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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
27 May 2004 (27.05.2004)

PCT

(10) International Publication Number
WO 2004/044681 A2

(51) International Patent Classification[7]: G06F

(21) International Application Number:
PCT/US2003/035022

(22) International Filing Date:
3 November 2003 (03.11.2003)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
10/288,973 5 November 2002 (05.11.2002) US

(71) Applicant: **SONICS, INC.** [US/US]; 2440 West El Camino Real, Suite #620, Mountain View, CA 94040 (US).

(72) Inventors: **CHOU, Chien-Chun**; 1709 Peony Lane, San Jose, CA 95124 (US). **TOMLINSON, Jay, S.**; 5516 Del Oro Drive, San Jose, CA 95124 (US). **WEBER, Wolf-Dietrich**; 5851 Meander Drive, San Jose, CA 95120 (US). **WINGARD, Drew, E.**; 1059 Hewitt Drive, San Carlos, CA 90470 (US). **KASETTI, Sricharan**; 3351 Alma Street #216, Palo Alto, CA 94031 (US).

(74) Agents: **MALLIE, Michael, J.** et al.; Blakely, Sokoloff, Taylor & Zafman LLP, 12400 Wilshire Boulevard, 7th floor, 7th Floor, Los Angeles, CA 90025 (US).

(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, UZ, VC, VN, YU, ZA, ZM, ZW.

(84) Designated States *(regional)*: ARIPO patent (BW, GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IT, LU, MC, NL, PT, RO, SE, SI, SK, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:
— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

WO 2004/044681 A2

(54) Title: METHOD AND APPARATUS FOR CONFIGURABLE ADDRESS MAPPING AND PROTECTION ARCHITECTURE AND HARDWARE FOR ON-CHIP SYSTEMS

(57) Abstract: A method and apparatus of a configurable address mapping and protection architecture and hardware for on-chip systems have been described.

METHOD AND APPARATUS FOR CONFIGURABLE ADDRESS MAPPING AND PROTECTION ARCHITECTURE AND HARDWARE FOR ON-CHIP SYSTEMS

FIELD OF THE INVENTION

[0001] The present invention pertains to on-chip systems. More particularly, the present invention relates to a method and apparatus for a configurable address mapping and protection architecture and hardware for on-chip systems.

BACKGROUND OF THE INVENTION

[0002] The operational model for most computer and/or on-chip systems involves the sending of requests from one or more processing units to one or more service modules in the system. Upon receiving a request (i.e., an instruction) from a processing unit, a service module completes the task as requested. Then, there may be responses to be returned from the service module back to the processing unit. It is also very common to have a component in the system to act as both a processing unit and a service module.

[0003] Many different ways may be used to deliver requests and responses between processing units and servicing modules. One of the most frequently used methods, for delivering the requests, is by addressing (plus, protection checking). For instance, a request is tagged with a "destination address" and a "source protection identification (ID)". The destination address tells where the service module(s) is (are) located, and/or how to deliver the request to the service module(s). The source protection ID identifies the processing unit and is used to determine whether the service module(s) should execute the request, or whether the request can be delivered to the service module(s), thus providing access to the service module selectively depending on source identity. Usually, the number of transistors (often referred to as "gates") and the resulting gate size (and thus area) of the hardware module (on for example, an integrated circuit) devoted to address decoding and protection ID checking are comparatively large. Additional circuitry, which consumes more power, may also be needed in order to make this decoding and checking hardware dynamic (i.e., configurable) during operation. For a wireless device, especially, where the demand for a smaller chip die size and a lower power consumption is high, a large and power-consuming address decoding and protection-checking module is unacceptable. This presents problems.

BRIEF DESCRIPTION OF THE DRAWINGS

[0004] The invention is illustrated by way of example and not limitation in the figures

of the accompanying drawings, in which like references indicate similar elements and in which:

[0005] Figure 1 illustrates a network environment in which the method and apparatus of the present invention may be implemented;

[0006] Figure 2 is a block diagram of a computer system;

[0007] Figure 3 illustrates one embodiment of the invention showing in a block diagram form an on-chip system with N processing units, M service modules, an address mapping and protection module, and a request-delivery module;

[0008] Figure 4 illustrates one embodiment of the invention showing in a flowchart form the process in generating optimized address mapping and protection hardware;

[0009] Figure 5 illustrates one embodiment of the invention showing in block diagram form a hardware implementation for the configurable address mapping and protection architecture;

[0010] Figure 6 illustrates one embodiment of the invention showing in table form, a definition of some possible read-only, read-write, or not-accessible configuration parameters; and

[0011] Figure 7 illustrates one embodiment of the invention showing in table form, a definition of some possible not-accessible configuration parameters.

DETAILED DESCRIPTION

[0012] A method and apparatus for a configurable address mapping and protection architecture and hardware for on-chip systems are described.

[0013] Figure 1 illustrates a network environment 100 in which the techniques described may be applied. The network environment 100 has a network 102 that connects S servers 104-1 through 104-S, and C clients 108-1 through 108-C. More details are described below.

[0014] Figure 2 illustrates a computer system 200 in block diagram form, which may be representative of any of the clients and/or servers shown in Figure 1. More details are described below.

[0015] The term IP as used in this document denotes Intellectual Property. The term IP, may be used by itself, or may be used with other terms such as core, to denote a design having a functionality. For example, an IP core or IP for short, may consist of circuitry, buses, communication links, a microprocessor, etc. Additionally, IP may be implemented in a variety of ways, and may be fabricated on an integrated circuit, etc. The term flooding is used to denote a communication in which an incoming packet is duplicated and sent out on every outgoing pathway throughout most of a chip, system,

etc.

[0016] In this disclosure, a method and apparatus for a configurable address mapping and protection architecture and hardware for on-chip systems are described. In one embodiment of the invention, circuitry for providing the necessary address mapping and protection functionality is provided in hardware. In another embodiment the invention allows a product designer to configure the address mapping and protection module at design time, such that, only the minimum specified mapping and protection is implemented into the hardware. Thus, the final gate size and power consumption of the address mapping and protection hardware module is determined by the specification of the product. Moreover, the address width and data word width for each of the service modules may also be considered and used to minimize the number of signal wires to/from the service modules. This may result in hardware that is not over designed and may more easily meet the gate count and power consumption requirements of a product.

[0017] The disclosed invention "configurable address mapping and protection architecture and hardware for on-chip systems" may provide:

1. A centralized, configurable address mapping and protection architecture for an on-chip system.
2. A set of configuration parameters that may lead to overall gate size reduction, power consumption reduction, and/or the elimination of unnecessary signal wires for a final address mapping and protection hardware module.
3. The ability for a designer to configure the address mapping and protection hardware module at design time using a specification language and achieve the goal of producing a minimized hardware implementation.

[0018] Figure 4 shows the processing flow that can be used in generating optimized address mapping and protection hardware. First, a product designer must understand the configurable architecture (403). The architecture (423) includes the definition of the address mapping scheme, the definition of the protection scheme, the definition of the set of configuration parameters, and the definition of the specification language. Then, the designer can design the address mapping and protection module with minimum hardware based upon the product requirements (405). Based on the product requirements and user models, the designer should be able to decide, for instance, the number of service modules in the system, the number of address regions for a service module, and how the information of an address region is going to be used (443).

[0019] Next the designer needs to specify the address mapping and protection

hardware module using the provided specification language (407 and 425). At the end (409), a post-processing tool, which takes the design specification as input, is used and generates an optimized hardware gate-level netlist (427) for the address mapping and protection hardware.

[0020] Figure 3 shows a block diagram of one embodiment of the invention having an on-chip system with N processing units at the top (302-0, 302-1, through 302-N), and M service modules at the bottom (318-0, 318-1, through 318-M). In the middle of the system is an address mapping and protection module 306, showing the address mapping 308 and protection key map 310 followed by a request-delivery module 314. Note that Figure 3 only shows the request delivery side of the system (i.e., the request side and does not show the return side). In one embodiment of the invention, the use model of the system is the following:

- A request is sent from a processing unit (such as 320-N) to the address mapping and protection module (306).
- In the address mapping and protection module 306, the "destination address" and "source protection ID" are extracted out of the request (such as that illustrated at 304). The address is decoded and compared against the address map (308) to find out where the service module is and how to deliver the request to the module. The given protection ID is checked against the protection key map (310) to determine whether the request should be delivered to the service module because the protection allows it. Note that other possible fields inside a request, not shown in Figure 3, are "request type field", "data field", "data type field" (e.g., to indicate that it is a burst data stream), and "user provided request information" (e.g., user can use this field to provide a proprietary, sub request type).
- Next, the request (such as that illustrated at 312 and 316) is sent to the service module (such as 318-2) by the request-delivery module (314).

