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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
22 February 2018 (22.02.2018)

WIPO | PCT

(10) International Publication Number
WO 2018/034879 A1

(51) International Patent Classification:
G06F 9/30 (2006.01)

(21) International Application Number:
PCT/US2017/045702

(22) International Filing Date:
07 August 2017 (07.08.2017)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
15/241,488 19 August 2016 (19.08.2016) US

(71) Applicant: **WISCONSIN ALUMNI RESEARCH FOUNDATION** [US/US]; 614 Walnut Street, Madison, WI 53726 (US).

(72) Inventors: **SANKARALINGAM, Karthikeyan**; 4809 Hillview Terrace, Madison, WI 53711 (US). **NOWATZKI, Anthony**; 4723 Sheboygan Avenue, Apt. 221, Madison, WI 53705 (US).

(74) Agent: **BAXTER, Keith, M.**; Boyle Fredrickson, S.C., 840 North Plankinton Avenue, Milwaukee, WI 53203 (US).

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

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

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

(54) Title: COMPUTER ARCHITECTURE WITH SYNERGISTIC HETEROGENEOUS PROCESSORS

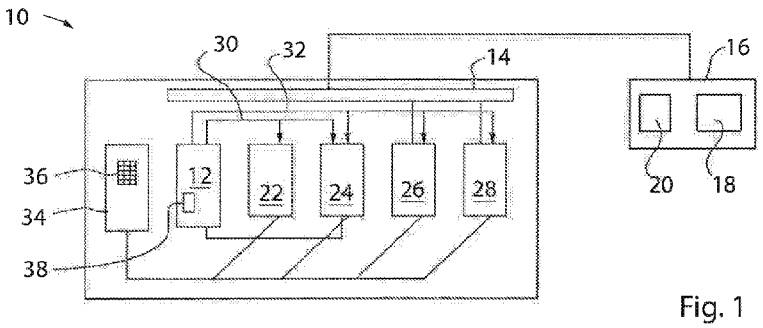


Fig. 1

(57) Abstract: A computer architecture employs multiple special-purpose processors having different affinities for program to execute substantial portions of general-purpose programs to provide improved performance with respect to a general-purpose processor executing the general-purpose program alone. The computer architecture comprises: a memory cache; a set of heterogeneous processors sharing the memory cache and providing relatively different performances with respect to different portions of a program; and switch unit alternately switching between heterogeneous processors for different portions of the program based on an assessed comparative advantage of the heterogeneous processors in executing the different portion.

WO 2018/034879 A1

Computer Architecture with Synergistic Heterogeneous Processors

STATEMENT REGARDING FEDERALLY SPONSORED RESEARCH OR DEVELOPMENT

CROSS REFERENCE TO RELATED APPLICATION

This application claims the benefit of U.S. non-provisional application 15/241,488 filed August 19, 2016, and hereby incorporated by reference.

BACKGROUND OF THE INVENTION

The present invention relates to a computer architecture having multiple, highly specialized processors and in particular to an architecture capable of switching operating modes based on anticipated program behavior.

General-purpose processors, such as out-of-order processors providing speculative execution, are designed to efficiently execute a wide range of computer programs. It is known, however, that improved performance can be obtained in particular program domains or with particular program behaviors using specialized computer processors. For example, "single instruction multiple data" or SIMD processors have been developed for programs that require rapid execution of a common instruction on multiple data elements. Such SIMD processors are beneficial in the domain of graphics processing, for example, where in an adjustment of the brightness of an image, multiple data elements of pixels in the image are all scaled by a common factor. Similarly, specialized neural network processors can provide improved performance in machine learning or pattern recognition applications.

The improved performance, measured in energy consumption and/or speed, provided by these specialized processors normally disappears as these processors are required to execute a general-purpose program having a variety of behaviors and/or spanning different program domains. For this reason, specialized processors are normally paired and work under the control of a general-purpose processor. In this situation, the general-purpose processor may execute the program until the specialized processor is invoked through special commands or instructions embedded in the executed program, for example, instructions accessing an application programmer interface (API) associated with the specialized processor.

SUMMARY OF THE INVENTION

The present inventors have recognized that a properly selected group of specialized processors can operate synergistically to provide improved performance (speed and/or energy efficiency) over a wide range of general-purpose programs. Importantly, the appropriate set of specialized processors can provide complementary "affinities" that cover a significant percentage of the instructions of a general-purpose program. As a result, the general-purpose processor can be simplified, providing a net savings in integrated circuit area or complexity.

In one embodiment, the complexity of partitioning a program among multiple specialized processors having different affinities is handled through a run time profiling process permitting program allocation to be largely transparent to the user/programmer.

In a simulation across a wide range of benchmarks, a 2-wide OOO processor with three special-purpose processors selected per the present invention was able to match the performance of a conventional 6-wide OOO processor while using 40 percent lower chip area and being 2.6 times more energy efficient.

Specifically, in one embodiment, the invention provides a computer architecture employing a set of heterogeneous processors providing relatively different performances with respect to portions of a program having different degrees of data parallelism. The heterogeneous processors are controlled by a switch unit alternately switching between the heterogeneous processors for different portions of the program based on the assessed degree of data parallelism. The heterogeneous processors include: (a) a general-purpose processor providing speculative execution; (b) at least one processor providing higher performance execution than the general-purpose out-of-order processor for first given program portions exhibiting a predetermined level of data parallelism; and (c) at least one processor providing higher performance execution than the general-purpose out-of-order processor for second given program portions exhibiting less data parallelism than the predetermined level of data parallelism.

In some embodiments, the program may have different degrees of control flow complexity and the heterogeneous processors may further include: (d) at least one processor providing higher performance execution than the general-purpose out-of-order processor for third given program portions having lower control flow complexity than the first given program portions.

In some embodiments, the program may have different degrees of control bias and the heterogeneous processors may further include: (e) at least one processor providing higher performance execution than the general-purpose out-of-order processor for fourth given program portions having more biased control flow than the second given program portions.

It is thus a feature of at least one embodiment of the invention to identify specialized processors that can be synergistically combined to improve processor performance under the parameters of execution speed, energy consumption, and chip area. The present inventors have determined that the proper combination of multiple specialized processors significantly shifts the performance frontier to higher performance without the need for greater chip area.

In some embodiments, the invention may use a combination of program analysis and profiling data from a previous execution of the program to predict which of the heterogeneous processors will have the most performance. Based on these predictions, the decisions of which processor to execute on may be statically determined and encoded into the program binary.

In one embodiment, the invention may provide a real-time program profiler monitoring execution of different portions of the program on different of the heterogeneous processors to assess data parallelism during an earlier execution of the different portions. The switch unit alternately switches between the heterogeneous processors for a later execution of the different portions of the program based on the assessed degree of data parallelism by the real-time program profiler during the earlier execution of the different portions. Similarly, the profiler may monitor control flow complexity and control flow criticality.

