Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2920528 A1 2015/02/12

(21) 2 920 528

(12) DEMANDE DE BREVET CANADIEN
CANADIAN PATENT APPLICATION

(13) A1

(86) Date de dépôt PCT/PCT Filing Date: 2014/07/30
(87) Date publication PCT/PCT Publication Date: 2015/02/12
(85) Entrée phase nationale/National Entry: 2016/02/05
(86) N° demande PCT/PCT Application No.: CN 2014/083322
(87) N° publication PCT/PCT Publication No.: 2015/018290
(30) Priorité/Priority: 2013/08/06 (CN201310339295.0)

(51) Cl.Int./Int.Cl. *G06F 12/02* (2006.01)

(71) Demandeur/Applicant:
HUAWEI TECHNOLOGIES CO., LTD., CN

(72) Inventeurs/Inventors:
FAN, DONGRUI, CN;
SONG, FENGLONG, CN;
WANG, DA, CN;
YE, XIAOCHUN, CN

(74) Agent: GOWLING LAFLEUR HENDERSON LLP

(54) Titre : METHODE, APPAREIL ET SYSTEME DE TRAITEMENT D'ACCES MEMOIRE
(54) Title: MEMORY ACCESS PROCESSING METHOD AND APPARATUS, AND SYSTEM

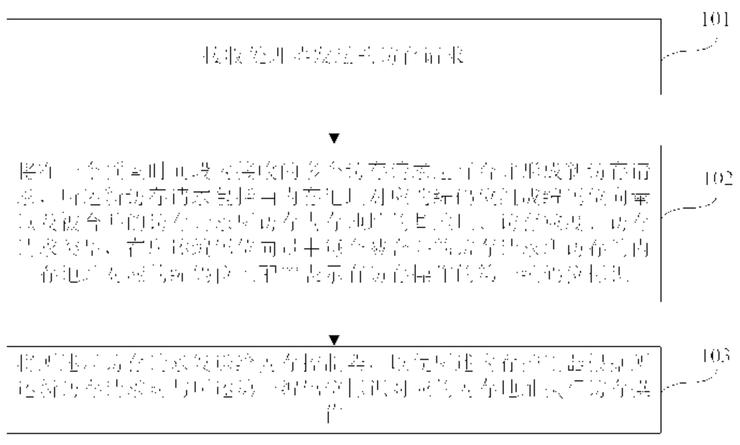

图 1 / FIG. 1

101 RECEIVE A ACCESS REQUEST SENT BY A PROCESSOR
102 COMBINE MULTIPLE ACCESS REQUESTS RECEIVED IN A PRESET TIME PERIOD INTO A NEW ACCESS REQUEST. THE NEW ACCESS REQUEST COMPRISING A CODE BIT VECTOR FORMED BY CODE BITS CORRESPONDING TO MEMORY ADDRESSES. AND BASE ADDRESSES OF THE MEMORY ADDRESSES ACCESSED BY THE COMBINED ACCESS REQUESTS. ACCESS GRANULARITY AND ACCESS REQUEST TYPES. AND CONFIGURE A FIRST CODE BIT IDENTIFICATION INDICATING A ACCESS OPERATION ON A CODE BIT CORRESPONDING TO A MEMORY ADDRESS ACCESSED BY EACH OF THE COMBINED MEMORY REQUESTS IN THE CODE BIT VECTOR
103 SEND THE NEW ACCESS REQUEST TO A MEMORY CONTROLLER, SO THAT THE MEMORY CONTROLLER EXECUTES, ACCORDING TO THE NEW ACCESS REQUEST. A ACCESS OPERATION ON A MEMORY ADDRESS CORRESPONDING TO THE FIRST CODE BIT IDENTIFICATION

(57) Abrégé/Abstract:
A memory access processing method, apparatus, and system. The method comprises: receiving a access request sent by a processor; combining multiple access requests received in a preset time period into a new access request, the new access request

(21) **2 920 528**

(13) **A1**

(57) **Abrégé(suite)/Abstract(continued):**
comprising a code bit vector formed by code bits corresponding to memory addresses, and base addresses of the memory addresses accessed by the combined access requests, access granularity and access request types, and configuring a first code bit identification indicating a access operation on a code bit corresponding to a memory address accessed by each of the combined memory requests in the code bit vector; and sending the new access request to a memory controller, so that the memory controller executes a access operation on a memory address corresponding to the first code bit identification. The method can combine multiple access requests sent by a processor into one access request, and send the access request to a memory controller to execute a access operation, thereby effectively improving utilization of memory bandwidth.

ABSTRACT

The present invention provides a memory access processing method and apparatus, and a system, where the method includes: receiving a memory access request sent by a processor; combining multiple memory access requests received within a preset time period to form a new memory access request, where the new memory access request includes a code bit vector consisting of code bits corresponding to memory addresses, a base address of the memory addresses accessed by the memory access requests that are combined, a memory access granularity, and a memory access request type; and a first code bit identifier indicating there is a memory access operation is configured for the code bits that are in the code bit vector and corresponding to the memory addresses accessed by the memory access requests that are combined; and sending the new memory access request to a memory controller, so that the memory controller executes a memory access operation on a memory address corresponding to the first code bit identifier. According to the present invention, multiple memory access requests sent by the processor can be combined into a memory access request and sent to the memory controller in a unified manner, so as to execute a memory access operation, which effectively improves memory bandwidth utilization.

MEMORY ACCESS PROCESSING METHOD AND APPARATUS, AND SYSTEM

TECHNICAL FIELD

[0001] The present invention relates to computer technologies, and in particular, to a memory access processing method and apparatus, and a system.

BACKGROUND

[0002] During a running process of a computer processor, a speed for acquiring data from an off-chip memory by the processor directly affects efficiency of the processor.

[0003] A read/write speed of an off-chip memory is much lower than a data processing speed of a processor. Therefore, in order to reduce latency for a processor to read data, a caching technique (Cache) is used in the prior art by taking advantage of temporal locality and spatial locality of a program, that is, a cache is disposed on a processor chip to cache data commonly used by the processor. A data read/write speed of the cache is relatively high. When reading data, the processor accesses the cache first; when the accessed data is not in the cache, the processor accesses an off-chip memory by using a memory controller. With the cache, work efficiency of the processor can be effectively improved. To facilitate data management, data in a cache is managed with a granularity of a cache line, such as 64 bytes (Byte). When data is read or written between the cache and an off-chip memory, data is also read into an on-chip cache together with a granularity of a cache line.

[0004] However, when the foregoing processor reads or writes data, for an application program with poor data locality, the processor needs to repeatedly access the off-chip memory by using the memory controller, which wastes relatively large access bandwidth. In addition, when a multi-core processor concurrently sends a large quantity of memory access operations to the memory controller, because a quantity of memory access requests that can be received

and processed concurrently by the memory controller is limited, some memory access requests are congested in the memory controller and cannot be processed in a timely and efficient manner.

SUMMARY

[0005] Embodiments of the present invention provide a memory access processing method and apparatus, and a system, which can improve timeliness of processing a memory access request, and can improve effective bandwidth utilization of a memory controller.

[0006] A first aspect of an embodiment of the present invention provides a memory access processing method, including:

receiving a memory access request sent by a processor;

combining multiple memory access requests received within a preset time period to form a new memory access request, where the new memory access request includes includes a base address of the memory addresses accessed by the memory access requests that are combined, a memory access granularity, and a memory access request type, and a code bit vector consisting of code bits corresponding to the memory addresses; and a first code bit identifier indicating there is a memory access operation is configured for the code bits that are in the code bit vector and corresponding to the memory addresses accessed by the memory access requests that are combined; and

sending the new memory access request to a memory controller, so that the memory controller executes, according to the new memory access request, a memory access operation on a memory address corresponding to the first code bit identifier.

[0007] With reference to the memory access processing method of the first aspect, in a first implementation manner, the combining multiple memory access requests received within a preset time period to form a new memory access request, where the new memory access request includes a base address of the memory addresses accessed by the memory access requests that are combined, a memory access granularity, and a memory access request type, and a code bit vector consisting of code bits corresponding to the memory addresses, includes:

collecting, to a same row of an address index table, the multiple memory access requests that are received within the preset time period and that have a same memory access request type, a same base address of the memory addresses corresponding to the memory access requests, and a same memory access granularity, where each row of the address index table includes a memory access request type, a base address of memory addresses accessed by memory access requests, a memory access granularity, and a code bit vector, where memory addresses corresponding to all code bits that form a code bit vector in each row have a same base address; and

extracting the memory access request type, the base address, the memory access granularity, and a code bit vector of the collected memory access requests from the same row of the address index table to form the new memory access request.

[0008] With reference to the first implementation manner of the memory access processing method of the first aspect, in a second implementation manner, if the memory access request type of the new memory access request is a read memory operation, after the sending the new memory access request to a memory controller, the method further includes:

writing data, which is returned after the memory controller executes the read memory operation according to the new memory access request, to a cache integrated in the processor; and

updating the first code bit identifier, which is in the code bit vector of the address index table and is corresponding to the executed memory access request, to a second code bit identifier that indicates there is no memory access request.

[0009] With reference to the first implementation manner or the second implementation manner of the memory access processing method of the first aspect, in a third implementation manner, if the memory access request type of the new memory access request is a write memory operation, after the sending the new memory access request to a memory controller, the method further includes:

sending data, which is corresponding to the write memory operation and is read from a cache of the processor, to the memory controller, so that the memory controller writes the data corresponding to the write memory operation to a memory address corresponding to the new memory access request; and

updating the first code bit identifier, which is in the code bit vector of the address index table and is corresponding to the executed memory access request, to a second code bit identifier that indicates there is no memory access request.

[0010] With reference to the first implementation manner, the second implementation manner or the third implementation manner of the memory access processing method of the first aspect, in a fourth implementation manner, if a quantity of bits of each memory address is A, a quantity of bits of the code bit vector of the address index table is N, and the memory access granularity is L, the base address of the collected memory access requests whose memory access granularity is L in the row of the address index table has (A – log2(N*L)) bits.

[0011] A second aspect of an embodiment of the present invention provides a memory access processing method, including:

receiving a new memory access request, where the new memory access request is formed by combining multiple memory access requests received within a preset time period, and the new memory access request includes a base address of the memory addresses accessed by the memory access requests that are combined, a memory access granularity, and a memory access request type, and a code bit vector consisting of code bits corresponding to the memory addresses; and and a first code bit identifier indicating there is a memory access operation is configured for the code bits that are in the code bit vector and corresponding to the memory addresses accessed by the memory access requests that are combined; and

obtaining, by parsing the new memory access request, the memory addresses accessed by the memory access requests that are combined, and executing a memory access operation on the memory addresses that are obtained by the parsing.

[0012] With reference to the memory access processing method of the second aspect, in a first implementation manner, the new memory access request is formed by combining the multiple memory access requests that are received within the preset time period and that have a same memory access request type, a same base address of the memory addresses corresponding to the memory access requests, and a same memory access granularity, and the obtaining, by parsing the new memory access request, the memory addresses accessed by the memory access requests that are combined includes:

acquiring the base address and the memory access granularity of the memory

access requests that are combined in the new memory access request, and position information of each first code bit identifier in the code bit vector of the new memory access request; and

if the acquired position information indicates that the first code bit identifier is in the i^{th} bit of the code bit vector, obtaining, by adding an offset address to the base address of the new memory access request, the memory addresses accessed by the memory access requests that are combined, where the offset address is a product of the acquired memory access granularity and i, and if a quantity of bits of the code bit vector of the new memory access request is N, a value of i ranges from 0 to N–1.

[0013] With reference to the first implementation manner of the memory access processing method of the second aspect, in a second implementation manner, if the memory access request type of the new memory access request is a read memory operation, executing a memory access operation on the corresponding memory addresses that are in an off-chip memory and are obtained by the parsing includes:

reading data in the accessed memory addresses that are obtained by the parsing; and

returning the read data.

[0014] With reference to the first implementation manner of the memory access processing method of the second aspect, in a third implementation manner, if the memory access request type of the new memory access request is a write memory operation, executing a memory access operation on the corresponding memory addresses that are in an off-chip memory and are obtained by the parsing includes:

acquiring data corresponding to the write memory operation; and

writing the data corresponding to the write memory operation to the accessed memory addresses that are obtained by the parsing.

[0015] A third aspect of an embodiment of the present invention provides a memory access processing apparatus, including:

an acquiring unit, configured to receive a memory access request sent by a processor;

a combining unit, configured to combine multiple memory access requests

received within a preset time period to form a new memory access request, where the new memory access request includes a code bit vector consisting of code bits corresponding to memory addresses, a base address of the memory addresses accessed by the memory access requests that are combined, a memory access granularity, and a memory access request type; and a first code bit identifier indicating there is a memory access operation is configured for the code bits that are in the code bit vector and corresponding to the memory addresses accessed by the memory access requests that are combined; and

a sending unit, configured to send the new memory access request to a memory controller, so that the memory controller executes, according to the new memory access request, a memory access operation on a memory address corresponding to the first code bit identifier.

[0016] With reference to the memory access processing apparatus of the third aspect, in a first implementation manner, the combining unit is specifically configured to: collect, to a same row of an address index table, the multiple memory access requests that are received within the preset time period and that have a same memory access request type, a same base address of the memory addresses accessed by the memory access requests, and a same memory access granularity, where each row of the address index table includes a memory access request type, a base address of memory addresses corresponding to memory access requests, a memory access granularity, and a code bit vector, where memory addresses corresponding to all code bits that form a code bit vector in each row have a same base address; and extract the memory access request type, the base address, the memory access granularity, and the code bit vector of the collected memory access requests from the same row of the address index table to form the new memory access request.

[0017] With reference to the first implementation manner of the memory access processing apparatus of the third aspect, in a second implementation manner, if the memory access request type of the new memory access request is a read memory operation, the apparatus further includes:

a data writing unit, configured to write data, which is returned after the memory controller executes the read memory operation according to the new memory access request, to a cache integrated in the processor; and

a first updating unit, configured to update the first code bit identifier, which is in the code bit vector of the address index table and is corresponding to the executed memory access request, to a second code bit identifier that indicates there is no memory access request.

[0018] With reference to the first implementation manner or the second implementation manner of the memory access processing apparatus of the third aspect, in a third implementation manner, if the memory access request type of the new memory access request is a write memory operation, the apparatus further includes:

a data reading unit, configured to send data, which is corresponding to the write memory operation and is read from a cache of the processor, to the memory controller, so that the memory controller writes the data corresponding to the write memory operation to a memory address corresponding to the new memory access request; and

a second updating unit, configured to update the first code bit identifier, which is in the code bit vector of the address index table and is corresponding to the executed memory access request, to a second code bit identifier that indicates there is no memory access request.

[0019] With reference to the first implementation manner, the second implementation manner or the third implementation manner of the memory access processing apparatus of the third aspect, in a fourth implementation manner, if a quantity of bits of each memory address is A, a quantity of bits of the code bit vector of the address index table is N, and the memory access granularity is L, the base address of the collected memory access requests whose memory access granularity is L in the row of the address index table has (A – log2(N*L)) bits.

[0020] A fourth aspect of an embodiment of the present invention provides a memory controller, including:

a second acquiring unit, configured to receive a new memory access request, where the new memory access request is formed by combining multiple memory access requests received within a preset time period, and the new memory access request includes a base address of the memory addresses accessed by the memory access requests that are combined, a memory access granularity, and a memory access request type; and a first code bit identifier indicating there is a memory access operation is configured for the code bits that are in the code bit vector and corresponding to the memory addresses accessed by the memory access requests that are combined; and

a memory access unit, configured to: obtain, by parsing the new memory access request, the memory addresses accessed by the memory access requests that are combined, and execute a memory access operation on the memory addresses that are obtained by the parsing.

[0021] With reference to the memory controller of the fourth aspect, in a first implementation manner, if the new memory access request is formed by combining the multiple memory access requests that are received within the preset time period and that have a same memory access request type, a same base address of the memory addresses corresponding to the memory access requests, and a same memory access granularity, the memory access unit is specifically configured to: acquire the base address and the memory access granularity of the memory access requests that are combined in the new memory access request, and position information of each first code bit identifier in the code bit vector of the new memory access request; if the acquired position information indicates that the first code bit identifier is in the i^{th} bit of the code bit vector, obtain, by adding an offset address to the base address of the new memory access request, the memory addresses accessed by the memory access requests that are combined, where the offset address is a product of the acquired memory access granularity and i, and if a quantity of bits of the code bit vector of the new memory access request is N, a value of i ranges from 0 to N–1; and execute a memory access operation on the obtained memory addresses accessed by the memory access requests that are combined.