[0021] One embodiment of the invention having a configurable address mapping and protection architecture using segmentation and address regions for the on-chip system mentioned above is described here. Conceptually, the address space for the entire on-chip system may be divided into R address regions. Each service module in the system may have multiple address regions. In order to reduce the complexity of matching to 1 of the R address regions, multiple address regions may be grouped into one address segment. There may be up to S segments in the system. To send a request to a specific service module, a processing unit needs to tag the request with an address that is within

an address region of the module. The information about an address region is kept in a region register stored in the address map hardware. Information about an address segment is kept in a segment register, which is also stored in the address map hardware.

[0022] There may be up to K different protection keys in the system; each protection key is kept in one protection key register, which resides in the protection key map hardware. Each address region is associated with at least one protection key, and each processing unit is assigned with one or more protection ID. When a request is sent from a processing unit, the request is tagged with a protection ID (such as 515 shown in Figure 5) of the processing unit. After the request's destination address is decoded and a service module's address region is identified, the protection ID is checked against those protection keys associated with that address region to see whether it is safe to forward the request to the service module.

[0023] Figure 5 shows one embodiment of the invention having a hardware implementation of the address mapping and protection architecture mentioned above. In Figure 5, a request is shown with its destination address tag and protection ID tag (such as illustrated in Figure 3 at 304). The destination address contains three parts, a segment (base) address part 501a, a region (base) address part 501b, and the offset within the region 501c. The address mapping hardware contains the S segment registers (503) and the R region registers (505a, b, and c, for example; and there can be more region pages as shown in Figure 3). Each address segment may contain multiple address regions – this is illustrated by having one page of region registers associated with each segment register (pages 505a, 505b, and 505c are associated with, for example, segment register 0, segment register 1, and segment register 2, respectively. Figure 5 also shows that the address segment 0 has five region registers (505a).

[0024] Moreover, the segment address of a request is used to match one of the segment registers (509), and the region address is used to match one or more of the region registers kept in the address map. Combining the two matching signals, in a normal case, one single match happens (511). Note that a duplicate match and a failed match may be detected, if desirable, as errors.

[0025] If a single match occurs, the protection key register number associated with the region is returned (519); and the routing information for the targeting service module is also returned (513). The protection key register number (519) is used to filter out unrelated matches coming out of the protection key map module (517). A positive

security okay signal (521) indicates that the request can be delivered to the servicing module.

[0026] Table 1 shows a summary of the combinations and the results of a normal matching case (i.e., a single address match is identified and security check is also okay), and error cases. When a single match is identified, Information (saved in the matched region register) about the destination service module is forwarded to the downstream modules. For instance,

- The protection key ID is forwarded to the "Security Check Okay" circuit in order to complete the security check.
- The destination service module's data word width and physical target ID (the physical target ID contains the physical location information of the service module) are forwarded to the request-delivery module.

Table 1

Matching and checking on fields	Normal Case	Error Case 1	Error Case 2	Error Case 3
Request's segment (base) address	Single match	Single match	Single match	Double match
Request's region (base) address	Single match on the region register page of the matched segment	Single match on the region register page of the matched segment	Double match on the region register page of the matched segment	Don't care
Request's protection ID	Okay	Not okay	Don't care	Don't care
Results	Information about the destination service module is generated and forwarded to the request-delivery module	Protection violation	Error found in the region address map	Error found in the segment address map

[0027] A set of configuration parameters and registers (plus register fields in each type of register) are also identified for the architecture mentioned above such that a designer can adjust them in order to build the address mapping and protection module with minimum hardware. The final goal is to reduce the hardware module's gate size, power consumption, and signal wires. The following lists the configuration parameters,

their definition, and how they can affect hardware:

- DataWidth: This parameter represents the data word size of a request. Different data word sizes can be allowed for requests coming from different processing units. However, only a single data width parameter is used here; thus, requests coming from the processing units have the same data word size. Setting this parameter to only the needed data word size can save gates and wires in the hardware module.
- AddrWidth: This parameter represents the address tag width for the on-chip system; i.e, the dimension of the address mapping and protection module. Setting this parameter to only the needed address width can save a great number of gates and wires in the hardware module.
- NumSegments: This parameter indicates how many segment registers can exist in the system and is used to removed un-needed segment registers.
- For each segment register:

  SegmentSize register field: This field tells the size of a segment; it can be used to reduce the number of bits for a segment register.

  SegmentBase register field: This field indicates the segment base address of an address segment.
- NumRegions: This parameter indicates how many region registers can exist in the system and is used to remove un-needed region registers.
- For each region register:

  RegionSize register field: This field tells the size of an address region; it can be used to reduce the number of bits for a region register.

  RegionBase register field: This field indicates the region base address of an address region.

  RegionProtectionKeyRegisterNum (RPKRN) register field: This field tells which protection key register is to be used by the security checking logic when a single match occurs on this address region. Multiple register fields of this type can exist, however, only one is used here.

  RegionDataWidth register field: This field tells the data word width of the service module that links to an address region. It can be used to trim data bus wires, if possible, connecting to the service module. It can also be used to indicate whether data packing or unpacking is needed; packing or unpacking may be needed when the data word size of a request's source

processing unit is different from the data word size of the request's destination service module.

RegionPhysicalTargetID (RPTID) register field: This field describes the physical linkage between an address region and a service module. This physical linkage can be, for example: (1) hardware routing information to be passed on to the request-delivery module in order to deliver a request to the service module; or (2) a hardware signal bit position such that, when the request-delivery module asserts the signal, a request is sent to the service module.

RegionAddressSpace (RAS) register field: This field allows an address region of a service module to be further partitioned.

RegionEnable register field: This field indicates whether or not this region register is used for the current design or to indicate whether the region is currently available.

- NumProtectionKeys: This parameter indicates how many protection key registers can exist in the system and is used to remove un-needed protection key registers. In addition, it can also save bits in each of the region registers, where a protection key number is stored.
- For each protection key register:

  ProtectionKeyBitVector register field: This bit vector tells which protection IDs are allowed to access the service modules that are linked by region registers pointing to this protection key register. A bit 1 in position N indicates that a request tagged with protection ID of N is okay to access the request's destination service module.
- NumProtectionIDs: This parameter indicates how many different protection IDs can exist in the system and is used to removed un-needed protection key bits in the ProtectionKeyBitVector register field.
- Endianess: This parameter tells whether the big endianess or little endianess is applied in the architecture; it determines the address byte location and the data byte sequence coming out of data packing/unpacking.

[0028] Moreover, the register fields of each of the registers can also be specified to be one of the following three usage types so that a minimum logic design can be applied to construct the hardware to save area and power:

- Non-Accessible (NA) Register Field: A register field is hardwired to a power-on value

and cannot be read, nor written.

- Read-Only (RO) Register Field: A register field is hardwired to a power-on value and needs to be software visible (read-only) during operation. In this case, extra gates are needed in order to allow the software read access of the register field.
- Read-Write (RW) Register Field: A register field can be read and written by software dynamically. For this type of register field, extra circuitry (for example, in the form of flip-flops and gates) is needed in order to allow software changes.

[0029] Additionally, each register field can also be specified as an "exporting constant" (EC) register field such that the netlist portion of the register field is exported to the top-level of the final netlist. It makes the power-on value of a register field more easily to be manually modified, as needed by a product, late during the full chip generation process. For instance, the ProtectionKeyBitVector register field of each of the protection key registers can be declared as "exporting constant" field; therefore, it allows a final protection key map to be put into the chip late in the product generation process.

[0030] Figure 6 and Figure 7 show two almost identical address mapping and protection configurations. The register fields in Figure 6 are of RO, RW, or NA type; however, for Figure 7, all register fields are of NA type, plus, the un-used region register 1 is removed. From a first-order estimation, the number of flip-flops (also referred to as flops) saved in Figure 7 is 82; that is no flops are used in the address map (308) and the protection key map (310) as shown in the center of Figure 3. Also note that, if (1) the address width of the hardware module is reduced from 20 to 17, and (2) the number of protection IDs is scaled down to 8, additional buffer register bits and signal wires can be saved versus the Figure 6 case.

