

(19) World Intellectual Property Organization
International Bureau



(43) International Publication Date
22 June 2006 (22.06.2006)

PCT

(10) International Publication Number
WO 2006/064962 A1

(51) International Patent Classification:
G06F 13/40 (2006.01) **G06F 15/167** (2006.01)
G06F 5/06 (2006.01)

(21) International Application Number:
PCT/JP2005/023430

(22) International Filing Date:
14 December 2005 (14.12.2005)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
11/014,226 15 December 2004 (15.12.2004) US

(71) Applicant (for all designated States except US): **SONY COMPUTER ENTERTAINMENT INC.** [JP/JP];
2-6-21, Minami-Aoyama, Minato-ku, Tokyo, 1070062 (JP).

(72) Inventor: **YAMAZAKI, Takeshi**; c/o SONY COMPUTER ENTERTAINMENT INC., 2-6-21, Minami-Aoyama, Minato-ku, Tokyo, 1070062 (JP).

(74) Agent: **MORISHITA, Sakaki**; 2-11-12, Ebisu-Nishi, Shibuya-ku, Tokyo, 1500021 (JP).

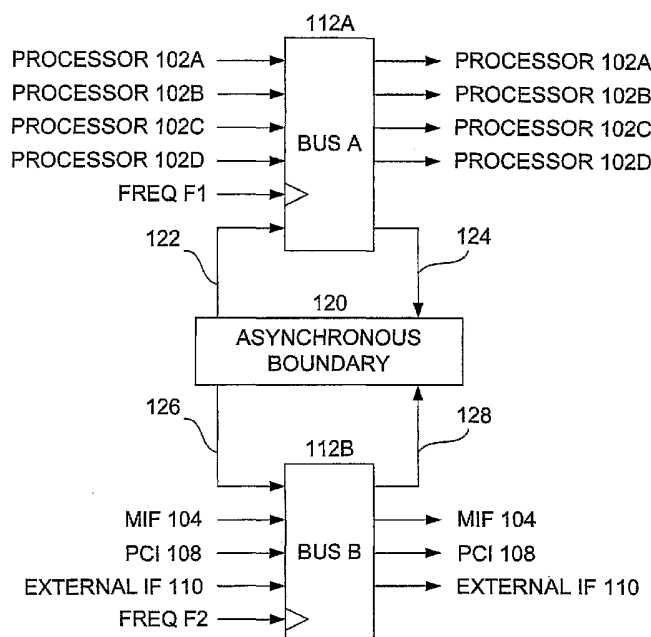
(81) Designated States (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LC,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

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

Published:
— with international search report

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

(54) Title: METHODS AND APPARATUS FOR PROVIDING AN ASYNCHRONOUS BOUNDARY BETWEEN INTERNAL BUSES IN A MULTI-PROCESSOR DEVICE



(57) Abstract: Methods and apparatus provide for transferring data to and from one or more processors of a multi-processor system over a first bus at a first frequency; transferring data to and from one or more interface circuits over a second bus at a second frequency; and isolating the first and second busses by transferring data between the first and second busses at the respective first and second frequencies with the help of an asynchronous boundary circuit.

WO 2006/064962 A1

DESCRIPTION

**METHODS AND APPARATUS FOR PROVIDING AN ASYNCHRONOUS
BOUNDARY BETWEEN INTERNAL BUSES IN A MULTI-PROCESSOR DEVICE**

5

TECHNICAL FIELD

The present invention relates to methods and apparatus for intra-processing system data transfers in multi-processing system.

10

RELATED ART

In recent years, there has been an insatiable desire for faster computer processing data throughputs because cutting-edge computer applications involve real-time, multimedia functionality. Graphics applications are among those that place the highest demands on a processing system because they require such vast numbers of data accesses, data computations, and data manipulations in relatively short periods of time to achieve desirable visual results. These applications require extremely fast processing speeds, such as many thousands of megabits of data per second. While some processing systems employ a single processor to achieve fast processing speeds, others are implemented utilizing multi-

processor architectures. In multi-processor systems, a plurality of sub-processors can operate in parallel (or at least in concert) to achieve desired processing results.

