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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
17 March 2011 (17.03.2011)

PCT

(10) International Publication Number
WO 2011/031361 A1

(51) International Patent Classification:
G06F 7/38 (2006.01)

(21) International Application Number:
PCT/US2010/038350

(22) International Filing Date:
11 June 2010 (11.06.2010)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
12/556,440 9 September 2009 (09.09.2009) US

(71) Applicant *(for all designated States except US)*: **BOARD OF REGENTS, UNIVERSITY OF TEXAS SYSTEM** [US/US]; The University Of Texas At Austin, 3925 West Braker Lane, Suite 1.9a, Austin, TX 78759 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **BURGER, Doug** [US/US]; 5002 Lucas Lane, Austin, TX 78731 (US). **KECKLER, Stephen, W.** [US/US]; 11304 Chessington Drive, Austin, TX 78750 (US). **ESMAEILZADEH, Hadi** [IR/US]; 2501 Lake Austin Boulevard, #e102, Austin, TX 78703 (US).

(74) Agents: **MILLS, Alicia, G.** et al.; Dorsey & Whitney LLP, Suite 1500, 50 South Sixth Street, Minneapolis, MN 55402-1498 (US).

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

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

Published:

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

(54) Title: METHOD, SYSTEM AND COMPUTER-ACCESSIBLE MEDIUM FOR PROVIDING A DISTRIBUTED PREDICATE PREDICTION

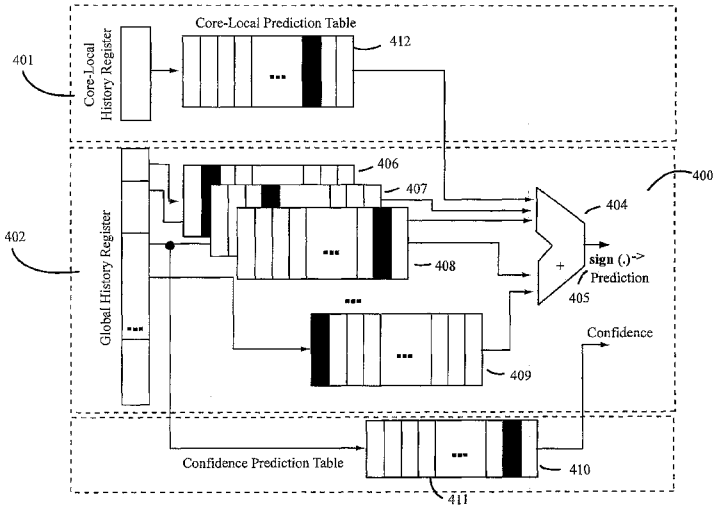

FIG. 4

(57) Abstract: Examples of a system, method and computer accessible medium are provided to generate a predicate prediction for a distributed multi-core architecture. Using such system, method and computer accessible medium, it is possible to intelligently encode approximate predicate path information on branch instructions. Using this statically generated information, distributed predicate predictors can generate dynamic predicate histories that can facilitate an accurate prediction of high-confidence predicates, while minimizing the communication between the cores.

WO 2011/031361 A1

Method, System and Computer-Accessible Medium for Providing a Distributed Predicate Prediction

Statement Regarding Government Sponsored Research

[0001] This invention was made with government support under F33615-03-C-4106 awarded by the Air Force. The U.S. government has certain rights in the invention.

Background

[0002] In distributed, large-window processors a predication is a technique used to convert branches, which cause frequent changes in the control flow of the programs, to data values, which can guard instructions, and may determine which instructions are executed and which are not. Predication can linearize control flow, facilitating instructions to be provided down both possible paths which a branch may take to be collapsed, and fetching all of the instructions and only executing some of them depending on the predicate. While this model can be effective to generate large blocks of code to spread out over many execution units, it can create a problem in that the predicates, which would have been branches in a non-predicated architecture, may be evaluated at execute time (unlike branches, which are predicted shortly after they are fetched). This deferral of the evaluation of the predicates may reduce performance significantly.

[0003] Current technologies adopt one of two possibly undesirable options. First, such technologies prefer to avoid predication, which can leave every control decision as a branch, and may preclude distributing instructions over a large distributed processor. A second alternative can be to predicate instructions, but to centralize fetch and predicate prediction into a

single unit, resulting in low fetch bandwidth again precluding the distribution over a large distributed multi-core processor.

[0004] In hybrid dataflow architectures, such as Explicit Data Graph Execution (EDGE) architectures, a control flow can be a mixture of branches and predicates. Determining which branches can be if-converted to predicates may be a complex problem. It can be preferable for hard-to-predict branches to be predicated, and branches that facilitate sufficiently linearized control flow are predicated. The remaining control points can be left as branches. With such partitioning, the prediction scheme can predict all branches and the most predictable predicates, deferring the evaluation of hard-to-predict predicates to be preferred at execute time.

[0005] In a distributed dataflow machine, however, maintaining complete control histories to facilitate accurate predicate prediction can be difficult. Previous efforts in this area have typically relied on a compiler that applies "if-conversion" to hard-to-predict branches, and a particular microarchitectural mechanism to recover the cases, where the compiler makes a mistake due to a lack of run-time information. Consequently, most dataflow machines known to date have not employed a predicate prediction in a manner that can be effective for commercial applications.

Summary

[0006] Examples of a system, method and computer accessible medium are provided to generate a predicate prediction for a distributed multi-core architecture. Using such system, method and computer accessible medium, it is possible to intelligently encode approximate predicate path information on branch instructions. Using this statically generated information, distributed predicate predictors can generate dynamic predicate histories that can facilitate an

accurate prediction of high-confidence predicates, while minimizing the communication between the cores.

[0007] In one example, a computing system is provided. The computing system may include a multi-core processor that has a plurality of processor cores. Each of the cores may include a predicate predictor. The predicate predictor may be configured to generate a predicate prediction.

[0008] In another example, a method for providing a predicate prediction in a multi-core processor is provided. The method may include providing a predicate predictor for each of a plurality of processor cores in the multi-core processor and generating the predicate prediction from a plurality of branch instructions using the predicate predictors.

[0009] In yet another example, a computer accessible medium having stored thereon computer executable instructions for providing a predicate prediction within a multi-core processor computing system is provided. A processing arrangement may be configured to perform processing procedures when the processing arrangement executes the instructions. The processing procedures may include providing a predicate predictor for each of a plurality of processor cores in the multi-core processor, each of the processor cores including at least one predicate predictor, and generating the predicate prediction using the predicate predictors.