[0031] As mentioned previously, a specification language needs to be provided so that a designer may easily specify a minimum design for a product. The following lists, in one embodiment of the invention, an example specification as shown in Figure 6:

```
Address Mapping and Protection Module {
    DataWidth: 16
    AddrWidth: 20
    Endianess: little
    NumSegments: 2
    NumRegions: 5
    NumProtectionKeys: 2
    NumProtectionKeyIDs: 16
```

SEGMENT(0) {
SegmentSize: 64KB { access RO }
SegmentBase: 0x00000 { access RW }
}
SEGMENT(1) {
SegmentSize: 64KB { access RO }
SegmentBase: 0x10000 { access RW }
}
REGION(0) {
Inside SEGMENT(0)
RegionSize: 256B { access RO }
RegionBase: 0x00100 { access RW }
RegionProtectionKeyRegisterNum: 0 { access RW }
RegionDataWidth: 2B { access NA }
RegionPhysicalTargetID: link to ServiceModule 1 using "targetselect pin 1" { access NA }
RegionAddressSpace: 0 { access RO }
RegionEnable: Yes { access RW }
}
REGION(1) {
Inside SEGMENT(0)
RegionSize: 4KB { access RO }
RegionBase: 0x01000 { access RW }
RegionProtectionKeyRegisterNum: 1 { access RW }
RegionDataWidth: 2B { access NA }
RegionPhysicalTargetID: link to ServiceModule 1 using "targetselect pin 1" { access NA }
RegionAddressSpace: 1 { access RO }
RegionEnable: No { access RW }
}
REGION(2) {
Inside SEGMENT(0)
RegionSize: 16B { access RO }
RegionBase: 0x00000 { access RW }
RegionProtectionKeyRegisterNum: 0 { access RW }

RegionDataWidth: 4B { access NA }

RegionPhysicalTargetID: link to ServiceModule 0 using "targetselect pin 0" { access NA }

RegionAddressSpace: 0 { access RO }

RegionEnable: Yes { access RW }

}

REGION(3) {

Inside SEGMENT(1)

RegionSize: 4KB { access RO }

RegionBase: 0x10000 { access RW }

RegionProtectionKeyRegisterNum: 1 { access RW }

RegionDataWidth: 1B { access NA }

RegionPhysicalTargetID: link to ServiceModule 2 using "targetselect pin 2" { access NA }

RegionAddressSpace: 0 { access RO }

RegionEnable: Yes { access RW }

}

REGION(4) {

Inside SEGMENT(1)

RegionSize: 4KB { access RO }

RegionBase: 0x11000 { access RW }

RegionProtectionKeyRegisterNum: 0 { access RW }

RegionDataWidth: 4B { access NA }

RegionPhysicalTargetID: link to ServiceModule 3 using "targetselect pin 3" { access NA }

RegionAddressSpace: 0 { access RO }

RegionEnable: Yes { access RW }

}

PROTECTIONKEY(0) {

ProtectionKeyBitVector: 0x007B { access RW and EC }

}

PROTECTIONKEY(1) {

ProtectionKeyBitVector: 0x0085 { access RW and EC }

}

[0032] For the above example (also shown in Figure 6), there are 2 segments and 5 address regions; the address region 1 is disabled at the initialization time (i.e., the region register's RegionEnable field is set to "No"), but can be re-configured at run-time because the field is read/writable. There are two protection key registers and each has a 16-bit bit vector. The example also specifies the following at power-on:

- The request address width is 20 bits and data word size is 16 bits.
- There are four service modules: ServiceModule 0, 1, 2, and 3.
- Address region 0, 1, and 2 exist in the address segment 0 and are based at address 0x00100, 0x01000, and 0x0000, and of size 256 bytes, 4K bytes, and 16 bytes, respectively. The region register 1 is not enabled at the current time, but, can be used as a future addition.
- Address region 3 and 4 exist in the address segment 1 and are based at address 0x10000 and 0x11000, respectively; both are 4K-byte in size.
- Requests coming from processing units using Protection ID 0, 1, 3, 4, 5, and 6 (the ProtectionKeyBitVector of 0x007B) can go to ServiceModule 0, 1, and 3, depending on the request address. Requests coming from processing units using Protection ID 0, 2, and 7 (the ProtectionKeyBitVector of 0x0085) can go to ServiceModule 2, if the request address falls into the address region 3.

[0033] For the design shown in Figure 7, its specification looks like the following:

```
Address Mapping and Protection Module {
    DataWidth: 16
    AddrWidth: 20
    Endiness: little
    NumSegments: 2
    NumRegions: 4
    NumProtectionKeys: 2
    NumProtectionKeyIDs: 16
    SEGMENT(0) {
      SegmentSize: 64KB { access NA }
      SegmentBase: 0x00000 { access NA }
    }
    SEGMENT(1) {
      SegmentSize: 64KB { access NA }
      SegmentBase: 0x10000 { access NA }
    }
```

REGION(0) {

Inside SEGMENT(0)

RegionSize: 256B { access NA }

RegionBase: 0x00100 { access NA }

RegionProtectionKeyRegisterNum: 0 { access NA }

RegionDataWidth: 2B { access NA }

RegionPhysicalTargetID: link to ServiceModule 1 using "targetselect pin 1" { access NA }

RegionAddressSpace: 0 { access NA }

RegionEnable: Yes { access NA }

}

REGION(2) {

Inside SEGMENT(0)

RegionSize: 16B { access NA }

RegionBase: 0x00000 { access NA }

RegionProtectionKeyRegisterNum: 0 { access NA }

RegionDataWidth: 4B { access NA }

RegionPhysicalTargetID: link to ServiceModule 0 using "targetselect pin 0" { access NA }

RegionAddressSpace: 0 { access NA }

RegionEnable: Yes { access NA }

}

REGION(3) {

Inside SEGMENT(1)

RegionSize: 4KB { access NA }

RegionBase: 0x10000 { access NA }

RegionProtectionKeyRegisterNum: 1 { access NA }

RegionDataWidth: 1B { access NA }

RegionPhysicalTargetID: link to ServiceModule 2 using "targetselect pin 2" { access NA }

RegionAddressSpace: 0 { access NA }

RegionEnable: Yes { access NA }

}

REGION(4) {

Inside SEGMENT(1)

```
    RegionSize: 4KB { access NA }
    RegionBase: 0x11000 { access NA }
    RegionProtectionKeyRegisterNum: 0 { access NA }
    RegionDataWidth: 4B { access NA }
    RegionPhysicalTargetID: link to ServiceModule 3 using "targetselect pin 3" { access
NA }
    RegionAddressSpace: 0 { access NA }
    RegionEnable: Yes { access NA }
  }
  PROTECTIONKEY(0) {
    ProtectionKeyBitVector: 0x007B { access NA and EC }
  }
  PROTECTIONKEY(1) {
    ProtectionKeyBitVector: 0x0085 { access NA and EC }
  }
```

[0034] As mentioned in the description, and as shown in Figure 4, at one of the final stages, a post-processing tool, which takes the specified design (such as the specification text shown above) as input, is used and generates an optimized hardware netlist for the address mapping and protection hardware.

[0035] Thus, what has been disclosed is a method and apparatus of a configurable address mapping and protection hardware for on-chip systems.

[0036] Referring back to Figure 1, Figure 1 illustrates a network environment 100 in which the techniques described may be applied. The network environment 100 has a network 102 that connects S servers 104-1 through 104-S, and C clients 108-1 through 108-C. As shown, several systems in the form of S servers 104-1 through 104-S and C clients 108-1 through 108-C are connected to each other via a network 102, which may be, for example, an on-chip communication network. Note that alternatively the network 102 might be or include one or more of: inter-chip communications, an optical network, the Internet, a Local Area Network (LAN), Wide Area Network (WAN), satellite link, fiber network, cable network, or a combination of these and/or others. The servers may represent, for example: a master device on a chip; a memory; an intellectual property core, such as a microprocessor, communications interface, etc.; a disk storage system; and/or computing resources. Likewise, the clients may have computing, storage, and viewing capabilities. The method and apparatus described herein may be applied to essentially any type of communicating means or device whether local or remote, such

as a LAN, a WAN, a system bus, on-chip bus, etc. It is to be further appreciated that the use of the term client and server is for clarity in specifying who initiates a communication (the client) and who responds (the server). No hierarchy is implied unless explicitly stated. Both functions may be in a single communicating device, in which case the client-server and server-client relationship may be viewed as peer-to-peer. Thus, if two devices such as 108-1 and 104-S can both initiate and respond to communications, their communication may be viewed as peer-to-peer. Likewise, communications between 104-1 and 104-S, and 108-1 and 108-C may be viewed as peer to peer if each such communicating device is capable of initiation and response to communication.