It is thus a feature of at least one embodiment of the invention to manage the complexity of a processor having many similar specialized processors with overlapping capabilities while improving processor performance. Real-time profiling allows program portions to be allocated in a manner that is largely transparent to the user/programmer, rendering the architectural complexity tractable.

The real-time program profiler may monitor data dependencies in the execution of a given portion of the program to assess a degree of data parallelism with larger numbers of dependencies associated with lesser data parallelism. In addition the real-time program profiler may monitor a number of branches in the execution of a given portion of the program to assess the amount of control flow with larger numbers of branches associated with more control flow.

It is thus a feature of at least one embodiment of the invention to provide a simple run

time assessment of the amount of data parallelism and control flow complexity.

The architecture may include a general-purpose out-of-order processor providing speculative execution, and only a single one of the heterogeneous processors other than the general-purpose out-of-order processor may execute at a time and the remaining processors other than the general-purpose processor maybe placed in a reduced energy consumption state.

It is thus a feature of at least one embodiment of the invention to largely eliminate the energy penalty of using multiple processors.

The heterogeneous processors may share a same memory and cache from which instructions and data are obtained.

It is thus a feature of at least one embodiment of the invention to provide a limited need for state transfer between processors to allow exploitation of relatively short program portions.

The heterogeneous processors may include a general-purpose processor providing configuration data to other heterogeneous processors before switching to another heterogeneous processor for execution of a program portion.

It is thus a feature of at least one embodiment of the invention to permit highly simplified special-purpose processors that can be assisted in certain functions by a general-purpose processor.

These particular objects and advantages may apply to only some embodiments falling within the claims and thus do not define the scope of the invention.

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 is a block diagram of a hybrid computer constructed according to the present invention providing a general-purpose processor and four special-purpose processors controlled by a switching unit;

Fig. 2 is a decision tree showing the allocation of program portions to different special-purpose processors;

Fig. 3 is a flowchart of the process of switching between the different special-purpose processors during runtime;

Fig. 4 is a logical representation of a program showing a mapping of data to program counter values to control the switching process of Fig. 3;

Fig. 5 is a detailed flowchart of a profiling step determining the allocation of special-purpose processors to program portions of Fig. 3; and

Fig. 6 is a flowchart showing three methods of determining the allocation of program portions to the special purpose processors.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

Referring now to Fig. 1, a heterogeneous processor computer 10 may include a general-purpose processor 12, for example, providing for out-of-order (OOO), speculative execution. The general-purpose processor 12 may employ a comprehensive instruction set suitable for execution of general-purpose program types having a wide range of behaviors and coming from a wide range of program domains. The general-purpose processor 12 may nevertheless be limited in performance, for example, to a two-wide OOO processor in contrast to a 6-wide OOO, where two-wide refers to a superscalar processor that can issue two instructions to the pipeline at once.

The general-purpose processor 12 communicates with a cache structure 14 (and memory management unit) allowing access to a main memory 16. The main memory 16 may employ a variety of different technologies including volatile and nonvolatile storage and may hold programs and data including an operating system 18, a compiler for the present heterogeneous processor computer 10, and a general-purpose program 20 for execution.

In addition to the general-purpose processor, the heterogeneous processor computer 10 employs four special-purpose processors including: a short vector, single instruction multiple data processor (SIMD) 22; a data parallel, course-grained reconfigurable architecture processor (DP-CGRA) 24; a non-speculative data flow processor (NS-DF) 26; and a trace-speculative processor (Trace-P) 28. These processors are generally highly specialized and may not be able to execute a full instruction set architecture of the general-purpose processor 12.

The SIMD 22 may be any of a variety of well-known architectures allowing a single instruction to be applied simultaneously to many data elements, the latter provided in the form of a vector from the general-purpose processor 12 over vector line 30 between the general-purpose processor 12 and the SIMD 22. After transfer of the vector, the general-purpose processor 12 provides the instructions to the SIMD 22 that should be applied to the vector and the SIMD 22 performs the necessary calculations and returns the modified vector. The SIMD 22 is suitable only for program portions having a limited amount of control (branches controlled by calculation results), for example, finding application in data parallel loops with little or no control.

The DP-CGRA 24 also provides parallel execution on data but tolerates additional control. An example of a DP-CGRA 24 is the Dyser architecture described by Venkatraman Govindaraju, Chen-Han Ho, Tony Nowatzki, Jatin Chhugani, Nadathur Satish, Karthikeyan Sankaralingam, and Changkyu Kim in "Dyser: Unifying functionality and parallelism specialization for energy efficient computing", IEEE Micro, 2012 and the Morphosys architecture described by H. Singh, Ming-Hau Lee, Guangming Lu, F. J. Kurdahi, N. Bagherzadeh, and E. M. Chaves Filho in: "Morphosys: an integrated reconfigurable system for data-parallel and computation-intensive applications", IEEE Transactions on Computers, 2000, both hereby incorporated by reference. The DP-CGRA 24 is suitable for parallel loops with separable computation and memory access. The DP-CGRA 24 may receive a vector directly from the general-purpose processor 12 over vector line 30 and may receive configuration (implementing instructions) and initialization data from the general-purpose processor 12 over control line 32.

The NS-DF 26 is suitable for program portions that are not data-parallel but still have high potential instruction level parallelism (ILP) where control flow does not lie on a critical path (the long chain of computational instructions that determines the execution time). In this regard the NS-DF 26 may not provide speculative execution to simplify its construction and boost performance for this type of program portion. An example of the NS-DF 26 is the SEED architecture described by Tony Nowatzki, Vinay Gangadhar, and Karthikeyan Sankaralingam in: "Exploring the potential of heterogeneous Von Neumann/Dataflow execution models, in ISCA, 2015, and the Wavescalar architecture described by Steven Swanson, Ken Michelson, Andrew Schwerin, and Mark Oskin in "Wavescalar" in MICRO, pages 291 et seq., 2003, both incorporated by reference. The NS-DF 26 will generally be configured by the general-purpose processor 12 through the configuration lines 32 and then will receive a program counter value to communicate directly through the cache structure 14 to obtain data and program instructions starting at that program counter value when activated.

The Trace-P 28 is suitable for programs that require some speculation where the speculation is highly biased along one "hot path", for example, through a loop. An example Trace-P 28 architecture is described in BERET architecture described by Shantanu Gupta, Shuguang Feng, Amin Ansari, Scott Mahlke, and David August in: "Bundled execution of recurring traces for energy-efficient general purpose processing" in MICRO, 2011, and the CCA

architecture described by Nathan Clark, Manjunath Kudlur, Hyunchul Park, Scott Mahlke, and Krisztian Flautner in: "Application-specific processing on a general-purpose core via transparent instruction set customization" in MICRO, 2004, both incorporated by reference. The Trace-P 28 processor will generally be configured by the general-purpose processor 12 and will receive a program counter value over control line 32 and may communicate directly through the cache structure 14 to obtain data and program instructions when activated.