[0010] While multiple examples are disclosed, still other examples will become apparent to those skilled in the art from the following detailed description. As will be apparent, the system, method and computer accessible medium is capable of modifications in various obvious aspects, all without departing from the spirit and scope of the teachings herein. Accordingly, the detailed description is to be regarded as illustrative in nature and not restrictive.

Brief Description of the Figures

[0011] The foregoing and other features of the present disclosure will become more fully apparent from the following description and appended claims, taken in conjunction with the accompanying drawings. Understanding that these drawings depict only several examples in accordance with the disclosure and are, therefore, not to be considered limiting of its scope, the disclosure will be described with additional specificity and detail through use of the accompanying drawings, in which:

FIG. 1 is a block diagram of a representative computing system in accordance with one example;

FIG. 2 is a block and flow diagram of a predicate prediction system in accordance with one example;

FIG. 3 is a block and flow diagram of the predicate prediction system in accordance with another example;

FIG. 4 is a block and flow diagram of a geometric history length predictor in accordance with one example;

FIG. 5 is a block and flow diagram of a suitable method for a predicate prediction in accordance with one example; and

FIGS. 6(a) and 6(b) are block and flow diagrams of a global history register configuration in accordance with some examples.

Detailed Description

[0012] In the following detailed description, reference is made to the accompanying drawings, which form a part hereof. In the drawings, similar symbols typically identify similar components, unless context dictates otherwise. The illustrative examples described in the

detailed description, drawings, and claims are not meant to be limiting. Other examples may be utilized, and other changes may be made, without departing from the spirit or scope of the subject matter presented herein. It will be readily understood that the aspects of the present disclosure, as generally described herein, and illustrated in the Figures, can be arranged, substituted, combined, separated, and designed in a wide variety of different configurations, all of which are implicitly contemplated herein.

[0013] This disclosure is drawn to methods, apparatus, computer programs and systems related to branch prediction. Certain preferred embodiments of one such system are illustrated in the figures and described below. Many other embodiments are also possible, however, time and space limitations prevent including an exhaustive list of those embodiments in one document. Accordingly, other embodiments within the scope of the claims will become apparent to those skilled in the art from the teachings of this patent.

[0014] The figures include numbering to designate illustrative components of examples shown within the drawings, including the following: a computer system 100, a processor 101, a system bus 102, an operating system 103, an application 104, a read-only memory 105, a random access memory 106, a disk adapter 107, a disk unit 108, a communications adapter 109, an interface adapter 110, a display adapter 111, a keyboard 112, a mouse 113, a speaker 114, a display monitor 115, a computing environment 201, an application program 202, an instruction data flow 203, a compiler 204, branch instructions 205, an approximate predicate path information 206, a second instruction data flow 207,a processor 210, processor cores 211–213, predicate predictors 214–216, a predicate prediction 220, block instructions 301–303, block starting addresses 304–306, a geometric history length predictor 400, a core-local history register 401, a global history register 402, a summation block 404, a

prediction sign 405, global prediction tables 406–409, a confidence prediction table 410, a counter 411, a core-local prediction table 412, a core-local predicate history register, and global history registers 601–603, 605.

[0015] FIG. 1 is a schematic illustration of a block diagram of a computing system 100 arranged in accordance with some examples. Computer system 100 is also representative of a hardware environment for an example of the present disclosure. For example, computer system 100 may have a processor 101 coupled to various other components by a system bus 102. Processor 101 may be a heterogeneous multi-core processor with a plurality of predicate predictors 214–216 arranged in accordance with the examples herein. A more detailed description of processor 101 is provided below in connection with a description of the example shown in FIG. 2. Referring to FIG. 1, an operating system 103 may run on processor 101, and provide control and coordinate the functions of the various components of FIG. 1. An application 104 in accordance with the principles of examples of the present disclosure may execute in conjunction with operating system 103, and provide calls and / or instructions to operating system 103 where the calls / instructions implement the various functions or services to be performed by application 104.

[0016] Referring to FIG. 1, a read-only memory ("ROM") 105 may be coupled to system bus 102, and can include a basic input/output system ("BIOS") that can control certain basic functions of computer device 100. A random access memory ("RAM") 106 and a disk adapter 107 may also be coupled to system bus 102. It should be noted that software components, including operating system 103 and application 104, may be loaded into RAM 106, which may be computer system's 100 main memory for execution. A disk adapter 107 may be provided which can be an integrated drive electronics ("IDE") or parallel advanced technology

attachment ("PATA") adapter, a serial advanced technology attachment ("SATA") adapter, a small computer system interface ("SCSI") adapter, a universal serial bus ("USB") adapter, an IEEE 1394 adaptor, or any other appropriate adapter that communicates with a disk unit 108, e.g., disk drive.

[0017] Referring to FIG. 1, computer system 100 may further include a communications adapter 109 coupled to bus 102. Communications adapter 109 may interconnect bus 102 with an external network (not shown) thereby facilitating computer system 100 to communicate with other similar and / or different devices.

[0018] Input / Output ("I/O") devices may also be connected to computer system 100 via a user interface adapter 110 and a display adapter 111. For example, a keyboard 112, a mouse 113 and a speaker 114 may be interconnected to bus 102 through user interface adapter 110. Data may be provided to computer system 100 through any of these example devices. A display monitor 115 may be connected to system bus 102 by display adapter 111. In this example manner, a user can provide data or other information to computer system 100 through keyboard 112 and / or mouse 113, and obtain output from computer system 100 via display 115 and / or speaker 114.

[0019] The various aspects, features, embodiments or implementations of examples of the present disclosure described herein can be used alone or in various combinations. The method examples of the present disclosure can be implemented by software, hardware or a combination of hardware and software (e.g., software stored on a computer-accessible medium).

[0020] Described herein is an example of a predicate prediction scheme for a distributed multi-core microarchitecture, which may be implemented on processor 101, and e.g., can be adapted to be used with an explicit data graph execution (EDGE) microarchitecture. The

example of the distributed scheme may rely on the compiler to intelligently encode approximate predicate path information in the branch instructions. Using this statically generated information, distributed predicate predictors may generate dynamic predicate histories that facilitate accurate prediction of high-confidence predicates, while reducing the communication between the cores. The example of the accurate and efficient distributed predicate prediction scheme facilitates the compiler to aggressively predicate the code, e.g., relying on the predicate predictor to mitigate the overhead computation.