[0037] Referring back to Figure 2, Figure 2 illustrates a system 200 in block diagram form, which may be representative of any of the clients and/or servers shown in Figure 1. The block diagram is a high level conceptual representation and may be implemented in a variety of ways and by various architectures. Bus system 202 interconnects a Central Processing Unit (CPU) 204, Read Only Memory (ROM) 206, Random Access Memory (RAM) 208, storage 210, display 220, audio, 222, keyboard 224, pointer 226, miscellaneous input/output (I/O) devices 228, and communications 230. The bus system 202 may be for example, one or more of such buses as an on-chip bus, a system bus, Peripheral Component Interconnect (PCI), Advanced Graphics Port (AGP), Small Computer System Interface (SCSI), Institute of Electrical and Electronics Engineers (IEEE) standard number 1394 (FireWire), Universal Serial Bus (USB), etc. The CPU 204 may be a single, multiple, or even a distributed computing resource. Storage 210, may be Compact Disc (CD), Digital Versatile Disk (DVD), hard disks (HD), optical disks, tape, flash, memory sticks, video recorders, etc. Display 220 might be, for example, a Cathode Ray Tube (CRT), Liquid Crystal Display (LCD), a projection system, Television (TV), etc. Note that depending upon the actual implementation of the system, the system may include some, all, more, or a rearrangement of components in the block diagram. For example, an on-chip communications system on an integrated circuit may lack a display 220, keyboard 224, and a pointer 226. Another example may be a thin client might consist of a wireless hand held device that lacks, for example, a traditional keyboard. Thus, many variations on the system of Figure 2 are possible.

[0038] For purposes of discussing and understanding the invention, it is to be understood that various terms are used by those knowledgeable in the art to describe techniques and approaches. Furthermore, in the description, for purposes of

explanation, numerous specific details are set forth in order to provide a thorough understanding of the present invention. It will be evident, however, to one of ordinary skill in the art that the present invention may be practiced without these specific details. In some instances, well-known structures and devices are shown in block diagram form, rather than in detail, in order to avoid obscuring the present invention. These embodiments are described in sufficient detail to enable those of ordinary skill in the art to practice the invention, and it is to be understood that other embodiments may be utilized and that logical, mechanical, electrical, and other changes may be made without departing from the scope of the present invention.

[0039] Some portions of the description may be presented in terms of algorithms and symbolic representations of operations on, for example, data bits within a computer memory. These algorithmic descriptions and representations are the means used by those of ordinary skill in the data processing arts to most effectively convey the substance of their work to others of ordinary skill in the art. An algorithm is here, and generally, conceived to be a self-consistent sequence of acts leading to a desired result. The acts are those requiring physical manipulations of physical quantities. Usually, though not necessarily, these quantities take the form of electrical, magnetic, or optical signals capable of being stored, transferred, combined, compared, and otherwise manipulated. It has proven convenient at times, principally for reasons of common usage, to refer to these signals as bits, values, elements, symbols, characters, terms, numbers, or the like.

[0040] It should be borne in mind, however, that all of these and similar terms are to be associated with the appropriate physical quantities and are merely convenient labels applied to these quantities. Unless specifically stated otherwise as apparent from the discussion, it is appreciated that throughout the description, discussions utilizing terms such as "processing" or "computing" or "calculating" or "determining" or "communicating" or "displaying" or the like, can refer to the action and processes of a computer system, or an electronic device, that manipulates and transforms data represented as physical (electronic) quantities within the electronic device or computer system's registers and memories into other data similarly represented as physical quantities within the electronic device and/or computer system memories or registers or other such information storage, transmission, or display devices.

[0041] The present invention can be implemented by an apparatus for performing the operations herein. This apparatus may be specially constructed for the required purposes, or it may comprise a general-purpose computer, selectively activated or

reconfigured by a computer program stored in the computer. Such a computer program may be stored in a computer readable storage medium, such as, but not limited to, any type of disk including floppy disks, hard disks, optical disks, compact disk- read only memories (CD-ROMs), digital versatile disk (DVD), and magnetic-optical disks, read-only memories (ROMs), random access memories (RAMs), electrically programmable read-only memories (EPROM)s, electrically erasable programmable read-only memories (EEPROMs), FLASH memories, magnetic or optical cards, etc., or any type of media suitable for storing electronic instructions either local to the computer or remote to the computer.

[0042] The algorithms and displays presented herein are not inherently related to any particular computer or other apparatus. Various general purpose systems may be used with programs in accordance with the teachings herein, or it may prove convenient to construct more specialized apparatus to perform the required method. For example, any of the methods according to the present invention can be implemented in hard-wired circuitry, by programming a general-purpose processor, or by any combination of hardware and software. One of ordinary skill in the art will immediately appreciate that the invention can be practiced with computer system configurations other than those described, including hand-held devices, multiprocessor systems, microprocessor-based or programmable consumer electronics, digital signal processing (DSP) devices, set top boxes, network PCs, minicomputers, mainframe computers, and the like. The invention can also be practiced in distributed computing environments where tasks are performed by remote processing devices that are linked through a communications network. This communications network is not limited by size, and may range from, for example, on-chip communications to WANs such as the Internet.

[0043] The methods of the invention may be implemented using computer software. If written in a programming language conforming to a recognized standard, sequences of instructions designed to implement the methods can be compiled for execution on a variety of hardware platforms and for interface to a variety of operating systems. It will be appreciated that a variety of programming languages may be used to implement the teachings of the invention as described herein. Furthermore, it is common in the art to speak of software, in one form or another (e.g., program, procedure, application, driver,...), as taking an action or causing a result. Such expressions are merely a shorthand way of saying that execution of the software by a computer causes the processor of the computer to perform an action or produce a result.

[0044] It is to be understood that various terms and techniques are used by those knowledgeable in the art to describe communications, protocols, applications, implementations, mechanisms, etc. One such technique is the description of an implementation of a technique in terms of an algorithm or mathematical expression. That is, while the technique may be, for example, implemented as executing code on a computer, the expression of that technique may be more aptly and succinctly conveyed and communicated as a formula, algorithm, or mathematical expression. Thus, one of ordinary skill in the art would recognize a block denoting A+B=C as an additive function whose implementation in hardware and/or software would take two inputs (A and B) and produce a summation output (C). Thus, the use of formula, algorithm, or mathematical expression as descriptions is to be understood as having a physical embodiment in at least hardware and/or software (such as a computer system in which the techniques of the present invention may be practiced as well as implemented as an embodiment).

[0045] A machine-readable medium is understood to include any mechanism for storing or transmitting information in a form readable by a machine (e.g., a computer). For example, a machine-readable medium includes read only memory (ROM); random access memory (RAM); magnetic disk storage media; optical storage media; flash memory devices; electrical, optical, acoustical, or other form of propagated signals (e.g., carrier waves, infrared signals, digital signals, etc.); etc.

[0046] Thus, a method and apparatus for a configurable address mapping and protection architecture and hardware for on-chip systems have been described.

CLAIMS

What is claimed is:

1. A method comprising:

receiving a request with parameters;

checking said parameters against criteria; and

generating a second request with second parameters if meeting said criteria.

2. The method of claim 1 wherein said second request with second parameters is selected from the group consisting of said request with parameters, parts of said request with parameters, and parts of said request with parameters and new parameters.

3. The method of claim 1 wherein said criteria are selected from the group consisting of predefined, and dynamically generated when a system is operating.

4. The method of claim 3 wherein said predefined criteria are defined at design time for an integrated circuit.

5. The method of claim 1 wherein said criteria are selected from the group consisting of protection ID, destination address, request type, data, data type, and user provided request information.

6. The method of claim 1 wherein said parameters are selected from the group consisting of protection ID, destination address, request type, data, data type, and user provided request information.

7. The method of claim 5 further comprising comparing said destination address with a predefined system address map.

8. The method of claim 7 wherein said request further comprises an address.

9. The method of claim 8 wherein said address has a field selected from the group consisting of a segment address, a region address, and an offset inside a region.

