/or non-volatile memory, such as read only memory (ROM) or flash RAM. Computer-readable media includes volatile and non-volatile, removable and non-removable media implemented according to any technology or techniques for storage of information such as computer-readable instructions, data structures, program modules, or other data for execution by one or more processors of a computing device. Examples of computer-readable media include, but are not limited to, phase change memory (PRAM), static random-access memory (SRAM), dynamic random-access memory (DRAM), other types of random access memory (RAM), read-only memory (ROM), electrically erasable programmable read-only memory (EEPROM), flash memory or other memory technology, compact disk read-only memory (CD-ROM), digital versatile disks (DVD) or other optical storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other non-transmission medium that can be used to store information for access by a computing device.

[0024] For purposes herein, a computer-readable media may include all or part of an application specific integrated circuit (ASIC) or other hardware device. Such a hardware device may be configured to include other functionality, including functions performed in a bandwidth management in an AMI. Accordingly, within such an integrated circuit, one or more processors are configured with executable instructions, which may be defined by logic, transistors or other components, or on-board memory.

[0025] In the illustrated example, memory 118 includes encrypted data 122 and a one or more applications 124, which may be defined by processor-executable

instructions executable by actions of the processor(s) 116 and/or the MCU 120. Accordingly, the application(s) 124 may be considered to be software, subroutines, programs, etc. Alternatively, the processor 116 and/or 120, memory 118 and/or applications 124 may be defined by operation of one or more hardware devices such as ASICs.

Example Microcontroller Unit

[0026] FIG. 2 is a schematic diagram of an example 200 of an arrangement of a microcontroller unit (MCU) 120 and an external memory device 118. The MCU 120 may be configured for decryption of data contained in the external memory device 118. The data may include information, software application program statements, and the like.

[0027] A memory device 118 is external to the MCU 120, and is therefore inherently more vulnerable to attack by those who would steal its data than if the memory device were completely enclosed within the MCU 120. However, the external memory configuration allows the memory device 118 to be much less expensive, and also allows it to be upgraded and/or replaced more easily. To protect against attack, one or both of the data 122 and software program statements (e.g., software commands) of the application 124 may be encrypted. Accordingly, while the memory device 118 is external to the MCU 120, the data and application(s) contained within the device may be protected by encryption.

[0028] An external memory interface 202 is configured to connect to an address bus 204 and a data bus 206, and to thereby interface with the external memory device 118. In a second example, the external memory could be connected to the microcontroller unit 120 by a serial link, as opposed to a parallel bus. Either configuration could allow use of the same decryption mechanism. The external memory interface 202 receives an address input 208, which controls the address

output 204. A data bus 210 transfers data obtained from the external memory device 118 to a cache memory 212.

[0029] The cache memory 212 may be configured to hold a block of encrypted data read from the external memory 202. The block of data may be read over a data bus 210. The data bus may be read at a time indicated by an input 214, which may be controlled by an address generator. In one example, the data read into the cache 212 is stored in memory locations or registers 216. Collectively, the memory locations 216 may be configured to contain one block of data. The block of data may be 128 bits, 256 bits or other quantity. In one example, a 128-bit block is used, which matches the 128-bit block of data used by AES encryption. In such an example, the registers 216 may organize the 128-bit block as N words, such as 4 words of 32-bit length. Such an example may utilize an electronic codebook (ECB), wherein message may be divided into blocks, and each block may be encrypted separately. Alternatively, non-ECB encryption techniques may be used, which may provide better security.

[0030] The cache memory device 212 may be configured to provide the block of data to a decryption engine 218. The decryption engine 218 may be a special purpose hardware device or a general purpose device configured to operate decryption software. To obtain higher throughput, a single purpose hardware device configured to decrypt AES-encrypted data is used. A symmetric key may be built into the device 218 or otherwise provided to it. In the example of FIG. 2, an external clock 220 may be used to clock the decryption engine 218, although an internal clock could alternatively be used.

[0031] Output from the decryption engine may be transferred to a second cache 222. The second cache 222 may be configured to contain a block of decrypted data organized as N words in a plurality of registers 224. In one example, the registers 224 organize the decrypted block of data as 4 words of 32-bit length.