[0021] Thus, as shown in FIG. 2, a computing environment 201 can be provided in the computer system 100 and may include a software application program 202 and a compiler 204. Application program 202 may produce an instruction data flow 203, some or many of which may be branch instructions. Compiler 204 may encode branch instructions 205 with an approximate predicate path information 206. A resulting instruction data 207 may flow to a processor 101, 210 for execution. Processor 210 may include a plurality of processor cores (for simplicity, e.g., three processor cores are depicted as cores 211–213), each of which can include a respective one of predicate predictors 214–216. These predictors 214–216 may then use compiler-encoded predicate path information 206 to facilitate accurate predictions of high-confidence predicates [block 220].

[0022] Provided below is a further description of the EDGE instruction set architecture (ISA). However, it should be appreciated that examples of the present invention may be similarly used with other ISAs. The EDGE ISA has two example characteristics, e.g., a block-atomic execution (either all of the instructions of a block complete and commit, or none of them do), and a direct instruction communication (the ISA can encode the intra-block dependences in the instructions). Using this example model, a block of instruction data flow,

e.g., instruction data flow 203, can complete its operation when it produces a consistent set of outputs. For example, with each round of execution, a block can write or provide data to the same number of registers, generate the same number of stored data as being statically encoded in the block header in addition to generating exactly one branch output. Instructions may communicate between blocks through registers and memory 105, 106.

[0023] The instructions may be interleaved across some or all cores aggregated as one processor 101, 210 based on their index in the block of instructions. When the processor core configuration changes, e.g., when the cores are operating as distinct processing units, the interleaving procedure may change accordingly, which can facilitate the processor to execute the block on one or more of the cores. Each instruction block may have a predicate operand, which can make the EDGE ISA fully predicated. Using the dataflow execution model (e.g., a direct operand communication), an instruction block can be executed when it receives all its operands, and if it is predicated, the corresponding predicate.

[0024] In an architecture according to one example, each block of instructions may have a particular core (e.g., an owner core) which coordinates the connected or participating cores to facilitate the execution of such block. For example, the owner core may be identified by a block starting address, similar or equivalent to a program counter in conventional architectures. Turning to FIG. 3, a plurality of blocks of instructions (e.g., three blocks being depicted for simplicity as blocks 301–303) are provided, each block including a respective one of block starting addresses 304–306. Such addresses 304–306 can be used to identify appropriate owner cores 211–213. As depicted in FIG 3, block 301 can be associated with (e.g., owned by) core 211 through address respective 304, block 302 can be associated with core 213 through respective address 305, and block 303 can be associated with core 212 through address

respective 306. The respective owner cores 211–213 may be responsible for transmitting the fetch command, predicting the next block address and passing an ownership token back thereto, gathering a completion information (e.g., including register writes, stores, exit/branch address, etc.), transmitting commit or flush command(s), and eventually obtaining the commit acknowledgement.

[0025] Furthermore, one or more of the respective owner cores 211–213 may facilitate the prediction of the next block address. Each one of the cores 211–213 can include a respective one of fully functional block predictors 214–216, and the predictors may be identical across the cores or different from one another. The next one of block predictors 214–216 may include an exit predictor that can predict which branch is to be taken out from a block, and the associated target one of predictors 214–216 that can predict the address of the next one of blocks 214–216 based on the predicted exit target. Referring to FIG. 4, the exit predictor of predictors 214–216 may include a two-level local predictor, a global predictor, and/or a choice predictor, which can use local exit history 401 and global exit history 402. Exit histories 401, 402 may be generated from the cores 211–213 using the approximate predicate path information 206 assigned statically to each branch instruction in the respective block. The approximate predicate path information 206 may be encoded in the branch instructions, and can identify the specific branches of a particular block. Compiler 204 may originally assign the approximate predication path information 206 based on the sequential order of the branch instructions in the respective block. The cores 211–213 may use the exit to generate the local and global histories in the associated predictor instead of a taken/not taken information (as may be used in conventional architectures).

[0026] Each of the cores 211–213 may be augmented with a predicate prediction arrangement that may predict the output of the predicate instructions mapped on that core. A global history based predictor, which may include a base predictor and a global history register, may be used in each core. Such global history predictor may attempt to maintain the global history information updated in each of the cores 211–213, while reducing the communication among cores 211–213. First, referring to FIG. 4, in one example, a geometric history length (GEHL) predictor 400 may be used as the base predictor for performing a distributed predicate prediction. GEHL predictor 400 may include several prediction tables 406–409 indexed by independent functions of a global branch history register 402 and branch address (e.g., a core-local history register 401). The prediction may be based on the sign 405 of the summation 404 over the values retrieved from tables 406–409, as well as a value from a core-local prediction table 412 associated with core-local history register 401. In this example, a large part of the storage may be used to capture a correlation of the recent branch histories, while still facilitating a capture of the correlation with certain old branches.

[0027] Turning now to global predicate history information, several suitable examples can be provided. Certain examples may achieve a high degree of accuracy, while reducing communication among the cores 211–213. In one such example, referring to FIG. 6a, a core-local predicate history register (CLPHR) 600 can be used. Thus, e.g., the predictor may be provided to merely use the information which is only available in the core, without communicating any information with other cores. In this example, each of the cores 211–213 can have its own exclusive global history register 601–603 which can keep track of the predicate instructions mapped to such core. Compiler 204 can then attempt to map dependent instructions

to that core. Consequently, dependent predicate instructions may be mapped to the same core, which can facilitate the CLPHR 600 to utilize the correlation between those instructions.

[0028] In another example, referring to FIG. 6b, a global block history register (GBHR) 605 may be provided. As the distributed exit predictor predicts the exit codes of the blocks, such exit predictor can also concatenate the exit prediction numbers or codes, thus producing the GBHR 605. In this example, compiler 204 may assign 3-bit exit codes to the branch instructions in each block, according to their sequential order in the program. Hence, the GBHR 605 can be used as the global history information. Without using any information from the predicates in the blocks, both exit and predicate predictors may share the same history information, which can reduce or eliminate the need for an additional communication mechanism.

[0029] The examples of CLPHR 600 and GBHR 605 may be combined in any suitable manner. For example, GBHR 605 may be augmented by adding another table which can be indexed by CLPHR rather than the main global history register. The prediction retrieved from this table is can be combined through an adder tree with the predictions retrieved from the GEHL tables.