For example, a multi-processor system may include a plurality of processors all sharing a common system memory, where each processor also has a local memory in which to execute instructions. The multi-processor system may also include one or more interfaces, for example, to external devices, to the shared memory, etc. The data traffic among the processors and the interfaces may be supported by a common data bus. While this configuration has many advantages, it has been found to be beneficial to operate the processors at a different frequency than the interfaces. Indeed, when the processors operate at a lower frequency, the power dissipation in the system reduces significantly. Unfortunately, if the frequency of the common data bus were lowered to achieve lower power dissipation, the resultant reduction in memory bandwidth, for example, as seen by interfaces, external devices, etc. would be undesirable.

20

DISCLOSURE OF THE INVENTION

The present invention provides for an asynchronous

boundary between two separate busses in a multi-processor system. The processors communicate over the first bus and the interfaces communicate over the second bus. By reducing the frequency of operation of the first bus, the aforementioned power dissipation goals may be achieved without impacting the memory bandwidth, etc. as seen by the interfaces and/or other devices.

In accordance with one or more features described herein, an apparatus includes: a first bus operatively coupled to one or more processors of a multi-processor system, the first bus being adapted to transfer data to and from the processors at a first frequency; a second bus operatively coupled to one or more interface circuits, the second bus being adapted to transfer data to and from the interface circuits at a second frequency; and an asynchronous boundary circuit operable to transfer data between the first and second busses at the respective first and second frequencies.

The first frequency is preferably lower than the second frequency.

The asynchronous boundary circuit may include at least a first asynchronous FIFO and a second asynchronous FIFO, where the first asynchronous FIFO is operable to receive data from the second bus at the second frequency and to

transfer the data to the first bus at the first frequency, and the second asynchronous FIFO is operable to receive data from the first bus at the first frequency and to transfer the data to the second bus at the second frequency.

5 The one or more interface circuits may include: a memory interface circuit operable to facilitate data transfers between the one or more processors and a shared memory; an external interface circuit operable to facilitate data transfers between the multi-processor system and an external
10 device; a peripheral component interconnect circuit operable to facilitate data transfers between the multi-processor system and a peripheral device; and/or any other interface circuit.

 In accordance with one or more features described
15 herein, an apparatus includes: a plurality of processors capable of being coupled to a memory that is operable to store data; a first bus operable to couple the processors together and to transfer data to and from the processors at a first frequency; a second bus operatively coupled to at least a
20 memory interface circuit, the memory interface circuit being adapted to facilitate data transfers between the processors and the memory, and the second bus being adapted to transfer data to and from the memory interface circuit at a second

frequency; and an asynchronous boundary circuit operable to transfer data between the first and second busses at the respective first and second frequencies.

The apparatus may also include a respective local
5 memory that is not a hardware cache memory coupled to each of the processors, each local memory being adapted to execute instructions therein. The apparatus may also include the shared memory. Preferably, the processors and the local memories are disposed on a common semiconductor substrate.
10 Alternatively, the processors, the associated local memories, and the memory may be disposed on a common semiconductor substrate.

The apparatus may also include a main processor operatively coupled to the processors over the first bus,
15 which transfers data to and from the main processor at the first frequency. A hardware cache memory may be associated with the main processor and operable cache data obtained from at least one of the memory and one or more of the local memories of the processors. The first bus is preferably
20 adapted to transfer data to and from the cache memory processor at the first frequency. The processors, the local memories, and the main processor are preferably disposed on a common semiconductor substrate.

In accordance with one or more further features described herein, a method includes: transferring data to and from one or more processors of a multi-processor system over a first bus at a first frequency; transferring data to and from
5 one or more interface circuits over a second bus at a second frequency; and isolating the first and second busses by transferring data between the first and second busses at the respective first and second frequencies. The first frequency is preferably lower than the second frequency.

10 The isolating step may include: receiving data from the second bus at the second frequency and transferring the data to the first bus at the first frequency; and receiving data from the first bus at the first frequency and
transferring the data to the second bus at the second
15 frequency.

The method may also include transferring data to and from: (i) a memory interface circuit over the second bus at the second frequency, the memory interface circuit being adapted to facilitate data transfers between the processors
20 and a shared memory; (ii) an external interface circuit over the second bus at the second frequency, the external interface circuit being operable to facilitate data transfers between the processors and an external device; (iii) a peripheral

component interconnect circuit, the peripheral component interconnect circuit being operable to facilitate data transfers between the multi-processor system and a peripheral device; (iv) and/or any other interface circuit.

5

BRIEF DESCRIPTION OF THE DRAWINGS

For the purposes of illustrating the various aspects of the invention, there are shown in the drawings forms that are presently preferred, it being understood, however, that the invention is not limited to the precise arrangements and instrumentalities shown.