Generally, it will be appreciated that each of the specialized processors SIMD 22, DP-CGRA 24, NS-DF 26, and Trace-P 28 share the same virtual memory system and cache hierarchy (either directly or through the general-purpose processor 12) allowing fast switching between processors without the need to migrate state information. This switching between the general-purpose processor 12 and the specialized processors SIMD 22, DP-CGRA 24, NS-DF 26, and Trace-P 28 is performed by a switch unit 34 communicating with each of the specialized processors SIMD 22, DP-CGRA 24, NS-DF 26, and Trace-P 28 to switch them between an active state in which they may execute instructions and a dormant state providing low power consumption. During the dormant state, the specialized processors SIMD 22, DP-CGRA 24, NS-DF 26, and Trace-P 28 are allowed to lose their architectural state and thus may be essentially shut down completely for zero power consumption. When any of the specialized processors SIMD 22, DP-CGRA 24, NS-DF 26, and Trace-P 28 are operating, the general-purpose processor 12 may be moved to a low-power state but retains architectural state information so that it can be reactivated without loss of state information.

Generally, the switch unit 34 and the general-purpose processor 12 inter-communicate so that the general-purpose processor 12 may properly initialize each of the specialized processors SIMD 22, DP-CGRA 24, NS-DF 26, and Trace-P 28 before it is activated and to receive it and stored data from the SIMD 22, DP-CGRA 24 before they are deactivated.

Referring now to Fig. 6, three basic strategies can be used for allocating portions of the program 20 to different ones of the specialized processors SIMD 22, DP-CGRA 24, NS-DF 26, and Trace-P 28 e. All approaches have the same basic steps. The first step is finding program "phases" being regions of the program 20, typically thousands to millions of execution-time instructions, where allocation to a particular one of the specialized processors is indicated. In a second step, the program phases are characterized for example by profiling. Next, in a third step, a combination of program intermediate representation (IR) analysis and/or profiling information

is used to predict which specialized processor SIMD 22, DP-CGRA 24, NS-DF 26, and Trace-P 28 will give the highest performance. With that decision made, the fourth step is to compile or interpret the program IR into the architectural interface of the chosen specialized processor (or determine the necessary configuration of the chosen specialized processor). The fifth step is to inject the invocation of the processor into the program execution of the program 20, so that it can switch on a future activation.

For a quasi-static approach, the phase assignment decisions are made before the execution of the relevant program begins. Program phases may be identified, for example as indicated by process block 74, by profiling the program and tracking the number of instructions executed in loops and functions. Next profiling, per process block 76, may be used to collect information collected from one or several previous executions of the identified program phases, which may use test program inputs and program instrumentation to discover these properties of the program phases. For example, the profiling information collected may be length of a critical path through the instructions according to the program IR. One prediction technique, then, involves the estimation of execution time of the program phases on a particular specialized processor by formulating this execution time as the length of the critical path through the instructions according to the Program IR, in light of expected instruction execution times and dependences, and microarchitectural constraints unique to the given specialized processor. The specialized processor that has the least execution time estimation (or alternatively energy) would be selected per process block 78. Alternatively, a set of rules and thresholds can be developed based directly on the properties through empirical testing and validation, and these rules may be tuned through statistical techniques like regression. For example, the number of control instructions may measured in the profiling may be applied directly to an empirically derived threshold to decide whether or not execution on the program phase using a SIMD processor would be profitable or not. After a specialized processor is selected, the region of the phase in question would be compiled, per process block 80, for that specialized processor using well known code generation techniques. Finally, special jump instructions would be inserted into the program 20 at the region entry point which would indicate to the switch unit to transfer execution to the selected processor, per process block 82.

A second approach is to employ a virtual machine (e.g., Java), which typically interpret instructions in the virtual machine's instruction set until frequently executed regions are found,

then compile them into native code. The virtual machine approach is quite similar to the static approach, except that the compilation decisions are based on profiling during the current execution of the program. Thus at process block 84, loops and functions may be monitored to identify program phases using the virtual machine software. The same process may be used at process block 86 to collect profile data used at process block 88 estimate execution times and select an appropriate specialized processor. Compilation of the program phases for the particular processor may occur at process block 90 and the necessary transition instructions inserted per process block 92.

A final approach is to use hardware-based dynamic translation. Here, hardware mechanisms are used for detecting program phases. In one example, represented by process block 94, retiring instructions linked to particular program counter values can be used to detect program loops and to invoke an analysis of the program portions holding these loops once a certain threshold loop count has been reached. This technique is described, for example in Matthew A. Watkins, Tony Nowatzki, Anthony Carno, "Software transparent dynamic binary translation for coarse-grain reconfigurable architectures", HPCA, 2016, 2016 IEEE International Symposium on High Performance Computer Architecture (HPCA) pp. 138-150 . Similar hardware monitoring may be employed to collect data and to determine the best specialized processor to use for each program phase per process block 96 and process block 98. Necessary configuration or translation processes may be incorporated into the architecture of the computer 10 (so that each specialized processor may work with a common instruction set) as indicated by process block 100. For example, hardware may monitor execution of loops or function calls in the pipeline to detect frequently executed regions and may operate (optionally using an additional microprocessor) to compile code to different heterogeneous processors as required. Hardware in the front-end of the general processor pipeline may detect starts of regions and invoke the accelerator.

Finally, as indicated by process block 102, a hardware system may be used to invoke switching between the different specialized processors and the general-purpose processor. This approach will be now described in greater detail below

In the dynamic translation embodiment of the invention, the general-purpose processor 12 may manage an activation table 36 controlling the switch unit 34. Referring now to Fig. 4, the activation table 36 may logically provide a table with multiple rows representing portions 40

of the program 20 executed by the heterogeneous processor computer 10, these different portions 40 such as will be assigned to different of the specialized processors SIMD 22, DP-CGRA 24, NS-DF 26, and Trace-P 28 . A first column 39 of each row, for example, may hold a program counter value of program 20 demarcating a beginning of a portion 40. The portions 40, for example, may be natural divisions of the program 20 representing loops or function calls within the program 20. In the case of nested loops or nested function calls, the program portions 40 may be the lowest most level of loop and function calls, although the invention contemplates that portions 40 may be combined (for example, all of the loops within a loop) if they tend to have the same affinity to a particular specialized processor.

For each row associated with the program portion 40, a second column 42 will designate a particular core of the specialized processors SIMD 22, DP-CGRA 24, NS-DF 26, and Trace-P 28 or the general-purpose processor 12 which will be activated to execute the instructions of that program portion 40. For this purpose the switch unit 34 may monitor the current program counter of the general-purpose processor 12 or the various specialized cores that have a program counter and switch among the cores appropriately using the guidance of column 42, for example, through the use of an exception type mechanism.