[0030] In some examples, referring again to FIG. 4, a confidence prediction table 410 may be provided which may be configured to perform an estimation of the accuracy of a predicate prediction. Confidence prediction table 410 may thus facilitate the predictor to "filter out" hard-to-predict predicates (e.g., predicates with low confidence). The confidence prediction table 410 entries may be resetting counters, e.g., counter 411, that can count the number of correct consecutive predictions in a branch. For example, if a misprediction occurs, counter 411 may be reset to zero. A predicate can be selected to be predicted if the corresponding counter's

value is higher than a certain threshold. In some examples, the entries in the confidence table (e.g., counter 411) can be 3-bit resetting counters which may support a high confidence threshold for configuration with a fewer number of cores.

[0031] An example of a method of the present invention is depicted as FIG. 5. Computer system 100 can include processor 101, which when it is executed, may be configured to perform the following procedures. In particular, processor 101 can cause compiler 204 to encode the branch instructions with approximate predicate path information [procedure 501]. Next, the resulting instruction data may be provided to processor 101 for execution [procedure 502]. These predictors associated with each processor core may then use compiler-encoded path information to facilitate accurate predictions of high-confidence predicates [procedure 503].

[0032] Disclosed in some examples is a computing system comprising a multi-core processor comprising a plurality of processor cores, each of the cores comprising at least one predicate predictor, wherein the predicate predictors generate a predicate prediction. In some examples, the computing system may further comprise an application program comprising one or more branch instructions having a predicate path information encoded thereon. In other examples, the encoding of predicate path information is accomplished by a compiler. In various other examples, the block address on each of the one or more branch instructions determines which processor core in the multi-core processor is to execute the respective branch instruction. In further examples, the multi-core processor comprises an explicit data graph execution microarchitecture. In still further examples, the one or more predicate predictors comprise a base predictor and a global history register. In other examples, the base predictor is a geometric history length predictor. While in some examples, the global history register is a core-local predicate history register. In further examples, the global history register is a global block

history register. In yet further examples, the global history register comprises a core-local predicate history register and a global block history register. In still other examples, the computing system further comprises a confidence prediction table.

[0033] Disclosed in other examples is a method for providing a predicate prediction in a multi-core processor comprising providing one or more branch instructions via a plurality of processor cores in the multi-core processor, each of the processor cores comprising at least one predicate predictor, and generating the predicate prediction using the predicate predictors. In some examples, the method may further comprise encoding approximate predicate path information in one or more branch instructions. In other examples, the encoding of predicate path information is performed by a compiler. In further examples, the method may additionally comprise determining which processor core is to execute a branch instruction using a block address for each of the one or more branch instructions. While in other examples, the one or more predicate predictors comprise a base predictor and a global history register. In still further examples, the base predictor is a geometric history length predictor. In yet other examples, the global history register is a core-local predicate history register. In various other examples, the global history register is a global block history register.

[0034] Disclosed in yet other examples is a computer accessible medium having stored thereon computer executable instructions for providing a predicate prediction within a multi-core processor computing system, wherein when a processing arrangement executes the instructions, the processing arrangement is configured to perform procedures comprising encoding approximate predicate path information in one or more branch instructions, executing the one or more branch instructions on one or more processor cores in the multi-core processor,

each of the one or more processor cores comprising one or more predicate predictors, and generating a predicate prediction using the one or more predicate predictors.

[0035] The present disclosure is not to be limited in terms of the particular examples described in this application, which are intended as illustrations of various aspects. Many modifications and examples can be made without departing from its spirit and scope, as will be apparent to those skilled in the art. Functionally equivalent methods and apparatuses within the scope of the disclosure, in addition to those enumerated herein, will be apparent to those skilled in the art from the foregoing descriptions. Such modifications and examples are intended to fall within the scope of the appended claims. The present disclosure is to be limited only by the terms of the appended claims, along with the full scope of equivalents to which such claims are entitled. It is to be understood that this disclosure is not limited to particular methods, reagents, compounds compositions or biological systems, which can, of course, vary. It is also to be understood that the terminology used herein is for the purpose of describing particular examples only, and is not intended to be limiting.

[0036] With respect to the use of substantially any plural and/or singular terms herein, those having skill in the art can translate from the plural to the singular and/or from the singular to the plural as is appropriate to the context and/or application. The various singular/plural permutations may be expressly set forth herein for sake of clarity.

[0037] It will be understood by those within the art that, in general, terms used herein, and especially in the appended claims (e.g., bodies of the appended claims) are generally intended as "open" terms (e.g., the term "including" should be interpreted as "including but not limited to," the term "having" should be interpreted as "having at least," the term "includes" should be interpreted as "includes but is not limited to," etc.). It will be further understood by

those within the art that if a specific number of an introduced claim recitation is intended, such an intent will be explicitly recited in the claim, and in the absence of such recitation no such intent is present. For example, as an aid to understanding, the following appended claims may contain usage of the introductory phrases "at least one" and "one or more" to introduce claim recitations. However, the use of such phrases should not be construed to imply that the introduction of a claim recitation by the indefinite articles "a" or "an" limits any particular claim containing such introduced claim recitation to examples containing only one such recitation, even when the same claim includes the introductory phrases "one or more" or "at least one" and indefinite articles such as "a" or "an" (e.g., "a" and/or "an" should be interpreted to mean "at least one" or "one or more"); the same holds true for the use of definite articles used to introduce claim recitations. In addition, even if a specific number of an introduced claim recitation is explicitly recited, those skilled in the art will recognize that such recitation should be interpreted to mean at least the recited number (e.g., the bare recitation of "two recitations," without other modifiers, means at least two recitations, or two or more recitations). Furthermore, in those instances where a convention analogous to "at least one of A, B, and C, etc." is used, in general such a construction is intended in the sense one having skill in the art would understand the convention (e.g., "a system having at least one of A, B, and C" would include but not be limited to systems that have A alone, B alone, C alone, A and B together, A and C together, B and C together, and/or A, B, and C together, etc.). In those instances where a convention analogous to "at least one of A, B, or C, etc." is used, in general such a construction is intended in the sense one having skill in the art would understand the convention (e.g., "a system having at least one of A, B, or C" would include but not be limited to systems that have A alone, B alone, C alone, A and B together, A and C together, B and C together, and/or A, B, and C together, etc.). It will be

further understood by those within the art that virtually any disjunctive word and/or phrase presenting two or more alternative terms, whether in the description, claims, or drawings, should be understood to contemplate the possibilities of including one of the terms, either of the terms, or both terms. For example, the phrase "A or B" will be understood to include the possibilities of "A" or "B" or "A and B."

[0038] In addition, where features or aspects of the disclosure are described in terms of Markush groups, those skilled in the art will recognize that the disclosure is also thereby described in terms of any individual member or subgroup of members of the Markush group.