FIG. 1 is a block diagram illustrating the structure of a multi-processing system having two or more sub-processors accordance with one or more aspects of the present invention;

FIG. 2 is a diagram illustrating a bus configuration for the processing system of FIG. 1;

FIG. 3 is a block diagram illustrating further details of the bus configuration for the processing system;

FIG. 4 is a schematic diagram illustrating further details of the bus configuration for the processing system;

FIG. 5 is a diagram illustrating a preferred

processor element (PE) that may be used to implement one or more further aspects of the present invention;

FIG. 6 is a diagram illustrating the structure of an exemplary sub-processing unit (SPU) of the system of FIG. 5 in accordance with one or more further aspects of the present invention; and

FIG. 7 is a diagram illustrating the structure of an exemplary processing unit (PU) of the system of FIG. 5 in accordance with one or more further aspects of the present invention.

BEST MODE FOR CARRYING OUT THE INVENTION

With reference to the drawings, wherein like numerals indicate like elements, there is shown in FIG. 1 a processing system 100 suitable for implementing one or more features of the present invention. For the purposes of brevity and clarity, the block diagram of FIG. 1 will be referred to and described herein as illustrating an apparatus 100, it being understood, however, that the description may readily be applied to various aspects of a method with equal force.

The processing system 100 includes a plurality of

processors 102A, 102B, 102C, 102D, it being understood that any number of processors 102 may be employed. The system 100 also includes a memory interface circuit 104, a shared memory 106, a peripheral component interconnect circuit (PCI) 108 and
5 an external interface circuit 110. These components are preferably coupled to one another over a bus system 112 that is operable to transfer data to and from each component in accordance with suitable protocols.

Each of the processors 102 may be of similar
10 construction or of differing construction. The processors 102 may be implemented utilizing any of the known technologies that are capable of requesting data from the system memory 106, and manipulating the data to achieve a desirable result. For example, the processors 102 may be implemented using any of
15 the known microprocessors that are capable of executing software and/or firmware, including standard microprocessors, distributed microprocessors, etc. By way of example, one or more of the processors 102 may be a graphics processor that is capable of requesting and manipulating data, such as pixel
20 data, including gray scale information, color information, texture data, polygonal information, video frame information, etc.

One or more of the processors 102 of the system 100

may take on the role as a main (or managing) processor. The main processor may schedule and orchestrate the processing of data by the other processors 102.

The memory interface circuit 104 is preferably
5 operable to facilitate data transfers between the processors 102 and the shared memory 106 such that the processors 102 may execute application programs and the like. By way of example, the memory interface circuit 104 may provide one or two high-bandwidth channels into the shared memory and may be adapted
10 to be a slave to the bus system 112. Any of the known memory interface technologies may be employed to implement the memory interface circuit 104.

The system memory 106 is preferably a dynamic random access memory (DRAM) coupled to the processors 102 through the
15 memory interface circuit 104. Although the system memory 106 is preferably a DRAM, the memory 106 may be implemented using other means, e.g., a static random access memory (SRAM), a magnetic random access memory (MRAM), an optical memory, a holographic memory, etc.

20 The peripheral component interconnect circuit 108 provides an interconnection scheme between the processors 102 and one or more peripheral devices, such as a printer, a monitor, etc. over a communications channel 116. Any of the

known PCI technologies may be employed to implement the peripheral component interconnect circuit 108.

The external interface circuit 110 is operable to facilitate data transfers between the system 100 and one or
5 more external devices over a communications channel 118. Preferably, the external interface circuit is adapted to exchange non-coherent traffic with an external device and/or operate coherently by extending the bus system 112 to another, external device, such as another processing system. Although
10 any of the known external interface technologies may be employed to implement the external interface circuit 110, it is preferred that the circuit combines command and data into packetized envelopes and insures successful delivery of the envelopes to/from the external device.

15 Turning again to the processors, each processor 102 preferably includes a processor core and a local memory in which to execute programs. These components may be integrally disposed on a common semi-conductor substrate or may be separately disposed as may be desired by a designer. The
20 processor core is preferably implemented using a processing pipeline, in which logic instructions are processed in a pipelined fashion. Although the pipeline may be divided into any number of stages at which instructions are processed, the

pipeline generally comprises fetching one or more instructions, decoding the instructions, checking for dependencies among the instructions, issuing the instructions, and executing the instructions. In this regard, the processor
5 core 102A may include an instruction buffer, instruction decode circuitry, dependency check circuitry, instruction issue circuitry, and execution stages.