[0032] A program counter 228 increments to a next or appropriate address, the contents of which indicate an instruction to be read into the instruction register 226. The program counter 228 may be controlled by an operating system or other device. A signal 230 from the program counter 228 may be received by the address alignment module 232. The address alignment module 232 may calculate an offset pointer 234, wherein the offset pointer indicates which of the registers 224 of the second cache 222 having decrypted data should be sent to the instruction register 226.

[0033] In one example, the program counter 228 includes an address of the external memory 118 having the desired program instruction. However, data was previously read from the external memory 118, stored in the first cache 212, decrypted by the decryption engine 218 and then stored in the second cache 222 in decrypted form. Accordingly, a decrypted form of the program instruction indicated by the program counter 228 is contained in one of the registers 224 of the second cache memory 222. The address alignment module 232, using the offset pointer 234, indicates the appropriate register.

[0034] An address generator 238 may receive an input 236 from the address alignment module 232. The input may indicate an address, which the address generator 238 may translate into an address of a block of data to fetch from the external memory 118 for storage in the first cache 212. The address generator 238 may place an address on the address bus 208/204, to select desired data from the external memory 118. In one example, the address generator 238 is configured to adapt to address buses of different widths, such as by having sufficient address space to accommodate varying memory device sizes. When the address is on the bus, an output 214 allows the address generator 238 to notify the first cache when it is time to read the data bus 210/206.

[0035] Optionally, the functionality of the address alignment module 232 and the address generator 238 may be combined. In such a configuration, a unified address alignment and generator module could be configured to: provide an offset pointer 234 to the second cache memory 222; to generate an output 208 indicating an address of a block of memory to be fetched from external memory; and to provide a signal 214 to the first cache 212 to receive data on the data bus 210.

[0036] A clock 240 may be used to provide clock signal(s) to the instruction register 226, the program counter 228 and other microcontroller functionality 242. The microcontroller functionality 242 may include a plurality of functions that are present in known microcontrollers, and which are not described herein. In one example, the clock 220 driving the decryption engine 218 may be of a higher frequency than the clock 240, although the exact design parameters can vary depending on the implementation.

Example Methods

[0037] The example methods of FIG. 3 may be implemented at least in part by the configurations of FIGS. 1 and 2. However, FIG. 3 contains general applicability, and are not limited by other drawing figures and/or prior discussion. Each method described herein is illustrated as a collection of acts, blocks or operations in a logical flow graph, which represent a sequence of operations that can be implemented in hardware, software, or a combination thereof. In the context of software, the operations represent computer-executable instructions stored on one or more computer-readable storage media that, when executed by one or more processors, perform the recited operations. Such storage media, processors and computer-readable instructions can be located within a MCU (e.g., MCU 120 of FIGS. 1 and 2) according to a desired design or implementation.

The storage media seen in FIGS. 1 and 2 is representative of storage media generally, both removable and non-removable, and of any technology. Thus, the recited operations represent actions, such as those described in FIG. 3, and are taken under control of one or more processors configured with executable instructions to perform actions indicated. Generally, computer-executable instructions include routines, programs, objects, components, data structures, and the like that perform particular functions or implement particular abstract data types. The order in which the operations are described is not intended to be construed as a limitation, and the described operations may be combined in different orders and/or in parallel to implement the method. The above discussion may apply to other methods described herein.

Example Methods of External Memory Decryption

[0038] FIG. 3 is a flow diagram illustrating an example method 300 by which a microcontroller unit (MCU) may operate to provide decryption of data contained in external memory. At operation 302, a first address in memory whose contents are to be fetched is generated. In the example of FIG. 2, the program counter 228 may give an address 230 to the address alignment module 232. The address alignment module 232 may generate an address 236, which is sent to the address generator 238.

[0039] At operation 304, a block of encrypted data is fetched from an external memory device. The fetching may be performed in response to completion of the decrypting of the data, incrementing the offset pointer to the last word in the second cache, or other timing event. In the context of example of FIG. 2 a 128-bit block, sized to be compatible with AES encryption, is fetched from external memory device 118. The block of data may comprise encrypted computer software program statements or data information. The block may be fetched

through operation of an external memory interface 202. The external memory interface may be attached to the address bus 204 and a data bus 206. Address(es) are put on to the address bus, and corresponding data is read off the data bus.

[0040] At operation 306, the fetched block of encrypted data is stored in a first cache memory. In the context of example of FIG. 2, the N words of the block of encrypted data may be stored in N registers 216 in the first cache 212. In one example, the 128-bit block is stored in 4 registers, each of 32-bit length.