10. The method of claim 9 wherein said checking said parameters against criteria further comprises checking selected from the group consisting of checking said segment address against segment registers, and checking said region address against region registers.

11. The method of claim 1 wherein said checking said parameters against criteria further comprises checking selected from the group consisting of checking a segment address against segment registers, checking a region address against region registers, checking a protection key against protection registers, checking a segment address against one or more segment register fields, checking a region address against one or more region register fields, and checking a protection key against one or more protection register fields.

12. The method of claim 11 wherein said registers and said one or more register fields are selected from the group consisting of not-accessible registers, read-only registers, read-write registers, not-accessible register fields, read-only register fields, and read-write register fields.

13. A processing system comprising a processor, which when executing a set of requests performs the method of claim 1.

14. A machine-readable medium having stored thereon requests, which when executed performs the method of claim 1.

15. An apparatus comprising:

a plurality of source units, each source unit having a communication link;

a checking module having an input communication link and an output communication link, said input communication link coupled to said plurality of source communication links;

a delivery module having an input communication link and an output communication link, said delivery module input communication link coupled to said checking module output communication link; and

a plurality of receiver units, each receiver unit having a communication link, said plurality of receiver communication links coupled to said delivering module output communication link.

16. The apparatus of claim 15 wherein said checking module further comprises checking requests received from said plurality of source units against an address map and a protection key map.

17. The apparatus of claim 16 wherein said delivering module further comprises delivering requests to said receiver units based upon said address map and protection key map if checking was successful.

18. The apparatus of claim 16 wherein said address map and said protection key map are determined at design time and/or dynamically during operation for an integrated circuit.

19. A method comprising:

receiving a map definition;

receiving a set of configuration parameters;

receiving a product specification; and

generating a netlist.

20. The method of claim 19 wherein said map further comprises an address map and a protection key map.

21. The method of claim 20 wherein said netlist represents an optimized address mapping and hardware protection.

22. The method of claim 20 wherein said product specification has parameters selected from the group consisting of a specified number of service modules, address width of a service module, data width of a service module, a specified number of different address regions.

23. The method of claim 22 wherein said different address regions have characteristics selected from the group consisting of fixed, invisible to software routines, and changeable by software routines.

24. An apparatus comprising:

means for receiving requests from one or more processing units;

means for checking said requests against an address map and a protection key map;

means for generating one or more second requests;

means for delivering said one or more second requests to one or more service modules.

25. The apparatus of claim 24 wherein said received requests further comprise a field selected from the group consisting of a destination address, and a protection identification.

26. A machine-readable medium having stored thereon information representing the apparatus of claim 24.

27. A method comprising:

generating an address match signal indicative of a successful or unsuccessful address match;

generating a service module signal indicative of a present or absent service module;

generating a security check signal indicative of passing or failing security; and

forwarding information based on a successful address match, a present service module, and passing security.

28. The method of claim 27 wherein generating said security check further comprises:

receiving a protection key;

checking said protection key against one or more protection registers; and

generating a match or no match signal.

29. The method of claim 28 wherein generating said security check signal further comprises combining said match or no match signal and said present or absent service module signal.

30. A method comprising:

inputting an address map;

inputting protection parameters;

inputting configuration parameters; and

generating a database for configuring circuitry.

31. The method of claim 30 wherein said configuration parameters are selected from the group consisting of address width, number of segments, segment size, segment base, and number of regions.

32. The method of claim 31 further comprising for each region of said number of regions a field selected from the group consisting of region address size, region base address, region protection key register number, region enable, region address space, width of a service module connected to said region, and physical linkage information of a service module connected to said region.

33. The method of claim 32 wherein said physical linkage information of a service module connected to said region further comprises information selected from the group consisting of routing information, and a position.

34. The method of claim 33 wherein said routing information comprises hardware routing information passed to a request-delivery module in order to deliver a request to said service module.

35. The method of claim 33 wherein said position comprises a hardware signal bit position.

36. The method of claim 35 wherein if a request-delivery module asserts a signal at said hardware signal bit position a request is sent to said service module.

37. The method of claim 30 wherein said circuitry further comprises a register.

38. The method of claim 37 wherein said configuring results in an operative mode selected from the group consisting of non-accessible register, read only register, and read-write register.

39. The method of claim 37 wherein said configuring said register further comprises specifying said register as an export constant so that in said database a netlist portion

of said register is exported to a top level of a netlist.

40. The method of claim 39 wherein said configuring further comprises configuring said netlist in time after said configuring of said register.

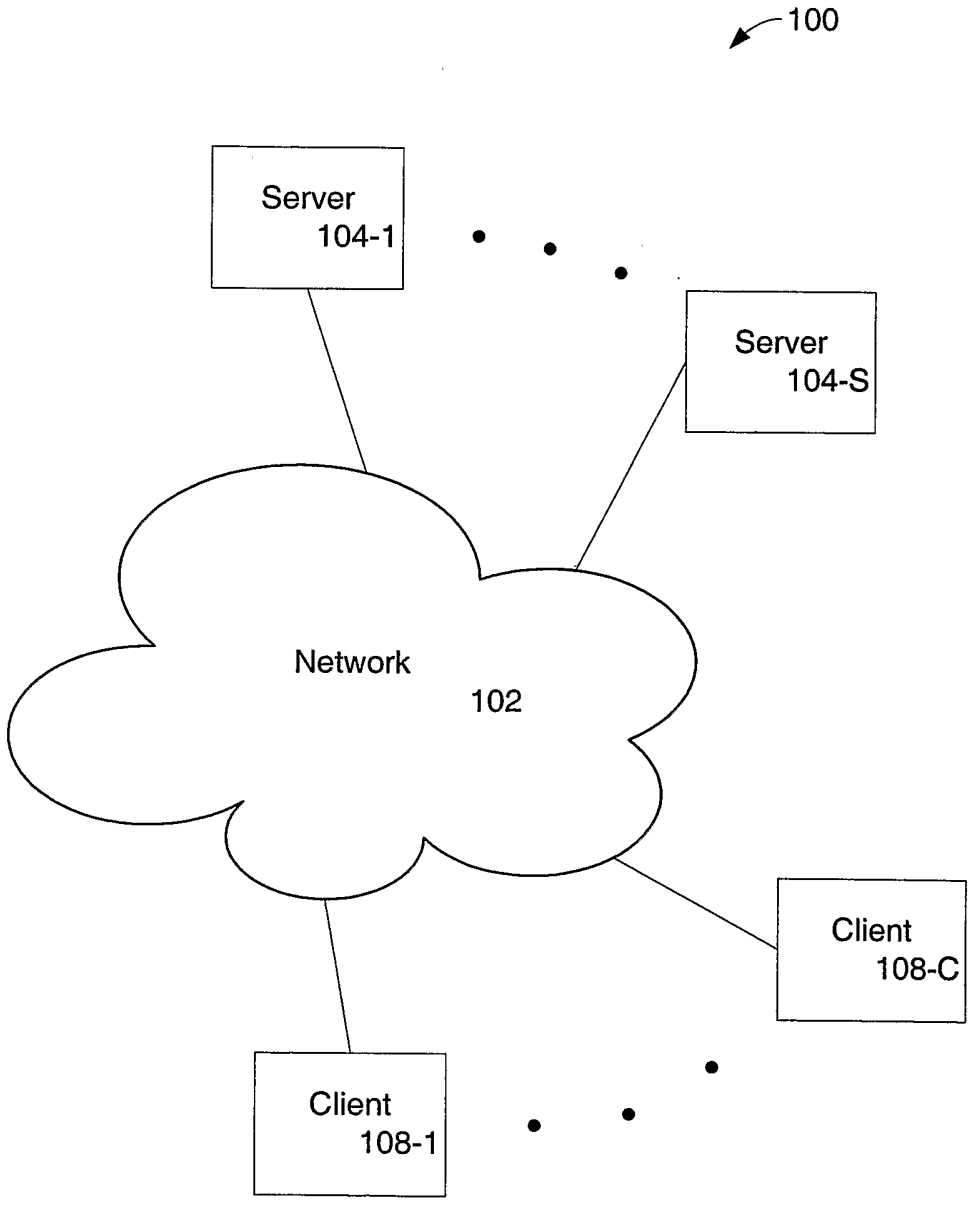
100
Server
104-1
Server
104-S
Network
102
Client
108-C
Client
108-1