The local memory is coupled to the processor core via a bus and is preferably located on the same chip (same
10 semiconductor substrate) as the processor core. The local memory is preferably not a traditional hardware cache memory in that there are no on-chip or off-chip hardware cache circuits, cache registers, cache memory controllers, etc. to implement a hardware cache memory function. As on chip space
15 is often limited, the size of the local memory may be much smaller than the shared memory 106.

The processors preferably provide data access requests to copy data (which may include program data) from the system memory 106 over the bus system 112 into their
20 respective local memories for program execution and data manipulation. The mechanism for facilitating data access may be implemented utilizing any of the known techniques, for example the direct memory access (DMA) technique.

Reference is now made to FIG. 2, which is a simplified block diagram of the bus system 112. As shown, the bus system 112 is operable to receive data from the processors 102, the memory interface circuit 104, the peripheral component interconnect circuit 108, and the external interface circuit 110. The bus system 112 is also operable to transmit data to the processors 102, the memory interface circuit 104, the peripheral component interconnect circuit 108, and the external interface circuit 110. The bus system 112 manages the transfer of data between these components to achieve the desired data flow.

With reference to FIG. 3, the bus system 112 preferably includes a first bus 112A and a second bus 112B operatively coupled and isolated from one another by way of an asynchronous boundary circuit 120. The first bus 112A is operatively coupled to the processors 102 such that data may be transferred to and from the processors 102 at a first frequency F1. The second bus 112B is operatively coupled to the memory interface circuit 104, the peripheral component interconnect circuit 108, and the external interface circuit 110 such that data may be transferred to and from the interface circuits at a second frequency F2.

The asynchronous boundary circuit 120 is operable to

transfer data between the first and second busses 112A, 112B at the respective first and second frequencies. Preferably, the first frequency is lower than the second frequency such that the frequency of operation of the processors 102 may be
5 reduced and power dissipation may likewise be reduced. Advantageously, however, the frequency of operation of the interface circuits need not be reduced.

Reference is now made to FIG. 4, which is block diagram of a suitable circuit configuration that is capable of
10 carrying out the functionality of the asynchronous boundary circuit 120. The circuit 120 may include at least a first asynchronous FIFO 202 and a second asynchronous FIFO 204 each including dual clock inputs. One clock input of each asynchronous FIFO receives the first frequency F1, which
15 matches the frequency of operation of the first bus 112A. The other frequency input of each FIFO receives the second frequency F2, which matches the frequency of operation of the second bus 112B. As the general operation of an asynchronous FIFO is well known, it suffices to say that the first
20 asynchronous FIFO 202 is preferably operable to receive data from the second bus 112B over line 128 at the second frequency and to transfer the data to the first bus 112A at the first frequency over line 122. Similarly, the second asynchronous FIFO 204 is preferably operable to receive data from the first

bus 112A at the first frequency over line 124 and to transfer the data to the second bus 112B at the second frequency over line 126.

In an alternative embodiment, the system 100 may
5 include a main processor (not shown) operatively coupled to the other processors 102 and capable of being coupled to the shared memory 106 over the bus system 112. The main processor may schedule and orchestrate the processing of data by the other processors 102. Unlike the other processors 102,
10 however, the main processor may be coupled to a hardware cache memory, which is operable cache data obtained from at least one of the shared memory 106 and one or more of the local memories of the processors 102. The main processor may provide data access requests to copy data (which may include program
15 data) from the system memory 106 over the bus system 112 into the cache memory for program execution and data manipulation utilizing any of the known techniques, such as DMA techniques.

A description of a preferred computer architecture for a multi-processor system will now be provided that is
20 suitable for carrying out one or more of the features discussed herein. In accordance with one or more embodiments, the multi-processor system may be implemented as a single-chip solution operable for stand-alone and/or distributed

processing of media-rich applications, such as game systems, home terminals, PC systems, server systems and workstations. In some applications, such as game systems and home terminals, real-time computing may be a necessity. For example, in a
5 real-time, distributed gaming application, one or more of networking image decompression, 3D computer graphics, audio generation, network communications, physical simulation, and artificial intelligence processes have to be executed quickly enough to provide the user with the illusion of a real-time
10 experience. Thus, each processor in the multi-processor system must complete tasks in a short and predictable time.