[0039] As will be understood by one skilled in the art, for any and all purposes, such as in terms of providing a written description, all ranges disclosed herein also encompass any and all possible subranges and combinations of subranges thereof. Any listed range can be easily recognized as sufficiently describing and enabling the same range being broken down into at least equal halves, thirds, quarters, fifths, tenths, etc. As a non-limiting example, each range discussed herein can be readily broken down into a lower third, middle third and upper third, etc. As will also be understood by one skilled in the art all language such as "up to," "at least," "greater than," "less than," and the like include the number recited and refer to ranges which can be subsequently broken down into subranges as discussed above. Finally, as will be understood by one skilled in the art, a range includes each individual member. Thus, for example, a group having 1-3 cells or cores refers to groups having 1, 2, or 3 cells or cores. Similarly, a group having 1-5 cells or cores refers to groups having 1, 2, 3, 4, or 5 cells or cores, and so forth.

[0040] While various aspects and examples have been disclosed herein, other aspects and examples will be apparent to those skilled in the art. The various aspects and examples

disclosed herein are for purposes of illustration and are not intended to be limiting, with the true scope and spirit being indicated by the following claims.

Claims

What Is Claimed Is:

1. A computing system, comprising a multi-core processor that includes a plurality of processor cores, each of the cores comprising a predicate predictor, wherein the predicate predictors are configured to generate a predicate prediction.

2. The computing system of claim 1, wherein at least one of the cores executes an application program comprising a branch instruction having predicate path information encoded thereon, and the predicate predictor is configured to generate the predicate prediction based on the predicate path information.

3. The computing system of claim 2, wherein at least one of the cores is a compiler which is configured to encode the predicate path information on the branch instruction.

4. The computing system of claim 2, wherein the branch instruction has a block address that determines which of the cores in the multi-core processor is assigned to execute the branch instruction.

5. The computing system of claim 1, wherein the multi-core processor comprises an explicit data graph execution microarchitecture.

6. The computing system of claim 1, wherein the predicate predictor comprises a base predictor and a global history register.

7 The computing system of claim 6, wherein the base predictor comprises a geometric history length predictor.

8. The computing system of claim 6, wherein the global history register comprises a core-local predicate history register.

9. The computing system of claim 6, wherein the global history register comprises a global block history register.

10. The computing system of claim 6, wherein the global history register comprises a core-local predicate history register and a global block history register.

11. The computing system of claim 1, wherein the predicate predictors are configured to generate a plurality of predicate predictions, and at least one of the cores is configured to obtain a confidence prediction indicative of the accuracy of the predicate predictions, and to determine which predicates should be subsequently predicted based on the confidence prediction.

12. A method for providing a predicate prediction in a multi-core processor, comprising:

providing a predicate predictor for each of a plurality of processor cores in the multi-core processor; and

generating the predicate prediction from a plurality of branch instructions using the predicate predictors.

13. The method of claim 12, further comprising executing an application program by at least one of the cores, which program comprises one of the plurality of branch instruction having predicate path information encoded thereon.

14. The method of claim 13, further comprising encoding the predicate path information on the branch instruction using a compiler.

15. The method of claim 13, wherein the program comprises a plurality of branch instructions, the method further comprising determining which processor core is to execute the branch instructions using a block address for each of the branch instructions.

16. The method of claim 12, wherein the at least one predicate predictor comprises a base predictor and a global history register.

17. The method of claim 16, wherein the base predictor is a geometric history length predictor.

18. The method of claim 16, wherein the global history register is a core-local predicate history register.

19. The method of claim 16, wherein the global history register is a global block history register.

20. A computer accessible medium having stored thereon computer executable instructions for providing a predicate prediction within a multi-core processor computing system, wherein a processing arrangement is configured to perform processing procedures when the processing arrangement executes the instructions, the processing procedures comprising:

providing a predicate predictor for each of a plurality of processor cores in the multi-core processor, each of the processor cores comprising at least one predicate predictor; and

generating the predicate prediction using the predicate predictors.

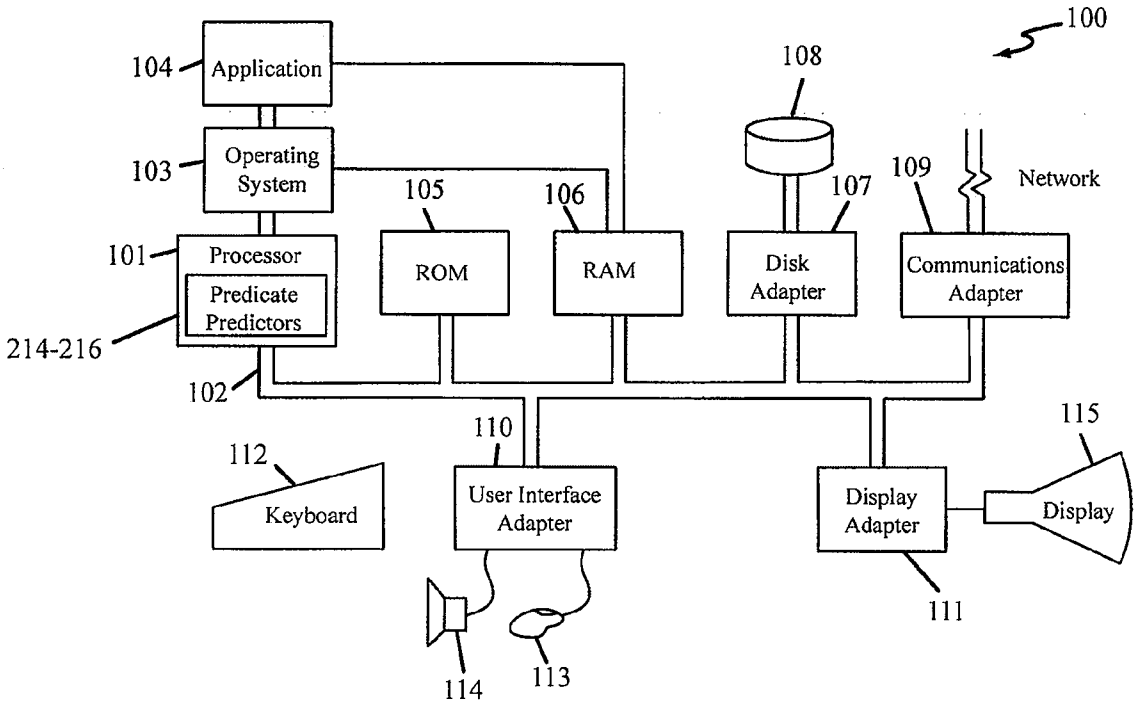
100
104
Application
103
Operating System
101
Processor
Predicate Predictors
214-216
102
105
ROM
106
RAM
108
107
Disk Adapter
109
Network
Communications Adapter
110
User Interface Adapter
112
Keyboard
114
113
115
Display Adapter
Display
111