[0041] At operation 308, data obtained from the first cache memory is decrypted. In the context of example of FIG. 2, the data from the first cache 212 is transferred to the decryption engine 218. The decryption engine 218 may be configured to decrypt a 128-bit block of encrypted data using a key (e.g., a symmetric key) built into the engine. The decryption engine 218 may be configured as a fast-operating hardware device, which may use the fastest clock 220 available in the MCU. The clock 220 may provide a clock signal at the decryption engine that is faster than a clock signal provided to the program counter or other areas of the microcontroller unit.

[0042] At operation 310, the decrypted data is stored in a second cache memory. In the context of example of FIG. 2, decrypted data is output from the decryption engine 218 for storage in the second cache memory 222. The address alignment module, the address generator and/or another device may signal the decryption engine to overwrite the second cache with newly decrypted data. The overwriting of the second cache may reset the offset pointer 234. In the example shown, a decrypted 128-bit block of data is stored as 4 words of 32-bit length in registers 224. However, other data block and word length sizes are possible, depending on the encryption technology used and the design of the decryption engine.

[0043] At operation 312, an address from a program counter is received. The address indicates a next statement to be executed by, and/or fetched for, the

microcontroller. In the context of example of FIG. 2, the program counter 228 may be incremented by an operating system, application of other device and/or object. Output 230 of the program counter 228 is transferred to an address alignment module 232.

[0044] At operation 314, an offset to a word (e.g. a data element) in the second cache is calculated, wherein the word represents the instruction or data indicated by the address provided by the program counter. The calculating may include incrementing an offset pointer to indicate the word contained within the second cache, wherein the incrementing is based on action by the program counter. In the context of example of FIG. 2, the address alignment module 232 processes the address received from the program counter 228. In particular, the address alignment module 232 maps the received address from a location in the external memory 118 to a register 224 in the second cache 222. Accordingly, the register 224 located in the second cache 222 contains a decrypted version of the address in the external memory 118, and is therefore usable by the MCU. The particular register 224 may be indicated by an offset or offset pointer 234. Thus, the data in the register indicated by the offset pointer 234 is a decrypted version of the data indicated by the program counter 228, which originally resided in the external memory 118. That is, the output 230 of the program counter 228 is an address to a location in external memory 118 that has been decrypted and moved to a register 224 in the second cache 222, pointed to by the offset pointer 234.

[0045] At operation 316, the data word indicated by the offset is provided to the instruction register. Accordingly, the data indicated by the address provided by the program counter, in a decrypted form found in the second cache, is provided to the program counter. In the context of example of FIG. 2, the appropriate register 224 in the second cache 222 is copied to the instruction register 226.

[0046] At operation 318, an address in the external memory is calculated for use in fetching an additional block of data. In the context of example of FIG. 2, the address alignment module 232 provides an address in output 236 to the address generator 238. The address generator 238 calculates an address to a next block of data to be fetched, based on the received address. Thus, the calculation of the address in the external memory 118 may be associated with an address received from the program counter 228 may include determining a block of data in the external memory that follows a block of data in the external memory that was most recently fetched.

[0047] At operation 320, a second block of encrypted data is fetched from the external memory. The second block of memory is associated with the address calculated at operation 316. In the context of example of FIG. 2, the address generator 208 provides appropriate address(es) to the external memory interface 202, and may provide a signal 214 to the first cache memory 212 to receive data from the external memory device 118. In one example, the second block may be fetched after some timing event, such as when the offset pointer 234 has pointed to a last word or register 224 in the second cache 222. In a further example, the second block may be fetched after the offset pointer has pointed to each register in the second cache. The offset pointer may then be reset to point at a first register, such as in response to an increment of the offset pointer when the offset point is pointing to the last register.

[0048] Note that a number of operations may be performed in parallel. As one example for purposes of illustration only, fetching the block of encrypted data (operation 302, 318), decrypting the data obtained from the first cache memory (operation 306) and providing the word indicated by the offset (operation 314) may all be performed simultaneously.

Conclusion

[0049] Although the subject matter has been described in language specific to structural features and/or methodological acts, it is to be understood that the subject matter defined in the appended claims is not necessarily limited to the specific features or acts described. Rather, the specific features and acts are disclosed as exemplary forms of implementing the claims.

