



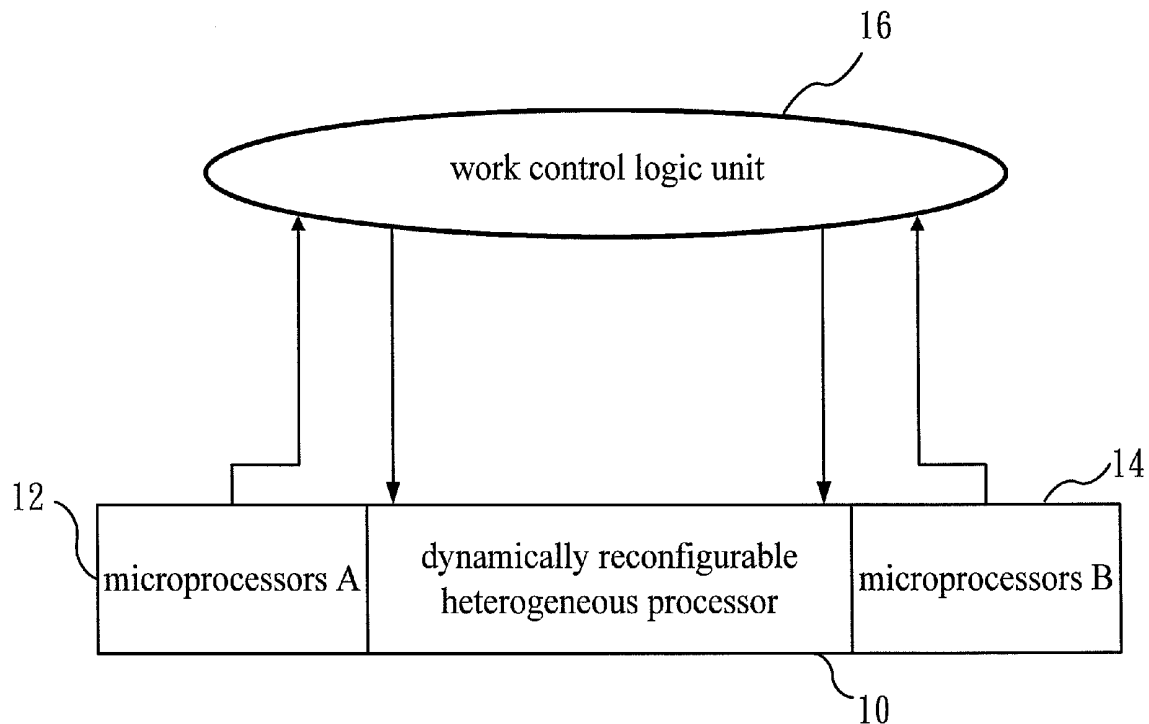
US 20110197048A1

(19) **United States**(12) **Patent Application Publication**
CHUNG et al.(10) **Pub. No.: US 2011/0197048 A1**(43) **Pub. Date: Aug. 11, 2011**(54) **DYNAMIC RECONFIGURABLE
HETEROGENEOUS PROCESSOR
ARCHITECTURE WITH LOAD BALANCING
AND DYNAMIC ALLOCATION METHOD
THEREOF**(76) Inventors: **Chung-Ping CHUNG**, Hsinchu
City (TW); **Hui-Chin Yang**,
Hsinchu County (TW); **Yi-Chi
Chen**, Taipei County (TW)(21) Appl. No.: **13/020,571**(22) Filed: **Feb. 3, 2011**(30) **Foreign Application Priority Data**

Feb. 11, 2010 (TW) 099104390

Publication Classification(51) **Int. Cl.**
G06F 15/76 (2006.01)
G06F 9/02 (2006.01)(52) **U.S. Cl.** **712/30; 712/E09.002**(57) **ABSTRACT**

A dynamic reconfigurable heterogeneous processor architecture with load balancing and dynamic allocation method thereof is disclosed. The present invention uses a work control logic unit to detect load imbalance between different types of processors, and employs a number of dynamically reconfigurable heterogeneous processors to offload the heavier loaded processors. Hardware utilization of such design can be enhanced, and variation in computation needs among different computation phases can be better handled. To design the dynamic reconfigurable heterogeneous processors, a method of how to choose the basic building blocks and place the routing components is included. With the present invention, performance can be maximized at a minimal hardware cost. Hence the dynamic reconfigurable heterogeneous processor(s) so constructed and the load balancing and dynamic allocation method together will have the best performance at least cost.