[0022] With reference to the memory controller of the fourth aspect or the first implementation manner of the memory controller, in a second implementation manner, if the memory access request type of the new memory access request is a read memory operation, the memory access unit is specifically configured to: read data in the accessed memory addresses that are obtained by the parsing; and return the read data.

[0023] With reference to the memory controller of the fourth aspect or the first implementation manner of the memory controller, in a third implementation manner, if the memory access request type of the new memory access request is a write memory operation, the memory access unit is specifically configured to: acquire data corresponding to the write memory operation; and write the data corresponding to the write memory operation to the

accessed memory addresses that are obtained by the parsing.

[0024] A fifth aspect of an embodiment of the present invention provides a memory access system, including at least one processor and an off-chip memory, where the system further includes a memory access processing apparatus and a memory controller, where:

the memory access processing apparatus is configured to: combine multiple memory access requests that are sent by the processor and received within a preset time period, to form a new memory access request, where the new memory access request includes a base address of the memory addresses accessed by the memory access requests that are combined, a memory access granularity, and a memory access request type, and a code bit vector consisting of code bits corresponding to the memory addresses; and a first code bit identifier indicating there is a memory access operation is configured for the code bits that are in the code bit vector and corresponding to the memory addresses accessed by the memory access requests that are combined; and send the new memory access request to the memory controller; and

the memory controller is configured to: receive the new memory access request; and obtain, by parsing the new memory access request, the memory addresses accessed by the memory access requests that are combined, and execute a memory access operation on the memory addresses that are obtained by the parsing.

[0025] In the embodiments of the present invention, multiple memory access requests received within a preset time period are combined into a new memory access request, so that multiple memory access requests with relatively low bandwidth utilization are combined into a corresponding new memory access request with high bandwidth utilization during a memory access process, so as to execute an access operation on a memory controller in a unified manner, which reduces a quantity of memory access requests that are sent to the memory controller, helps to improve bandwidth utilization of a memory, and further ensures that the memory controller performs timely processing on a memory access request sent by a processor. When the multiple memory access requests are combined to obtain the new memory access request, a first code bit identifier is configured for a memory address accessed by each memory access request that is combined, so that when the memory controller performs memory access according to the new memory access request, the memory controller

can accurately execute a memory access operation on a corresponding memory address according to the first code bit identifier, thereby implementing effective data exchange between the processor and an off-chip memory. As a result, in the embodiments of the present invention, timeliness of processing a memory access request sent by the processor is improved, and bandwidth utilization is effectively improved when the memory controller is accessed only once, and memory bandwidth utilization is effectively improved .

BRIEF DESCRIPTION OF DRAWINGS

[0026] To describe the technical solutions in the embodiments of the present invention or in the prior art more clearly, the following briefly introduces the accompanying drawings required for describing the embodiments or the prior art. Apparently, the accompanying drawings in the following description show some embodiments of the present invention, and a person of ordinary skill in the art may still derive other drawings from these accompanying drawings without creative efforts.

[0027] FIG. 1 is a flowchart of Embodiment 1 of a memory access processing method according to the present invention;

[0028] FIG. 2 is a flowchart of Embodiment 2 of a memory access processing method according to the present invention;

[0029] FIG. 3 is a schematic structural diagram of an address index table according to an embodiment of the present invention;

[0030] FIG. 4 is a flowchart of Embodiment 3 of a memory access processing method according to the present invention;

[0031] FIG. 5 is a schematic structural diagram of Embodiment 1 of a memory access processing apparatus according to the present invention;

[0032] FIG. 6 is a schematic structural diagram of Embodiment 2 of a memory access processing apparatus according to the present invention;

[0033] FIG. 7 is a schematic structural diagram of Embodiment 3 of a memory access processing apparatus according to the present invention;

[0034] FIG. 8 is a schematic structural diagram of an embodiment of a memory controller according to the present invention;

[0035] FIG. 9 is a schematic structural diagram of Embodiment 1 of a memory access system according to the present invention;

[0036] FIG. 10 is a schematic structural diagram of Embodiment 2 of a memory access system in actual application according to the present invention;

[0037] FIG. 11 is a schematic diagram for a result of combining read memory access requests in an address index table of a memory access processing apparatus;

[0038] FIG. 12A is a schematic diagram of a memory access process of a new memory access request obtained by means of combining in FIG. 11;

[0039] FIG. 12B is a schematic diagram of a data refilling process after memory access is performed for the new memory access request obtained by means of combining in FIG. 11;

[0040] FIG. 12C is a schematic diagram of a data write-back completed state after the memory access is performed for the new memory access request obtained by means of combining in FIG. 11;

[0041] FIG. 13 is a schematic diagram for a result of combining write memory access requests in an address index table of a memory access processing apparatus; and

[0042] FIG. 14 is a schematic diagram of a memory access process of a new memory access request obtained by means of combining in FIG. 13.

DESCRIPTION OF EMBODIMENTS

[0043] To make the objectives, technical solutions, and advantages of the embodiments of the present invention clearer, the following clearly and completely describes the technical solutions in the embodiments of the present invention with reference to the accompanying drawings in the embodiments of the present invention. Apparently, the described embodiments are some but not all of the embodiments of the present invention. All other embodiments obtained by a person of ordinary skill in the art based on the embodiments of the present invention without creative efforts shall fall within the protection scope of the present invention.

[0044] FIG. 1 is a flowchart of Embodiment 1 of a memory access processing method according to the present invention. As shown in FIG. 1, the method in this embodiment is executed by a memory access processing apparatus disposed between a processor and a memory controller, and the method in this embodiment includes:

[0045] Step 101: Receive a memory access request sent by the processor.

[0046] During a running process of a computer processor such as a central processing unit (CPU), when data exchange is performed between the computer processor and an off-chip memory, the computer processor sends a memory access request to the memory controller disposed between the processor and the off-chip memory, and the memory controller reads data into an on-chip cache from the off-chip memory according to a read instruction, or writes data in the on-chip cache to the off-chip memory according to a write instruction. Herein, memory access refers to an operation of reading data from or writing data to a memory address in the off-chip memory by the processor. Therefore, a type of a memory access request sent by the processor may be a read memory access request or a write memory access request. Because data of read requests or write requests sent by the processor has different granularities and the read requests or the write requests of the data with different granularities occur randomly, if a data granularity of a read request or a write request meets a size of a cache line in the on-chip cache, the memory access request needs to be executed only once, and the processor can directly exchange data of the memory access request with the on-chip cache subsequently; if a data granularity of a read request or a write request is less than a size of a cache line, because the data granularity of the memory access operation does not meet the size of the cache line in the on-chip cache, each time the processor sends a memory access request, the processor needs to use the memory controller to exchange data of the memory access request with the off-chip memory. In this case, the memory controller frequently executes memory access operations. As a result, it is not conducive to improvement on effective bandwidth utilization of the memory controller, and some memory access requests are congested in the memory controller and cannot be processed in a timely manner. In this embodiment of the present invention, the memory access processing apparatus disposed between the processor and the memory controller can acquire a memory access request sent by the processor, and can perform, in the following processing manner, combination

processing on memory access requests with different granularities sent by the processor, so as to improve bandwidth utilization of the memory controller, and relieve a problem that memory access requests are congested in the memory controller.

[0047] Step 102: Combine multiple memory access requests received within a preset time period to form a new memory access request, where the new memory access request includes a code bit vector consisting of code bits corresponding to memory addresses, a base address of the memory addresses accessed by the memory access requests that are combined, a memory access granularity, and a memory access request type; and a first code bit identifier indicating there is a memory access operation is configured for the code bits that are in the code bit vector and corresponding to the memory addresses accessed by the memory access requests that are combined.

[0048] Currently, the memory controller separately communicates with the processor and the off-chip memory by using a limited quantity of pins. Therefore, a quantity of memory access requests simultaneously received and processed by the memory controller is limited. However, each time a memory access request whose data granularity does not meet the size of the cache line in the on-chip cache is received, the memory controller is required to execute a memory access operation on the off-chip memory. Therefore, on the premise that the memory has a limited quantity of pins, when the processor sends a large quantity of memory access requests with relatively small granularities or when a multi-core processor concurrently sends a large quantity of memory access requests to the memory controller, memory access operation for each of the memory access requests occupies some memory access bandwidth, so that some memory access requests are congested in the memory controller and fail to be processed in a timely manner. In this embodiment of the present invention, in order to ensure that a memory access request sent by the processor can be received and processed by the memory controller in a timely manner, the memory access processing apparatus disposed between the processor and the memory controller combines multiple memory access requests received within a period of time, that is, the preset time period, to form a new memory access request. The new memory access request includes a base address of the memory addresses accessed by the memory access requests that are combined, a memory access granularity, and a memory access request type, and a code bit vector consisting of code bits corresponding to

the memory addresses. Memory addresses mapped to corresponding code bits in the code bit vector include the memory addresses accessed by the memory access requests that are combined, may include a memory address that is not accessed. The first code bit identifier indicating that there is a memory access operation is configured for the code bits that are in the code bit vector and that are corresponding to the memory addresses accessed by the memory access requests that are combined. That is, the memory access processing apparatus can combine the multiple memory access requests, which need to be sent to the memory controller in many times, to form a memory access request with a relatively large granularity, and send the formed memory access request to the memory controller once, which implements fast processing on the multiple memory access requests sent by the processor, and can relieve a problem of a restriction by a pin quantity of the memory controller. In this way, access operations of the multiple memory access requests on the memory controller can be completed by executing an access operation on the memory controller only once, thereby improving bandwidth utilization of the memory controller, ensuring that memory access requests sent by the processor can be processed in a timely manner, and enhancing processor performance.

[0049] The memory access request type of the new memory access request may be a read memory operation or a write memory operation. A base address (Base_Addr) is the first address of a block of memory space, and all addresses in the block of the memory space can be obtained by adding the base address and an intra-block offset address. The memory access granularity is a volume of data that is accessed by a memory access request sent by the processor, and the memory access request may be a memory access request with different granularities, such as 8 Bytes or 16 Bytes.

[0050] In this embodiment of the present invention, to ensure that the formed new memory access request corresponds to the combined multiple memory access requests and that the new memory access request, when being processed, can accurately access data of the memory addresses corresponding to the combined multiple memory access requests, when the new memory access request is obtained by means of combination, a corresponding first code bit identifier indicating that there is a memory access operation is configured for a memory address accessed by each combined memory access request; therefore, when the memory

controller performs memory access according to the new memory access request, the memory controller can determine, according to the first code bit identifier carried in the code bit vector of the new memory access request, that there is a memory access operation in a corresponding memory address, and execute a memory access operation on the corresponding memory address.

[0051] In this embodiment of the present invention, a corresponding first code bit identifier is configured in the code bit vector of the new memory access request for the memory addresses accessed by the memory access requests that are combined, so that the memory controller can correctly obtain, by means of parsing, multiple memory addresses to be accessed by the processor. These memory addresses may be non-contiguous, and the range of memory addresses of the memory access requests that are combined are not limited by the size of the cache line in the on-chip Cache. Therefore, the method in this embodiment of the present invention can further support irregular and leaping memory access, which improves flexibility of configuring a memory address. In an existing Cache, data writing, replacement and removal are all performed in unit of a cache line (Cache Line). If the cache line is set improperly, a memory access request with a relatively small granularity cannot be processed in a timely manner. Compared with the existing cache, in this embodiment of the present invention, a memory access granularity of a memory access request sent by the processor is not limited. In specific application, the memory access processing apparatus may classify and combine multiple memory access requests by a same granularity, or a same request type, or a same range of memory access addresses, to form a memory access request with a relatively large granularity, and send, in a unified manner, the formed memory access request to the memory controller for executing a memory access operation. In addition, code bit identifiers configured for the code bit vector of the new memory access request are mapped to the memory addresses of the combined multiple memory access requests, so that during a memory access process, a processing situation of the memory access requests that are combined can be determined according to the code bit identifiers, which ensures correct execution of the memory access requests sent by the processor. In an actual operation, multiple memory access requests with relatively small granularities can be combined into a memory access request with a relatively large granularity, so that memory access operations

of multiple memory access requests on the memory controller can be implemented by accessing the memory controller only once, which improves memory bandwidth utilization, ensures that the memory controller can process, in a timely manner, a memory access request sent by the processor, and further improves flexibility of configuring a memory address in the memory access request.

[0052] Step 103: Send the new memory access request to the memory controller, so that the memory controller executes, according to the new memory access request, a memory access operation on a memory address corresponding to the first code bit identifier.

[0053] After combining the multiple memory access requests to form the new memory access request, the memory access processing apparatus can send the new memory access request as one memory access request to the memory controller, which prevents a problem that memory bandwidth utilization is relatively low because the memory controller is accessed separately the multiple memory access requests and this leads to a decrease in the throughput rate of processing the memory access requests by the memory controller.

[0054] In this embodiment of the present invention, memory access requests, sent by the processor, with a relatively small memory access granularity can be combined into a memory access request with a relatively large granularity, and memory access requests with a relatively large memory access granularity can also be combined. Memory access requests with a relatively large granularity can be combined into a memory access request with a larger granularity by using the method in this embodiment of the present invention. Likewise, multiple access operations on the memory controller by the multiple memory access requests sent by the processor can be implemented by accessing the memory controller only once.

[0055] In this embodiment of the present invention, multiple memory access requests received within a preset time period are combined into a new memory access request, so that multiple memory access requests with relatively low bandwidth utilization are combined into a corresponding new memory access request with high bandwidth utilization during a memory access process, so as to execute an access operation on a memory controller in a unified manner, which reduces a quantity of memory access requests that are sent to the memory controller, helps to improve memory bandwidth utilization, and further ensures that the memory controller performs timely processing on a memory access request sent by a

processor. When the multiple memory access requests are combined to obtain the new memory access request, a corresponding first code bit identifier is configured for a memory address accessed by each combined memory access request, so that the memory controller can accurately execute a memory access operation on a corresponding memory address according to the first code bit identifier when performing memory access according to the new memory access request, thereby implementing effective data exchange between the processor and an off-chip memory. Therefore, in this embodiment of the present invention, timeliness of processing a memory access request sent by the processor is improved, and bandwidth utilization is further effectively improved when the memory controller is accessed only once.

[0056] FIG. 2 is a flowchart of Embodiment 2 of a memory access processing method according to the present invention. As shown in FIG. 2, the method in this embodiment includes:

[0057] Step 201: Receive a memory access request sent by a processor.

[0058] Step 202: Collect, to a same row of an address index table, multiple memory access requests that are received within a preset time period and that have a same memory access request type, a same base address of memory addresses accessed by the memory access requests, and a same memory access granularity.

[0059] Step 203: Configure, according to the base address of the memory addresses accessed by the memory access requests, a first code bit identifier, which indicates there is a memory access operation, for code bits that are in a code bit vector in each row of the address index table and that are corresponding to the memory addresses accessed by the memory access requests that are combined in the row.

[0060] In this embodiment of the present invention, to enable a memory controller to fast parse and process a memory address to be accessed by a memory access request collected by a memory access processing apparatus and to simplify a structure of the memory access processing apparatus, the multiple memory access requests that are received within the preset time period, for example, 20 milliseconds, and that have the same memory access request type, the same base address of the memory addresses corresponding to the memory access requests, and the same memory access granularity, are collected to the same row of the address index table, and the first code bit identifier indicating that there is a memory access operation is

configured for the code bits that are in the code bit vector in the row of the address index table and that are corresponding to the memory addresses accessed by the memory access requests that are combined.

[0061] In this embodiment, after the memory access request sent by the processor is acquired, the multiple memory access requests that have the same memory access request type, the same base address, and the same memory access granularity are collected to the same row of the address index table, and subsequently, the memory access processing apparatus may combine the multiple memory access requests collected in the same row of the address index table to form a new memory access request, where the new memory access request corresponds to one time of access to the memory controller. After the memory controller receives the new memory access request, the memory controller executes a memory access operation for each of the memory access requests that are combined, according to each first code bit identifier configured in the new memory access request. Memory access requests concurrently sent by the processor, especially a multi-core processor, may have different granularities, may include both a read memory access request and a write memory access request, and may further be used by the processor to access data in different base address areas. Therefore, after combination processing is performed on a large quantity of memory access requests according to a same memory access granularity, a same memory access request type, and a same base address, multiple new memory access requests are formed, where each new memory access request is corresponding to one access operation on the memory controller. When the memory access processing apparatus sends a new memory access request corresponding to a row to the memory controller, the memory controller can execute a memory access operation corresponding to the memory access requests that are combined in the row. An address index table consisting of one row is used as an example for description in the following.