CLAIMS

What is claimed is:

1. A node in a network, comprising:

a meter configured to generate encrypted data representing consumption of a resource and to record the encrypted data in a memory device located externally to a microcontroller unit;

a first cache, located within the microcontroller unit, and configured to store a block of encrypted data obtained from the memory device;

a decryption engine, located within the microcontroller unit, and configured to obtain the block of encrypted data from the first cache and to decrypt the block of encrypted data, thereby producing a block of decrypted data;

a second cache, located within the microcontroller unit, and configured to receive the block of decrypted data from the decryption engine;

an address alignment module configured to receive input from a program counter within the microcontroller unit and to calculate an offset pointer to a particular word in the block of decrypted data within the second cache;

an address generator configured to receive input from the address alignment module and to indicate a block of memory in the memory device to be loaded into the first cache; and

an application having program statements configured for operation on the microcontroller unit, the program statements being stored on the external memory in encrypted form and being executed in decrypted form in an order indicated by the offset pointer.

2. The node of claim 1, wherein the address alignment module calculates the offset pointer to the particular word based on a mapping of an address in the memory device external to the microcontroller unit to a register in the second cache.

3. A processing device, comprising:

a first cache configured to store N words of encrypted data obtained from a memory device external to the processing device;

a decryption engine configured to obtain the N words of encrypted data from the first cache and to decrypt the N words;

a second cache, configured to receive the N words of decrypted data from the decryption engine;

an address alignment and generator module, configured to:

receive input from a program counter and to calculate an offset pointer to a particular word of the N words of decrypted data within the second cache; and

indicate N words of encrypted data in the memory device external to the processing device to be loaded into the first cache.

4. The processing device of claim 3, wherein the address alignment and generator module translates the input from the program counter, which indicates a word from a particular location in external memory, into an offset to a register in the second cache.

5. The processing device of claim 3, wherein the address alignment and generator module is additionally configured to:

increment the offset pointer in response to input from the program counter, wherein the offset pointer points to a decrypted word of data for transmission to the instruction register.

6. The processing device of claim 3, wherein:

calculating the offset pointer includes translating an address, the address originally indicating a location in the memory device external to the processing device, to indicate a register in the second cache; and

indicating N words of encrypted data in the memory device includes mapping an address obtained from the program counter to indicate a block of memory in the external memory device.

7. The processing device of claim 3, wherein:

the address alignment and generator module signals the decryption engine to overwrite the second cache with newly decrypted data; and

the offset pointer is reset when decrypted data from the decryption engine overwrites the second cache.

8. The processing device of claim 3, wherein address alignment and generator module is additionally configured to:

adapt address buses of different widths; and

notify the first cache when it is time to read the data bus.

9. The processing device of claim 3, additionally comprising:

a clock configured to send a signal to the decryption engine; and

a second clock to send a signal to the instruction register and the program counter.

10. The processing device of claim 3, wherein:

the first cache contains one block of encrypted data;

the decryption engine contains one block of data in the process of decryption; and

the second cache contains one block of decrypted data.

11. A method of executing encrypted code from an external memory, comprising:

fetching a block of encrypted data from the external memory;

storing the fetched block of encrypted data in a first cache memory;

decrypting data obtained from the first cache memory;

storing the decrypted data in a second cache memory;

receiving an address from a program counter, the address indicating a next statement to be executed by a microcontroller;

calculating an offset to a word in the second cache memory, the offset based on the address received from the program counter;

providing the word indicated by the offset to an instruction register;

calculating an address in the external memory associated with the address received from the program counter; and

fetching a second block of encrypted data, the second block being associated with the calculated address.

12. The method of claim 11, wherein fetching the block of encrypted data from the external memory is in response to:

completing the decrypting of the data; and

incrementing the offset to the last word in the second cache.

13. The method of claim 11, wherein decrypting the data comprises:

decrypting the data using a decryption engine configured in hardware for AES decryption.

14. The method of claim 13, wherein decrypting the data comprises:

receiving a clock signal at the decryption engine that is faster than a clock signal provided to the program counter.

15. The method of claim 11, wherein calculating the offset to the word in the second cache memory comprises:

incrementing an offset pointer to indicate the word contained within the second cache, wherein the incrementing is based on action by the program counter.