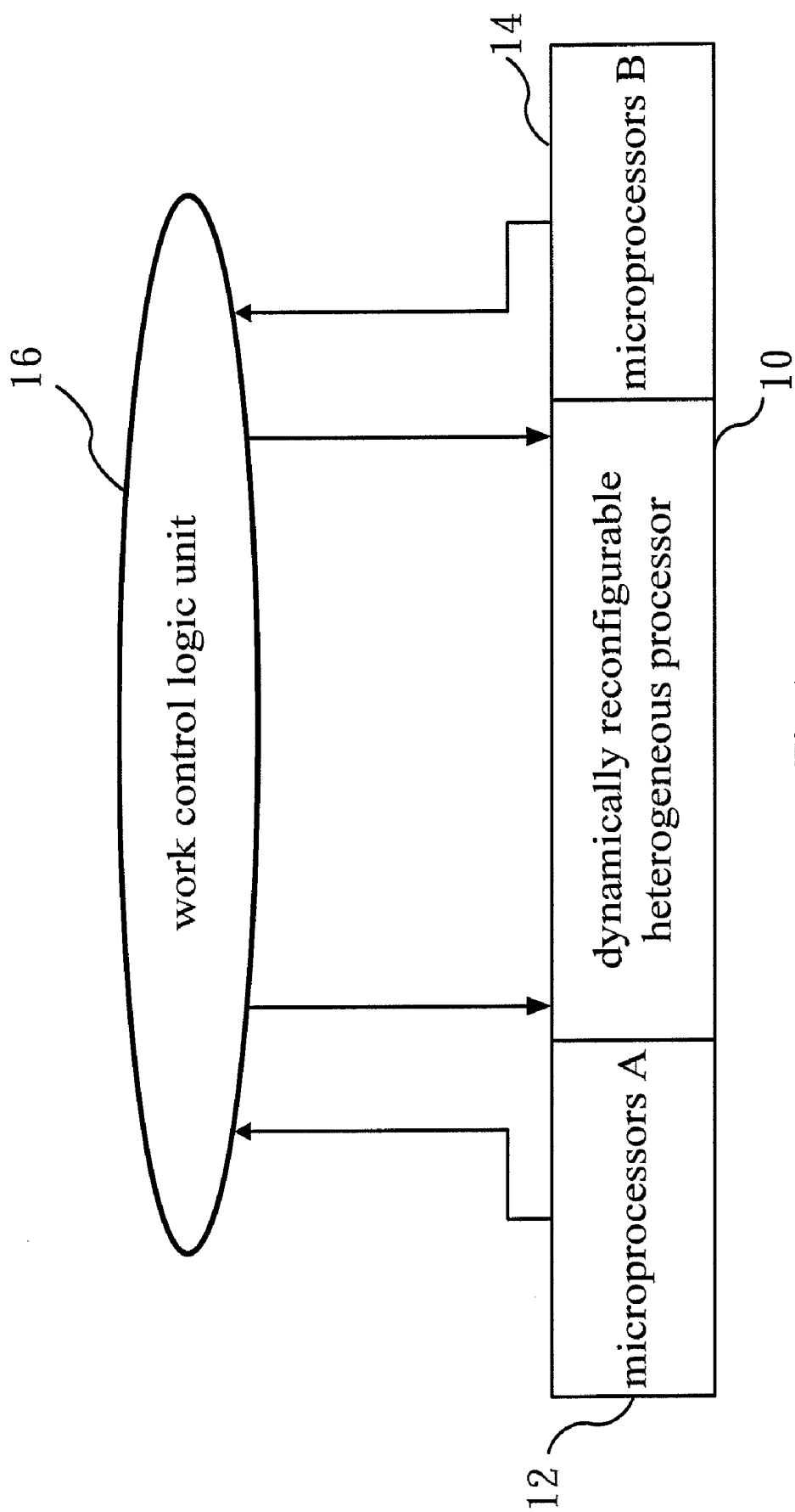


Fig.1

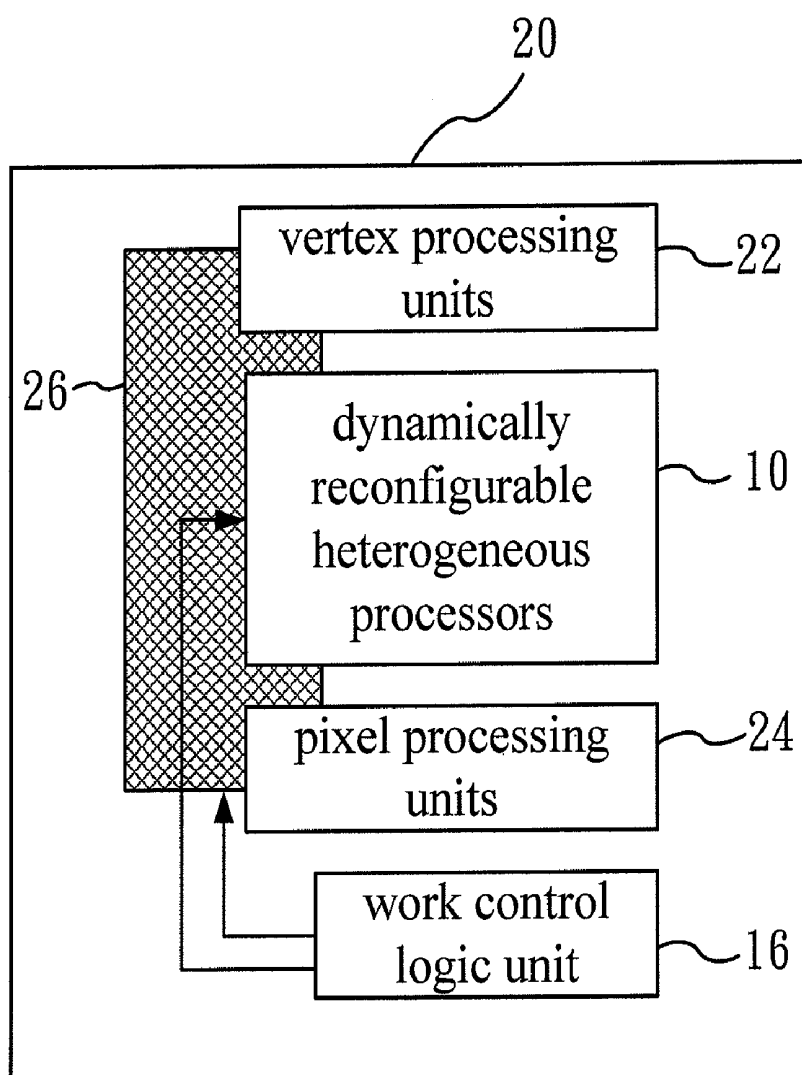


Fig.2

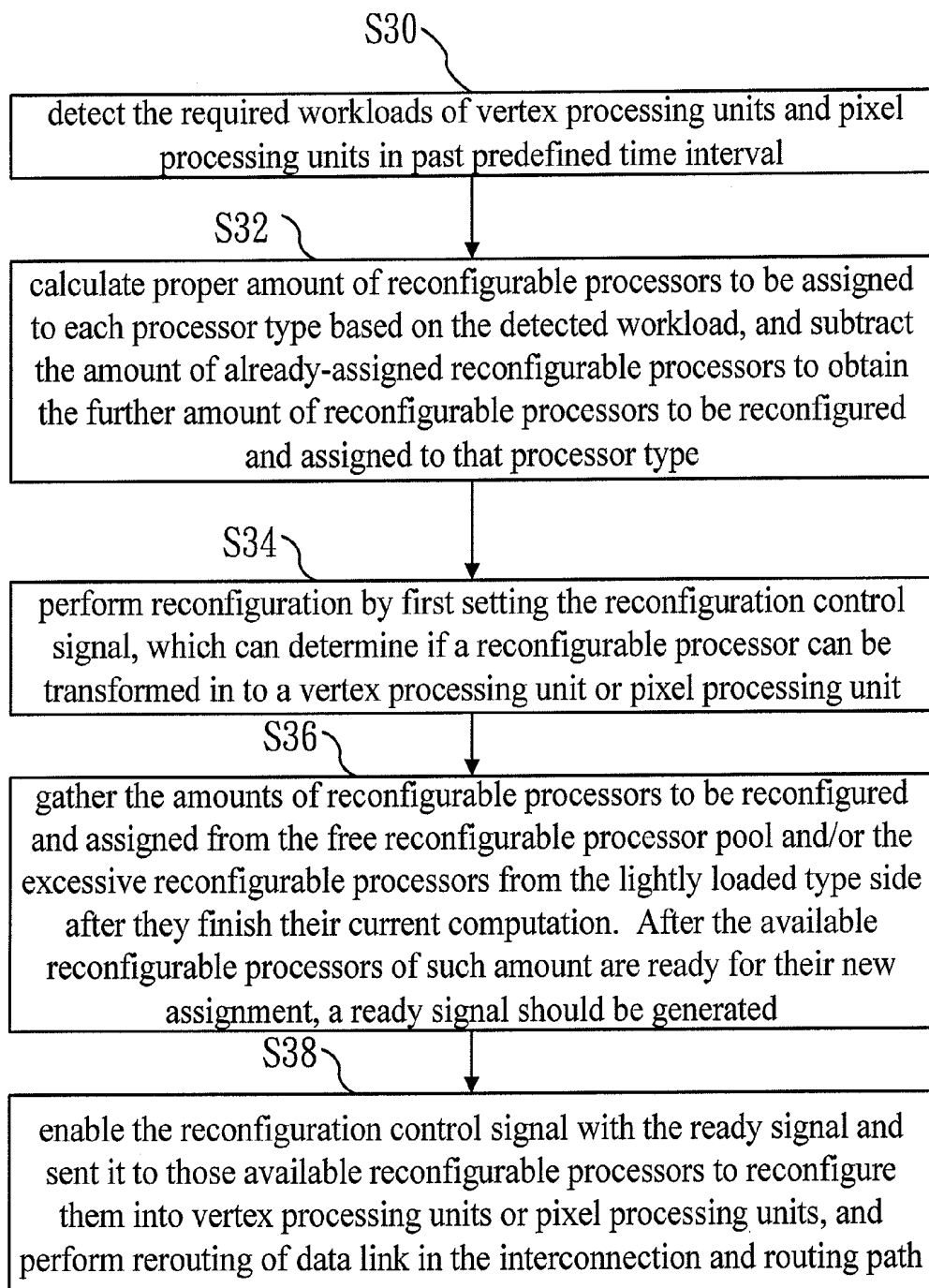


Fig.3

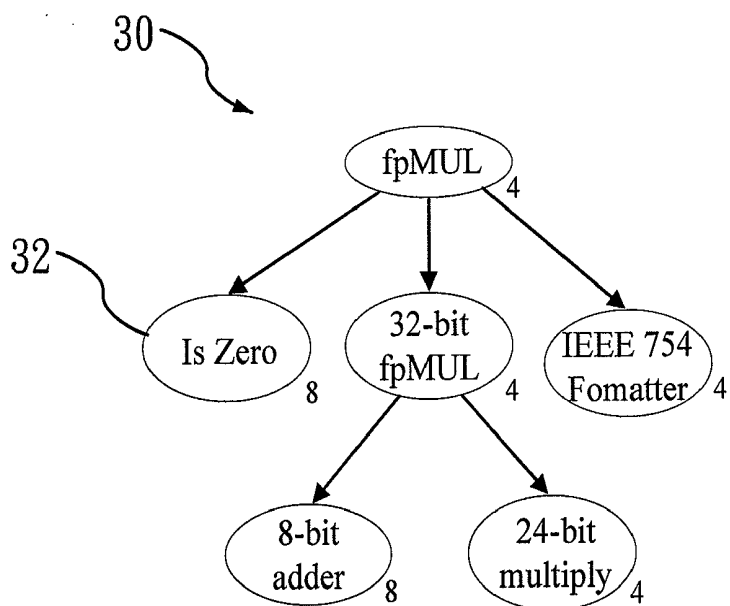


Fig.4(a)