[0062] FIG. 3 is a schematic structural diagram of an address index table according to an embodiment of the present invention. As shown in FIG. 3, a memory access request type in the address index table may be a read memory operation or a write memory operation, which are represented by 0 and 1 respectively. The memory access request type indicates that memory access request types of the memory access requests that are combined are all read

memory operations or all write memory operations; a base address is the base address of memory addresses accessed by the memory access requests that are combined; a memory access granularity indicates a volume of data accessed by the memory access requests that are combined; each code bit in a code bit vector is corresponding to a storage address of memory space, where the memory addresses accessed by the memory access requests that are combined can be encoded by encoding the code bits in the code bit vector.

[0063] In actual application, if a memory access request is received within a preset time period, and the memory access request indicates access to a memory address, a code bit that is in the code bit vector and that is corresponding to the memory address is marked as 1, which indicates that: in a new memory access request formed by combining the memory access requests received within the preset time period, the memory address corresponding to the code bit is to be accessed during a memory access operation corresponding to the new memory access request; if no memory access request that indicates access to a memory address is received within a preset time period, a code bit that is in the code bit vector and that is corresponding to the memory address is marked as 0, which indicates that: in a new memory access request formed by combining the memory access requests received within the preset time period, the memory address corresponding to the code bit is not to be accessed during a memory access operation corresponding to the new memory access request. Therefore, after the new memory access request is formed by the combination, the memory controller can determine, by determining whether a value of the code bit in the code bit vector is 1 or 0, whether the memory address corresponding to the code bit is accessed or not during the memory access operation. Therefore, the code bit vector in the address index table is corresponding to a block of an address space that starts from a base address, each code bit in the code bit vector is mapped to an address range whose volume of data is a memory access granularity in a memory, and the code bit identifier indicates whether the data within the address range is accessed when the memory controller executes the memory access operation. In this embodiment of the present invention, the first code bit identifier indicates that there is a memory access operation in the memory address mapped to the code bit, and a second code bit identifier indicates that there is no memory access operation in the memory address mapped to the code bit.

[0064] When the combined multiple memory access requests have the same memory access request type, the same base address of the memory addresses corresponding to the memory access requests, and the same memory access granularity, a memory address mapped to the i^{th} bit of the code bit vector is a memory address that is obtained by adding an offset "granularity * i" to the base address of the address index table, that is, "base address + granularity * i". In this way, when the memory controller receives the new memory access request, the memory addresses accessed by the memory access requests that are combined and that are sent by the processor can be computed conveniently. That is, when the memory access processing apparatus uses the foregoing method to perform encoding, the memory controller can determine, according to a correspondence between the code bits and the memory addresses, the memory addresses of data to be accessed by the memory access requests sent by the processor, so that a memory access operation is accurately executed.

[0065] In this embodiment of the present invention, if a quantity of bits of a memory address in a computer system is A, a quantity of bits of the code bit vector of the address index table is N, and the memory access granularity is L, the base address of the collected memory access requests whose memory access granularity is L in the row of the address index table has (A – log2(N*L)) bits. Therefore, after the quantity of bits of the memory address in the system is determined, a quantity of bits of the base address can be computed according to the quantity of bits of the code bit vector and the memory access granularity, and the first address of a memory address segment can be further determined. When the memory access requests are received, the base address of the memory addresses can be determined according to the memory addresses accessed by the memory access requests. The following uses a system with a 32-bit (bit) memory address and memory access for reading data with an 8-Byte memory access granularity as an example for description, where memory access with a granularity of (n*8) Bytes may be considered as n 8-Byte memory access with a same base address. That is, a new memory access request with a (n*8)-Byte granularity executed by the memory controller may be formed by combining n 8-Byte memory access requests with a same base address. In actual application, a correspondence between a granularity value of the address index table and a memory access granularity of a memory access request may be preset, and the memory controller can determine the memory access granularity of the

memory access requests that are combined according to the correspondence when receiving the new memory access request sent by the memory access processing apparatus, so as to compute the base address and a memory address of each memory access request. For example, when a granularity value of the address index table is set to 0, it indicates that multiple memory access requests with an 8-Byte granularity are combined, and a memory range mapped to any 1-bit code bit in the code bit vector of the address index table is 8 Bytes. If the code bit vector has 128 bits, the memory range mapped to the code bit vector of the formed new memory access request is 1 KB (1024 bits), therefore it can be determined that the base address in the address index table has 22 bits, and the value of the base address is the upper 22 bits in the 32-bit memory address. If a memory access granularity mapped to any 1-bit code bit in the code bit vector is 32 Bytes, that is, a memory range mapped to any 1-bit code bit is 32 Bytes, and the code bit vector has 128 bits, the memory range mapped to the code bit vector of the formed new memory access request is 4 KB, and it can be determined that the base address has 20 bits and a value of the base address is the upper 20 bits in the 32-bit memory address.

[0066] Step 204: Extract the memory access request type, the base address, the memory access granularity, and the code bit vector of the collected memory access requests from the same row of the address index table to form a new memory access request.

[0067] To process, in a timely manner, the memory access requests sent by the processor and to reduce memory access latency, the memory access processing apparatus in this embodiment of the present invention extracts, from the address index table, the memory access request type, the base address of the memory addresses, the memory access granularity and the code bit vector of the memory access requests that are collected within the preset time period, to form the new memory access request.

[0068] Step 205: Send the new memory access request to a memory controller, so that the memory controller executes, according to the new memory access request, a memory access operation on a memory address that is in an off-chip memory and that is corresponding to the first code bit identifier.

[0069] Because the memory addresses accessed by the memory access requests that are combined have the same base address and the same memory access granularity, and the

memory addresses accessed by the memory access requests that are combined are addresses in a memory block, the memory controller can rapidly obtain, by parsing the new memory access request, the memory addresses accessed by the memory access requests that are combined, and execute a memory access operation. That is, the memory controller can rapidly compute the accessed memory addresses according to the base address of the new memory access request and the code bit identifier in the code bit vector. In addition, by using this combination manner in this embodiment of the present invention, the memory access processing apparatus only needs to store, when receiving the memory access requests sent by the processor, the memory access requests sequentially according to the memory access request type, the base address of the memory addresses accessed by the memory access requests and the memory access granularity, and perform real-time encoding on the code bit in the code bit vector according to the memory addresses accessed by the memory access requests. Therefore, the memory access processing apparatus can implement this embodiment of the present invention only by setting a data table with a simple structure. The simple structure is conducive to implementation of the structure of the memory access processing apparatus. In actual application, multiple rows may be set for the address index table in the memory access processing apparatus, where each row is used to collect, within a preset time period, and combine memory access requests that meet different memory access granularities, memory access request types or base addresses. When a memory access request type, a base address, a memory access granularity and a code bit vector are extracted from a same row, a new memory access request corresponding to the row can be obtained.

[0070] Step 206: Determine whether the memory access request type is a read memory operation or a write memory operation. If the memory access request type is the read memory operation, execute step 207 and step 208; if the memory access request type is the write memory operation, execute step 209 and step 210.

[0071] Step 207: Write data, which is returned after the memory controller executes the read memory operation according to the new memory access request, to a cache integrated in the processor.

[0072] Step 208: Update the first code bit identifier, which is in the code bit vector of the address index table and is corresponding to the executed memory access request, to a second

code bit identifier that indicates there is no memory access request.

[0073] After the new memory access request is sent to the memory controller, the memory controller implements a memory access operation on the off-chip memory by interacting with the off-chip memory. If the memory access request type is the read memory operation, the new memory access operation is to read data of the memory access request in the off-chip memory into an on-chip cache. During a memory access process, the memory controller returns the request data, which is returned by the off-chip memory, to the memory access processing apparatus, so that the memory access processing apparatus can write the returned data to the cache integrated in the processor, that is, cache the data in a home node (Home Node) of the data in the on-chip cache, so as to complete the memory access request sent by the processor. During this memory access process, the memory controller can parse the code bit vector according to the new memory access request to acquire a memory address mapped to each code bit, so as to acquire the memory addresses accessed by the memory access requests that are combined, and access the off-chip memory by using the acquired memory addresses. A code bit, configured with the first code bit identifier, in the code bit vector of the new memory access request is mapped to a memory address accessed by each memory access request sent by the processor. Therefore, the memory controller may return the data of these memory access requests to the on-chip cache in any sequence in several times, and record a processed memory access request by using the memory access processing apparatus to update a code bit that is in the code bit vector and that is corresponding to the memory access request whose data is returned. A situation of returning data of the read memory operation to the on-chip cache can be maintained in real time in a manner of updating a code bit. Specifically, when the code bit identifier of the code bit is 1, it indicates that the code bit is mapped to a memory access request for the off-chip memory. When the data of the memory access request has been written to the on-chip cache, the memory access processing apparatus update the code bit, which is corresponding to the memory access request whose data is returned, from 1 to 0. When code bit identifiers of all code bits in the code bit vector of the new memory access request are 0, it indicates that all the memory access requests that are combined into the new memory access request have been executed. Therefore, all entry content of the address index table can be deleted, so as to collect a new memory access request.

[0074] Step 209: Send data, which is corresponding to the write memory operation and is read from a cache of the processor, to the memory controller, so that the memory controller writes the data corresponding to the write memory operation to a memory address corresponding to the new memory access request.

[0075] Step 210: Update the first code bit identifier, which is in the code bit vector of the address index table and is corresponding to the executed memory access request, to a second code bit identifier that indicates there is no memory access request.

[0076] If the memory access request type is the write memory operation, the write memory operation is to write data of the memory access request in the on-chip cache to the off-chip memory. During a memory access process, the memory access processing apparatus extracts, according to the memory access requests and from the on-chip cache, data to be written to the off-chip memory; the memory controller generates a memory address of the off-chip memory according to the new memory access request obtained by combination, and writes the data extracted by the memory access processing apparatus to the corresponding off-chip memory to complete the write memory operation. During the write operation, the memory access processing apparatus updates, in a timely manner, the code bit that is in the code bit vector of the address index table and that is corresponding to the memory access request for which the write memory operation has been completed, so as to record an execution situation of the write memory operation, that is, update the code bit for which the write memory operation has been completed to the second code bit identifier that indicates there is no memory access request.

[0077] In this embodiment of the present invention, multiple memory access requests that are received within a preset time period and that have a same memory access request type, a same base address of memory addresses corresponding to the memory access requests, and a same memory access granularity, are collected to a same row of an address index table, and are combined to form a new memory access request, so that a memory access processing apparatus can collect and combine the memory access requests by using the address index table that has a simple structure and is easy to implement. Because the memory addresses of the multiple memory access requests that are combined to form the new memory access request have the same base address, addressing for memory access needs to be performed

only in a specific memory area, so that memory access efficiency is relatively high. The memory access requests collected and combined within the preset time period are sent to the memory controller in a timely manner, which enables the memory controller to process a corresponding memory access operation in a timely manner, helps to reducing memory access latency.

[0078] In actual application, a cache that can be integrated in a processor chip includes a programmable on-chip memory (Programmable on-chip Memory, PoM for short), such as scratch-pad memory (Scratch-Pad Memory, SPM for short), or the like， and/or includes an on-chip Cache. In this kind of cache, a PoM is corresponding to a segment of memory address space, and a storage location of data in the PoM can be determined by using a memory address, where the storage location is a home node of the data in the cache. Therefore, whether the data accessed by the processor is in the on-chip cache can be determined by using the home node. When the on-chip cache includes only the PoM, for data cached on the PoM, a location of the data in the cache can be determined according to the home node of the data. Therefore, when the processor sends a read memory access request, whether the to-be-accessed data is on the PoM can be determined according to the memory address accessed by the read memory access request sent by the processor. When the data of the read memory access request is on the PoM, the processor can directly obtain the accessed data; when the data of the read memory access request is not on the PoM, the read memory access request needs to be sent to the memory controller for accessing the off-chip memory. Therefore, when the on-chip cache includes only the PoM, the memory access processing apparatus disposed between the processor and the memory controller is used to perform combination processing on the multiple memory access requests, and the memory access processing apparatus returns and caches the data, which is obtained after the memory controller performs memory access, to each home node of the PoM. When the processor sends a write memory access request, to-be-written data is cached on the PoM according to the memory address accessed by the write memory access request, and the write memory access request is sent to the memory controller for writing the cached to-be-written data to the off-chip memory. During this write memory process, multiple write memory access requests are combined and processed by the memory access processing apparatus. After the memory

controller executes the write memory operation, data stored in the off-chip memory is consistent with the data on the on-chip PoM, and processing of the write memory operation is implemented.

[0079] When the on-chip cache includes only an on-chip Cache, and when a memory access request sent by the processor does not hit Cache data, the memory controller may be directly accessed, and the memory controller executes a memory access operation on the off-chip memory, so as to directly write data of the memory access request to the off-chip memory or read the data to the Cache from the off-chip memory. In actual application, when multiple memory access requests that do not hit cache data are concurrently sent to the memory controller, a problem of memory access congestion still occurs. In order to ensure that the memory access requests that do not hit the Cache data can also be processed in a timely manner, in this embodiment of the present invention, a management policy of the on-chip Cache is further adjusted properly. For example, an existing Cache is improved, and a home node of data is set in the Cache, so as to ensure that whether the data is in the Cache status can also be determined according to a storage location when the Cache is accessed. For the Cache on which the home node of the data is set, a manner of processing the data of the memory access request sent by the processor is similar to that of the foregoing PoM. Therefore, when the memory access request does not hit the Cache data, the Cache can send the memory access request to the memory access processing apparatus for combination processing, so as to access the memory controller in a unified manner.

[0080] When the cache has both a PoM and an on-chip Cache, an address selector disposed on the processor chip can determine, according to a memory address of a memory access request, whether the memory access request is for accessing the PoM or accessing the on-chip Cache. When the memory address belongs to PoM address space, the request is a PoM request; otherwise, the request is a Cache request. In this case, the address selector can filter a request for accessing the PoM. With respect to the request for accessing the PoM, the PoM can perform processing in the foregoing processing manner that is used when the on-chip cache includes only the PoM; with respect to the request for accessing the on-chip Cache, the on-chip Cache can perform processing in the foregoing processing manner that is used when the on-chip cache includes only the on-chip Cache.

[0081] FIG. 4 is a flowchart of Embodiment 3 of a memory access processing method according to the present invention. As shown in FIG. 4, the method in this embodiment is executed by a memory controller, and the method in this embodiment includes:

[0082] Step 301: Receive a new memory access request, where the new memory access request is formed by combining multiple memory access requests received within a preset time period, and the new memory access request includes a code bit vector consisting of code bits corresponding to memory addresses, a base address of the memory addresses accessed by the memory access requests that are combined, a memory access granularity, and a memory access request type; and a first code bit identifier indicating there is a memory access operation is configured for the code bits that are in the code bit vector and corresponding to the memory addresses accessed by the memory access requests that are combined.

[0083] Step 302: Obtain, by parsing the new memory access request, the memory addresses accessed by the memory access requests that are combined, and execute a memory access operation on the memory addresses that are obtained by the parsing.

[0084] In this embodiment, after the memory controller receives the new memory access request that is formed by combining the multiple memory access requests and sent by a memory access processing apparatus, the memory controller acquires, by parsing the new memory access request, the memory addresses accessed by the memory access requests that are combined, so as to execute a memory access operation on an off-chip memory according to the acquired memory addresses. The new memory access request, which is obtained by combining the multiple memory access requests, is sent as one memory access request in a unified manner when being sent by the memory access processing apparatus to the memory controller, thereby reducing occupation of memory bandwidth and improving utilization of the memory bandwidth. In addition, the code bits corresponding to the first code bit identifier configured by the memory access processing apparatus for the code bit vector are mapped to the memory addresses accessed by the memory access requests that are combined, so that the memory controller can obtain, by means of parsing, a memory address of the off-chip memory accessed by the memory access requests sent by a processor; therefore, the memory access requests sent by the processor can be accurately executed.