In one embodiment, the particular processor used for a given program portion 40 selected from the specialized processors SIMD 22, DP-CGRA 24, NS-DF 26, and Trace-P 28 and the general-purpose processor 12 may be determined by a real-time profiling process. This process may be implemented by special hardware or performed by the general-purpose processor 12 according to a profiling program 38 executed, for example, as one thread by the general-purpose processor 12. In this profiling process, statistics are collected with respect to the execution of each program portion including current statistics placed in column 44 related to the current processor executing the portion 40 and historical statistics in column 47 that may be optionally used in an AB comparison type process as will be discussed below.

Referring now to Fig. 2, generally the various specialized processors SIMD 22, DP-CGRA 24, NS-DF 26, and Trace-P 28 are selected to provide for different comparative advantages or affinities 46 in the execution of program portions 40 of particular domains or behavior. Ideally, together, the processors provide affinities that span a substantial portion of a variety of general-purpose programs 20 and which together with the general-purpose processor 12 span the entire domain of most general-purpose programs. A distribution of affinities 46 may

be obtained by viewing program portions 40 according to a taxonomy 51 that generally considers data parallelism, control complexity, and characteristics such as control consistency and control criticality.

First, program portions 40 may be separated according to program portion as largely data parallel as indicated by node 50 or not data parallel indicated by node 52. One determination of data parallelism relates to how much data dependency exists between data elements within the program portion 40. Less data dependency between data elements is associated with greater data parallelism. Generally data parallelism refers to the ability to independently operate on the data without data dependency problems. Data dependency, for example, occurs when data elements modified by earlier instructions affect the data read at later instructions (read-after-write dependency) as well as with other combinations of reading and writing of data that could change the results of these operations if order is not preserved. A variety of techniques may be used to assess data dependency including, for example, monitoring a pipeline in the general processor executing the program portion to evaluate dependencies at a time of memory disambiguation. This dynamic or profiling information can be incorporated into analysis of the program IR to determine the total amount and type of data dependency (e.g. loop iteration dependence analysis). If the original program IR is available (in a static compilation approach for example) variable scoping information or operation on distinct data objects may also be used to determine the degree of data parallelism.

For program portions 40 that exhibit a high degree of data parallelism, the program is evaluated with respect to the degree of control complexity, for example, by counting branch instructions. Program portions with high data parallelism and low control complexity are marked in activation table 36 for execution by the SIMD 22. Conversely program portions with high data parallelism and moderate levels of control complexity may be executed by the DP-CGRA 24. Program portions 40 with high data parallelism and high levels of control complexity remain with the general-purpose processor 12.

For program portions 40 that do not provide high levels of data parallelism, as indicated by node 52, in cases where the control flow is relatively consistent (e.g., biased along a single hot path) the program portion is marked for execution by the Trace-P 28. A determination of consistent control flow may be ascertained by monitoring the per-instruction branch directions in a given region over time.

When the control of the program portion 40 is not highly biased but noncritical, meaning that the control flow path is relatively isolated from the critical data flow path, the program portion may be executed by NS-DF 26. The assessment of noncritical control flow maybe by performed with dataflow analysis on the program IR to determine the critical path of instructions (through for example the inner loop of the region) with estimates of instruction latencies and dependence information from profiling. If the control is largely off the estimated critical instruction path, than that would constitute a non-critical control region. Alternatively, a purely hardware approach can be used to find the critical path, as in "Focusing processor policies via Critical Path Prediction" described by Brian Fields, Shai Rubin, and Rastislav Bodik. Again cases where there is little parallel data but highly variable control criticality will be executed by the general-purpose processor 12.

Each of these determinations may be recorded as statistics in column 44 of activation table 36 and then applied to static or dynamically established thresholds to core identification per column 42 of activation table 36 and used to guide the switching between the various specialized processors SIMD 22, DP-CGRA 24, NS-DF 26, and Trace-P 28 and the general-purpose processor 12.

Alternatively or in addition, the switch unit 34 may experiment with the execution of different program portions 40 on different of the specialized processors SIMD 22, DP-CGRA 24, NS-DF 26, and Trace-P 28 versus general-purpose processor 12 to dynamically evaluate which core provides the best performance which may be used to set the core identification of column 42 in activation table 36. Historical statistics obtained from execution on different processors SIMD 22, DP-CGRA 24, NS-DF 26, Trace-P 28 and the general-purpose processor 12 may be stored in column 47 of activation table 36 and used in this comparison process to evaluate the relative advantages of one core versus another core.

Referring now to Fig. 3, use of the heterogeneous processor computer 10 of the present invention may optionally begin with a static analysis of the program 20 indicated by process block 54, for example, attempting to evaluate degrees of data parallelism, control criticality, control complexity and control variability discussed above with respect to Fig. 2 to the extent determinable in static analysis. This analysis may be performed during compilation of the program and may optionally make use of tags embedded in the program identifying preferred processors for use by particular program portions 40. Alternatively, initial selections may be

made by the programmer/user or a random initialization may be used. This statically derived information may be used to initialize the activation table 36 as indicated by process block 56 with respect to column 42.

Program execution is then begun and transitions between program portions 40 monitored by switch unit 34 as indicated by process block 58 by monitoring the program counter logically shared by the specialized processors SIMD 22, DP-CGRA 24, NS-DF 26, and Trace-P 28 typically managed by the general-purpose processor 12. At each transition, as indicated by process block 60, the activation table 36 is interrogated to see if a processor different from the currently executing processor should be used. If so, that processor is then activated.

At process block 62 profiling may be conducted (shown as a serial operation but in fact implemented as a background task in parallel), for example, by the switch unit 34 in conjunction with the general-purpose processor 12 executing a profiling program 38.

Referring momentarily to Fig. 4, as discussed briefly above, profiling of process block 62 may, for example, count branches as indicated by process block 64 (to determine control complexity), assess data dependency as indicated by process block 66 (to determine degree of data parallelism), and measure the criticality of control per process block 68 by monitoring intersections of data flow and control flow.

As noted above statistics derived from this profiling may be stored in column 44 of the activation table 36 and analyzed according to a predetermined mapping indicated by process block 70 using the taxonomy 51 of Fig. 2. This mapping process, for example, may establish thresholds applied against the statistics to select edges of the taxonomy of Fig. 2 to identify a particular processor. At process block 72 the results of this analysis are used to update column 42 of the activation table 36.

When an A/B comparison is to be used, statistics for the current processor from column 44 are moved to column 47 and new profiling of a new processor is conducted for a period of time. After completion of that period of time, the statistics of columns 44 and 47 are compared to determine a preferred processor to be identified in column 42.

The concept of improved performance as discussed herein with respect to a computer architecture refers generally to a position of a frontier describing execution speed as a function of energy consumption as a function of chip area in comparison to a similar frontier of a reference architecture.

Certain terminology is used herein for purposes of reference only, and thus is not intended to be limiting. For example, terms such as "upper", "lower", "above", and "below" refer to directions in the drawings to which reference is made. Terms such as "front", "back", "rear", "bottom" and "side", describe the orientation of portions of the component within a consistent but arbitrary frame of reference which is made clear by reference to the text and the associated drawings describing the component under discussion. Such terminology may include the words specifically mentioned above, derivatives thereof, and words of similar import. Similarly, the terms "first", "second" and other such numerical terms referring to structures do not imply a sequence or order unless clearly indicated by the context.