16. The method of claim 11, wherein calculating the address in the external memory associated with the address received from the program counter comprises:

determining a block of data in the external memory that follows a block of data in the external memory that was most recently fetched.

17. The method of claim 11, wherein:

fetching the block of encrypted data, decrypting the data obtained from the first cache memory and providing the word indicated by the offset are performed simultaneously.

18. The method of claim 11, wherein:

calculating the offset to the word in the second cache memory comprises incrementing an offset pointer by one register; and

calculating the address in the external memory comprises calculating an address of a block of data in the external memory that includes the address received from the program counter.

19. The method of claim 11, wherein fetching the second block of encrypted data comprises:

fetching the second block after an offset pointer has pointed to a last word in the second cache.

20. The method of claim 11, wherein fetching the second block of encrypted data comprises:

fetching the second block after an offset pointer has pointed to each register in the second cache; and

resetting the offset pointer to point at a first register in response to an increment of the offset pointer when the offset point is pointing to the last register.

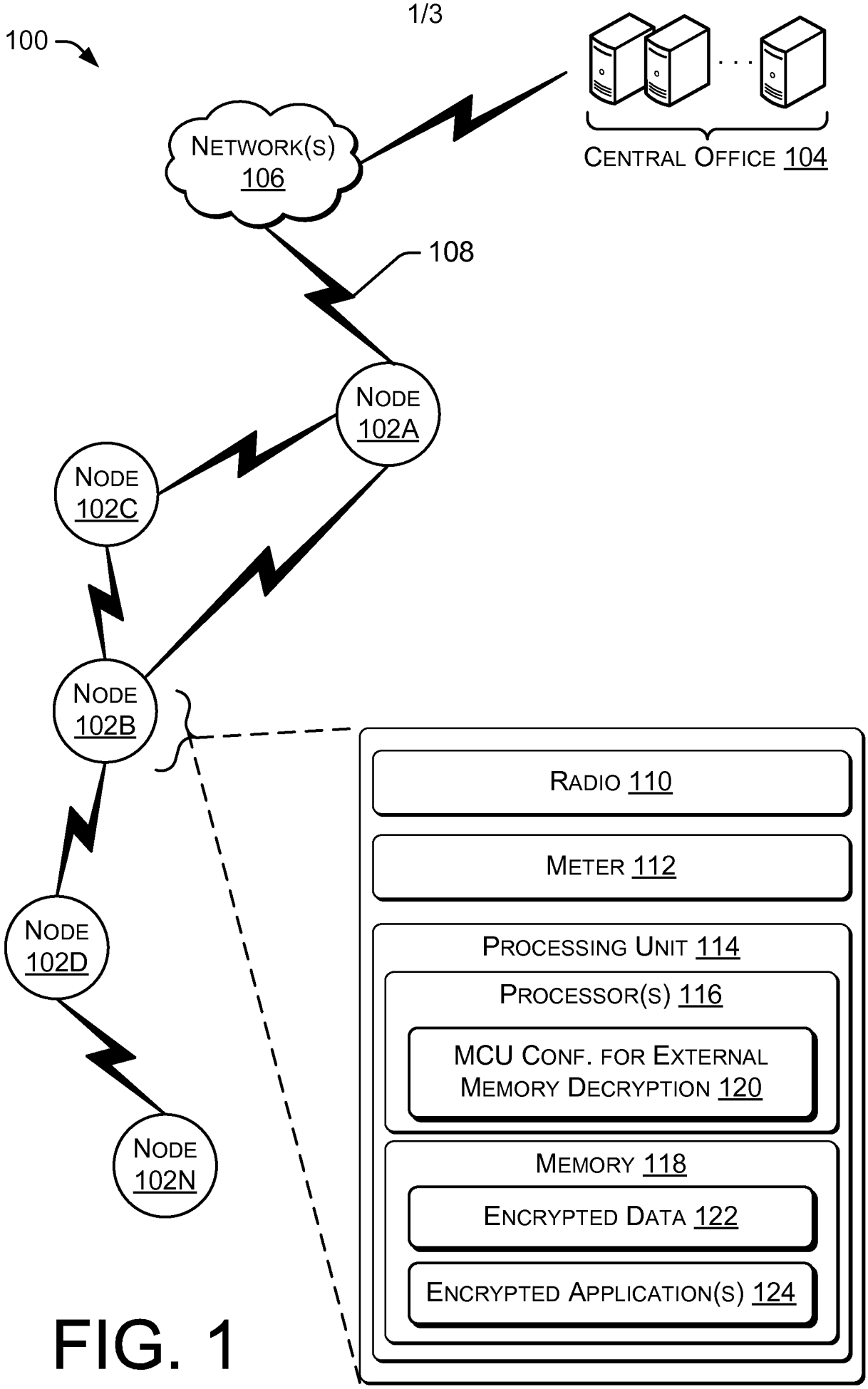
100
NETWORK(S) 106
CENTRAL OFFICE 104
108
NODE 102A
NODE 102C
NODE 102B
NODE 102D
NODE 102N
RADIO 110
METER 112
PROCESSING UNIT 114
PROCESSOR(S) 116
MCU CONF. FOR EXTERNAL MEMORY DECRYPTION 120
MEMORY 118
ENCRYPTED DATA 122
ENCRYPTED APPLICATION(S) 124