[0085] In actual application, to simplify a structure of the memory access processing

apparatus and to enable the memory controller to rapidly obtain, by means of parsing, the accessed memory address upon receipt of the new memory access request, the memory access processing apparatus in this embodiment of the present invention combines the multiple memory access requests that are collected within the preset time period and that have the same memory access request type, the same base address of the memory addresses corresponding to the memory access requests, and the same memory access granularity, to form the new memory access request. After the memory controller receives the new memory access request, the memory controller first obtains, by means of parsing, base address information and the memory access granularity of the memory access requests that are combined the new memory access request, and position information of each first code bit identifier in the code bit vector of the new memory access request, and then acquires, by means of parsing according to the first code bit identifier configured for the code bits in the code bit vector, the memory addresses accessed by the memory access requests combined by the memory access processing apparatus. A specific process of computing the accessed memory addresses is: if the acquired first code bit identifier is in the i^{th} bit of the code bit vector, obtaining, by adding an offset address to the base address of the new memory access request, the memory addresses accessed by the memory access requests that are combined, where the offset address is a product of the acquired memory access granularity and i, and if a quantity of bits of the code bit vector of the new memory access request is N, a value of i ranges from 0 to N–1. Therefore, after the memory controller parses the new memory access request to acquire the base address, the code bit vector and the memory access granularity, a memory address mapped to the i^{th} bit of the code bit vector, "base address + granularity * i", can be computed. Finally, a memory access operation corresponding to the memory access request type of the new memory access request is executed on the accessed memory address that is obtained by the parsing.

[0086] In this embodiment of the present invention, when receiving the new memory access request, the memory controller can determine, according to the memory access request type of the new memory access request, whether to execute a read memory operation or a write memory operation. If the memory access request type of the new memory access request is a read memory operation, the memory controller executes a memory access operation,

corresponding to the memory access request type of the new memory access request, on the accessed memory addresses that are obtained by the parsing, which specifically includes: reading data in the accessed memory addresses that are obtained by the parsing; returning the read data to the memory access processing apparatus, so that the memory access processing apparatus writes the returned data to a cache integrated on a processor chip; and updating a code bit identifier, which is in the code bit vector of an address index table and is corresponding to the memory access request whose data is returned, to a second code bit identifier that indicates there is no memory access request. If the memory access request type of the new memory access request is a write memory operation, the memory controller executes a memory access operation, corresponding to the memory access request type of the new memory access request, on the accessed memory addresses that are obtained by the parsing, which includes: acquiring data, sent by the memory access processing apparatus, corresponding to the write memory operation; and writing the data corresponding to the write memory operation to the accessed memory addresses that are obtained by the parsing.

[0087] In this embodiment, a memory controller executes, according to a received new memory access request, a memory access operation on a memory address that is in an off-chip memory and that is corresponding to a code bit for which a first code bit identifier is configured, thereby implementing processing on memory access requests combined by a memory access processing apparatus. During a memory access process, the memory controller implements a process of receiving and processing multiple memory access requests by receiving the memory access requests once, which effectively improves memory bandwidth utilization and ensures that a memory access request sent by a processor is processed in a timely manner.

[0088] A person of ordinary skill in the art can understand that all or a part of the steps of the method embodiments may be implemented by hardware relevant to program instructions . The program may be stored in a computer readable storage medium. When the program runs, a processor, such as a central processing unit (Central Processing Unit, CPU), performs the steps of the foregoing method embodiments. The foregoing storage medium includes: any medium that can store program code, such as a read-only memory (read-only memory, ROM), a random access memory (random access memory, RAM), a magnetic disk, or an optical disc.

[0089] FIG. 5 is a schematic structural diagram of Embodiment 1 of a memory access processing apparatus according to the present invention. As shown in FIG. 5, the apparatus in this embodiment of the present invention includes: a first acquiring unit 40, configured to receive a memory access request sent by a processor; a combining unit 41, configured to combine multiple memory access requests received within a preset time period to form a new memory access request, where the new memory access request includes a code bit vector consisting of code bits corresponding to memory addresses, a base address of the memory addresses accessed by the memory access requests that are combined, a memory access granularity, and a memory access request type, and a first code bit identifier indicating there is a memory access operation is configured for the code bits that are in the code bit vector and corresponding to the memory addresses accessed by the memory access requests that are combined; and a sending unit 42, configured to send the new memory access request to a memory controller, so that the memory controller executes, according to the new memory access request, a memory access operation on a memory address corresponding to the first code bit identifier.

[0090] The memory access processing apparatus in this embodiment of the present invention is disposed in a memory access path between the processor and the memory controller, and is configured to: combine multiple memory access requests sent by the processor to form a new memory access request, and send the new memory access request to the memory controller in a unified manner, so as to reduce memory bandwidth utilization; and use the memory controller to access an off-chip memory. The first acquiring unit 40 receives the memory access requests sent by the processor; the combining unit 41 combines the multiple memory access requests received by the first acquiring unit 40 within the preset time period, and configures the corresponding first code bit identifier for the memory addresses accessed by the memory access requests, so that the memory controller can accurately obtain, by means of parsing, the accessed memory addresses; the sending unit 42 sends, to the memory controller, the new memory access request that is obtained by combination performed by the combining unit 41, so that the memory controller executes the memory access operation on the off-chip memory according to the new memory access request.

[0091] In this embodiment of the present invention, a combining unit combines multiple

memory access requests received within a preset time period to form a new memory access request, so that multiple memory access requests with relatively low bandwidth utilization are combined into a corresponding new memory access request with high bandwidth utilization during a memory access process, so as to execute an access operation on a memory controller in a unified manner, which reduces a quantity of memory access requests that are sent to the memory controller, helps to improve memory bandwidth utilization, and further ensures that the memory controller performs timely processing on a memory access request sent by a processor. When the multiple memory access requests are combined to obtain the new memory access request, a corresponding first code bit identifier is configured for a memory address accessed by each combined memory access request, so that the memory controller can accurately execute a memory access operation on the corresponding memory address according to the first code bit identifier when performing memory access according to the new memory access request, thereby implementing effective data exchange between the processor and an off-chip memory. Therefore, in this embodiment of the present invention, timeliness of processing a memory access request sent by the processor is improved, and memory bandwidth utilization is further effectively improved during a memory access process.

[0092] FIG. 6 is a schematic structural diagram of Embodiment 2 of a memory access processing apparatus according to the present invention. As shown in FIG. 6, in this embodiment, a combining unit 41 is specifically configured to: collect, to a same row of an address index table, multiple memory access requests that are received within a preset time period and that have a same memory access request type, a same base address of memory addresses corresponding to the memory access requests, and a same memory access granularity, where each row of the address index table includes a memory access request type, a base address of memory addresses corresponding to memory access requests, a memory access granularity, and a code bit vector, and memory addresses corresponding to all code bits that form a code bit vector in each row have a same base address; and extract the memory access request type, the base address, the memory access granularity and the code bit vector of the collected memory access requests from the same row of the address index table to form a new memory access request. A determining unit 47 is configured to determine, according to

the memory access request type in each row of the address index table, whether the new memory access request corresponding to the row is a read memory operation or a write memory operation. A data writing unit 43 is configured to: when the memory access request type of the new memory access request is a read memory operation, write data, which is returned after a memory controller executes the read memory operation according to the new memory access request, to a cache integrated in a processor. A first updating unit 44 is configured to update a first code bit identifier, which is in the code bit vector of the address index table and is corresponding to the executed memory access request, to a second code bit identifier that indicates there is no memory access request. A data reading unit 45 is configured to: when the memory access request type of the new memory access request is a write memory operation, send data, which is corresponding to the write memory operation and is read from a cache of the processor, to the memory controller, so that the memory controller writes the data corresponding to the write memory operation to a memory address corresponding to the new memory access request. A second updating unit 46 is configured to update the first code bit identifier, which is in the code bit vector of the address index table and is corresponding to the executed memory access request, to the second code bit identifier that indicates there is no memory access request.

[0093] In this embodiment, the combining unit 41 collects, to a same row of the address index table, multiple memory access requests that are received within a period and that have a same memory access request type, a same base address of memory addresses corresponding to the memory access requests, and a same memory access granularity, and combines the multiple memory access requests into a new memory access request. The sending unit 42 sends, to the memory controller, the new memory access request that is acquired by combination. When the determining unit 47 determines that the corresponding new memory access request is read memory operation, that is, the memory access request sent by the processor indicates reading data from an off-chip memory into an on-chip cache, the memory controller interacts with the off-chip memory to execute the new memory access request, receives data returned by the off-chip memory, and returns the returned data to an acquiring unit 40 of the memory access processing apparatus, so that the data writing unit 43 of the memory access processing apparatus writes the memory access request data acquired by the

first acquiring unit 40 to a home node of data in the on-chip cache. During a data refilling process, data can be written to the cache progressively in many times, and the first updating unit 44 updates an identifier configured for a code bit in the code bit vector of the address index table, so as to record the memory access request whose data has been returned. When the determining unit 47 determines that the corresponding new memory access request is write memory operation, that is, the memory access request sent by the processor indicates writing data in an on-chip cache to an off-chip memory, the data reading unit 45 of the memory access processing apparatus reads the data from the on-chip cache; the sending unit 42 sends the read data to the memory controller; after receiving the new memory access request and the corresponding data, the memory controller generates a memory address of the off-chip memory according to the code bit vector and the base address, and writes the corresponding data to the generated memory address. During a data writing operation, the second updating unit 46 updates, in real time, a code bit identifier that is in the code bit vector and that is corresponding to the memory access request for which the write memory operation has been completed. During a process in which the memory access processing apparatus collects a memory access request to store it to the address index table, if a quantity of bits of a memory address is A, a quantity of bits of the code bit vector of the address index table is N, and a memory access granularity is L, the base address in a row of the collected memory access request with the memory access granularity L in the address index table has (A –log2(N*L)) bits. That is, after the quantity of bits of the memory address in a system is determined, a quantity of bits of the base address can be computed according to the quantity of bits of the code bit vector and the memory access granularity, and the first address of a memory address segment can be further determined. In this case, the received memory access request can be collected to the address index table corresponding to the base address and the memory access granularity. After the memory access processing apparatus combines the multiple memory access requests that have the same memory access request type, the same base address of the memory addresses corresponding to the memory access requests, and the same memory access granularity, the memory controller can accurately obtain, by means of parsing and according to the first code bit identifier configured for the code bit vector, the memory addresses accessed by the memory access requests that are combined, so as to

execute a memory access operation.

[0094] In this embodiment, multiple memory access requests that have a same memory access request type, a same base address of memory addresses corresponding to the memory access requests, and a same memory access granularity, are combined, which enables a memory controller to process the memory access requests sent by a processor in a timely and accurate manner, helps to improve bandwidth utilization of the memory controller, and further makes a structure of a memory access processing apparatus simple and easy to implement.

[0095] FIG. 7 is a schematic structural diagram of Embodiment 3 of a memory access processing apparatus according to the present invention. As shown in FIG. 7, the memory access processing apparatus in this embodiment includes a processor 80, a memory 82, a communications interface 84, and a bus 83, where the processor 80, the communications interface 84 and the memory 82 communicate with each other by using the bus 83; the memory 82 is configured to store a program instruction, for example, instruction code corresponding to operation steps of the memory access processing method in the foregoing embodiment; the processor 80 can be configured to execute the program instruction stored in the memory 82; the communications interface 84 can be configured to send or receive the program instruction between the processor and the memory or data that is generated according to the program instruction; and the bus 83 is configured to implement communication between function units inside the memory access processing apparatus.

[0096] FIG. 8 is a schematic structural diagram of an embodiment of a memory controller according to the present invention. As shown in FIG. 8, in this embodiment, the memory controller includes: a second acquiring unit 50, configured to receive a new memory access request, where the new memory access request is formed by combining multiple memory access requests received within a preset time period, and the new memory access request includes a code bit vector consisting of code bits corresponding to memory addresses, a base address of the memory addresses accessed by the memory access requests that are combined, a memory access granularity and a memory access request type, and a first code bit identifier indicating there is a memory access operation is configured for the code bits that are in the code bit vector and corresponding to the memory addresses accessed by the memory access requests that are combined; and a memory access unit 51, configured to obtain, by parsing the

new memory access request, the memory addresses accessed by the memory access requests that are combined, and execute a memory access operation on the memory addresses that are obtained by the parsing.

[0097] In this embodiment, after the second acquiring unit 50 of the memory controller receives the new memory access request that is formed by combination performed by a memory access processing apparatus, because the new memory access request is configured with the first code bit identifier corresponding to the memory addresses accessed by the memory access requests, the memory controller can obtain, by means of parsing, a to-be-accessed memory address according to the configured first code bit identifier, and execute a corresponding memory access operation according to the memory access granularity and the memory access request type of the new memory access request, so as to complete the memory access operation for a processor. By receiving one new memory access request in a unified manner, the memory controller can complete receiving processing on multiple memory access requests sent by the processor, thereby improving bandwidth utilization and reducing memory access latency.

[0098] To enable the memory controller to rapidly compute a memory address for a memory access operation and to implement a simple structure of the memory access processing apparatus, when the memory access processing apparatus combines the multiple memory access requests that are received within the preset time period and that have the same memory access request type, the same base address of the memory addresses corresponding to the memory access requests, and the same memory access granularity, to form the new memory access request, after the second acquiring unit 50 of the memory controller acquires the new memory access request sent by the memory access processing apparatus, the memory access unit 50 obtains, by means of parsing, the base address and the memory access granularity of the memory access requests that are combined in the new memory access request, and position information of each first code bit identifier in the code bit vector of the new memory access request; if the acquired position information indicates that the first code bit identifier is in the i^{th} bit of the code bit vector, the memory access unit 51 obtains, by adding an offset address to the base address of the new memory access request, the memory addresses accessed by the memory access requests that are combined, where the offset

address is a product of the acquired memory access granularity and i, and if a quantity of bits of the code bit vector of the new memory access request is N, a value of i ranges from 0 to N–1; the memory access unit 51 executes a memory access operation on the memory addresses accessed by the memory access requests that are combined that are obtained by the acquiring unit.

[0099] If the memory access request type of the new memory access request is a read memory operation, the memory access unit is specifically configured to read data in the accessed memory addresses that are obtained by the parsing, and return the read data to the memory access processing apparatus, so that the memory access apparatus writes the returned data to a cache integrated in the processor. If the memory access request type of the new memory access request is a write memory operation, the memory access unit is specifically configured to acquire data that is corresponding to the write memory operation and sent by the memory access processing apparatus, and write the data corresponding to the write memory operation to the accessed memory addresses that are obtained by the parsing.

[0100] In this embodiment of the present invention, after a second acquiring unit of a memory controller acquires a new memory access request that is formed by combination performed by a memory access processing apparatus, a memory access unit acquires, by means of parsing and according to a first code bit identifier configured for a code bit vector of the new memory access request, memory addresses accessed by memory access requests that are combined, and executes a memory access operation on the memory addresses that are obtained by the parsing. In this way, instead of separately performing an access operation for many times on the memory controller for multiple memory access requests sent by a processor, one access operation is accurately executed on an off-chip memory for one new memory access request, thereby improving memory bandwidth utilization and reducing memory access latency.

[0101] An embodiment of the present invention further provides a memory access system, including: at least one processor, an off-chip memory, the foregoing memory access processing apparatus shown in FIG. 5, FIG. 6 or FIG. 7, and the memory controller shown in FIG. 8. The memory access processing apparatus is configured to: combine multiple memory access requests that are sent by the processor and received within a preset time period, to form

a new memory access request, where the new memory access request includes a code bit vector consisting of code bits corresponding to memory addresses, a base address of the memory addresses accessed by the memory access requests that are combined, a memory access granularity, and a memory access request type; and a first code bit identifier indicating there is a memory access operation is configured for the code bits that are in the code bit vector and corresponding to the memory addresses accessed by the memory access requests that are combined; and send the new memory access request to the memory controller. The memory controller is configured to: receive the new memory access request, obtain, by parsing the new memory access request, the memory addresses accessed by the memory access requests that are combined, and execute a memory access operation on the memory addresses that are obtained by the parsing.