When introducing elements or features of the present disclosure and the exemplary embodiments, the articles "a", "an", "the" and "said" are intended to mean that there are one or more of such elements or features. The terms "comprising", "including" and "having" are intended to be inclusive and mean that there may be additional elements or features other than those specifically noted. It is further to be understood that the method steps, processes, and operations described herein are not to be construed as necessarily requiring their performance in the particular order discussed or illustrated, unless specifically identified as an order of performance. It is also to be understood that additional or alternative steps may be employed.

References to "a microprocessor" and "a processor" or "the microprocessor" and "the processor," can be understood to include one or more microprocessors that can communicate in a stand-alone and/or a distributed environment(s), and can thus be configured to communicate via wired or wireless communications with other processors, where such one or more processor can be configured to operate on one or more processor-controlled devices that can be similar or different devices. Furthermore, references to memory, unless otherwise specified, can include one or more processor-readable and accessible memory elements and/or components that can be internal to the processor-controlled device, external to the processor-controlled device, and can be accessed via a wired or wireless network.

It is specifically intended that the present invention not be limited to the embodiments and illustrations contained herein and the claims should be understood to include modified forms of those embodiments including portions of the embodiments and combinations of elements of different embodiments as come within the scope of the following claims. All of the publications described herein, including patents and non-patent publications, are hereby

incorporated herein by reference in their entireties.

CLAIMS

What we claim is:

1. A computer architecture for executing a program comprising:

a memory cache;

a set of heterogeneous processors sharing the memory cache and providing relatively different performances with respect to different portions of the program; and

a switch unit alternately switching between the heterogeneous processors for different portions of the program based on an assessed comparative advantage of the heterogeneous processors in executing the different portions;

wherein the heterogeneous processors include:

(a) a general-purpose processor providing speculative execution and capable of executing the program entirely;

(b) a first special purpose processor providing higher performance execution than the general-purpose processor for first given program portions; and

(c) a second special purpose processor providing higher performance execution than the general-purpose processor for second given program portions.

2. The computer architecture of claim 1 wherein the first special purpose processor provides higher performance execution than the general-purpose processor for a first given program portion having a predetermined level of control criticality and the second purpose processor provides higher performance execution than the general-purpose processor for a second given program portion exhibiting more control criticality than the first given program portion

3. The computer architecture of claim 2 wherein the second purpose processor provides higher performance execution than the general-purpose processor for the second given program providing a predetermined level of consistency of control flow and wherein the first special purpose processor provides higher performance execution than the general-purpose processor for the first given program portions exhibiting less control consistency than the second program portion.

4. The computer architecture of claim 3 wherein the second special purpose processor provides a level of speculation less than a level of speculation provided by the general-purpose processor.

5. The computer architecture of claim 1 wherein the set of heterogeneous processors includes a specialized processor not providing speculative execution.

6. The computer architecture of claim 1 further including a third special purpose processor providing higher performance execution than the general-purpose processor for a third program portion having a predetermined level of data parallelism and wherein the first and second special purpose processors provide higher performance execution than the general-purpose processor for the first and second given programs exhibiting less data parallelism than the third program portion

7. The computer architecture of claim 6 further including a fourth special purpose processor providing higher performance execution than the general-purpose processor for a fourth program portion having lower control complexity than the third program portion wherein the fourth special purpose processor provides higher performance execution than the general-purpose processor for the fourth program portion and the third special purpose processor provides higher performance execution than the general-purpose processor for the third program portion having greater control complexity than the fourth program portion.

8. The computer architecture of claim 7 wherein the fourth special purpose processor is a processor executing a single instruction in parallel on multiple data elements.

9. The computer architecture of claim 8 wherein the third and fourth special purpose processors are processors accepting a single instruction for execution in parallel on multiple data elements and providing comparative relative advantages on different lengths of data vectors.

10. The computer architecture of claim 1 wherein the general-purpose processor provides configuration data to given other heterogeneous processors before switching to the given other heterogeneous processors for execution of a program portion.

11. The computer architecture of claim 1 wherein the switch unit is controlled by one of the heterogeneous processors.

12. The computer architecture of claim 1 wherein only a single one of the heterogeneous processors other than the general-purpose processor executes at a time and remaining processors other than the general-purpose processor are placed in a reduced energy consumption state.

13. The computer architecture of claim 1 wherein the general-purpose processor is an out-of-order processor.

14. The computer architecture of claim 1 further including:

a real-time program profiler monitoring execution of different portions of the program on different of the processors to assess comparative advantages of the different processors in executing the different portions during an earlier execution of the different portions; and

wherein the switch unit alternately switches between the processors for a later execution of the different portions of the program based on the assessed comparative advantage as indicated by the real-time program profiler during the earlier execution of the different portions.

15. The computer architecture of claim 14 wherein the real-time program profiler monitors execution of a given portion of the program to assess a degree of control criticality.

16. The computer architecture of claim 14 wherein the real-time program profiler monitors data dependencies in the execution of a given portion of the program to assess a degree of data parallelism with larger numbers of dependencies associated with lesser data parallelism.

17. The computer architecture of claim 14 wherein the real-time program profiler

monitors a number of branches in the execution of a given portion of the program to assess control flow complexity with larger numbers of branches associated with more control flow complexity.

18. The computer architecture of claim 14 wherein the real-time program profiler is executed at least in part on one of the heterogeneous processors.

19. A method of executing a program on a computer architecture having:

a memory cache;

a set of heterogeneous processors sharing the memory cache and providing relatively different performances with respect to different portions of the program; and

a switch unit alternately switching between the heterogeneous processors for different portions of the program based on an assessed comparative advantage of the heterogeneous processors in executing the different portions;

wherein the heterogeneous processors include:

(a) a general-purpose processor providing speculative execution and capable of executing the program entirely;

(b) a first special purpose processor providing higher performance execution than the general-purpose processor for first given program portions; and

(c) a second special purpose processor providing higher performance execution than the general-purpose processor for second given program portions; the method comprising the steps of:

(a) allocating program portions to different of the heterogeneous processors according to a first allocation pattern;

(b) profiling the allocated program portions during execution on the heterogeneous processors according to the first allocation pattern; and

(c) based on the profiling, reallocate the program portions to the different heterogeneous processors according to a second allocation pattern providing improved performance.