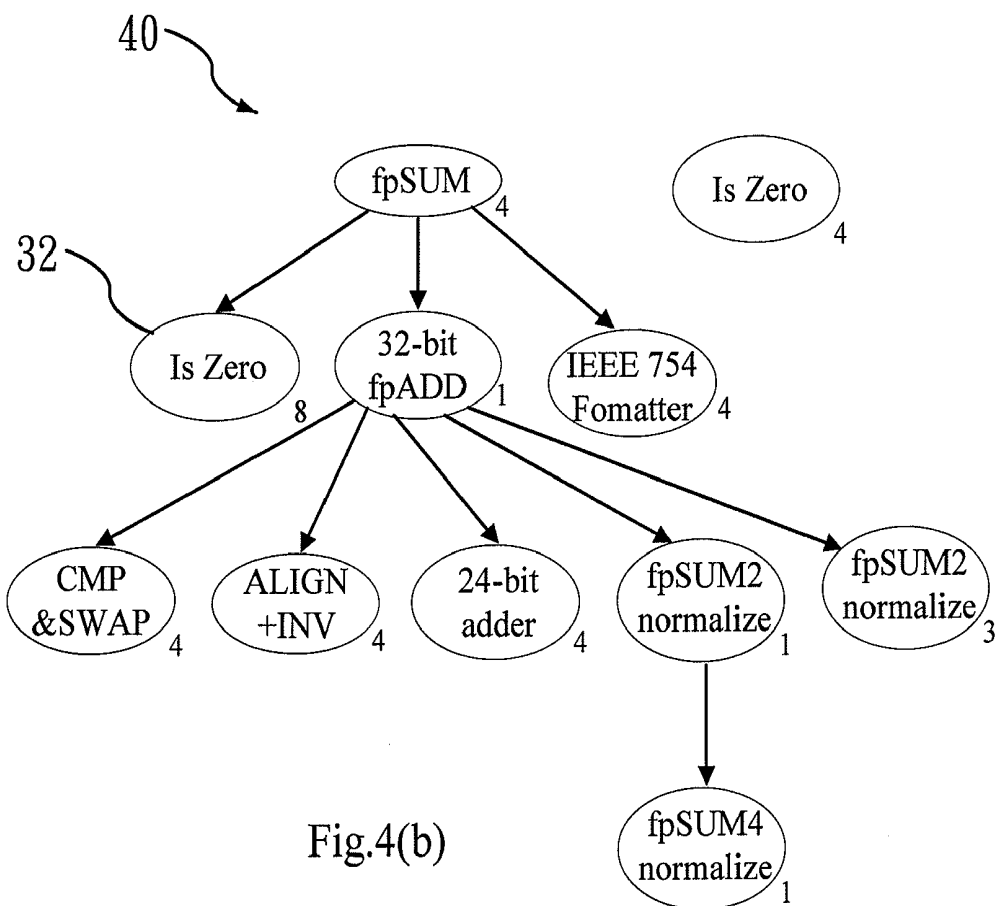


Fig.4(b)

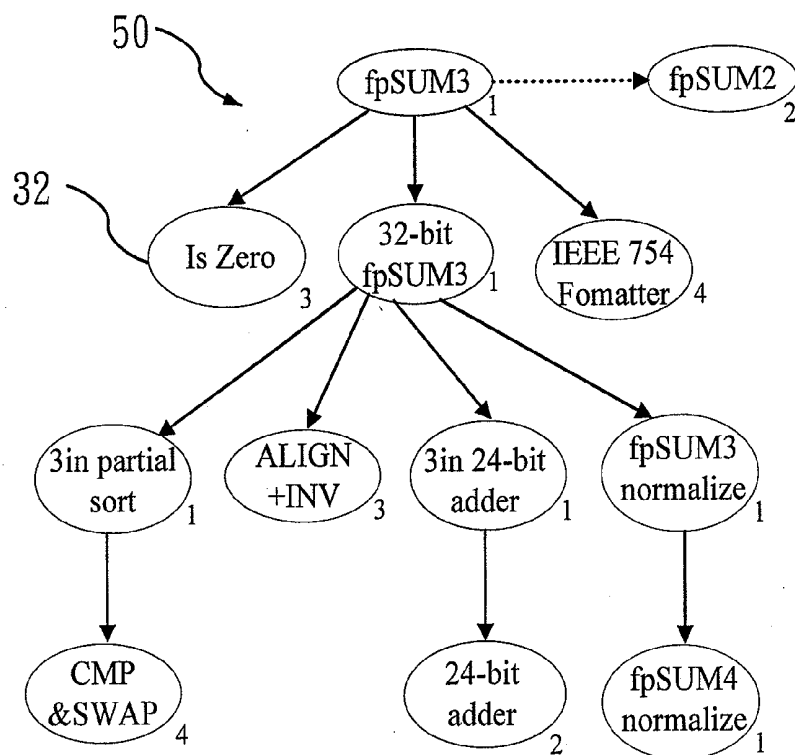


Fig.4(c)

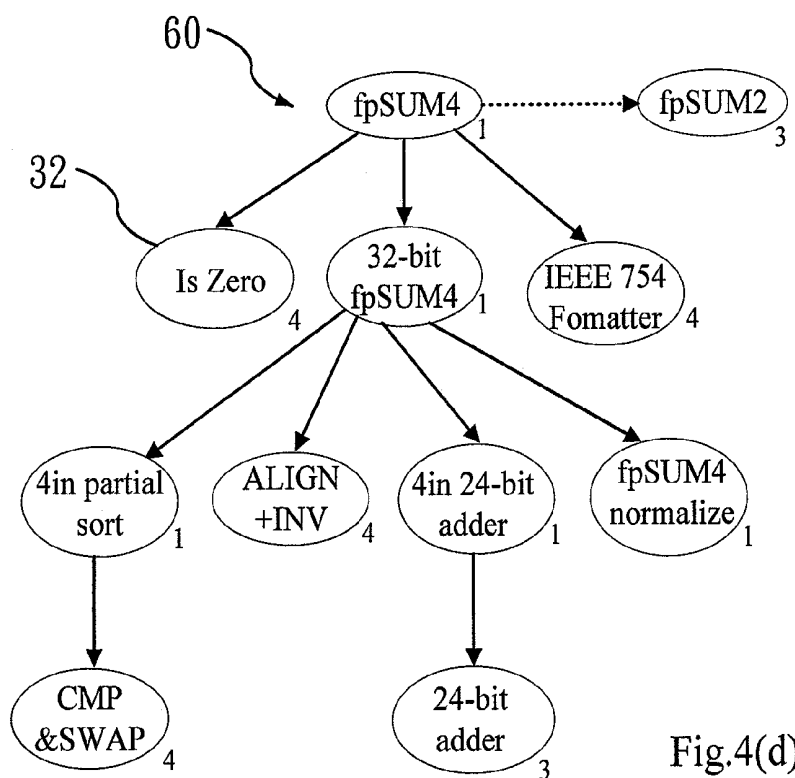


Fig.4(d)