[0102] FIG. 9 is a schematic structural diagram of Embodiment 1 of a memory access system according to the present invention. As shown in FIG. 9, the system in this embodiment includes multiple processing logic 601, multiple on-chip memories 602, a memory access processing apparatus 603, a memory controller 604, and an off-chip memory 605, where the multiple processing logic 601 may be corresponding to multiple processor cores; the multiple on-chip memories 602 are on-chip caches; and the memory access processing apparatus 603 is configured to combine memory access requests of different granularities, which are sent by the processing logic 601 through the on-chip memories 602, to form a memory access request with a relatively large granularity, where the process of combining memory access requests includes:

receiving a memory access request sent by a processor;

combining multiple memory access requests received within a preset time period to form a new memory access request, where the new memory access request includes a code bit vector consisting of code bits corresponding to memory addresses, a base address of the memory addresses accessed by the memory access requests that are combined, a memory access granularity, and a memory access request type, and a first code bit identifier indicating there is a memory access operation is configured for the code bits that are in the code bit vector and corresponding to the memory addresses accessed by the memory access requests that are combined; and

sending the new memory access request to the memory controller 604, so that the memory controller executes, according to the new memory access request, a memory access operation on a memory address that is in the off-chip memory 605 and that is corresponding to the first code bit identifier.

[0103] The memory access processing apparatus 603 can combine the multiple memory access requests and send the memory access requests to the memory controller in a unified manner, and then the memory controller performs the memory access operation on the off-chip memory, thereby improving memory bandwidth utilization effectively. The memory access processing apparatus 603 in this embodiment may be the foregoing apparatus shown in FIG. 5, FIG. 6 or FIG. 7.

[0104] In actual application, the foregoing on-chip memory in FIG. 9 may include a PoM and a Cache, and FIG. 10 is a schematic structural diagram of Embodiment 2 of a memory access system in actual application according to the present invention. As shown in FIG. 10, a processor chip in the system includes multiple address selectors 702 that are separately corresponding to multiple processing logic 701, where an address selector 702 is configured to determine, according to a memory access request sent by each processing logic 701, whether a memory access address for the memory access request is in a PoM address space; and if the memory access address for the memory access request is in the PoM address space, send the memory access request to a PoM 704; otherwise, send the memory access request to a Cache 703. The memory access request sent to the PoM 704 is sent to an address index table in a memory access processing apparatus 705 through the PoM 704, and the memory access request is collected and combined in the address index table. If a memory access request type, a base address, and a granularity of the current memory access request are different from those in any existing row in the address index table, a new row is allocated in the address index table according to the memory access request type of the current memory access request, the base address corresponding to the memory address accessed by the current memory access request, and the memory access granularity; and a code bit in a code bit vector of the current memory access request is encoded. If the memory access request type, the base address and the memory access granularity of the current memory access request are the same as those in an existing row in the address index table, the current memory access request is combined

into the row, and a code bit corresponding to the memory address of the current memory access request is encoded. When the address selector 702 determines that the memory address is not in the PoM address space, the address selector 702 sends the memory access request to the Cache 703, and the Cache 703 sends the memory access request to a memory controller 706 for executing a memory access operation.

[0105] In actual application, if the Cache 73 is an on-chip Cache in which a home node of data is set, that is, a management policy of the on-chip Cache is modified partially to add a home node of data, the Cache may also send the memory access request to the memory access processing apparatus for combination processing. Therefore, when the Cache 703 receives the memory access request, if data of the request is in the Cache, the data is directly returned to the processing logic 701; if data of the request is not in the Cache, that is, Cache data is not hit, the Cache may send the memory access request that does not hit the data (Cache Miss request) to the memory access processing apparatus 705. The memory access processing apparatus 705 may combine memory access requests that do not hit the data and send the memory access requests to the memory controller in a unified manner. For a process of combination processing performed by the memory access processing apparatus on the memory access requests that do not hit the data, refer to the embodiment shown in FIG. 1 or FIG. 2, and details are not described herein again. If all the foregoing on-chip memories in the embodiment shown in FIG. 9 are Caches, a manner in which the memory access processing apparatus performs, by setting a home node of the data, combination processing on the memory access requests that do not hit the Cache data may also be used. The memory access processing apparatus 705 sends new memory access requests, formed by combination, in the address index table to the memory controller 706, so that the memory controller interacts with an off-chip memory 707 in a timely manner to implement a memory access operation.

[0106] The following uses a system in which a 32-bit memory address is used, a read memory operation and a write memory operation with an 8-byte memory access granularity as an example for detailed description.

[0107] FIG. 11 is a schematic diagram for a result of combining read memory access requests in an address index table of a memory access processing apparatus; FIG. 12A is a schematic diagram of a memory access process of a new memory access request obtained by

means of combining in FIG. 11; FIG. 12B is a schematic diagram of a data refilling process after memory access is performed for the new memory access request obtained by means of combining in FIG. 11; FIG. 12C is a schematic diagram of a data write-back completed state after the memory access is performed for the new memory access request obtained by means of combining in FIG. 11. As shown in FIG. 11, and FIG. 12A to FIG. 12C, it can be determined, according to an 8-Byte granularity value and a 128-bit code bit vector, that a base address has 22 bits, and two read memory access requests with an 8-byte granularity are combined into the new memory access request, that is, a read memory operation is executed by the new memory access request on a memory address that is corresponding to a code bit whose identifier is 1 in the code bit vector in FIG. 11. Because the operation is a read memory operation, a memory controller can directly perform the read memory operation on an off-chip memory according to the new memory access request. Specifically, the memory controller computes the memory address according to the code bit whose identifier is 1 and the base address of the new memory access request, and performs a read operation on the off-chip memory; after data of a memory address corresponding to a code bit is refilled to a PoM, the corresponding code bit is updated. After data is refilled to the address index table in FIG. 11, the address index table is in a state shown in FIG. 12B. When all code bits in a code bit vector in a row of the address index table are 0, it indicates that data refilling of all memory access requests that are combined in the row is complete. As shown in FIG. 12C, entry content in the row of the address index table may be deleted.

[0108] FIG. 13 is a schematic diagram for a result of combining write memory access requests in an address index table of a memory access processing apparatus; FIG. 14 is a schematic diagram of a memory access process of a new memory access request obtained by means of combining in FIG. 13. As shown in FIG. 13 and FIG. 14, it may be determined, according to an 8-Byte granularity value and a 128-bit code bit vector, that a base address has 22 bits, and two write memory access requests with a 8-Byte granularity are combined into the new memory access request, that is, a write memory operation is executed by the new memory access request on a memory address that is corresponding to a code bit whose identifier is 1 in the code bit vector in FIG. 13; the memory access processing apparatus reads data of the memory address corresponding to the code bit from a PoM or a Cache. Because

the operation is a write memory operation, after the new memory access request is sent to a memory controller, the memory controller generates the memory address by means of computing according to the code bit in the code bit vector and the base address, and writes the data, which is read by the memory access processing apparatus, to the memory address of an off-chip memory generated by the memory controller. During the write memory operation, the memory access processing apparatus updates the code bit of the corresponding memory address on which the write operation has been completed by the memory controller, and the write memory operation is complete until all the code bits are 0 in the code bit vector. The memory access processing apparatus may delete entry content in a row, in which a code bit vector whose code bits are all 0 is located, of the address index table.

[0109] It should be noted that the foregoing embodiments are merely intended for describing the technical solutions of the present invention but not for limiting the present invention. Although the present invention is described in detail with reference to the exemplary embodiments, a person of ordinary skill in the art should understand that they may still make modifications or equivalent replacements to the technical solutions of the present invention without departing from the spirit and scope of the present invention.

CLAIMS

What is claimed is:

1. A memory access processing method, comprising:

receiving a memory access request sent by a processor;

combining multiple memory access requests received within a preset time period to form a new memory access request, wherein the new memory access request comprises a code bit vector consisting of code bits corresponding to memory addresses, a base address of the memory addresses accessed by the memory access requests that are combined, a memory access granularity, and a memory access request type; and a first code bit identifier indicating there is a memory access operation is configured for the code bits that are in the code bit vector and corresponding to the memory addresses accessed by the memory access requests that are combined; and

sending the new memory access request to a memory controller, so that the memory controller executes, according to the new memory access request, a memory access operation on a memory address corresponding to the first code bit identifier.

2. The method according to claim 1, wherein the combining multiple memory access requests received within a preset time period to form a new memory access request, wherein the new memory access request comprises a code bit vector consisting of code bits corresponding to memory addresses, a base address of the memory addresses accessed by the memory access requests that are combined, a memory access granularity, and a memory access request type, comprises:

collecting, to a same row of an address index table, the multiple memory access requests that are received within the preset time period and that have a same memory access request type, a same base address of the memory addresses corresponding to the memory access requests, and a same memory access granularity, wherein each row of the address index table comprises a memory access request type, a base address of memory addresses accessed by the memory access requests, a memory access granularity, and a code bit vector, wherein memory addresses corresponding to all code bits that form a code bit vector in each row have a same

base address; and

extracting the memory access request type, the base address, the memory access granularity, and the code bit vector of the collected memory access requests from the same row of the address index table to form the new memory access request.

3. The method according to claim 2, wherein: if the memory access request type of the new memory access request is a read memory operation, after the sending the new memory access request to a memory controller, the method further comprises:

writing data, which is returned after the memory controller executes the read memory operation according to the new memory access request, to a cache integrated in the processor; and

updating the first code bit identifier, which is in the code bit vector of the address index table and is corresponding to the executed memory access request, to a second code bit identifier that indicates there is no memory access request.

4. The method according to claim 2, wherein: if the memory access request type of the new memory access request is a write memory operation, after the sending the new memory access request to a memory controller, the method further comprises:

sending data, which is corresponding to the write memory operation and is read from a cache of the processor, to the memory controller, so that the memory controller writes the data corresponding to the write memory operation to a memory address corresponding to the new memory access request; and

updating the first code bit identifier, which is in the code bit vector of the address index table and is corresponding to the executed memory access request, to a second code bit identifier that indicates there is no memory access request.

5. The method according to any one of claims 2 to 4, wherein: if a quantity of bits of each memory address is A, a quantity of bits of the code bit vector of the address index table is N, and the memory access granularity is L, the base address of the collected memory access requests whose memory access granularity is L in the row of the address index table has (A – log2(N*L)) bits.

6. A memory access processing method, comprising:

receiving a new memory access request, wherein the new memory access request is

formed by combining multiple memory access requests received within a preset time period, and the new memory access request comprises a base address of the memory addresses accessed by the memory access requests that are combined, a memory access granularity, and a memory access request type, and a code bit vector consisting of code bits corresponding to the memory addresses; and a first code bit identifier indicating there is a memory access operation is configured for the code bits that are in the code bit vector and corresponding to the memory addresses accessed by the memory access requests that are combined; and

obtaining, by parsing the new memory access request, the memory addresses accessed by the memory access requests that are combined, and executing a memory access operation on the memory addresses that are obtained by the parsing.

7. The method according to claim 6, wherein that the new memory access request is formed by combining multiple memory access requests received within a preset time period comprises:

the new memory access request is formed by combining the multiple memory access requests that are received within the preset time period and that have a same memory access request type, a same base address of the memory addresses corresponding to the memory access requests, and a same memory access granularity.

8. The method according to claim 7, wherein the obtaining, by parsing the new memory access request, the memory addresses accessed by the memory access requests that are combined comprises:

acquiring the base address and the memory access granularity of the memory access requests that are combined in the new memory access request, and position information of each first code bit identifier in the code bit vector of the new memory access request; and

if the acquired position information indicates that the first code bit identifier is in the i^{th} bit of the code bit vector, obtaining, by adding an offset address to the base address of the new memory access request, the memory addresses accessed by the memory access requests that are combined, wherein the offset address is a product of the acquired memory access granularity and i, and if a quantity of bits of the code bit vector of the new memory access request is N, a value of i ranges from 0 to N–1.

9. The method according to claim 6, 7 or 8, wherein: if the memory access request type

of the new memory access request is a read memory operation, executing a memory access operation on the corresponding memory addresses that are in an off-chip memory and are obtained by the parsing comprises:

reading data in the accessed memory addresses that are obtained by the parsing; and

returning the read data.

10. The method according to claim 6, 7 or 8, wherein: if the memory access request type of the new memory access request is a write memory operation, executing a memory access operation on the corresponding memory addresses that are in an off-chip memory and are obtained by the parsing comprises:

acquiring data corresponding to the write memory operation; and

writing the data corresponding to the write memory operation to the accessed memory addresses that are obtained by the parsing.

11. A memory access processing apparatus, comprising:

a first acquiring unit, configured to receive a memory access request sent by a processor;

a combining unit, configured to combine multiple memory access requests received within a preset time period to form a new memory access request, wherein the new memory access request comprises a code bit vector consisting of code bits corresponding to memory addresses, a base address of the memory addresses accessed by the memory access requests that are combined, a memory access granularity, and a memory access request type; and a first code bit identifier indicating there is a memory access operation is configured for the code bits that are in the code bit vector and corresponding to the memory addresses accessed by the memory access requests that are combined; and

a sending unit, configured to send the new memory access request to a memory controller, so that the memory controller executes, according to the new memory access request, a memory access operation on a memory address corresponding to the first code bit identifier.

12. The apparatus according to claim 11, wherein the combining unit is specifically configured to: collect, to a same row of an address index table, the multiple memory access requests that are received within the preset time period and that have a same memory access request type, a same base address of the memory addresses accessed by the memory access

requests, and a same memory access granularity, wherein each row of the address index table comprises a memory access request type, a base address of memory addresses corresponding to memory access requests, a memory access granularity, and a code bit vector, wherein memory addresses corresponding to all code bits that form a code bit vector in each row have a same base address; and extract the memory access request type, the base address, the memory access granularity, and the code bit vector of the collected memory access requests from the same row of the address index table to form the new memory access request.

13. The apparatus according to claim 12, wherein: if the memory access request type of the new memory access request is a read memory operation, the apparatus further comprises:

a data writing unit, configured to write data, which is returned after the memory controller executes the read memory operation according to the new memory access request, to a cache integrated in the processor; and

a first updating unit, configured to update the first code bit identifier, which is in the code bit vector of the address index table and is corresponding to the executed memory access request, to a second code bit identifier that indicates there is no memory access request.

14. The apparatus according to claim 12, wherein: if the memory access request type of the new memory access request is a write memory operation, the apparatus further comprises:

a data reading unit, configured to send data, which is corresponding to the write memory operation and is read from a cache of the processor, to the memory controller, so that the memory controller writes the data corresponding to the write memory operation to a memory address corresponding to the new memory access request; and

a second updating unit, configured to update the first code bit identifier, which is in the code bit vector of the address index table and is corresponding to the executed memory access request, to a second code bit identifier that indicates there is no memory access request.

15. The apparatus according to any one of claims 11 to 14, wherein: if a quantity of bits of each memory address is A, a quantity of bits of the code bit vector of the address index table is N, and the memory access granularity is L, the base address of the collected memory access requests whose memory access granularity is L in the row of the address index table has (A – log2(N*L)) bits.

16. A memory controller, comprising:

a second acquiring unit, configured to receive a new memory access request, wherein the new memory access request is formed by combining multiple memory access requests received within a preset time period, and the new memory access request comprises a code bit vector consisting of code bits corresponding to memory addresses, a base address of the memory addresses accessed by the memory access requests that are combined, a memory access granularity, and a memory access request type; and a first code bit identifier indicating there is a memory access operation is configured for the code bits that are in the code bit vector and corresponding to the memory addresses accessed by the memory access requests that are combined; and

a memory access unit, configured to: obtain, by parsing the new memory access request, the memory addresses accessed by the memory access requests that are combined, and execute a memory access operation on the memory addresses that are obtained by the parsing.

17. The memory controller according to claim 16, wherein: if the new memory access request is formed by combining the multiple memory access requests that are received within the preset time period and that have a same memory access request type, a same base address of the memory addresses corresponding to the memory access requests, and a same memory access granularity, the memory access unit is specifically configured to: acquire the base address and the memory access granularity of the memory access requests that are combined in the new memory access request, and position information of each first code bit identifier in the code bit vector of the new memory access request; if the acquired position information indicates that the first code bit identifier is in the i^{th} bit of the code bit vector, obtain, by adding an offset address to the base address of the new memory access request, the memory addresses accessed by the memory access requests that are combined, wherein the offset address is a product of the acquired memory access granularity and i, and if a quantity of bits of the code bit vector of the new memory access request is N, a value of i ranges from 0 to N–1; and execute a memory access operation on the obtained memory addresses accessed by the memory access requests that are combined.