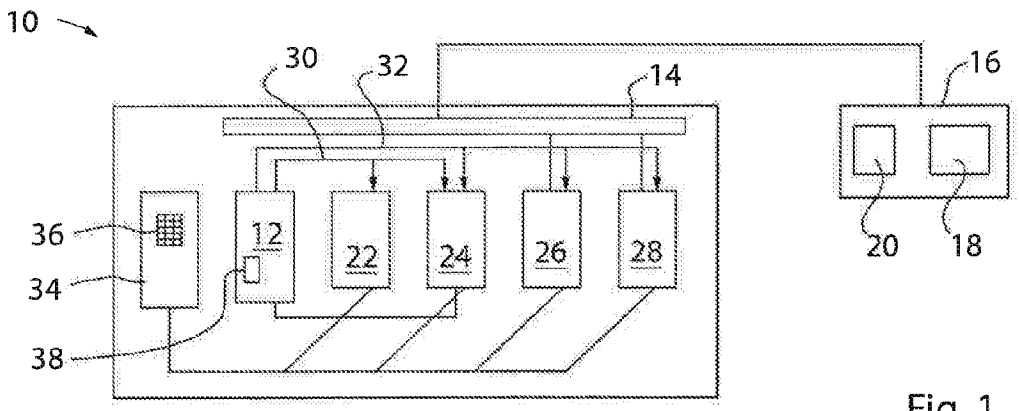


Fig. 1

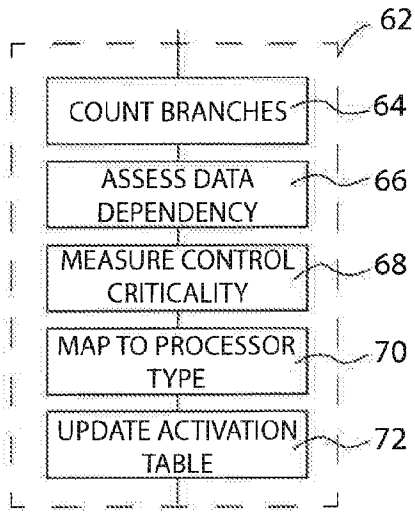
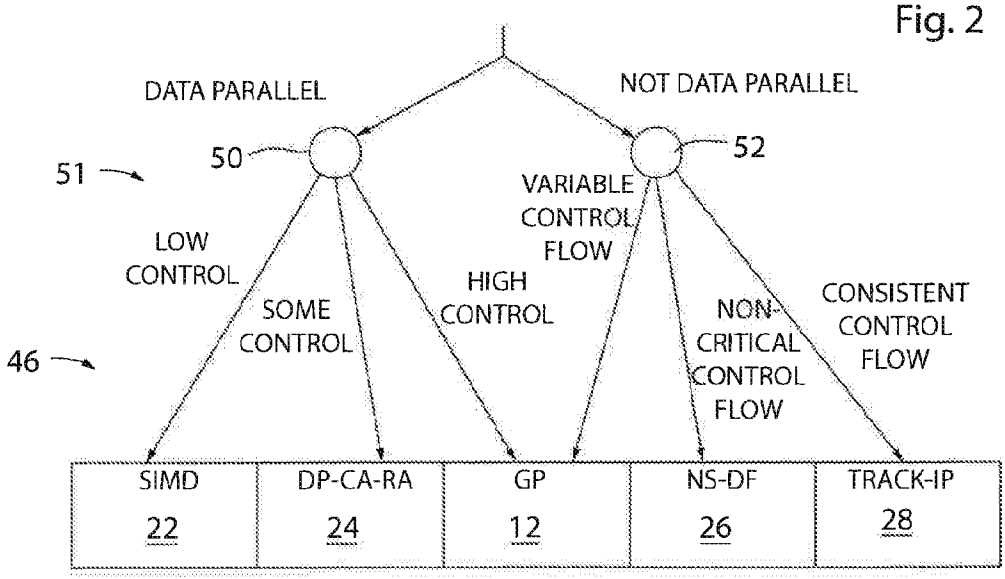