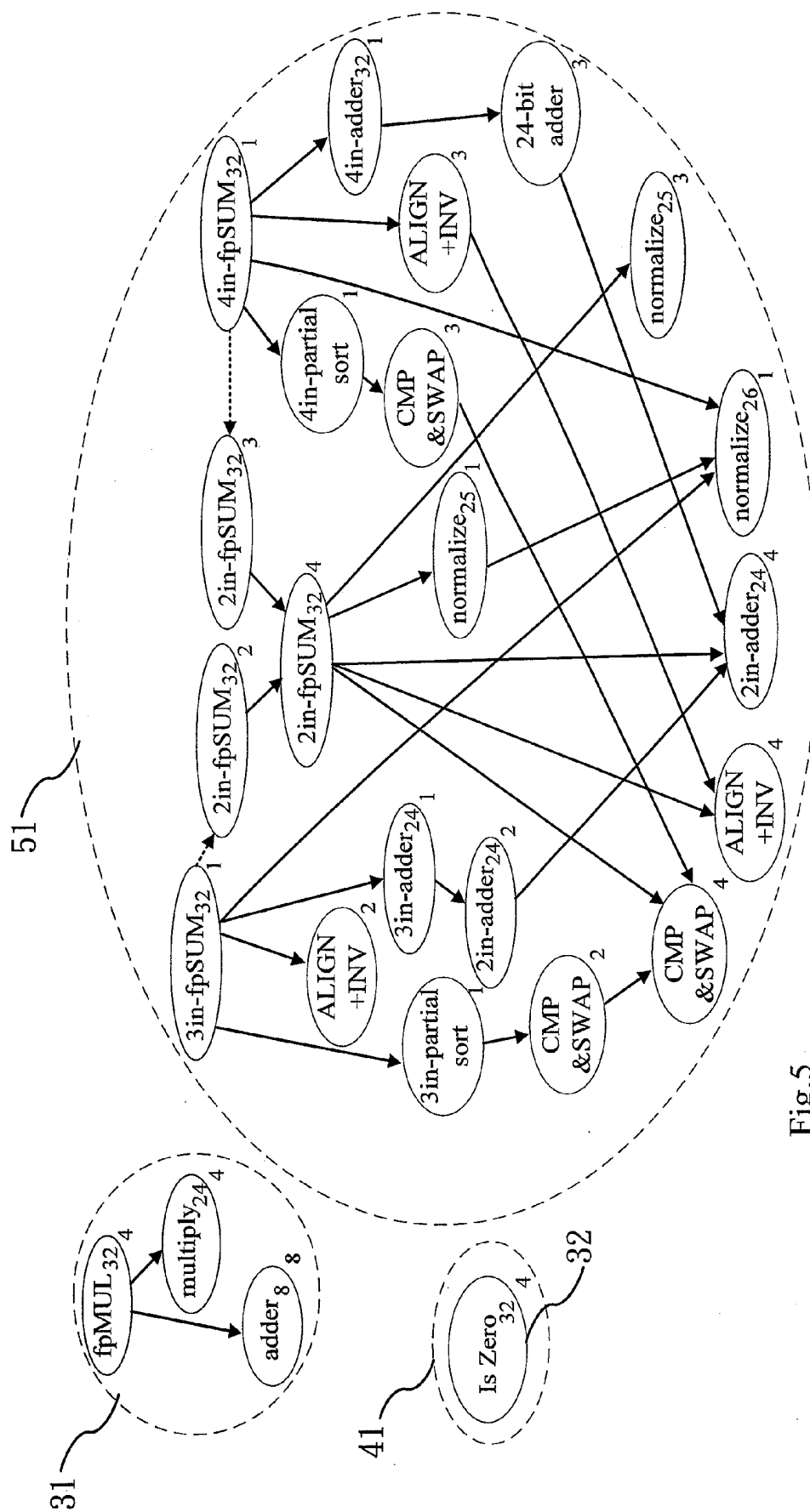


Fig.5

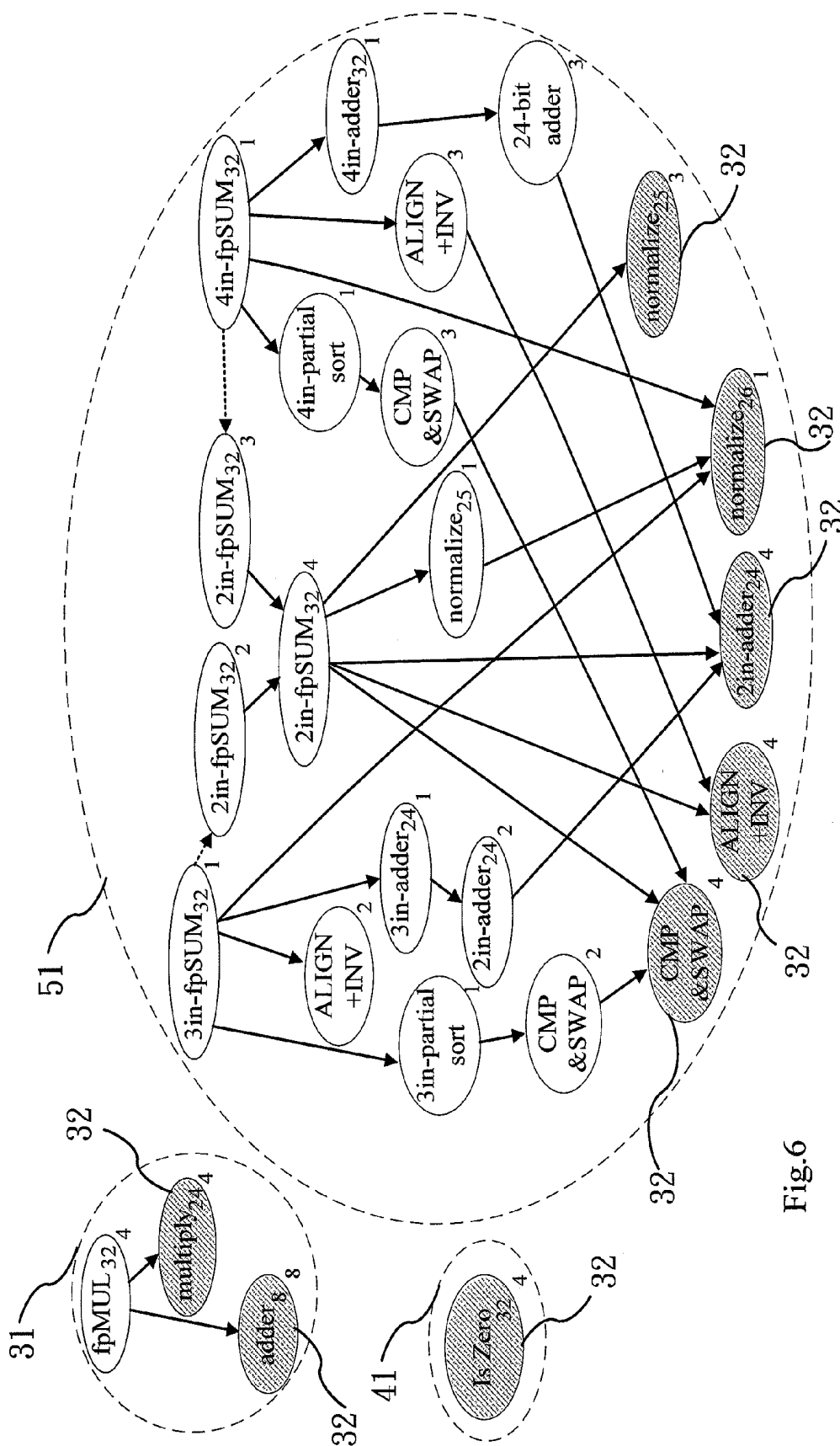


Fig.6

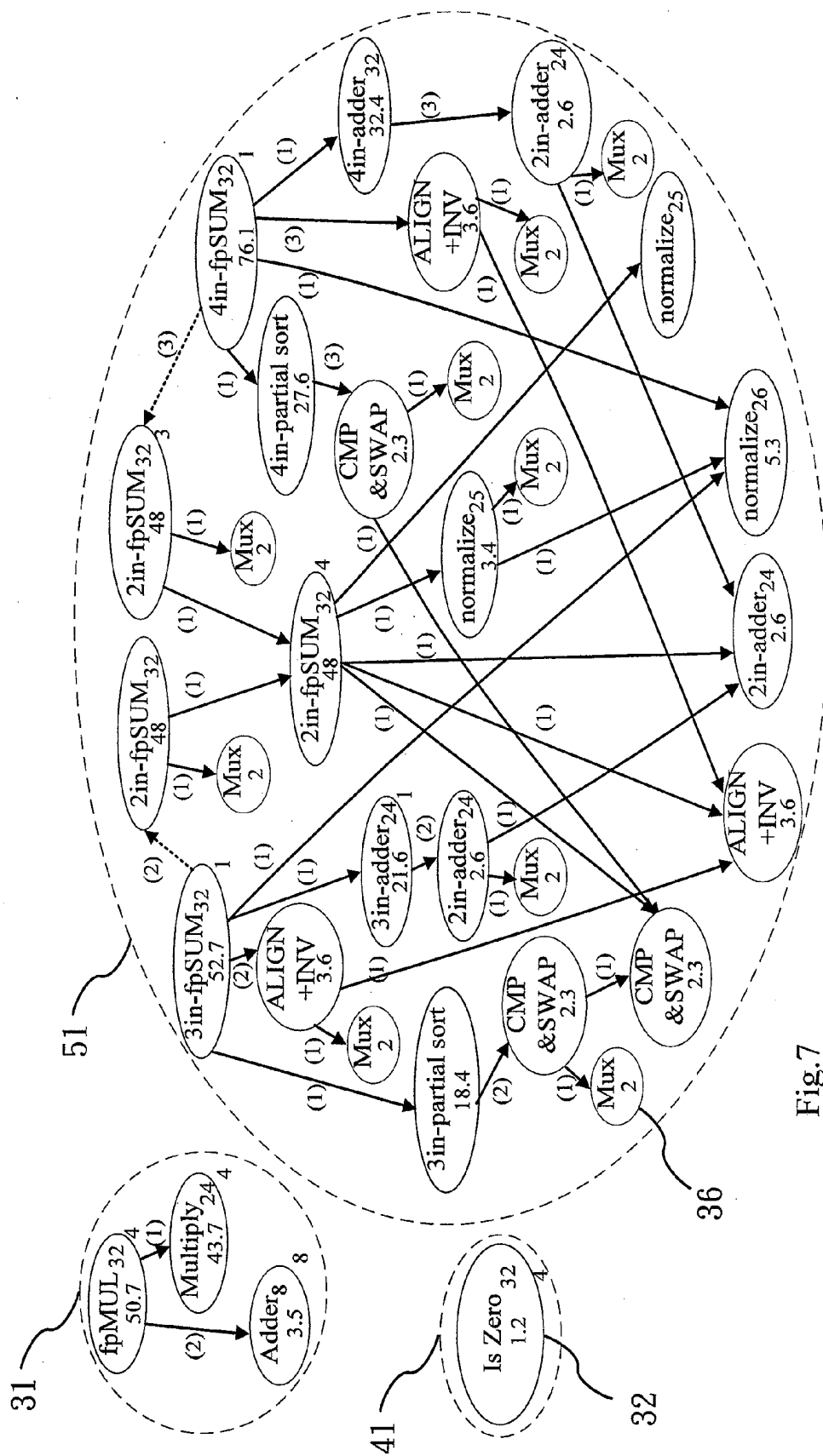


Fig.7

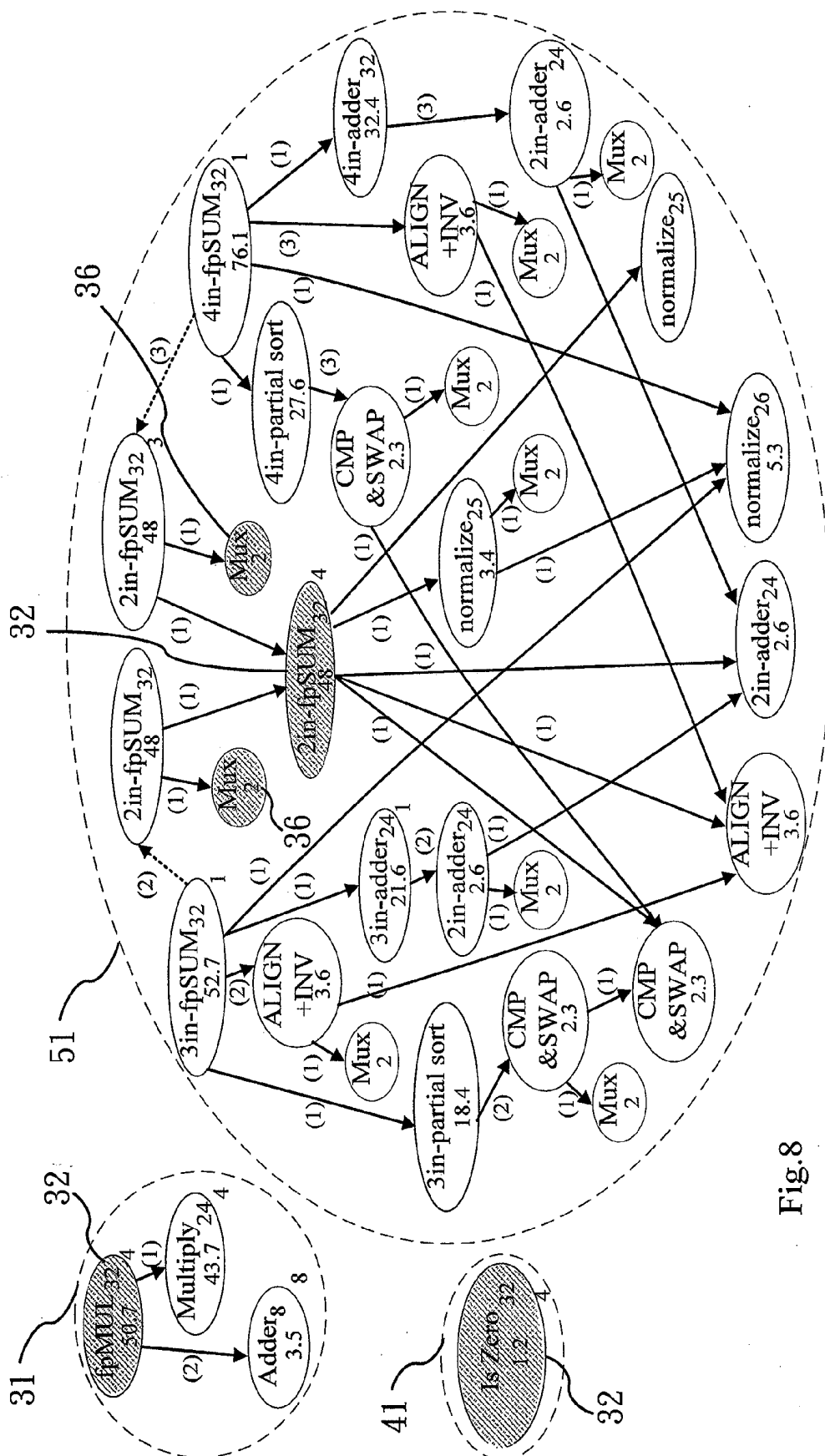


Fig.8

**DYNAMIC RECONFIGURABLE
HETEROGENEOUS PROCESSOR
ARCHITECTURE WITH LOAD BALANCING
AND DYNAMIC ALLOCATION METHOD
THEREOF**

BACKGROUND OF THE INVENTION

[0001] 1. Field of the Invention

[0002] The present invention is a kind of computer architecture, a load balancing reconfigurable heterogeneous processor architecture with dynamic allocation method for high performance in particular.