FIG. 1

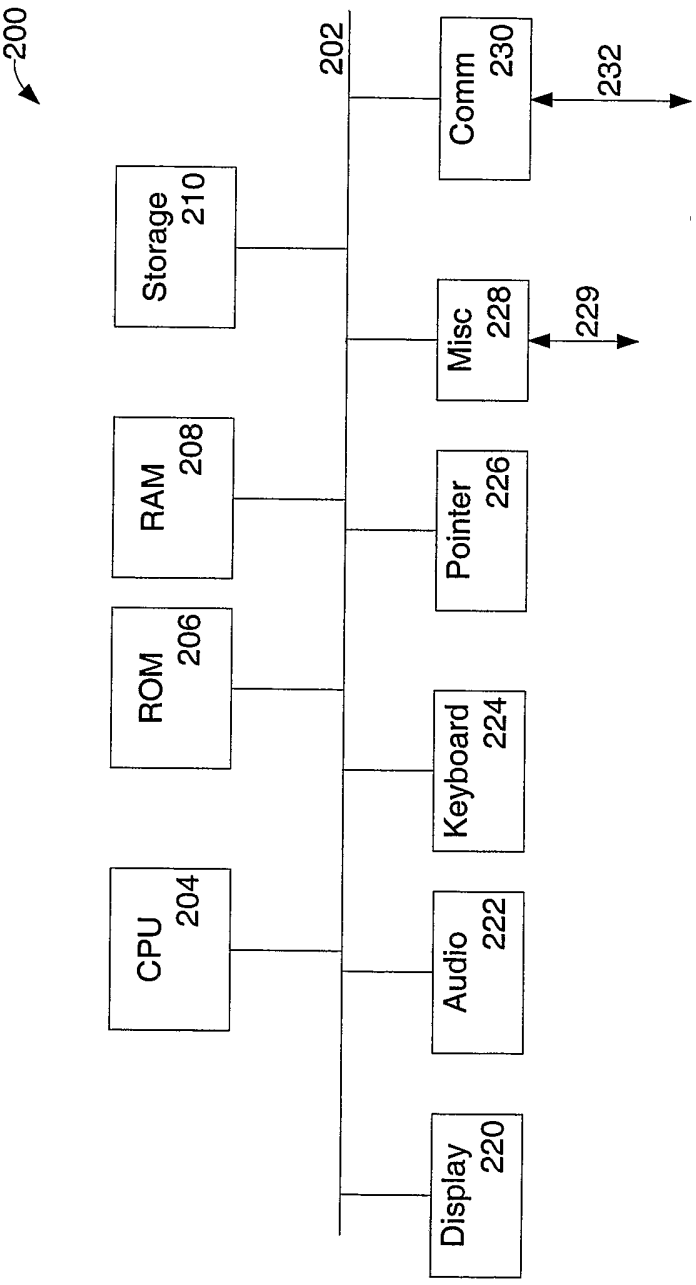
200
202
CPU 204
ROM 206
RAM 208
Storage 210
Display 220
Audio 222
Keyboard 224
Pointer 226
Misc 228
229
Comm 230
232

FIG. 2

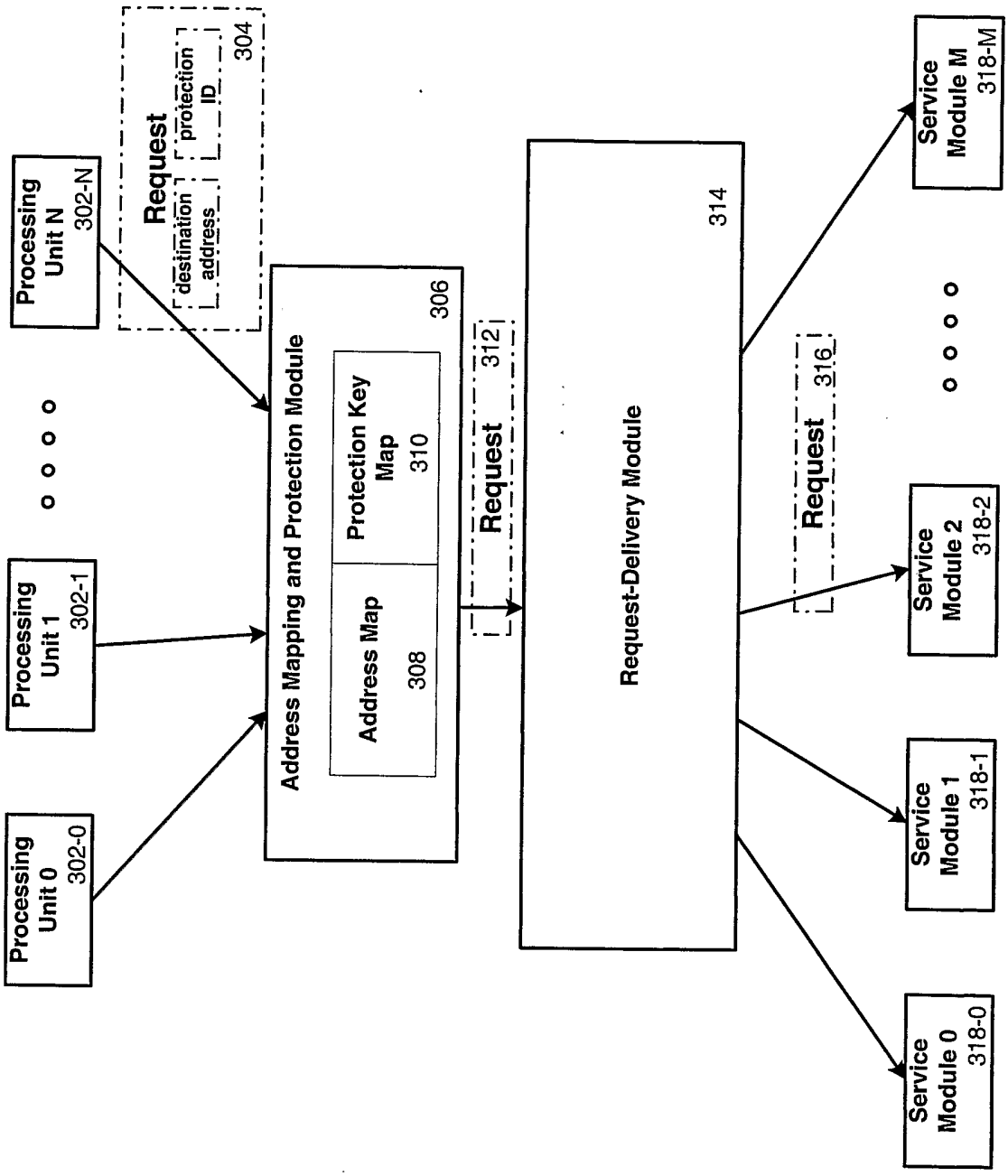
Processing Unit 0
302-0
Processing Unit 1
302-1
Processing Unit N
302-N
Request
destination address
protection ID
304
Address Mapping and Protection Module
306
Address Map
308
Protection Key Map
310
Request
312
Request-Delivery Module
314
Request
316
Service Module 0
318-0
Service Module 1
318-1
Service Module 2
318-2
Service Module M
318-M