Fig. 5

Fig. 2



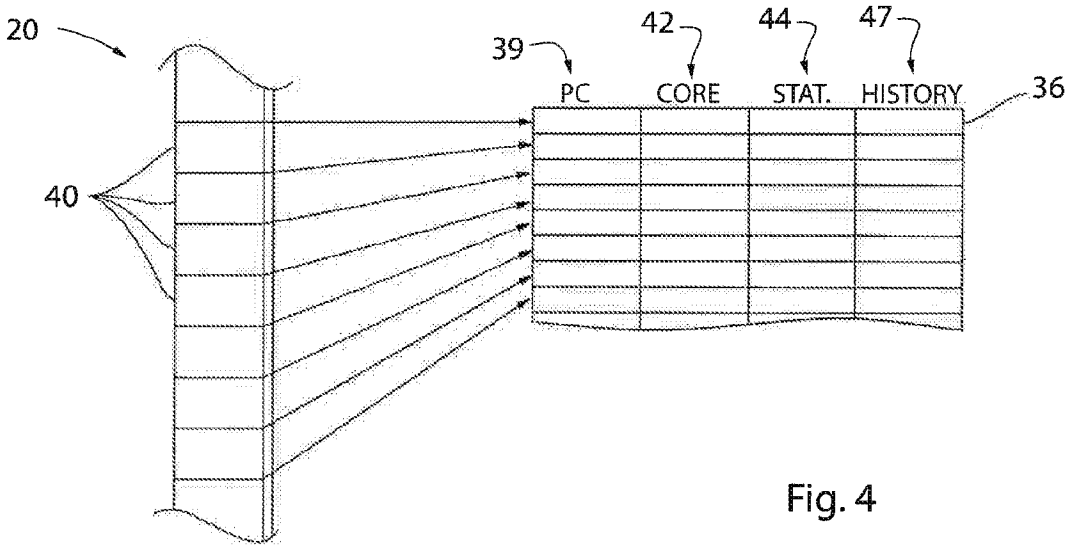

20
39
42
44
47
PC
CORE
STAT.
HISTORY
36
40

Fig. 4

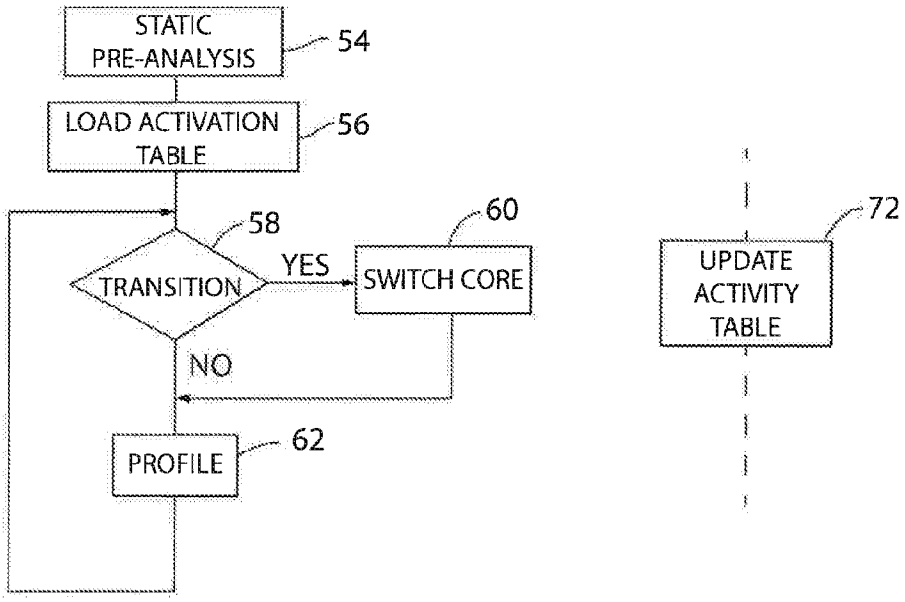

STATIC PRE-ANALYSIS
54
LOAD ACTIVATION TABLE
56
58
60
TRANSITION
YES
SWITCH CORE
NO
PROFILE
62
72
UPDATE ACTIVITY TABLE

Fig. 3

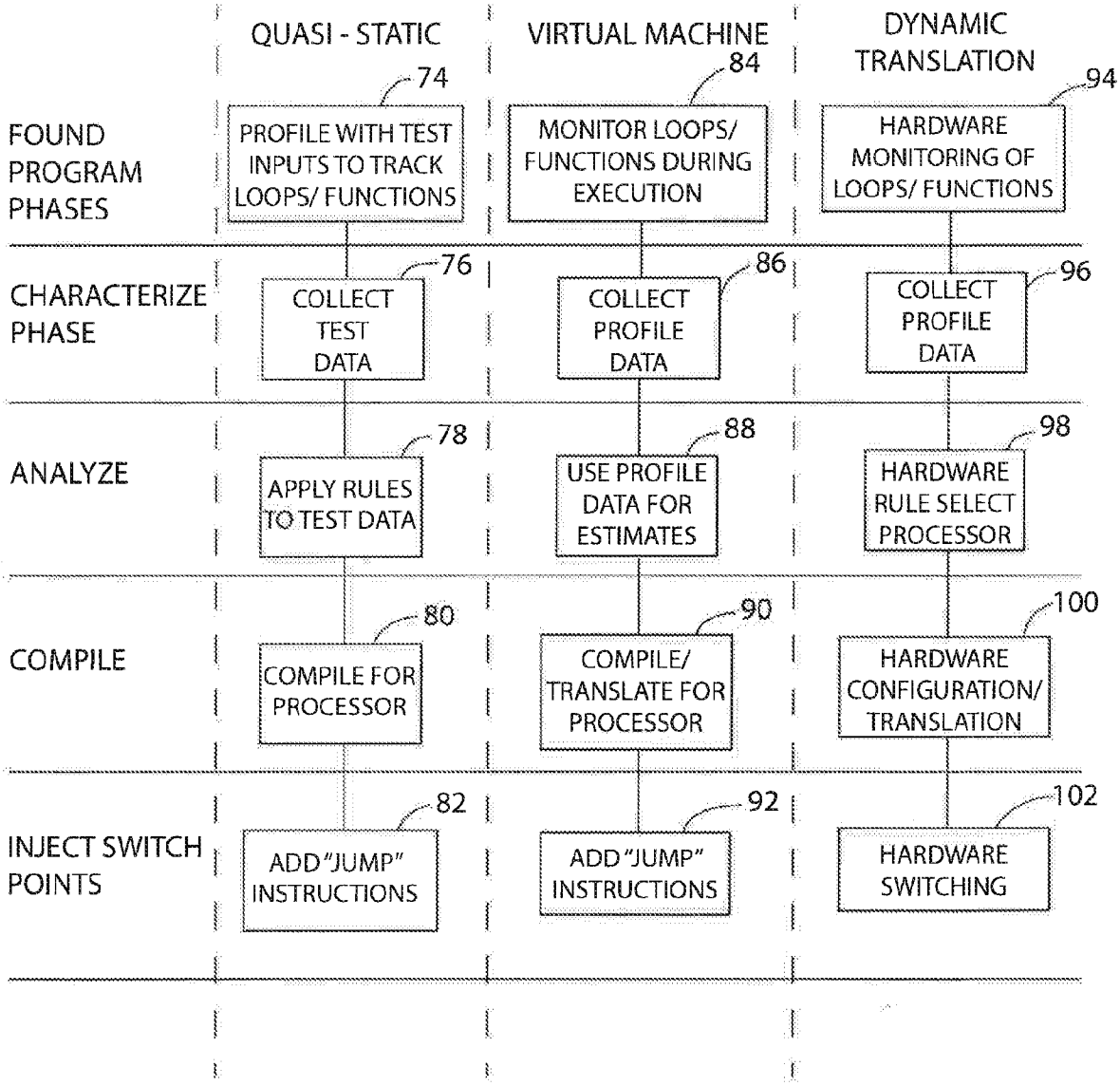

QUASI - STATIC
VIRTUAL MACHINE
DYNAMIC TRANSLATION
FOUND PROGRAM PHASES
CHARACTERIZE PHASE
ANALYZE
COMPILE
INJECT SWITCH POINTS
74
PROFILE WITH TEST INPUTS TO TRACK LOOPS/ FUNCTIONS
84
MONITOR LOOPS/ FUNCTIONS DURING EXECUTION
94
HARDWARE MONITORING OF LOOPS/ FUNCTIONS
76
COLLECT TEST DATA
86
COLLECT PROFILE DATA
96
COLLECT PROFILE DATA
78
APPLY RULES TO TEST DATA
88
USE PROFILE DATA FOR ESTIMATES
98
HARDWARE RULE SELECT PROCESSOR
80
COMPILE FOR PROCESSOR
90
COMPILE/ TRANSLATE FOR PROCESSOR
100
HARDWARE CONFIGURATION/ TRANSLATION
82
ADD "JUMP" INSTRUCTIONS
92
ADD "JUMP" INSTRUCTIONS
102
HARDWARE SWITCHING

Fig. 6

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2017/045702

A. CLASSIFICATION OF SUBJECT MATTER

G06F 9/30(2006.01)i

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

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F 9/30; G06F 15/00; G06F 9/44; G06F 9/46; G06F 13/24

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
Korean utility models and applications for utility models
Japanese utility models and applications for utility models

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
eKOMPASS(KIPO internal) & keywords: heterogeneous processors, switch, profiling, reallocate, performance, speculative execution, control criticality, consistency

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 2011-0067029 A1 (ANDREW WOLFE et al.) 17 March 2011 See paragraphs [0014], [0018]-[0019], [0027], [0029], [0032]; claims 1, 18, 22-24; and figures 1-2.	1-19
Y	US 2008-0250414 A1 (DANIEL ALAN BROKENSHIRE et al.) 09 October 2008 See paragraphs [0014], [0032], [0063], [0139], [0146]-[0147]; claims 1, 8; and figure 42.	1-19
Y	US 2008-0282064 A1 (MICHAEL NORMAN DAY et al.) 13 November 2008 See paragraphs [0022]-[0027]; claims 8-9; and figure 1.	4-5
A	US 2014-0013020 A1 (MATTHEW JAMES HORSNELL et al.) 09 January 2014 See paragraphs [0042], [0086]; claims 1, 12-16; and figure 7.	1-19
A	US 2013-0160016 A1 (JAYANTH GUMMARAJU et al.) 20 June 2013 See paragraphs [0037]-[0044]; and figure 2.	1-19