To this end, and in accordance with this computer architecture, all processors of a multi-processing computer system are constructed from a common computing module (or
15 cell). This common computing module has a consistent structure and preferably employs the same instruction set architecture. The multi-processing computer system can be formed of one or more clients, servers, PCs, mobile computers, game machines, PDAs, set top boxes, appliances, digital televisions and other
20 devices using computer processors.

A plurality of the computer systems may also be members of a network if desired. The consistent modular structure enables efficient, high speed processing of

applications and data by the multi-processing computer system, and if a network is employed, the rapid transmission of applications and data over the network. This structure also simplifies the building of members of the network of various sizes and processing power and the preparation of applications for processing by these members.

With reference to FIG. 5, the basic processing module is a processor element (PE) 500. The PE 500 comprises an I/O interface 502, a processing unit (PU) 504, and a plurality of sub-processing units 508, namely, sub-processing unit 508A, sub-processing unit 508B, sub-processing unit 508C, and sub-processing unit 508D. A local (or internal) PE bus 512 transmits data and applications among the PU 504, the sub-processing units 508, and a memory interface 511. The local PE bus 512 can have, e.g., a conventional architecture or can be implemented as a packet-switched network. If implemented as a packet switch network, while requiring more hardware, increases the available bandwidth.

The PE 500 can be constructed using various methods for implementing digital logic. The PE 500 preferably is constructed, however, as a single integrated circuit employing a complementary metal oxide semiconductor (CMOS) on a silicon substrate. Alternative materials for substrates include

gallium arsinide, gallium aluminum arsinide and other so-called III-B compounds employing a wide variety of dopants. The PE 500 also may be implemented using superconducting material, e.g., rapid single-flux-quantum (RSFQ) logic.

5 The PE 500 is closely associated with a shared (main) memory 514 through a high bandwidth memory connection 516. Although the memory 514 preferably is a dynamic random access memory (DRAM), the memory 514 could be implemented using other means, e.g., as a static random access memory
10 (SRAM), a magnetic random access memory (MRAM), an optical memory, a holographic memory, etc.

 The PU 504 and the sub-processing units 508 are preferably each coupled to a memory flow controller (MFC) including direct memory access DMA functionality, which in
15 combination with the memory interface 511, facilitate the transfer of data between the DRAM 514 and the sub-processing units 508 and the PU 504 of the PE 500. It is noted that the DMAC and/or the memory interface 511 may be integrally or separately disposed with respect to the sub-processing units
20 508 and the PU 504. Indeed, the DMAC function and/or the memory interface 511 function may be integral with one or more (preferably all) of the sub-processing units 508 and the PU 504. It is also noted that the DRAM 514 may be integrally or

separately disposed with respect to the PE 500. For example, the DRAM 514 may be disposed off-chip as is implied by the illustration shown or the DRAM 514 may be disposed on-chip in an integrated fashion.

5 The PU 504 can be, e.g., a standard processor capable of stand-alone processing of data and applications. In operation, the PU 504 preferably schedules and orchestrates the processing of data and applications by the sub-processing units. The sub-processing units preferably are single
10 instruction, multiple data (SIMD) processors. Under the control of the PU 504, the sub-processing units perform the processing of these data and applications in a parallel and independent manner. The PU 504 is preferably implemented using a PowerPC core, which is a microprocessor architecture that
15 employs reduced instruction-set computing (RISC) technique. RISC performs more complex instructions using combinations of simple instructions. Thus, the timing for the processor may be based on simpler and faster operations, enabling the microprocessor to perform more instructions for a given clock
20 speed.

It is noted that the PU 504 may be implemented by one of the sub-processing units 508 taking on the role of a main processing unit that schedules and orchestrates the

processing of data and applications by the sub-processing units 508. Further, there may be more than one PU implemented within the processor element 500.

In accordance with this modular structure, the
5 number of PEs 500 employed by a particular computer system is based upon the processing power required by that system. For example, a server may employ four PEs 500, a workstation may employ two PEs 500 and a PDA may employ one PE 500. The number of sub-processing units of a PE 500 assigned to processing a
10 particular software cell depends upon the complexity and magnitude of the programs and data within the cell.