FIG. 1

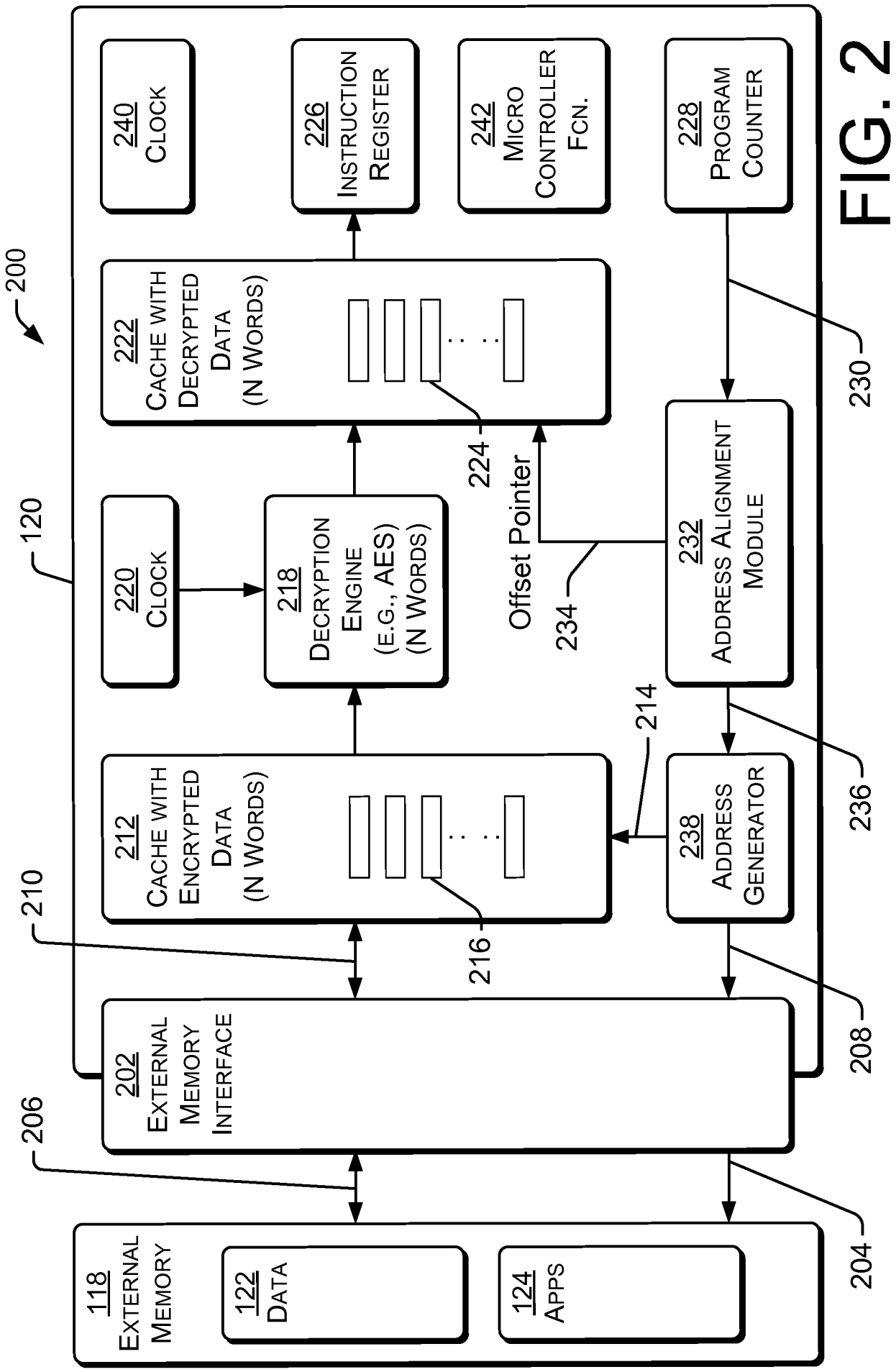
200
120
210
206
118 External Memory
122 Data
124 Apps
202 External Memory Interface
204
208
212 Cache with Encrypted Data (N Words)
216
214
238 Address Generator
236
220 Clock
218 Decryption Engine (e.g., AES) (N Words)
222 Cache with Decrypted Data (N Words)
224
Offset Pointer
234
232 Address Alignment Module
230
240 Clock
226 Instruction Register
242 Micro Controller Fcn.
228 Program Counter