[0003] 2. Description of the Related Art

[0004] As today's semiconductor technology advances at a rate sketched by the Moore's law, the assorted digital information apparatus tends to integrate processors with various functions into SoC (System-on-a-Chip) to suit the needs of versatility and small form factor. While such an SoC is at work, the characteristics of the application tend to use some processors of certain type intensively but leave those of other types idling from time to time, causing the abundant hardware resources often unevenly used. This ever-changing needs for different types of processors along time greatly lower the overall performance.

[0005] For example, the vastly used GPUs (Graphic Processing Units) in computer systems consist of large numbers of vertex shaders and pixel shaders. They process graphics through coordinate and light transformations, texture compression/decompression, bi-linear pixel shading, etc., to render graphics. The first task among these, vertex shading, shades vertices of geometries through coordinate and light transformation using a large number of vertex shaders. These shaded vertices are then passed on to another group of large number of pixel shaders and texture units for texture compression/decompression, bi-linear pixel shading, etc. As a result, often the number of pixels to be processed occasionally becomes much greater than the number of vertices, or while the vertex shaders are busy processing, the pixel shaders and texture units are idling; whereas while the pixel shaders and texture units are busy processing, the vertex shaders have little work to do. This fact makes the two sets of processors run unevenly along time, lowering the overall performance of the GPU. One solution may be to use unified shaders, but the costs are more complex shader circuits and routings.

[0006] To deal with such a deficiency, the US Patent US2007/0091089A1 proposes a dynamically allocatable GPU system with method, which is equipped with multiple sharable units such as a sharable vertex processor, a sharable geometry processor, and a sharable pixel processor. Through at least one control unit, the sharable processors are assigned execution tasks, and the workload of each processor is monitored. Those unloaded sharable processors can be assigned to assist the loaded sharable processors.

[0007] However, the aforementioned patent US2007/0091089A1 uses a plurality of shareable shaders to share the loads of various shading tasks, resulting in complicated hardware design and its associated monitoring and load sharing algorithm. The present invention is intended to resolve such difficulties. The present invention presents dynamic reconfig-

urable heterogeneous processors architecture with load balancing and dynamic allocation method

SUMMARY OF THE INVENTION

[0008] The primary objective of this invention is to propose a load-balancing, dynamically reconfigurable heterogeneous processors architecture with dynamic reconfiguration and allocation method. It uses a (plurality of) dynamically reconfigurable processor(s) to share the loads of heavily loaded processor(s) to improve overall system performance.

[0009] A secondary objective of this invention is that it should achieve a good cost/performance measure. This is due to the increased performance is the result of only very small silicon area and energy overheads.

[0010] A further objective of this invention is that it is easily applicable to the various digital system designs that process heterogeneous data and/or operations. The present invention's high compatibility with most such digital system designs is due to its efficient use of hardware and self-management.

[0011] To achieve the aforementioned objectives, the presented invention, the dynamic reconfigurable heterogeneous processor architecture with load balancing and dynamic allocation method thereof, consists of a plurality of processors, one or more dynamically reconfigurable heterogeneous processors, and a work control logic unit. The dynamically reconfigurable heterogeneous processor(s) are treated similarly to the other processors, and the work control logic unit is connected to all these heterogeneous and reconfigurable processors. By monitoring the workload of each processor (possibly through examining the usage of its associated data buffer), the work control logic unit analyzes the loadings of all processors, and determines if which reconfigurable processor should be assigned to assist which processor type. Hence the goal of balancing processor workloads and increasing performance can be achieved.

[0012] In the following, the embodiments of this invention are described in detail, together with schematic illustrations, to help understand the invention's objectives, its technical contents, special features, and how it achieves the goals.

BRIEF DESCRIPTION OF THE DRAWINGS

[0013] FIG. 1 is a schematic diagram showing a dynamically reconfigurable heterogeneous processor system according to an embodiment of the present invention;

[0014] FIG. 2 is a schematic diagram showing the system of a graphic processing unit according to an embodiment of the present invention;

[0015] FIG. 3 is a flowchart of the load balancing dynamic allocation method according to an embodiment of the present invention;

[0016] FIG. 4(a)-4(d) are schematic diagrams showing operation requirement trees of a dynamic reconfigurable heterogeneous processor design according to an embodiment of the present invention;

[0017] FIG. 5 is a schematic diagram showing block-selection trees of the dynamic reconfigurable heterogeneous processor design according to an embodiment of the present invention;

[0018] FIG. 6 is a schematic diagram showing the block-selection trees of the dynamic reconfigurable heterogeneous processor design choosing the sharable logic nodes according to an embodiment of the present invention;

[0019] FIG. 7 is a schematic diagram showing the block-selection trees of the dynamic reconfigurable heterogeneous processor design with multiplexer nodes added according to an embodiment of the present invention; and

[0020] FIG. 8 is a schematic diagram showing the block-selection trees of the dynamic reconfigurable heterogeneous processor design choosing upward composable logic nodes and multiplexer nodes according to an embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

[0021] The present invention reveals a heterogeneous processor architecture with dynamically reconfigurable processor(s) and load balancing mechanism. It uses a work control logic unit to dynamically assign reconfigure processor(s) to assist other processor(s) to balance the loads of the processors. In the following a design example is used to illustrate the technical features of this invention.

[0022] FIG. 1 is a schematic diagram showing a dynamically reconfigurable heterogeneous processor system according to an embodiment of the present invention. In FIG. 1, a dynamically reconfigurable heterogeneous processor 10 is placed in between microprocessors A 12 and microprocessors B 14. (Only one processor of each type is shown for simplicity, although the number of each type can be arbitrary.) Microprocessors A 12 and B 14 each may be a graphic processor, an embedded processor, a digital signal processor, or a multimedia processor, and assume that they are different from each other. The dynamically reconfigurable heterogeneous processor 10 can be configured to perform the function of either A 12 or B 14. A work control logic unit 16 is connected to microprocessors A 12, B 14, and the dynamically reconfigurable heterogeneous processor 10. The work control logic unit 16 examines the loadings of microprocessors A 12 and B 14 (possibly through examining the usages of their associated data buffers), and determines if the dynamically reconfigurable heterogeneous processor should be allocated to assist whichever microprocessor with noticeably heavier load. This allocation is done by configuring the dynamically reconfigurable heterogeneous processor 10 to the specified microprocessor function type, and reroute data links such that the configured processor 10 can receive data for the specified microprocessor and send results to proper destination, both dynamically.