FIG. 1

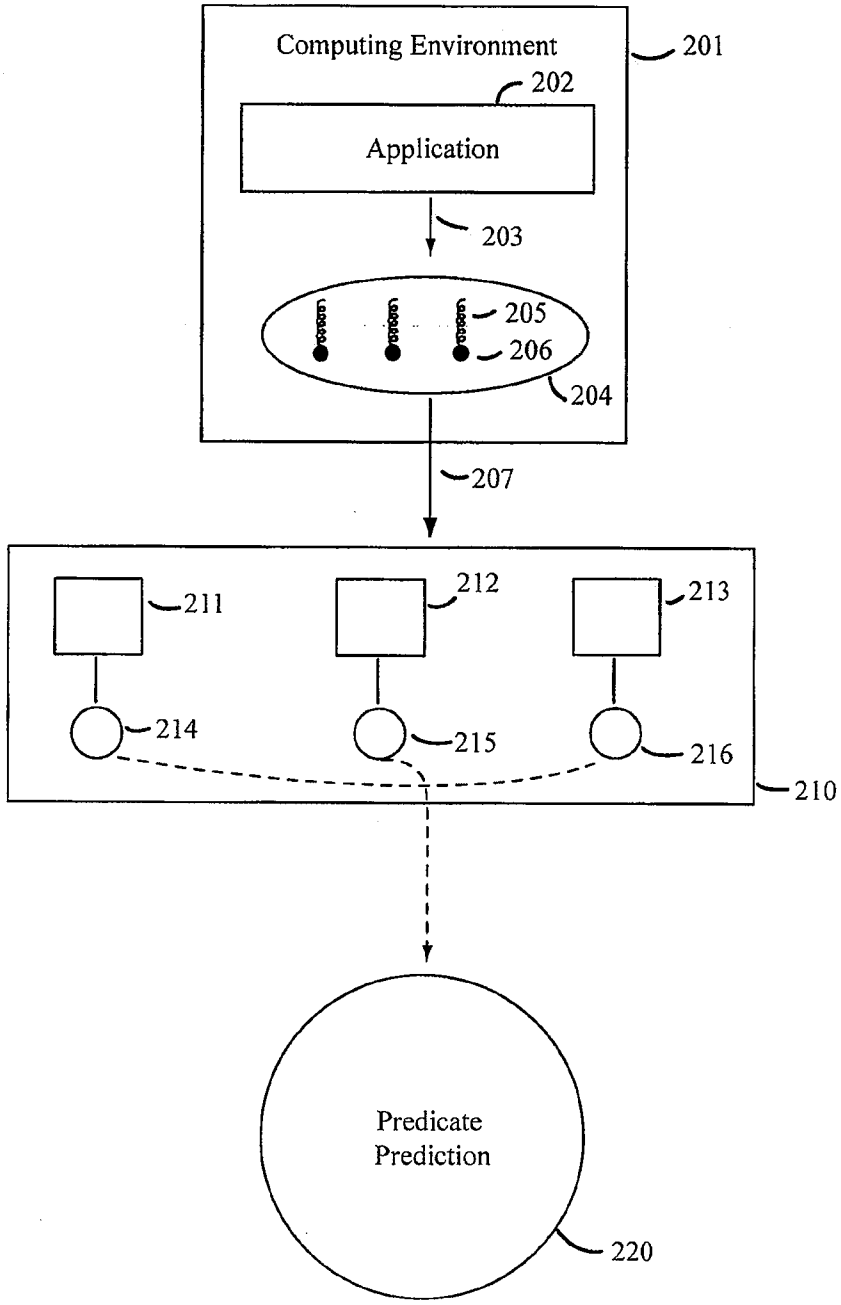
Computing Environment
201
202
Application
203
205
206
204
207
211
212
213
214
215
216
210
Predicate
Prediction
220