FIG. 2

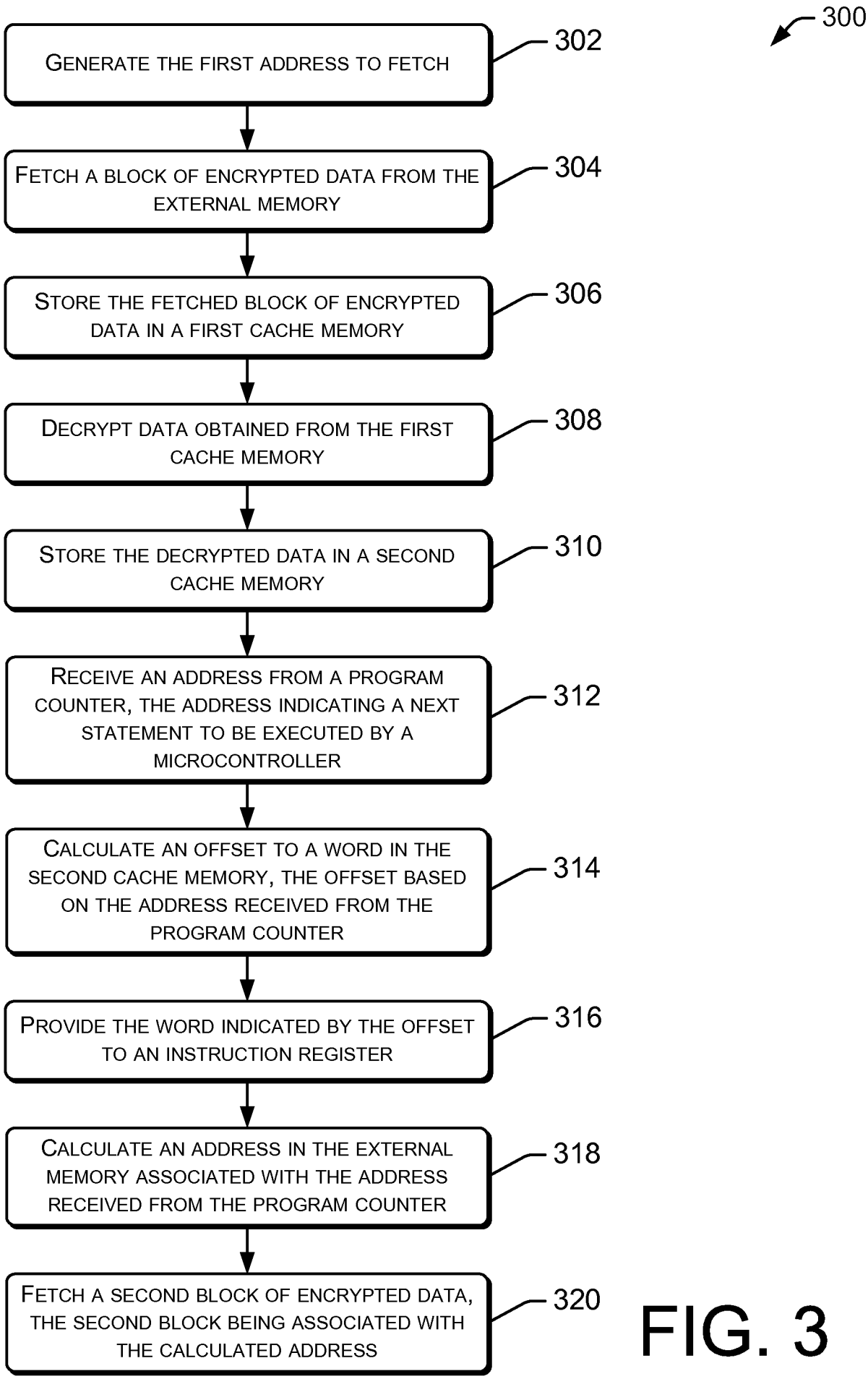
300
GENERATE THE FIRST ADDRESS TO FETCH
302
FETCH A BLOCK OF ENCRYPTED DATA FROM THE EXTERNAL MEMORY
304
STORE THE FETCHED BLOCK OF ENCRYPTED DATA IN A FIRST CACHE MEMORY
306
DECRYPT DATA OBTAINED FROM THE FIRST CACHE MEMORY
308
STORE THE DECRYPTED DATA IN A SECOND CACHE MEMORY
310
RECEIVE AN ADDRESS FROM A PROGRAM COUNTER, THE ADDRESS INDICATING A NEXT STATEMENT TO BE EXECUTED BY A MICROCONTROLLER
312
CALCULATE AN OFFSET TO A WORD IN THE SECOND CACHE MEMORY, THE OFFSET BASED ON THE ADDRESS RECEIVED FROM THE PROGRAM COUNTER
314
PROVIDE THE WORD INDICATED BY THE OFFSET TO AN INSTRUCTION REGISTER
316
CALCULATE AN ADDRESS IN THE EXTERNAL MEMORY ASSOCIATED WITH THE ADDRESS RECEIVED FROM THE PROGRAM COUNTER
318
FETCH A SECOND BLOCK OF ENCRYPTED DATA, THE SECOND BLOCK BEING ASSOCIATED WITH THE CALCULATED ADDRESS
320