FIG. 3

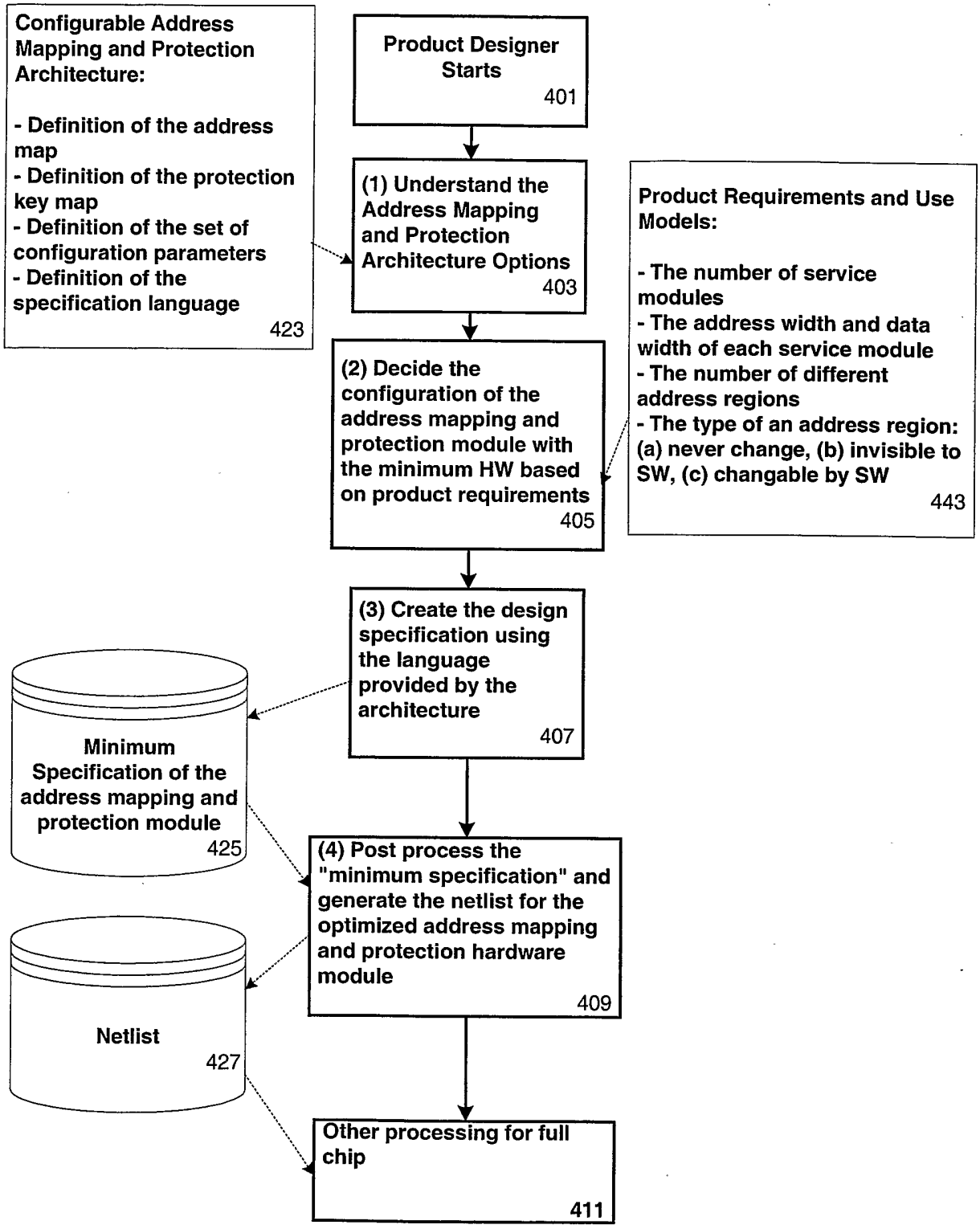
Configurable Address Mapping and Protection Architecture:
- Definition of the address map
- Definition of the protection key map
- Definition of the set of configuration parameters
- Definition of the specification language
423
Product Designer Starts
401
(1) Understand the Address Mapping and Protection Architecture Options
403
Product Requirements and Use Models:
- The number of service modules
- The address width and data width of each service module
- The number of different address regions
- The type of an address region: (a) never change, (b) invisible to SW, (c) changable by SW
443
(2) Decide the configuration of the address mapping and protection module with the minimum HW based on product requirements
405
(3) Create the design specification using the language provided by the architecture
407
Minimum Specification of the address mapping and protection module
425
(4) Post process the "minimum specification" and generate the netlist for the optimized address mapping and protection hardware module
409
Netlist
427
Other processing for full chip
411

FIG. 4

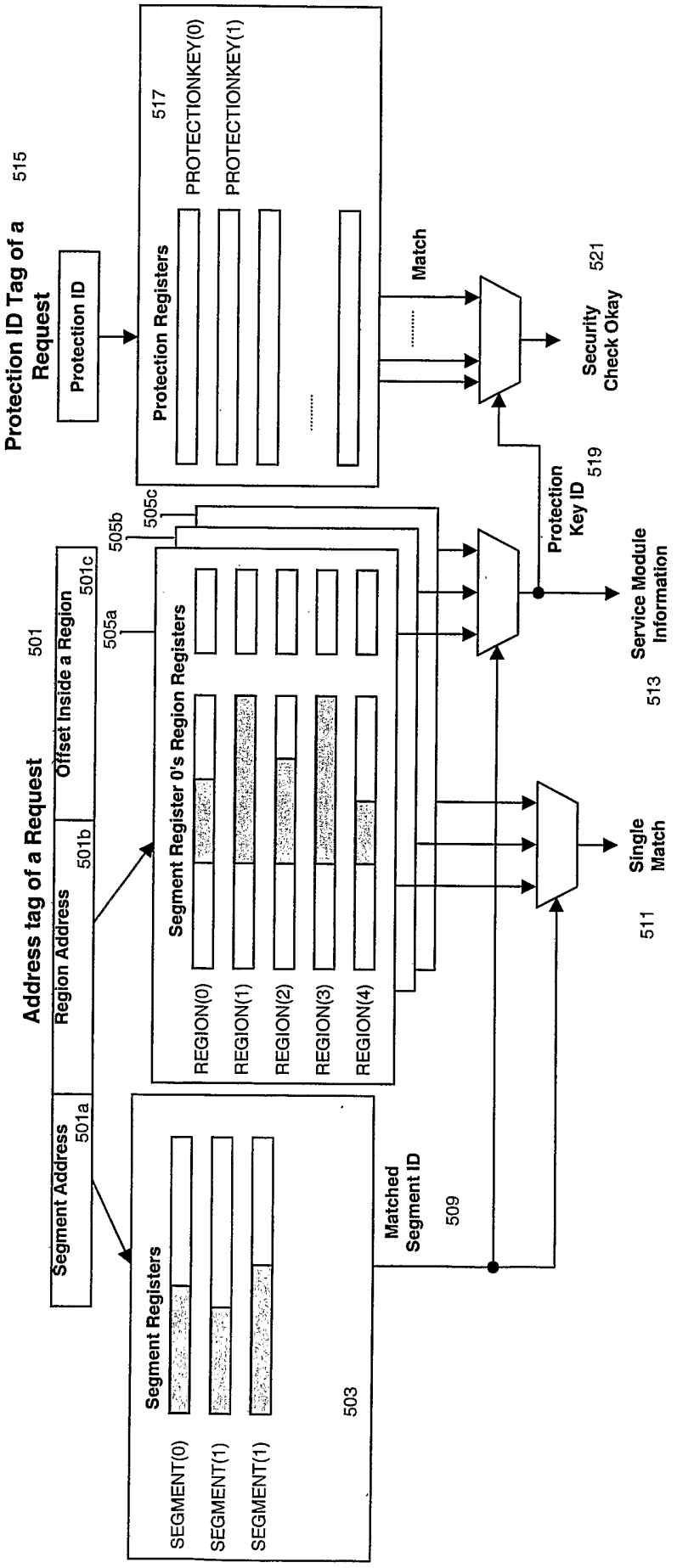
Protection ID Tag of a Request
515
Protection ID
Protection Registers
517
PROTECTIONKEY(0)
PROTECTIONKEY(1)
Match
Security Check Okay
521
Protection Key ID
519
Address tag of a Request
501
Segment Address
501a
Region Address
501b
Offset Inside a Region
501c
505a
505b
505c
Segment Register 0's Region Registers
REGION(0)
REGION(1)
REGION(2)
REGION(3)
REGION(4)
Service Module Information
513
Segment Registers
SEGMENT(0)
SEGMENT(1)
SEGMENT(1)
503
Matched Segment ID
509
Single Match
511