FIG. 2

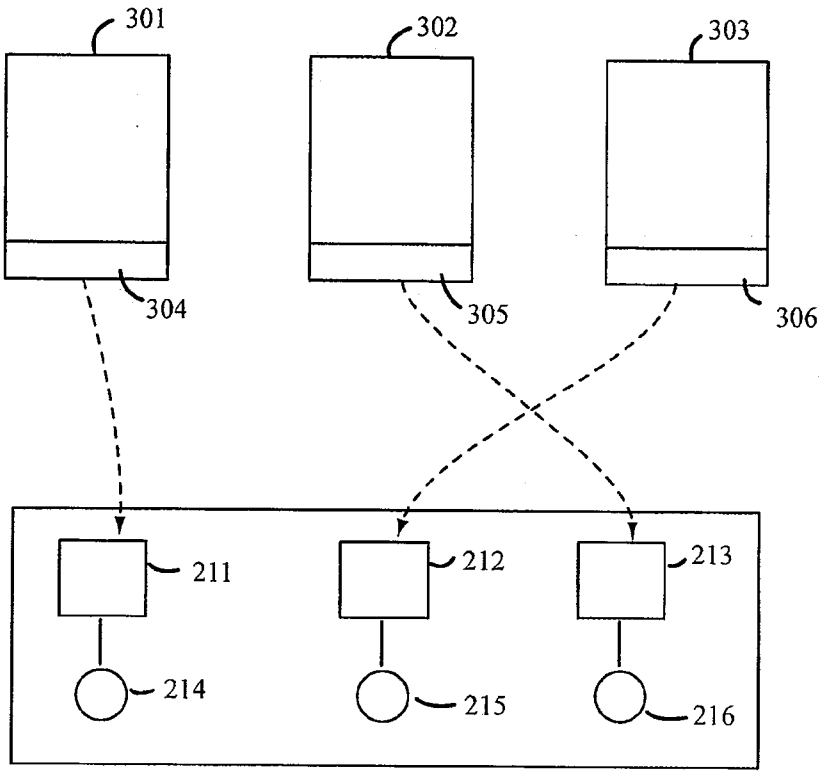

FIG. 3

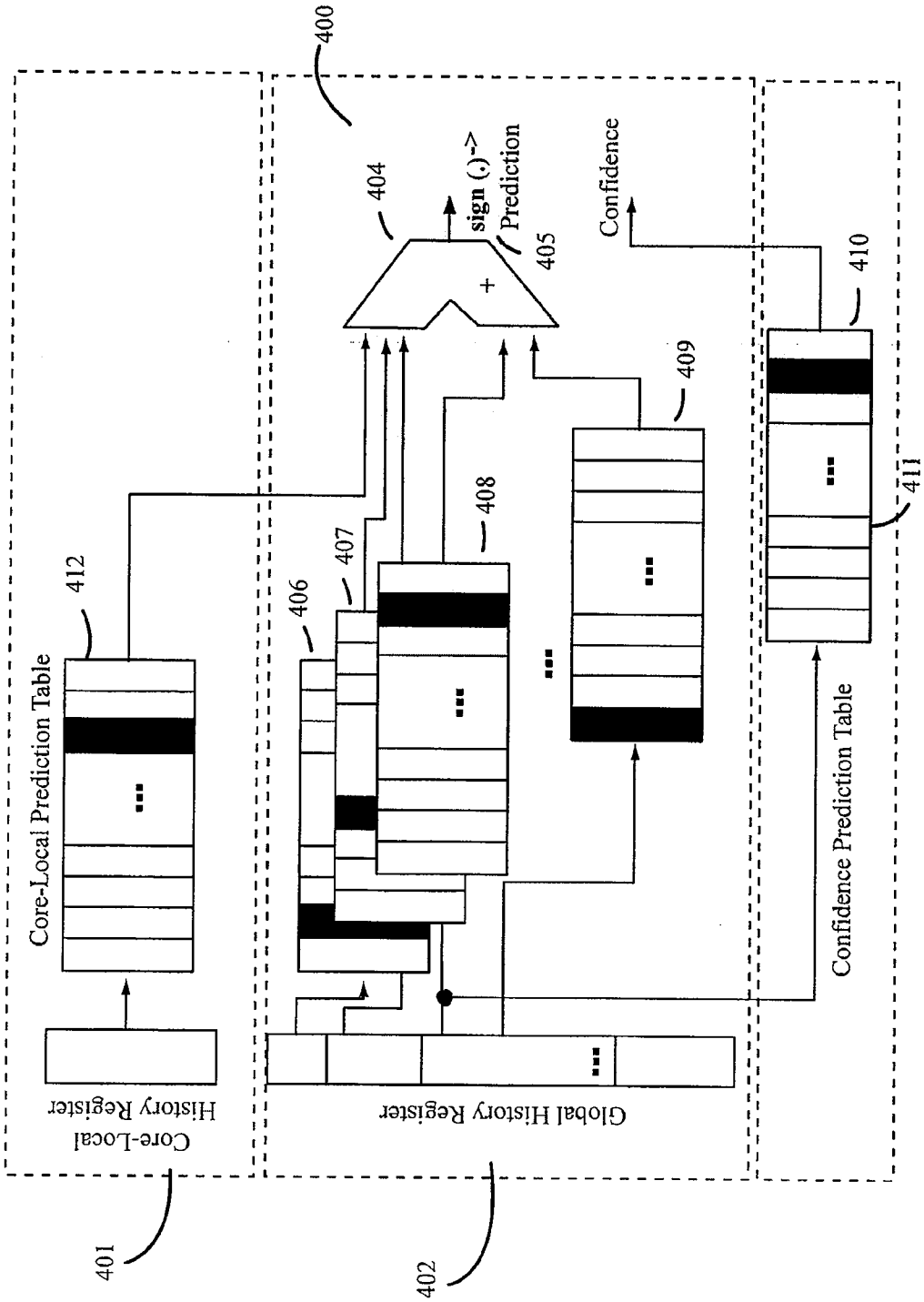
400
404
sign (.)->
Prediction
405
+
Confidence
410
409
408
407
406
412
411
Core-Local Prediction Table
Confidence Prediction Table
Core-Local
History Register
Global History Register
401
402

FIG. 4

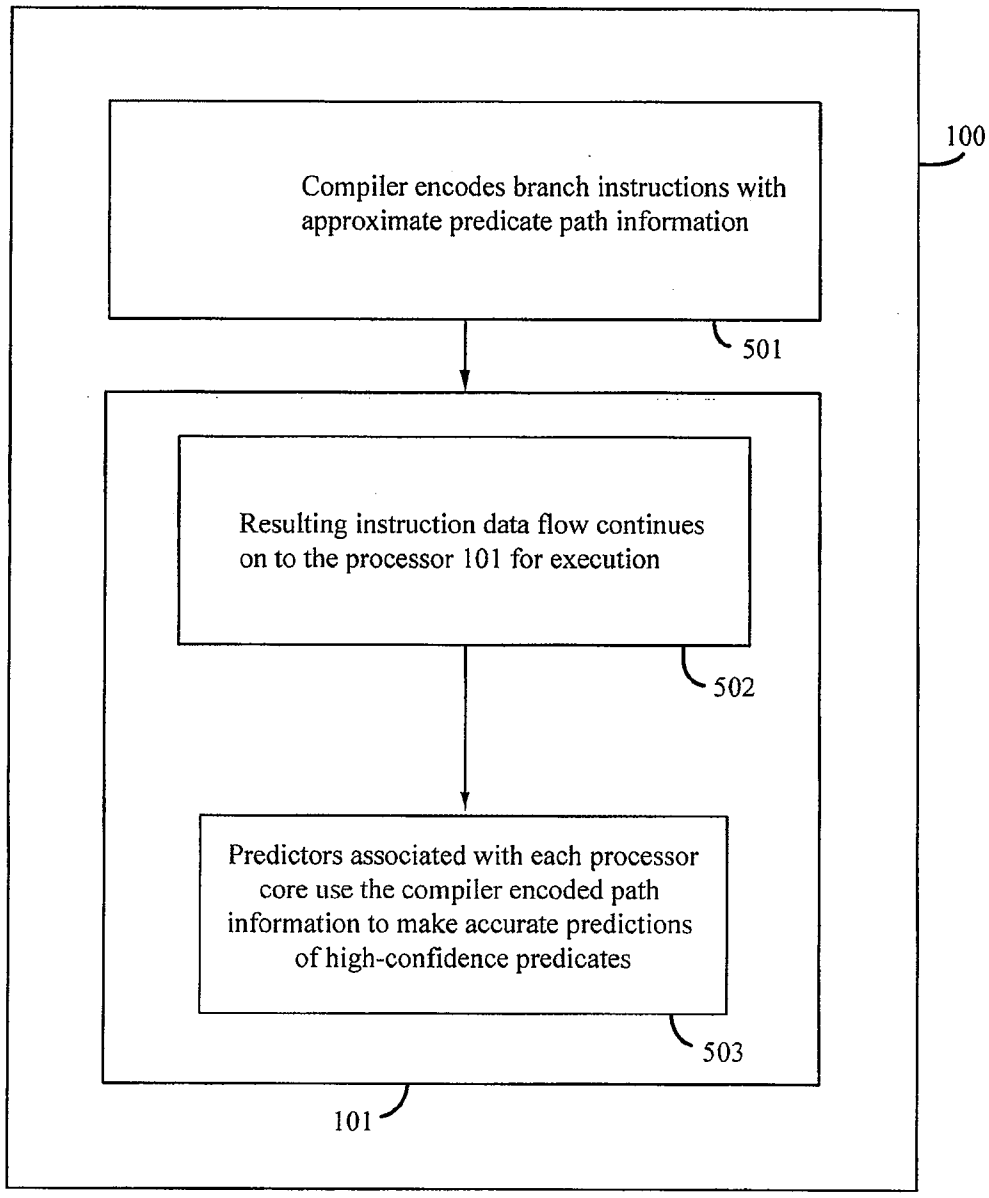
100
Compiler encodes branch instructions with approximate predicate path information
501
Resulting instruction data flow continues on to the processor 101 for execution
502
Predictors associated with each processor core use the compiler encoded path information to make accurate predictions of high-confidence predicates
503
101