18. The memory controller according to claim 16 or 17, wherein: if the memory access request type of the new memory access request is a read memory operation, the memory access unit is specifically configured to: read data in the accessed memory addresses that are

obtained by the parsing; and return the read data.

19. The memory controller according to claim 16 or 17, wherein: if the memory access request type of the new memory access request is a write memory operation, the memory access unit is specifically configured to: acquire data corresponding to the write memory operation; and write the data corresponding to the write memory operation to the accessed memory addresses that are obtained by the parsing.

20. A memory access system, comprising at least one processor and an off-chip memory, wherein the system further comprises a memory access processing apparatus and a memory controller, wherein:

the memory access processing apparatus is configured to: combine multiple memory access requests that are sent by the processor and received within a preset time period, to form a new memory access request, wherein the new memory access request comprises a code bit vector consisting of code bits corresponding to memory addresses, a base address of the memory addresses accessed by the memory access requests that are combined, a memory access granularity, and a memory access request type; and a first code bit identifier indicating there is a memory access operation is configured for the code bits that are in the code bit vector and corresponding to the memory addresses accessed by the memory access requests that are combined; and send the new memory access request to the memory controller; and

the memory controller is configured to: receive the new memory access request; and obtain, by parsing the new memory access request, the memory addresses accessed by the memory access requests that are combined, and execute a memory access operation on the memory addresses that are obtained by the parsing.

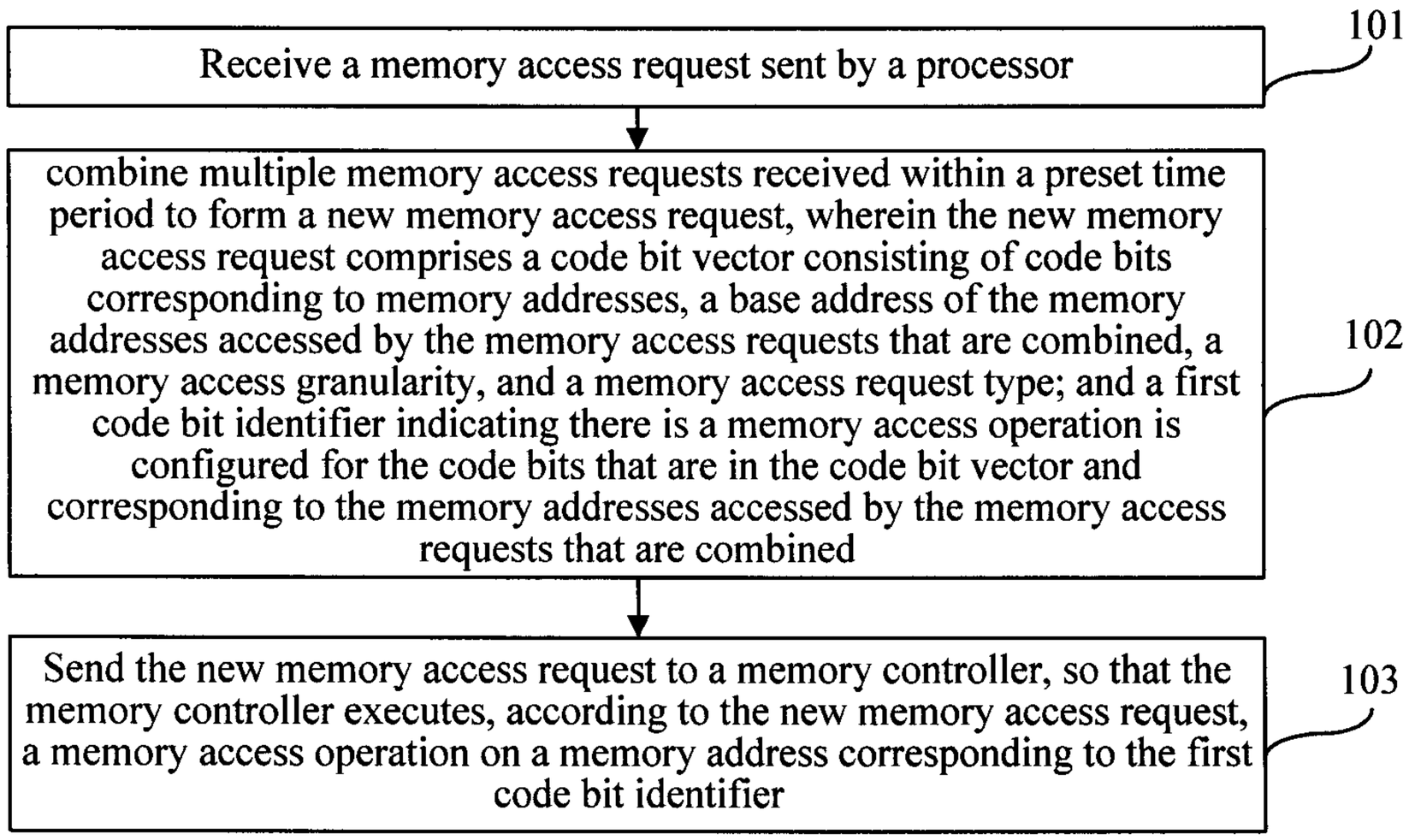
Receive a memory access request sent by a processor
101
combine multiple memory access requests received within a preset time period to form a new memory access request, wherein the new memory access request comprises a code bit vector consisting of code bits corresponding to memory addresses, a base address of the memory addresses accessed by the memory access requests that are combined, a memory access granularity, and a memory access request type; and a first code bit identifier indicating there is a memory access operation is configured for the code bits that are in the code bit vector and corresponding to the memory addresses accessed by the memory access requests that are combined
102
Send the new memory access request to a memory controller, so that the memory controller executes, according to the new memory access request, a memory access operation on a memory address corresponding to the first code bit identifier
103

FIG. 1

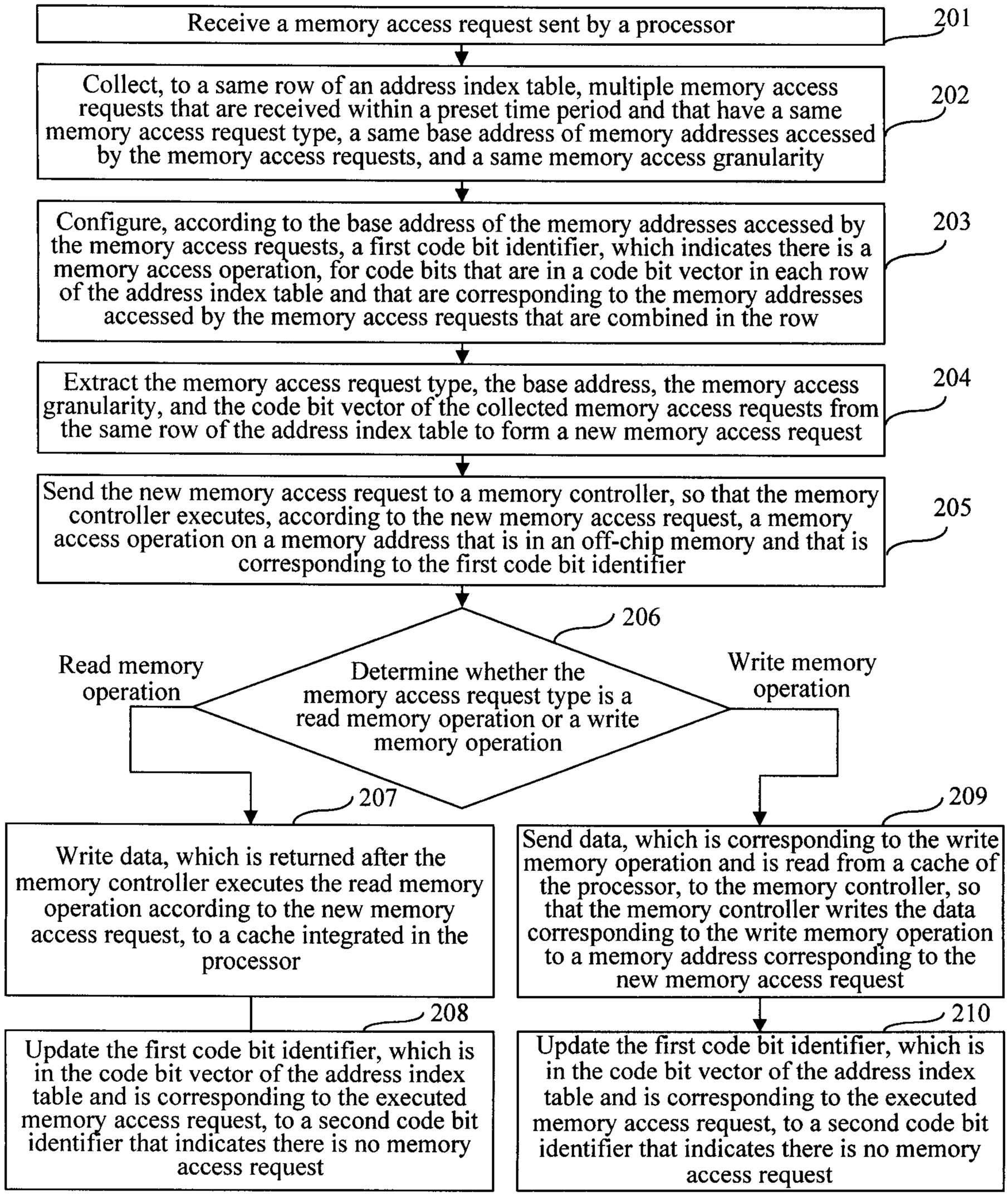
Receive a memory access request sent by a processor
201
Collect, to a same row of an address index table, multiple memory access requests that are received within a preset time period and that have a same memory access request type, a same base address of memory addresses accessed by the memory access requests, and a same memory access granularity
202
Configure, according to the base address of the memory addresses accessed by the memory access requests, a first code bit identifier, which indicates there is a memory access operation, for code bits that are in a code bit vector in each row of the address index table and that are corresponding to the memory addresses accessed by the memory access requests that are combined in the row
203
Extract the memory access request type, the base address, the memory access granularity, and the code bit vector of the collected memory access requests from the same row of the address index table to form a new memory access request
204
Send the new memory access request to a memory controller, so that the memory controller executes, according to the new memory access request, a memory access operation on a memory address that is in an off-chip memory and that is corresponding to the first code bit identifier
205
206
Determine whether the memory access request type is a read memory operation or a write memory operation
Read memory operation
Write memory operation
207
Write data, which is returned after the memory controller executes the read memory operation according to the new memory access request, to a cache integrated in the processor
208
Update the first code bit identifier, which is in the code bit vector of the address index table and is corresponding to the executed memory access request, to a second code bit identifier that indicates there is no memory access request
209
Send data, which is corresponding to the write memory operation and is read from a cache of the processor, to the memory controller, so that the memory controller writes the data corresponding to the write memory operation to a memory address corresponding to the new memory access request
210
Update the first code bit identifier, which is in the code bit vector of the address index table and is corresponding to the executed memory access request, to a second code bit identifier that indicates there is no memory access request