FIG. 3

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2012/037309

A. CLASSIFICATION OF SUBJECT MATTER

G06F 12/00(2006.01)i, G06F 12/08(2006.01)i, G06F 12/14(2006.01)i, G06F 21/24(2006.01)i

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)

G06F 12/00; H04L 9/14; H03M 13/05; H04L 9/28; G06F 12/14; H04L 9/32; H04L 9/00; G06F 21/00

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

Korean utility models and applications for utility models
Japanese utility models and applications for utility models

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

eKOMPASS(KIPO internal) & Keywords: AMI, encrypt, decrypt, cache, memory, meter, address, controller, align, program counter, offset

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2011-0131470 A1 (KAMBAYASHI, TORU et al.) 02 June 2011 See paragraphs [0064]-[0078], [0225]-[0231] and figures 1-6, 30-31.	1-20
A	US 2012-0036355 A1 (JANG, MOON-JONG et al.) 09 February 2012 See paragraphs [0054]-[0087] and figures 2-5.	1-20
A	US 2012-0069997 A1 (KAWABATA, TAKESHI et al.) 22 March 2012 See paragraphs [0079]-[0090], [0103]-[0111] and figures 13, 17, 24.	1-20
A	US 2011-0016517 A1 (KASAHARA, MIKA et al.) 20 January 2011 See paragraphs [0086]-[0089] and figures 1, 9.	1-20

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

☒ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of 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	Date of mailing of the international search report
13 MARCH 2013 (13.03.2013)	**14 MARCH 2013 (14.03.2013)**

Name and mailing address of the ISA/KR

Korean Intellectual Property Office
189 Cheongsa-ro, Seo-gu, Daejeon Metropolitan City, 302-701, Republic of Korea

Facsimile No. 82-42-472-7140

Authorized officer

JANG, Ho Keun

Telephone No. 82-42-481-8187

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2012/037309

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2011-0131470 A1	02.06.2011	WO 2011-064883 A1	03.06.2011
US 2012-0036355 A1	09.02.2012	KR 10-1055843 B1	09.08.2011
US 2012-0069997 A1	22.03.2012	JP 2012-070048 A	05.04.2012
US 2011-0016517 A1	20.01.2011	CN 101958889 A JP 2011-040040 A	26.01.2011 24.02.2011