FIG. 5

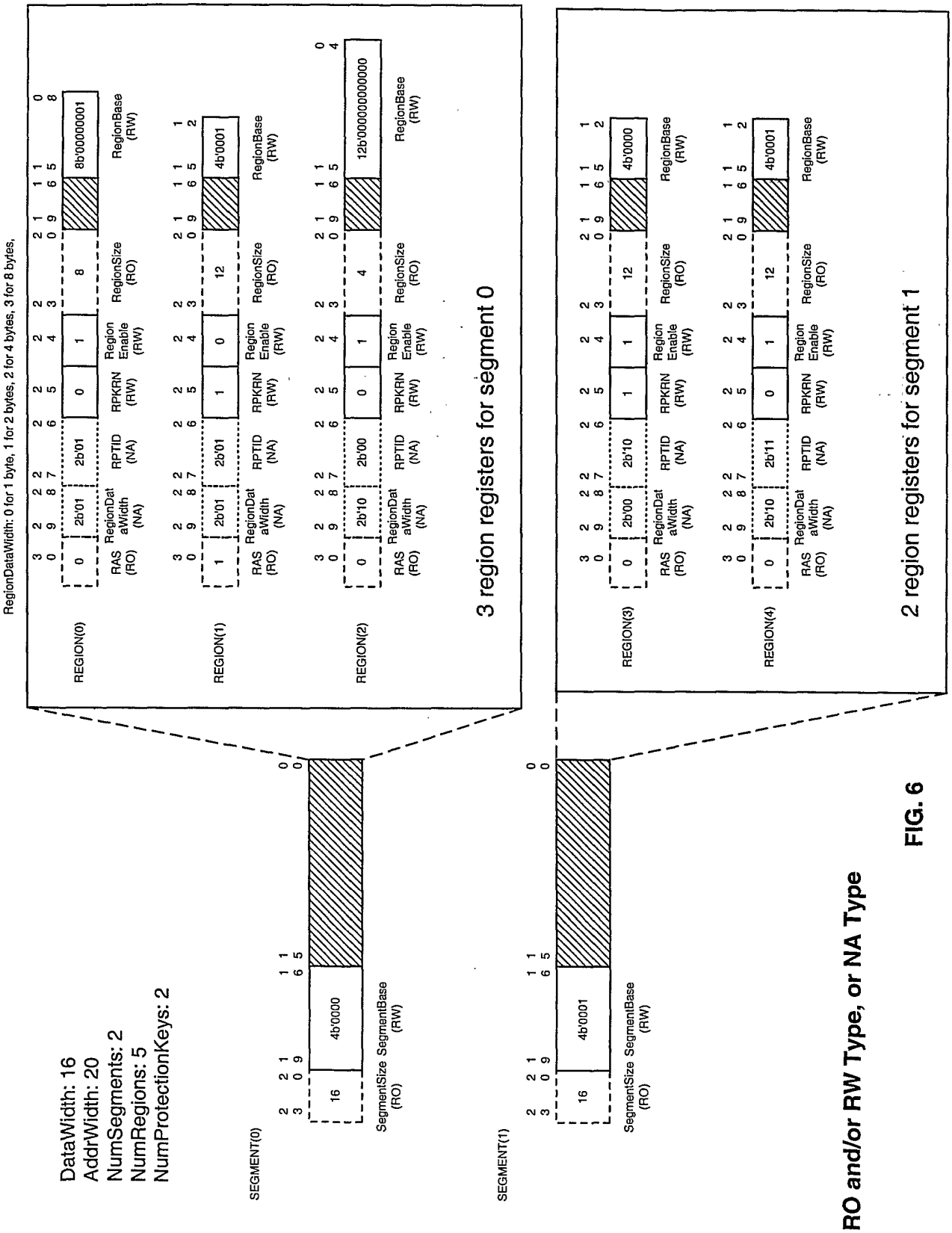
DataWidth: 16
AddrWidth: 20
NumSegments: 2
NumRegions: 5
NumProtectionKeys: 2
RegionDataWidth: 0 for 1 byte, 1 for 2 bytes, 2 for 4 bytes, 3 for 8 bytes,
SEGMENT(0)
SEGMENT(1)
SegmentSize (RO)
SegmentBase (RW)
REGION(0)
REGION(1)
REGION(2)
REGION(3)
REGION(4)
RAS (RO)
RegionDataWidth (NA)
RPTID (NA)
RPKRN (RW)
Region Enable (RW)
RegionSize (RO)
RegionBase (RW)
3 region registers for segment 0
2 region registers for segment 1
RO and/or RW Type, or NA Type

FIG. 6

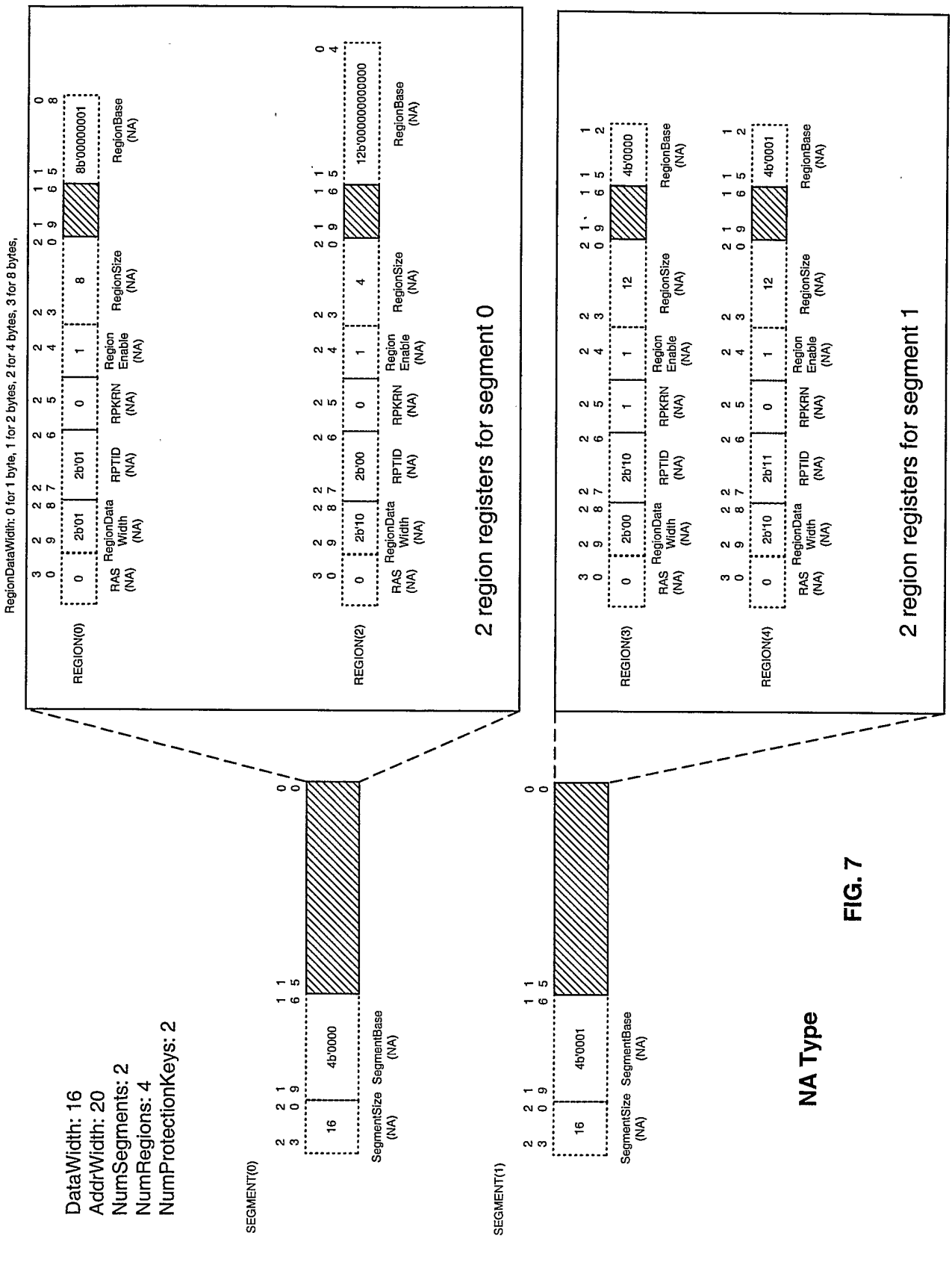
DataWidth: 16
AddrWidth: 20
NumSegments: 2
NumRegions: 4
NumProtectionKeys: 2
RegionDataWidth: 0 for 1 byte, 1 for 2 bytes, 2 for 4 bytes, 3 for 8 bytes,
SEGMENT(0)
SEGMENT(1)
SegmentSize (NA)
SegmentBase (NA)
REGION(0)
REGION(2)
REGION(3)
REGION(4)
RAS (NA)
RegionData Width (NA)
RPTID (NA)
RPKRN (NA)
Region Enable (NA)
RegionSize (NA)
RegionBase (NA)
2 region registers for segment 0
2 region registers for segment 1
NA Type

FIG. 7