[0023] The present invention is applicable to many digital system designs such as graphics processing unit design. FIG. 2 shows a way of applying the present invention in graphic processor unit design. The graphic processing unit 20 consists of vertex processing units 22, pixel processing units 24, and dynamic reconfigurable heterogeneous processors 10, interconnected with the interconnection and routing path 26. A work control logic unit 16 monitors the vertex processing units 22 and pixel processing units 24, and assigns the dynamic reconfigurable heterogeneous processors to whichever units with noticeably heavier load by dynamically reconfiguring the dynamic reconfigurable heterogeneous processors and rerouting data links in the interconnection and routing path 26. This helps to balance the loads of the vertex processing units 22 and pixel processing units 24.

[0024] Above is the explanation to the architecture of the dynamic reconfigurable heterogeneous processor. In the following the dynamic allocation method and the design flow of dynamic reconfigurable heterogeneous processor system architecture are to be introduced. FIG. 3 shows the flow of the

load balancing dynamic allocation method. Refer also to FIG. 2 when appropriate. In FIG. 3, first, in step S30, the work control logic unit 16 dynamically detects the required workloads of vertex processing units 22 and pixel processing units 24 in past predefined time interval. Then, in step S32, the work control logic unit 16 calculates the proper amount of reconfigurable processors 10 to be assigned to each processor type 22 or 24 based on the detected workloads, and subtracts the amount of already-assigned reconfigurable processors 10 to obtain the further amount of reconfigurable processors 10 to be reconfigured and assigned to that processor type 22 or 24. Then, in step S34, the reconfiguration is performed by first setting the reconfiguration control signal, which can determine if a reconfigurable processor 10 is to be transformed in to a vertex processing unit 22 or pixel processing unit 24. Then, in step S36, the amounts of reconfigurable processors 10 to be reconfigured and assigned are gathered from the free reconfigurable processor 10 pool and/or the excessive reconfigurable processors 10 from the lightly loaded type side after they finish their current computation. After the available reconfigurable processors 10 of such amounts are ready for their new assignments, a ready signal should be generated. Finally, in step S38, the reconfiguration control signal is enabled by the ready signal and sent to these available reconfigurable processors 10 to reconfigure them into vertex processing units 22 or pixel processing units 24. The rerouting of data links in the interconnection and routing path 26 is also performed by the work control logic unit 16 in this load balancing process; its details are not elaborated here to save space.

[0025] Above is the explanation to the dynamic allocation method. In this and subsequent paragraphs, the design flow of the dynamic reconfigurable heterogeneous processors 10 is introduced, and the graphic processing unit 20 is again used for example. With the work control logic unit 16, this invention dynamically allocate dynamic reconfigurable heterogeneous processors 10 to be vertex processing units 22 or pixel processing units 24, balancing processing time of vertices and pixels and enhancing hardware utilization of the graphic processing unit 20. The overall system performance is thus improved. Yet in order to achieve this advantage, such dynamically reconfigurable heterogeneous processors 10 must pay the cost for extra hardware compared with intrinsic vertex processing unit 22 or pixel processing unit 24. It is therefore important to derive a dynamic reconfigurable heterogeneous processor 10 design that is both low-cost and high-performance. FIG. 4(a) to 4(d) are schematic diagrams showing operation requirement trees of a dynamic reconfigurable heterogeneous processor 10 design. Refer also to FIG. 2 when appropriate. First, based on the functional requirements in vertex and pixel shading, four mutually independent operation requirement trees for the vertex processing units 22 and pixel processing units 24, operation requirement tree 30, operation requirement tree 40, operation requirement tree 50, and operation requirement tree 60 are constructed. These four mutually independent operation requirement trees each comprises a plurality of operation nodes 32, and higher-level operation nodes in these trees are each constructed using its descendent operation nodes 32. On the lower-right side of each operation node 32, a number indicates the amount of such operation nodes 32 to be needed. The operation requirement tree helps to show the underlying hardware requirements of those useful fundamental operations in targeted

applications. In the following, these operation requirement trees **30**, **40**, **50**, and **60**, shown in FIG. **4(a)** to **4(d)**, will be explained in detail.

[0026] FIG. **4(a)** shows that the operation requirement tree **30** has six operation nodes **32**. It consists of four floating-point multipliers (fpMUL), which in turn consists of eight zero detectors (IsZero), four 32-bit floating-point multipliers (32-bit fpMUL) and four IEEE 754 formatters (IEEE 754 Formatter). And the four 32-bit fpMULs are constructed with eight 8-bit adders (8-bit adder) and four 24-bit multipliers (24-bit multiply). FIG. **4(b)** shows that the operation requirement tree **40** consists of eleven operation nodes **32**. It consists of four floating-point adders (fpSUM), which in turn consists of eight zero detectors (IsZero), one 32-bit floating-point adder (32-bit fpADD), and four IEEE 754 formatters (IEEE 754 Formatter). And the 32-bit fpADD is constructed with four compare-and-swappers (CMP&SWAP), four align-and-inverters (ALIGN+INV), four 24-bit adders (24-bit adder), one floating-point normalizer for floating-point 2-operand adder (fpSUM2 normalize), one floating-point normalizer for floating-point 4-operand adder (fpSUM4 normalize), and three floating-point normalizer for floating-point 2-operand adder (fpSUM2 normalize). It also consists of four other zero detectors (IsZero).

[0027] FIG. **4(c)** shows that the operation requirement tree **50** has twelve operation nodes **32**. It consists of one floating-point 3-operand adder (fpSUM3), which in turn consists of three zero detectors (IsZero), one 32-bit floating-point 3-operand adder (32-bit fpSUM3) and four IEEE 754 formatters (IEEE 754 Formatter). And the 32-bit floating-point 3-operand adder (32-bit fpSUM3) is constructed with one 3-input partial sorter (3 in partial sort) which further consists of four compare-and-swappers (CMP&SWAP), three align-and-inverters (ALIGN+INV), one 3-input 24-bit adder (3 in 24-bit adder) which further consists of two 24-bit adders (24-bit adder), and one floating-point normalizer for floating-point 3-operand adder (fpSUM3 normalize) which further consists of one floating-point normalizer for floating-point 4-operand adder (fpSUM4 normalize). There are in addition two floating-point 2-operand adders (fpSUM2). The dotted arrow means that the operation node pointed at by the arrow can be used to substitute for the operation node at the origin of the arrow, and the lower right numbers indicate the corresponding amounts of the respective hardware units. FIG. **4(d)** shows that the operation requirement tree **60** also consists of eleven operation nodes **32**. It consists of one floating-point 4-operand adder (fpSUM4), which in turn consists of four zero detectors (IsZero), one 32-bit floating-point 3-operand adder (32-bit fpSUM3) and four IEEE 754 formatters (IEEE 754 Formatter). And the 32-bit floating-point 4-operand adder (32-bit fpSUM4) is constructed with one 4-input partial sorter (4 in partial sort) which further consists of four compare-and-swappers (CMP&SWAP), four align-and-inverters (ALIGN+INV), one 4-input 24-bit adder (4 in 24-bit adder) which further consists of three 24-bit adders (24-bit adder), and one floating-point normalizer for floating-point 4-operand adder (fpSUM4 normalize). There are in addition three floating-point 2-operand adders (fpSUM2). And a dotted arrow indicates that the three floating-point 2-operand adders (fpSUM2) can be used to substitute for the floating-point 4-operand adder (fpSUM4).