FIG. 5

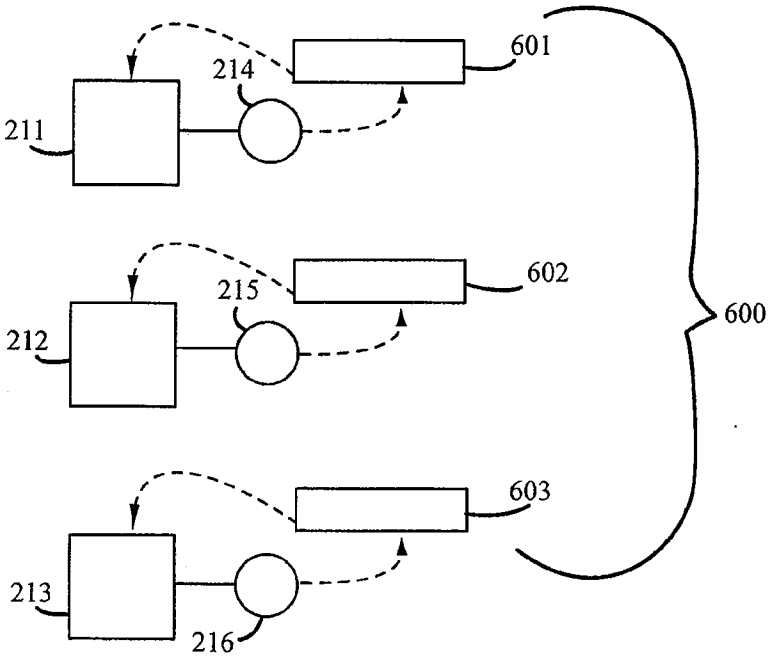
601
214
211
602
215
212
600
603
213
216

FIG. 6A

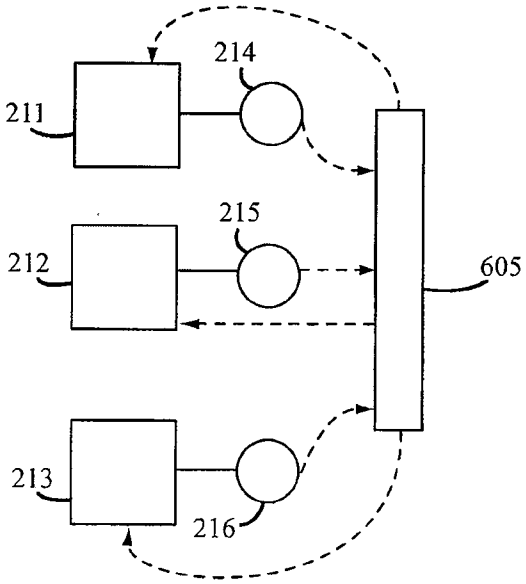
214
211
215
212
605
213
216

FIG. 6B

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US 10/38350

A. CLASSIFICATION OF SUBJECT MATTER
IPC(8) - G06F 7/38 (2010.01)
USPC - 712/233
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC(8): G06F 7/38 (2010.01)
USPC: 712/233

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
USPC: 712/239; 712/220 (keyword limited; terms below)

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
PubWest; Google Scholar; Google Patents; FreePatentsOnline. Search terms used:
architecture microarchitecture hardware-instruction, multi-core multiple-core multi-processor multiple-processors, predicate logical-expression, prediction predict forecast, branch-instruction, path predicate-path, compiler, block-address, graph...

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X -- Y	US 2009/0172371 A1 (JOAO et al.) 02 July 2009 (02.07.2009) entire document, especially Abstract; para [0026], [0027], [0030], [0034], [0042], [0055], [0067]	1,5,6,8,11,12,16,18,20 ---------- 2-4,7,9,10,13-15,17,19
Y	US 2005/0172277 A1 (CHHEDA et al.) 04 August 2005 (04.08.2005) entire document, especially Abstract; para [0029], [0066], [0067]	2-4, 9, 13-15, 19
Y	US 2009/0158017 A1 (MUTLU et al.) 18 June 2009 (18.06.2009) entire document, especially Abstract; para [0007], [0012]	7, 17
Y	US 2007/0288733 A1 (LUICK) 13 December 2007 (13.12.2007) entire document, especially Abstract; Figs.17A, 17B; para [0012], [0173] [0177], [0178]	10
A	US 7,487,340 B2 (LUICK) 03 February 2009 (03.02.2009) entire document	1 - 20
A	US 6,178,498 B1 (SHARANGPANI et al.) 23 January 2001 (23.01.2001) entire document	1 - 20

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

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
01 November 2010 (01.11.2010)	15 NOV 2010
Name and mailing address of the ISA/US Mail Stop PCT, Attn: ISA/US, Commissioner for Patents P.O. Box 1450, Alexandria, Virginia 22313-1450 Facsimile No. 571-273-3201	Authorized officer: Lee W. Young PCT Helpdesk: 571-272-4300 PCT OSP: 571-272-7774

Form PCT/ISA/210 (second sheet) (July 2009)