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

☒ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of 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
08 November 2017 (08.11.2017)	**08 November 2017 (08.11.2017)**

Name and mailing address of the ISA/KR
International Application Division
Korean Intellectual Property Office
189 Cheongsa-ro, Seo-gu, Daejeon, 35208, Republic of Korea
Facsimile No. +82-42-481-8578

Authorized officer
CHIN, Sang Bum
Telephone No. +82-42-481-8398

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2017/045702

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2011-0067029 A1	17/03/2011	CN 102473113 A	23/05/2012
		CN 102473113 B	10/06/2015
		GB 2485683 A	23/05/2012
		JP 2013-501297 A	10/01/2013
		JP 5615361 B2	29/10/2014
		KR 10-1362017 B1	11/02/2014
		KR 10-2012-0017471 A	28/02/2012
		US 8881157 B2	04/11/2014
		WO 2011-031356 A1	17/03/2011
US 2008-0250414 A1	09/10/2008	CN 101040268 A	19/09/2007
		CN 101040268 B	06/10/2010
		CN 1494690 A	05/05/2004
		CN 1494690 C	20/08/2008
		CN 1496511 A	12/05/2004
		CN 1496511 C	20/12/2006
		CN 1496516 A	12/05/2004
		CN 1496516 B	20/07/2011
		CN 1496517 A	12/05/2004
		CN 1496517 C	11/10/2006
		CN 1496518 A	12/05/2004
		CN 1496518 C	11/10/2006
		EP 1370948 A1	17/12/2003
		EP 1370948 B1	19/05/2010
		EP 1370961 A1	17/12/2003
		EP 1370961 B1	05/01/2011
		EP 1370968 A1	17/12/2003
		EP 1370968 B1	16/11/2011
		EP 1370969 A1	17/12/2003
		EP 1370969 B1	14/03/2012
		EP 1370971 A1	17/12/2003
		EP 1370971 B1	05/06/2013
		EP 1805626 A2	11/07/2007
		EP 1805626 B1	22/07/2009
		EP 2296090 A2	16/03/2011
		EP 2296090 A3	23/03/2011
		EP 2296090 B1	13/02/2013
		JP 2002-342165 A	29/11/2002
		JP 2002-351850 A	06/12/2002
		JP 2002-358289 A	13/12/2002
		JP 2002-366533 A	20/12/2002
		JP 2002-366534 A	20/12/2002
		JP 2003-271570 A	26/09/2003
		JP 2003-281107 A	03/10/2003
		JP 2003-303134 A	24/10/2003
		JP 2004-046861 A	12/02/2004
		JP 2004-078979 A	11/03/2004
		JP 2004-252990 A	09/09/2004
		JP 2006-107513 A	20/04/2006

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2017/045702

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
		JP 2006-107514 A	20/04/2006
		JP 3411273 B2	26/05/2003
		JP 3454808 B2	06/10/2003
		JP 3483877 B2	06/01/2004
		JP 3515985 B2	05/04/2004
		JP 3696563 B2	21/09/2005
		JP 4334901 B2	30/09/2009
		JP 4455822 B2	21/04/2010
		JP 4489399 B2	23/06/2010
		JP 4597553 B2	15/12/2010
		JP 4768386 B2	07/09/2011
		KR 10-0840113 B1	20/06/2008
		KR 10-0847982 B1	22/07/2008
		KR 10-0866739 B1	03/11/2008
		KR 10-0890134 B1	20/03/2009
		KR 10-0891063 B1	31/03/2009
		KR 10-0939398 B1	28/01/2010
		KR 10-0959748 B1	25/05/2010
		KR 10-2003-0085037 A	01/11/2003
		KR 10-2008-0108588 A	15/12/2008
		TW 200629053 A	16/08/2006
		TW 200634553 A	01/10/2006
		TW I227401 B	01/02/2005
		TW I266200 B	11/11/2006
		TW I317886 B	01/12/2009
		TW I319529 B	11/01/2010
		US 2002-0135582 A1	26/09/2002
		US 2002-0138637 A1	26/09/2002
		US 2002-0138701 A1	26/09/2002
		US 2002-0138707 A1	26/09/2002
		US 2002-0156993 A1	24/10/2002
		US 2003-0229765 A1	11/12/2003
		US 2005-0078117 A1	14/04/2005
		US 2005-0081181 A1	14/04/2005
		US 2005-0081209 A1	14/04/2005
		US 2005-0081213 A1	14/04/2005
		US 2005-0097302 A1	05/05/2005
		US 2005-0120187 A1	02/06/2005
		US 2005-0120254 A1	02/06/2005
		US 2005-0138325 A1	23/06/2005
		US 2005-0160097 A1	21/07/2005
		US 2005-0268048 A1	01/12/2005
		US 2006-0190614 A1	24/08/2006
		US 2007-0168538 A1	19/07/2007
		US 2007-0186077 A1	09/08/2007
		US 2007-0288701 A1	13/12/2007
		US 2008-0256275 A1	16/10/2008
		US 2011-0302591 A1	08/12/2011
		US 6526491 B2	25/02/2003
		US 6809734 B2	26/10/2004

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2017/045702

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
		US 6826662 B2	30/11/2004
		US 7093104 B2	15/08/2006
		US 7139882 B2	21/11/2006
		US 7231500 B2	12/06/2007
		US 7233998 B2	19/06/2007
		US 7392511 B2	24/06/2008
		US 7457939 B2	25/11/2008
		US 7496673 B2	24/02/2009
		US 7509457 B2	24/03/2009
		US 7516334 B2	07/04/2009
		US 7720982 B2	18/05/2010
		US 7999813 B2	16/08/2011
		US 8028288 B2	27/09/2011
		US 8091078 B2	03/01/2012
		US 8321866 B2	27/11/2012
		US 8434091 B2	30/04/2013
		WO 02-077826 A1	03/10/2002
		WO 02-077838 A1	03/10/2002
		WO 02-077845 A1	03/10/2002
		WO 02-077846 A1	03/10/2002
		WO 02-077848 A1	03/10/2002
		WO 2006-038714 A2	13/04/2006
		WO 2006-038714 A3	29/06/2006
		WO 2006-038717 A2	13/04/2006
		WO 2006-038717 A3	01/03/2007
US 2008-0282064 A1	13/11/2008	US 8214808 B2	03/07/2012
US 2014-0013020 A1	09/01/2014	US 9021172 B2	28/04/2015
US 2013-0160016 A1	20/06/2013	US 8707314 B2	22/04/2014
		WO 2013-090788 A1	20/06/2013