[0028] Next the block selection is explained. The purpose of block selection is to define the basic function blocks to be used in the dynamic reconfigurable heterogeneous processors

10. A good selection of the set of blocks both saves hardware cost and simplifies the reconfiguration and rerouting needed. As shown in FIG. **5**, the schematic diagram showing block-selection trees of the dynamic reconfigurable heterogeneous processor **10** design, some common operation nodes **32** in FIG. **4(a)** to **4(d)** are identified from the operation requirement trees **30**, **40**, **50** and **60**, and their block selection trees **31**, **41** and **51** are formed. These block selection trees **31**, **41** and **51** are independent sets of each other. Then, in FIG. **6**, the schematic diagram showing the block-selection trees of the dynamic reconfigurable heterogeneous processor design choosing the sharable logic nodes, required and sharable common hardware circuits—shown as the leaf nodes **32** in FIG. **6**—are identified. FIG. **7**, the schematic diagram showing the block-selection trees of the dynamic reconfigurable heterogeneous processor design with multiplexer nodes added, shows that necessary multiplexers logic nodes **36** are added at a level higher than those sharable logic nodes **32** marked in FIG. **6**. In addition, FIG. **7** relabels the block selection trees **31**, **41** and **51** with the following: every operation node **32** is marked with its hardware cost (absolute or normalized to a reference design) in the lower portion inside the node; and every edge is marked with the number of lower-level operation nodes **32** needed to construct the upper-level operation node **32** by the edge in a pair of parentheses. As an example, in FIG. **7**, the 32-bit floating-point multiplier (fpMUL₃₂) in block selection tree **31** has a hardware cost of 50.7 units, and it can be replaced or constructed using its descendants in the block selection tree **31**: two Adds₈ and one Multiplier₂₄.

[0029] Finally, as shown in FIG. **8**, the block selection trees **31**, **41** and **51** will be searched for those composable operation nodes **32** and associated multiplexers (Mux) **36**, through means of linear programming or similar. The selected operation nodes should fulfill all necessary reconfiguration requirements of the dynamic reconfigurable heterogeneous processors **10**, and the amounts of the selected operation nodes **32** and multiplexers (Mux) **36**, if not constrained, should fulfill the computation needs of the target dynamic reconfigurable heterogeneous processor system with load balancing and dynamic allocation as FIG. **1** shows. The goal is that the selected and equipped operation nodes and multiplexers can maximize the benefit of hardware sharing at a minimal cost. Hence the dynamic reconfigurable heterogeneous processor (s) **10** constructed in this way will have the best performance at their least cost.

[0030] According to the previous disclosure, the present invention uses a work control logic unit **16** to dynamically allocate the dynamic reconfigurable heterogeneous processor (s) **10** to balance the workloads of different processor types. The present invention provides a complete design picture, and it is highly compatible with most contemporary processor system designs. As long as the application requires noticeable amounts of varying types of operations, use of this invention in the system result in very good return on the investment.

[0031] The embodiments described above are only to exemplify the present invention but not to limit the scope of the present invention. Therefore, any equivalent modification or variation according to the shape, structures, characteristics and spirit disclosed in the present invention is to be also included within the scope of the present invention.

What is claimed is:

1. A dynamic reconfigurable heterogeneous processor architecture with load balancing, comprising:

a plurality of microprocessors;

at least one dynamically reconfigurable heterogeneous processor coupled to said microprocessors and assisting said microprocessors in executing operations; and

a work control logic unit coupled to said microprocessors and said dynamically reconfigurable heterogeneous processor, analyzing work proportion of each said microprocessor, dynamically allocating said dynamically reconfigurable heterogeneous processor to support said microprocessors to execute said operations, and balancing workload of each said microprocessor.

2. The dynamic reconfigurable heterogeneous processor architecture with load balancing according to claim 1, wherein

said work control logic unit detects a noticeable imbalance of data or job buffers of said microprocessors under detection, which is used as a basis to analyze said work proportion of each said microprocessor.

3. The dynamic reconfigurable heterogeneous processor architecture with load balancing according to claim 1, wherein

said work control logic unit changes routing paths connecting said dynamically reconfigurable heterogeneous processor and said microprocessors, whereby said dynamically reconfigurable heterogeneous processor is dynamically allocated to assist said microprocessors.

4. The dynamic reconfigurable heterogeneous processor architecture with load balancing according to claim 1, wherein

said dynamically reconfigurable heterogeneous processor assists at least two said microprocessors.

5. The dynamic reconfigurable heterogeneous processor architecture with load balancing according to claim 1, wherein

said microprocessors are graphic processors, embedded processors, digital signal processors, multimedia processors, or a combination of such.

6. The dynamic reconfigurable heterogeneous processor architecture with load balancing according to claim 1, wherein

said dynamically reconfigurable heterogeneous processor is a multi-functional processor.

7. The dynamic reconfigurable heterogeneous processor architecture with load balancing according to claim 1, wherein

a procedure of designing said dynamically reconfigurable heterogeneous processor further comprising steps of performing a plurality of hardware requirement analyses using operation requirement trees for basic operations of said microprocessors, wherein each said operation requirement tree comprises a plurality of operation nodes showing how a required operation is constructed in a variety of ways;

choosing common said operation nodes of said operation requirement trees and establishing a plurality of hardware breakdown lists of said common said operation nodes using block-selection trees;

choosing sharable said logic nodes of said block-selection trees and adding a multiplexer logic node at each sharable said logic node, respectively; and

searching all said block-selection trees and choosing said composable said operation nodes and associated said multiplexers that fulfill all necessary reconfiguration requirements of said dynamically reconfigurable heterogeneous processor.

8. The dynamic reconfigurable heterogeneous processor architecture with load balancing according to claim 7, wherein

in said step of searching all said block-selection trees, searching all said block-selection trees is based on linear programming.

9. The dynamic reconfigurable heterogeneous processor architecture with load balancing according to claim 7, wherein

said composable said operation nodes and said multiplexer logic nodes maximize a benefit of hardware sharing at a minimal cost.

10. The dynamic reconfigurable heterogeneous processor architecture with load balancing according to claim 7, wherein

an amounts of said composable said operation nodes and said multiplexer logic nodes meet hardware requirement to implement said basic operations of said microprocessors.

11. A dynamic allocation method with load balancing, comprising steps of:

detecting instruction execution loads of a plurality of microprocessors in past predefined time interval by a work control logic unit;

said work control logic unit calculating a proper amount of dynamically reconfigurable processors to be assigned to each processor type, and subtracts an amount of already-assigned said dynamically reconfigurable processors to obtain a further amount of said dynamically reconfigurable processors to be reconfigured and assigned to that processor type;

setting reconfiguration control signals which transform said dynamically reconfigurable heterogeneous processors into a desired processor type;

gathering an amount of said dynamically reconfigurable processors to be reconfigured and assigned from a free dynamically reconfigurable processor pool and/or excessive dynamically reconfigurable processors from a lightly loaded type side after they finish their current computation, and generating a ready signal after available said dynamically reconfigurable processors of such amount are ready for their new assignment; and

enabling said reconfiguration control signal using said ready signal such that said available said dynamic reconfigurable processors are properly reconfigured, and rerouting data links in interconnection and routing path according to a updated dynamic reconfigurable processor assignment.

12. The dynamic allocation method with load balancing according to claim 11, wherein in a step of said work control logic unit detecting said instruction execution loads of said microprocessors in past predefined time interval, said work control logic unit detects a noticeable imbalance of data/job buffers of said microprocessors under detection.

13. The dynamic allocation method with load balancing according to claim **11**, wherein said further amount of dynamically reconfigurable processors to be reconfigured and assigned to heavier loaded processor type is calculated.

14. The dynamic allocation method with load balancing according to claim **11**, wherein in step of allocating said dynamically reconfigurable heterogeneous processor to said microprocessors that requires assistance, said work control logic unit changes said routing paths connected with said dynamically reconfigurable heterogeneous processor and

said microprocessors whereby said dynamically reconfigurable heterogeneous processor is dynamically allocated to assist said microprocessors that require assistance.

15. The dynamic allocation method with load balancing according to claim **11**, wherein said ready signal and said control signals are used together to reconfigure said dynamically reconfigurable heterogeneous processor.

* * * * *