FIG. 2

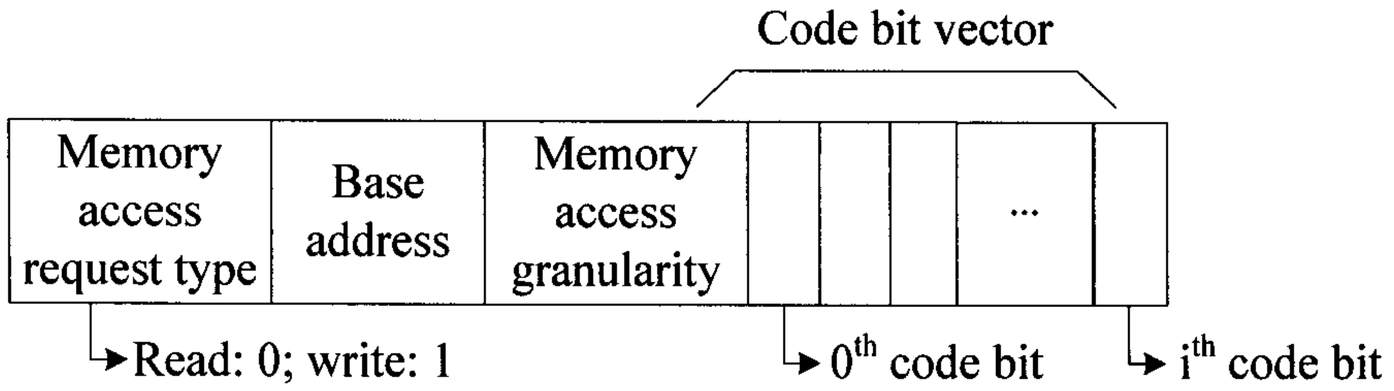

FIG. 3

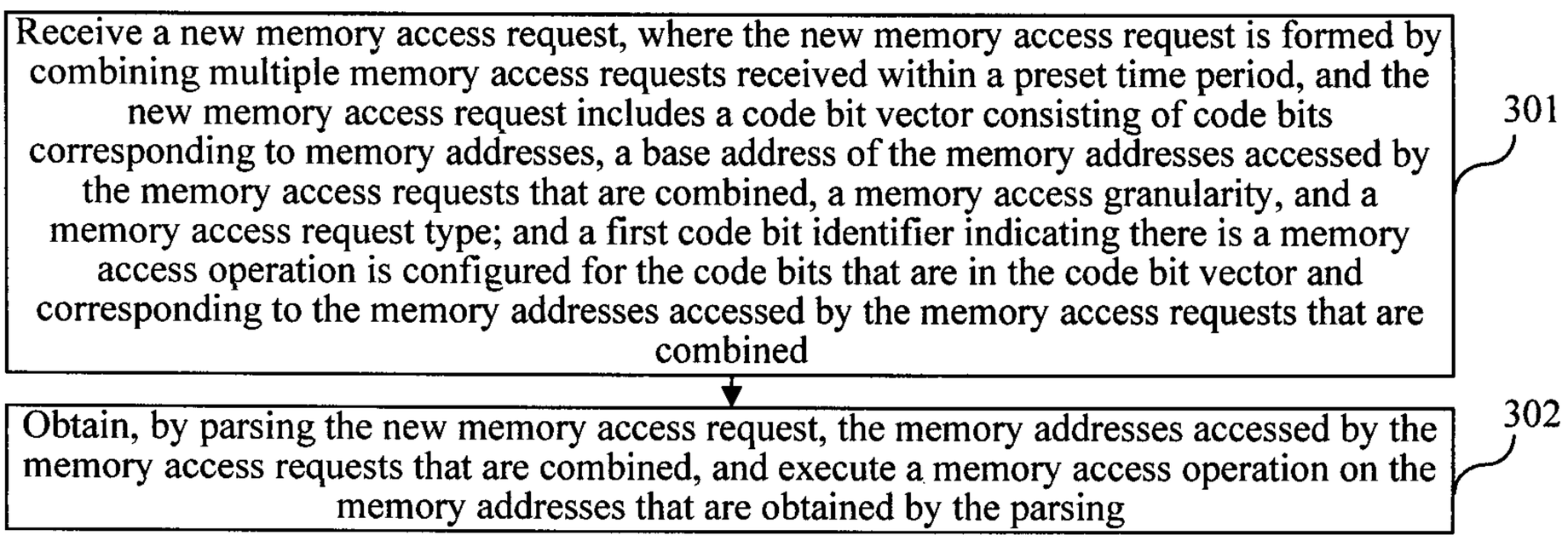

FIG. 4

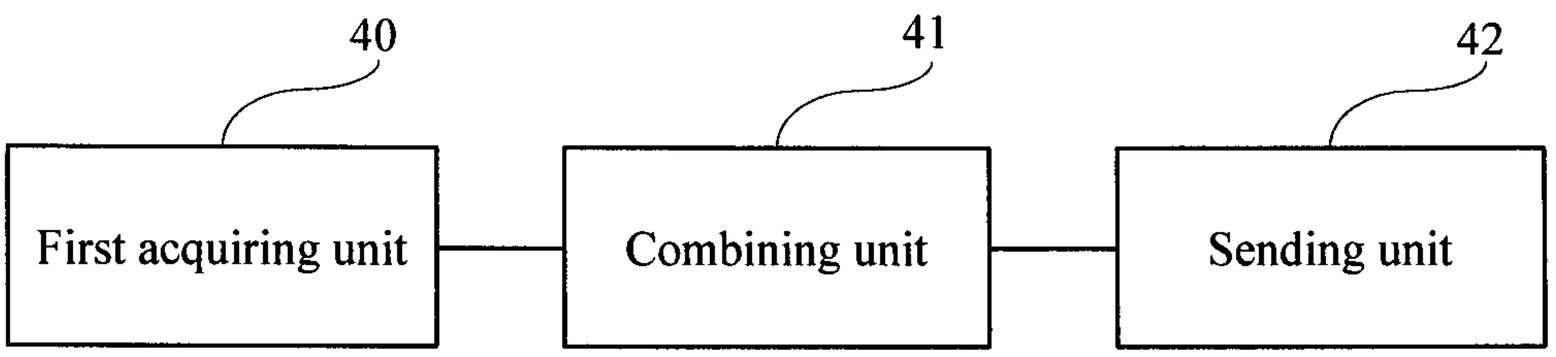

FIG. 5

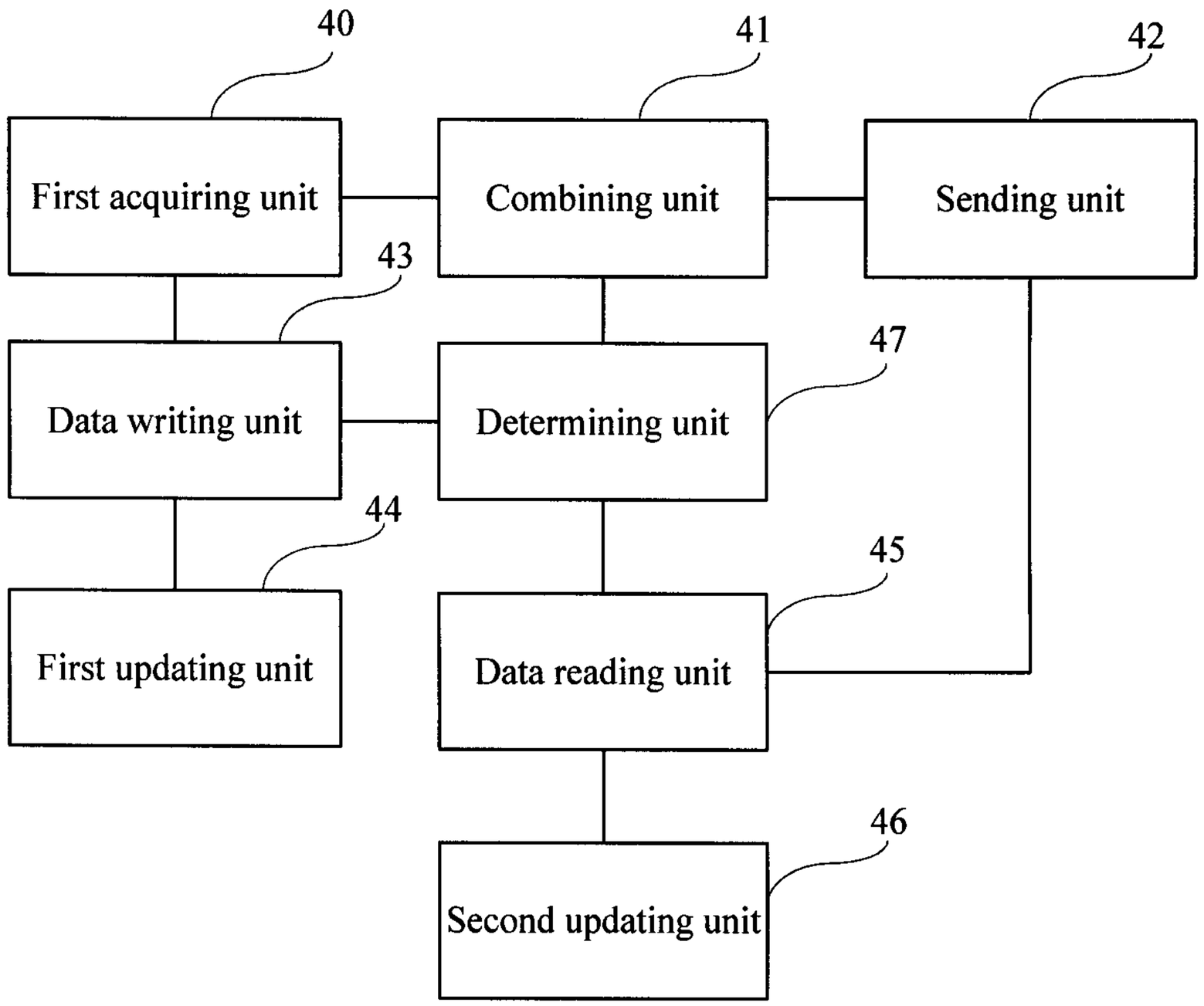
40
41
42
First acquiring unit
Combining unit
Sending unit
43
Data writing unit
47
Determining unit
44
45
First updating unit
Data reading unit
46
Second updating unit

FIG. 6

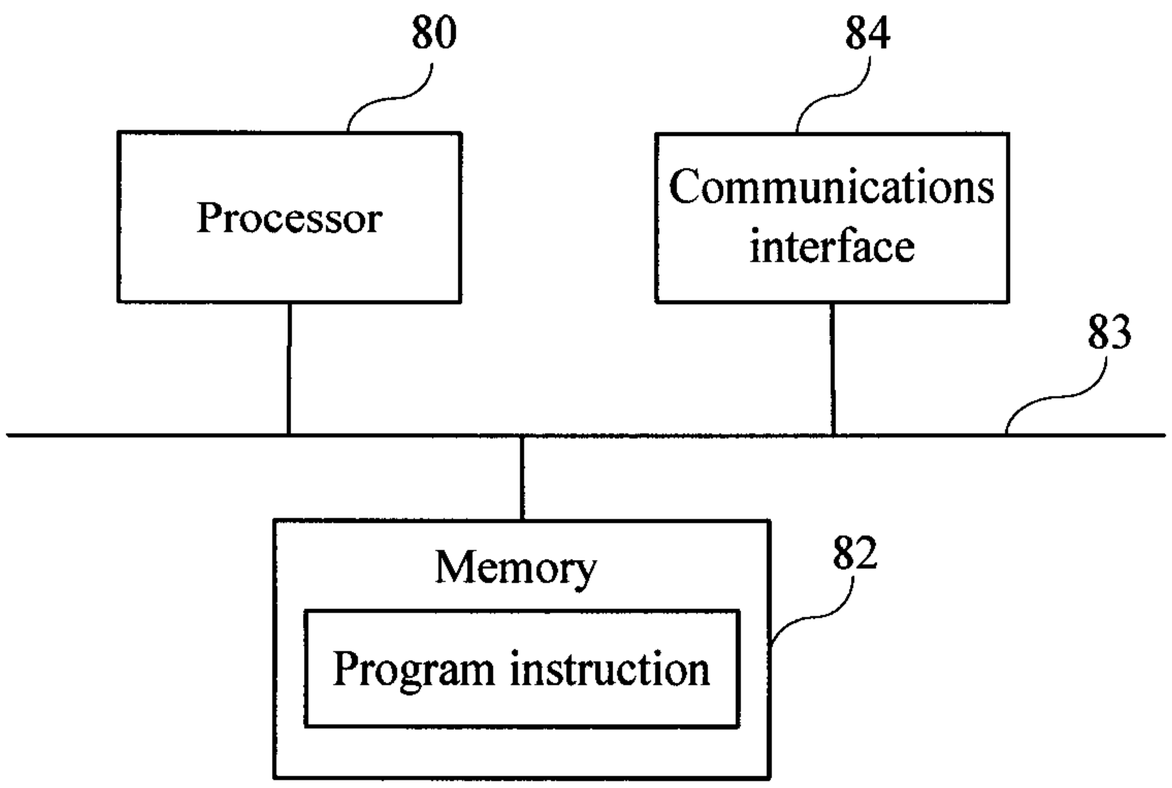
80
84
Processor
Communications interface
83
Memory
82
Program instruction

FIG. 7

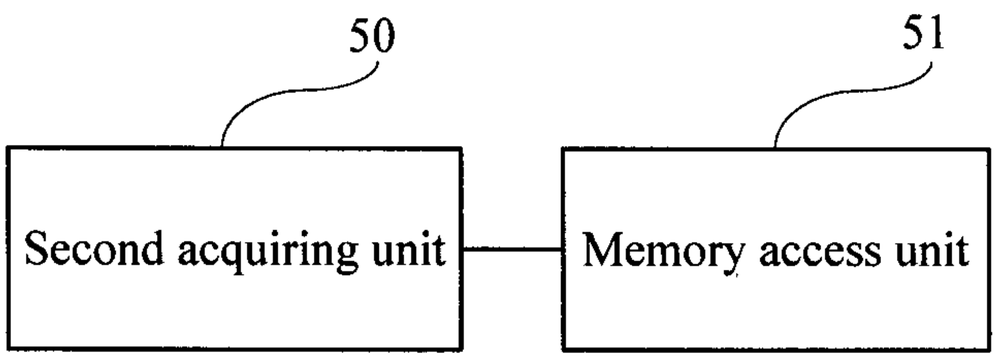
50
51
Second acquiring unit
Memory access unit

FIG. 8

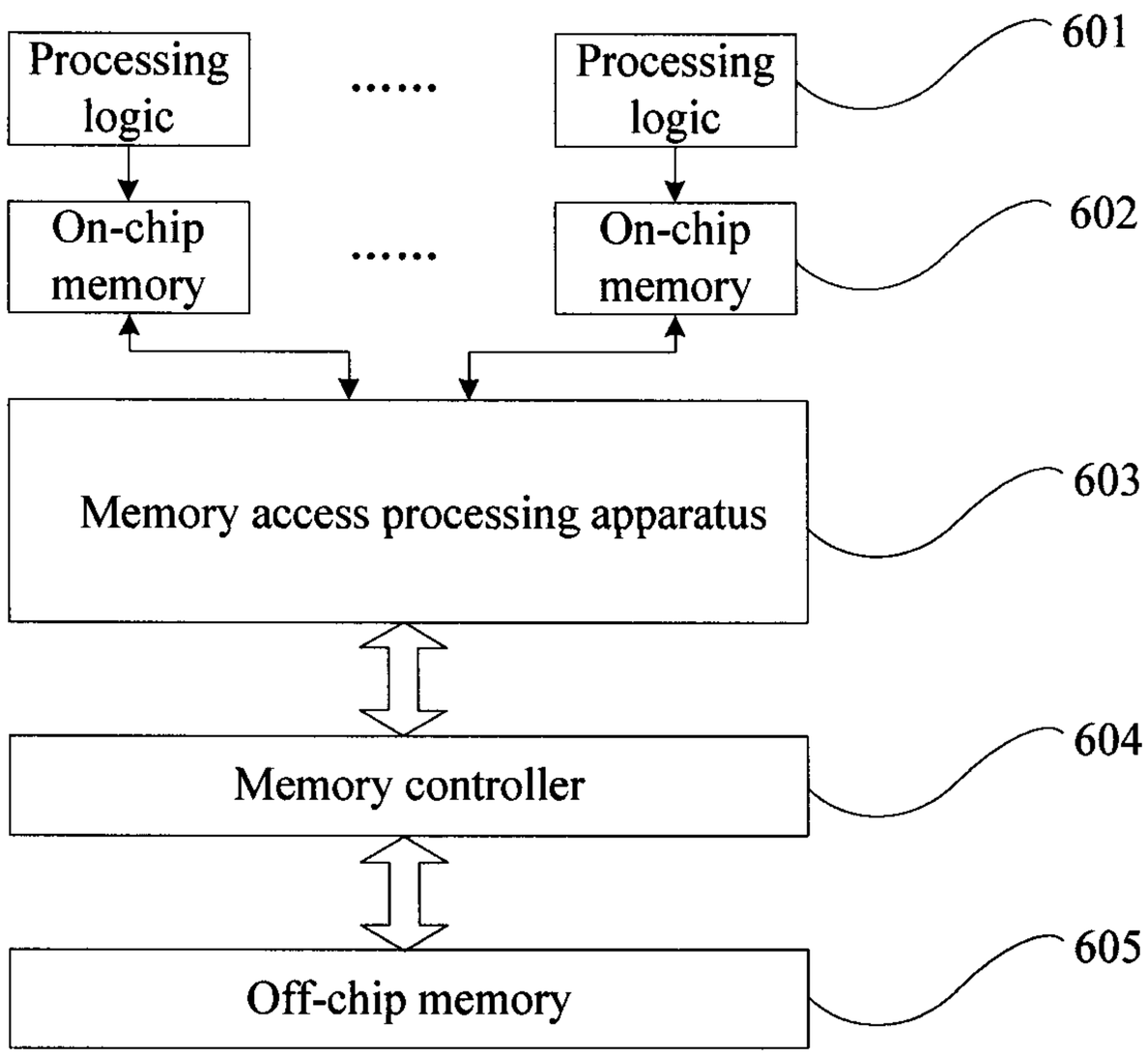
Processing logic
.....
Processing logic
601
On-chip memory
.....
On-chip memory
602
Memory access processing apparatus
603
Memory controller
604
Off-chip memory
605

FIG. 9

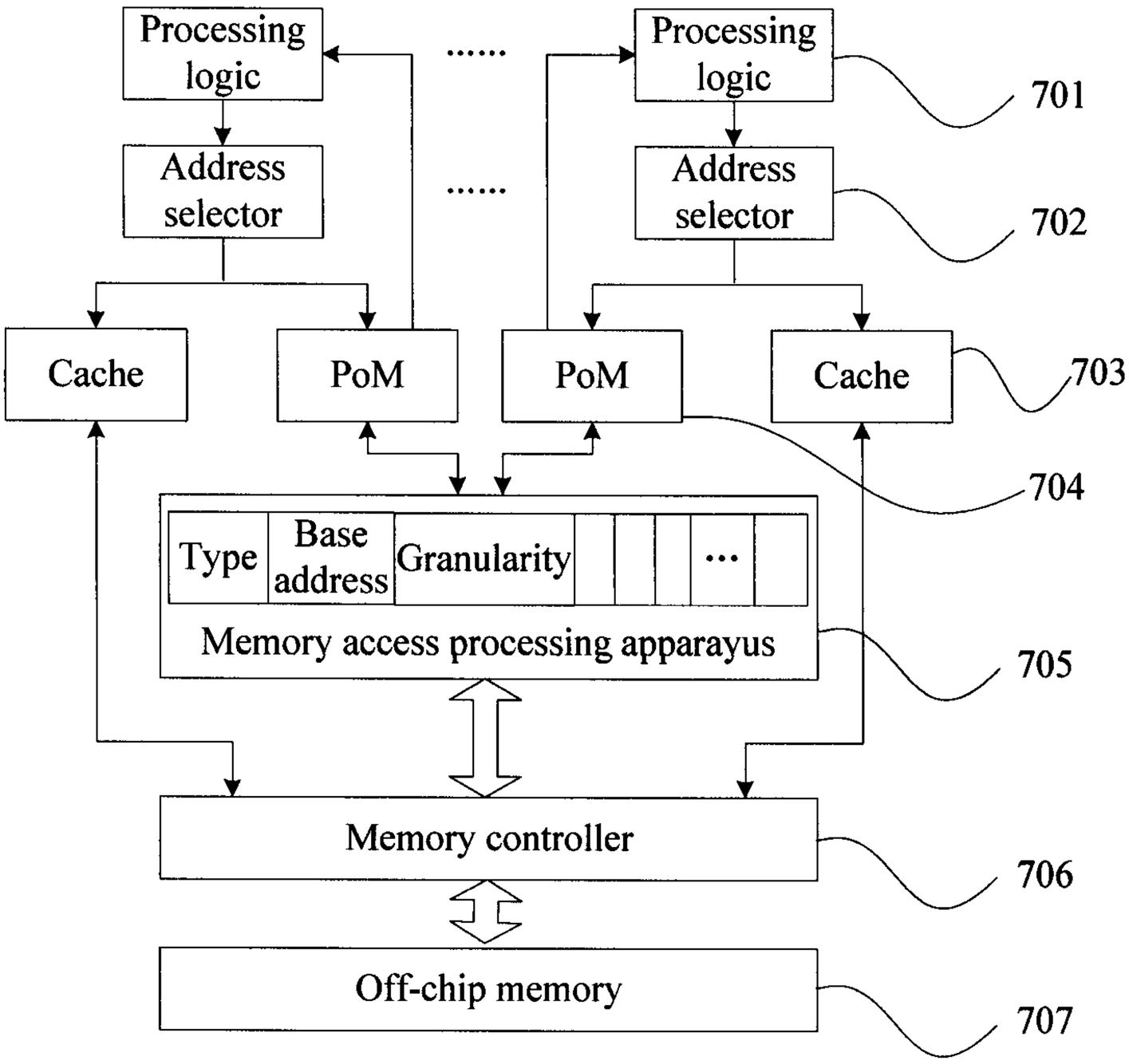
Processing logic
Address selector
Processing logic
Address selector
701
702
Cache
PoM
PoM
Cache
703
704
Type
Base address
Granularity
Memory access processing apparayus
705
Memory controller
706
Off-chip memory
707

FIG. 10

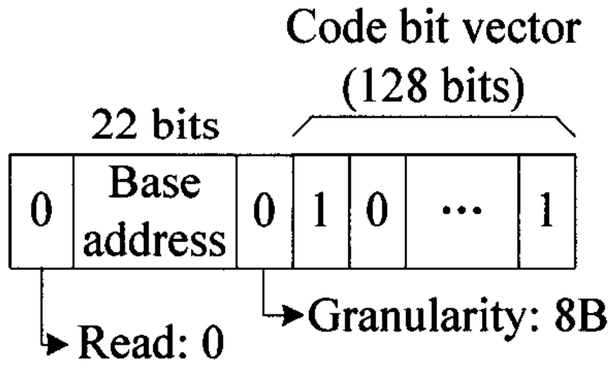
Code bit vector
(128 bits)
22 bits
0
Base address
0
1
0
1
Read: 0
Granularity: 8B

FIG. 11

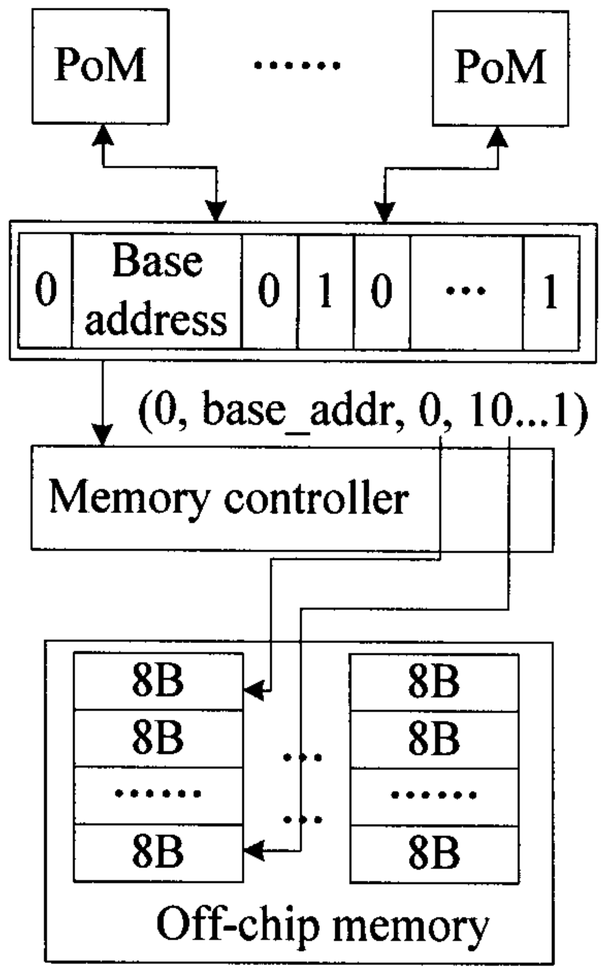

FIG. 12A

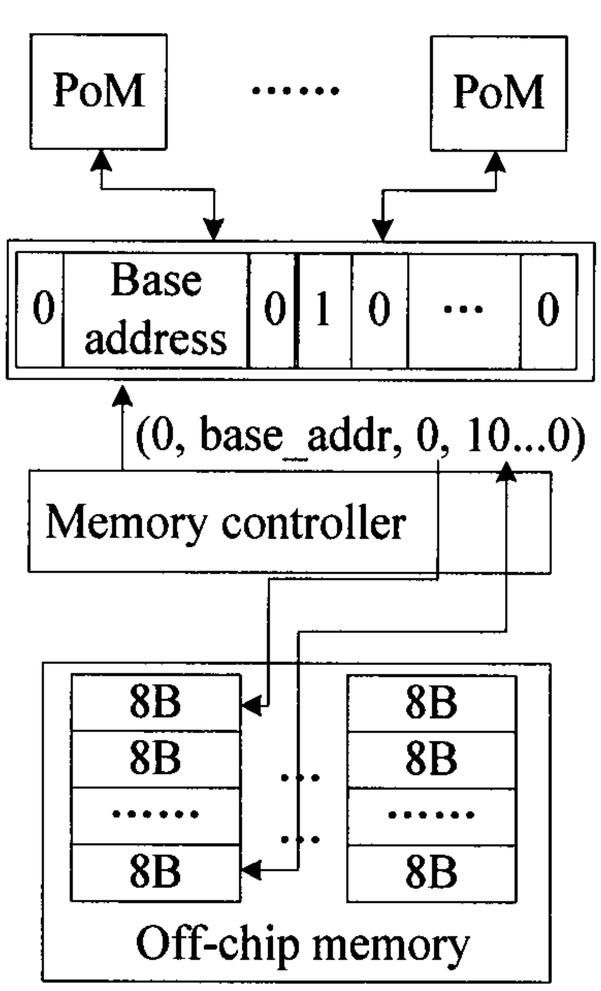

FIG. 12B

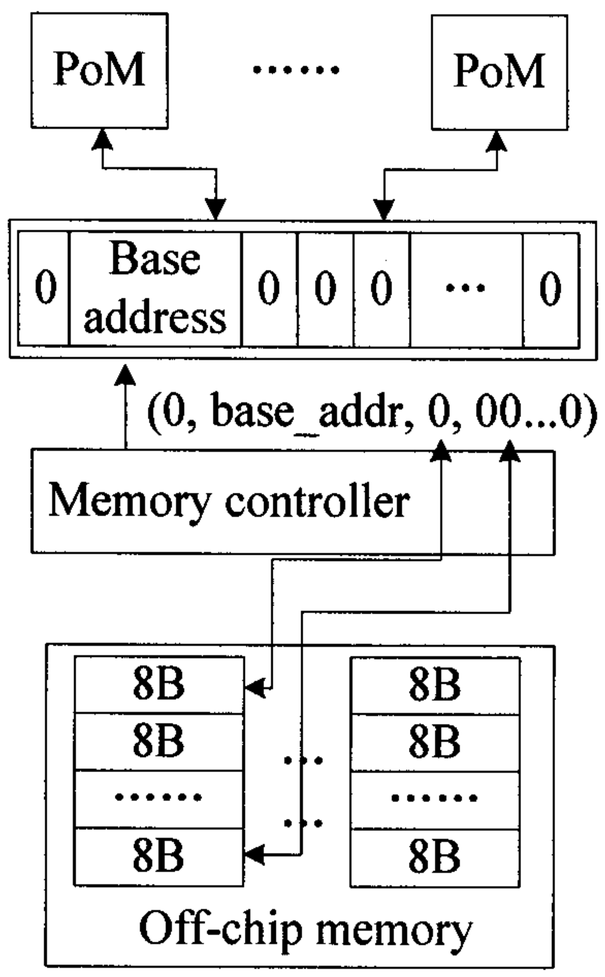

FIG. 12C

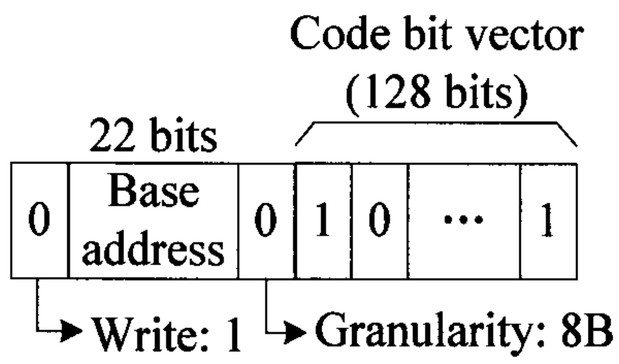

FIG. 13

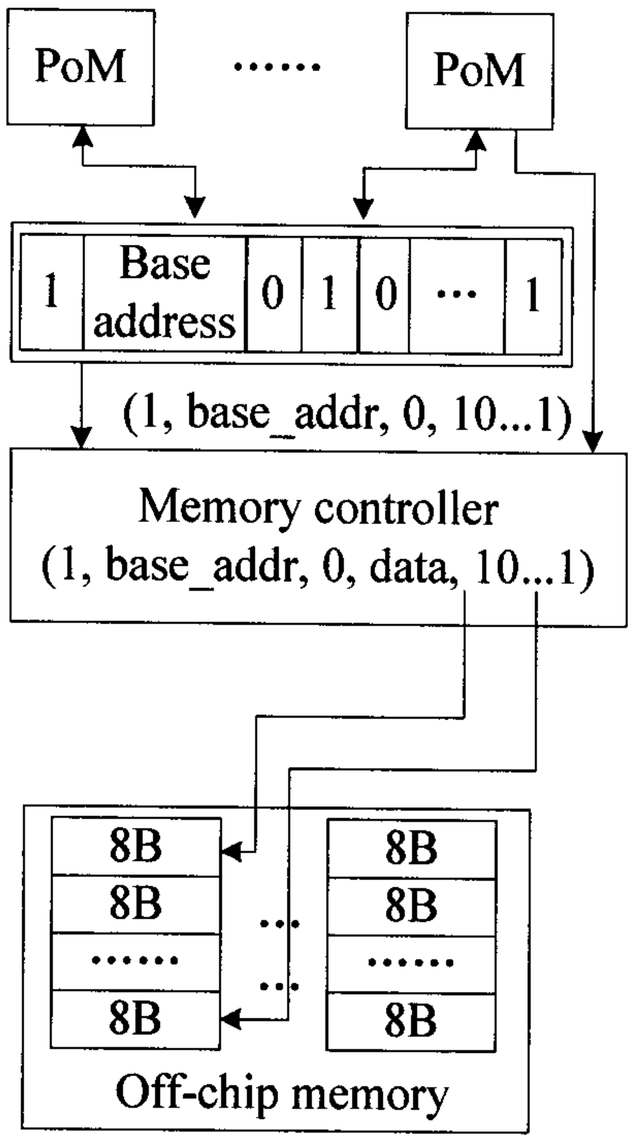
PoM
.....
PoM
1
Base address
0
1
0
...
1
(1, base_addr, 0, 10...1)
Memory controller
(1, base_addr, 0, data, 10...1)
8B
8B
.....
8B
8B
8B
8B
.....
8B
Off-chip memory

FIG. 14

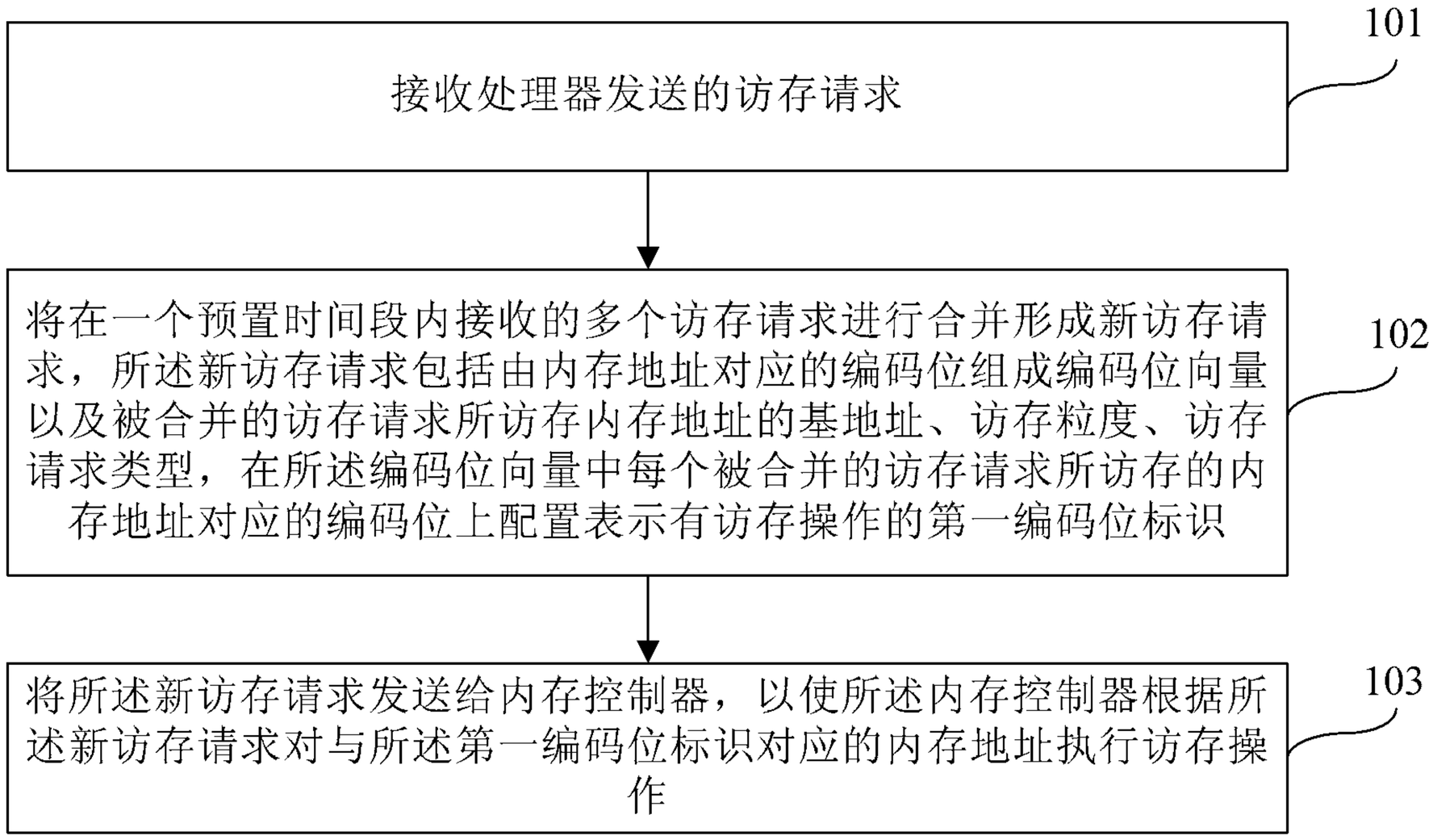

图 1 / FIG. 1

101 RECEIVE A ACCESS REQUEST SENT BY A PROCESSOR

102 COMBINE MULTIPLE ACCESS REQUESTS RECEIVED IN A PRESET TIME PERIOD INTO A NEW ACCESS REQUEST, THE NEW ACCESS REQUEST COMPRISING A CODE BIT VECTOR FORMED BY CODE BITS CORRESPONDING TO MEMORY ADDRESSES, AND BASE ADDRESSES OF THE MEMORY ADDRESSES ACCESSED BY THE COMBINED ACCESS REQUESTS, ACCESS GRANULARITY AND ACCESS REQUEST TYPES, AND CONFIGURE A FIRST CODE BIT IDENTIFICATION INDICATING A ACCESS OPERATION ON A CODE BIT CORRESPONDING TO A MEMORY ADDRESS ACCESSED BY EACH OF THE COMBINED MEMORY REQUESTS IN THE CODE BIT VECTOR

103 SEND THE NEW ACCESS REQUEST TO A MEMORY CONTROLLER, SO THAT THE MEMORY CONTROLLER EXECUTES, ACCORDING TO THE NEW ACCESS REQUEST, A ACCESS OPERATION ON A MEMORY ADDRESS CORRESPONDING TO THE FIRST CODE BIT IDENTIFICATION