FIG. 6 illustrates the preferred structure and function of a sub-processing unit (SPU) 508. The SPU 508 architecture preferably fills a void between general-purpose
15 processors (which are designed to achieve high average performance on a broad set of applications) and special-purpose processors (which are designed to achieve high performance on a single application). The SPU 508 is designed to achieve high performance on game applications, media
20 applications, broadband systems, etc., and to provide a high degree of control to programmers of real-time applications. Some capabilities of the SPU 508 include graphics geometry pipelines, surface subdivision, Fast Fourier Transforms, image

processing keywords, stream processing, MPEG encoding/decoding, encryption, decryption, device driver extensions, modeling, game physics, content creation, and audio synthesis and processing.

5 The sub-processing unit 508 includes two basic functional units, namely an SPU core 510A and a memory flow controller (MFC) 510B. The SPU core 510A performs program execution, data manipulation, etc., while the MFC 510B performs functions related to data transfers between the SPU
10 core 510A and the DRAM 514 of the system.

 The SPU core 510A includes a local memory 550, an instruction unit (IU) 552, registers 554, one ore more floating point execution stages 556 and one or more fixed point execution stages 558. The local memory 550 is preferably
15 implemented using single-ported random access memory, such as an SRAM. Whereas most processors reduce latency to memory by employing caches, the SPU core 510A implements the relatively small local memory 550 rather than a cache. Indeed, in order to provide consistent and predictable memory access latency
20 for programmers of real-time applications (and other applications as mentioned herein) a cache memory architecture within the SPU 508A is not preferred. The cache hit/miss characteristics of a cache memory results in volatile memory

access times, varying from a few cycles to a few hundred cycles. Such volatility undercuts the access timing predictability that is desirable in, for example, real-time application programming. Latency hiding may be achieved in the
5 local memory SRAM 550 by overlapping DMA transfers with data computation. This provides a high degree of control for the programming of real-time applications. As the latency and instruction overhead associated with DMA transfers exceeds that of the latency of servicing a cache miss, the SRAM local
10 memory approach achieves an advantage when the DMA transfer size is sufficiently large and is sufficiently predictable (e.g., a DMA command can be issued before data is needed).

A program running on a given one of the sub-processing units 508 references the associated local memory
15 550 using a local address, however, each location of the local memory 550 is also assigned a real address (RA) within the overall system's memory map. This allows Privilege Software to map a local memory 550 into the Effective Address (EA) of a process to facilitate DMA transfers between one local memory
20 550 and another local memory 550. The PU 504 can also directly access the local memory 550 using an effective address. In a preferred embodiment, the local memory 550 contains 256 kilobytes of storage, and the capacity of registers 552 is 128

X 128 bits.

The SPU core 510A is preferably implemented using a processing pipeline, in which logic instructions are processed in a pipelined fashion. Although the pipeline may be divided
5 into any number of stages at which instructions are processed, the pipeline generally comprises fetching one or more instructions, decoding the instructions, checking for dependencies among the instructions, issuing the instructions, and executing the instructions. In this regard, the IU 552
10 includes an instruction buffer, instruction decode circuitry, dependency check circuitry, and instruction issue circuitry.

The instruction buffer preferably includes a plurality of registers that are coupled to the local memory
550 and operable to temporarily store instructions as they are
15 fetched. The instruction buffer preferably operates such that all the instructions leave the registers as a group, i.e., substantially simultaneously. Although the instruction buffer may be of any size, it is preferred that it is of a size not larger than about two or three registers.

20 In general, the decode circuitry breaks down the instructions and generates logical micro-operations that perform the function of the corresponding instruction. For example, the logical micro-operations may specify arithmetic

and logical operations, load and store operations to the local memory 550, register source operands and/or immediate data operands. The decode circuitry may also indicate which resources the instruction uses, such as target register addresses, structural resources, function units and/or busses. The decode circuitry may also supply information indicating the instruction pipeline stages in which the resources are required. The instruction decode circuitry is preferably operable to substantially simultaneously decode a number of instructions equal to the number of registers of the instruction buffer.

The dependency check circuitry includes digital logic that performs testing to determine whether the operands of given instruction are dependent on the operands of other instructions in the pipeline. If so, then the given instruction should not be executed until such other operands are updated (e.g., by permitting the other instructions to complete execution). It is preferred that the dependency check circuitry determines dependencies of multiple instructions dispatched from the decoder circuitry simultaneously.

The instruction issue circuitry is operable to issue the instructions to the floating point execution stages 556 and/or the fixed point execution stages 558.

The registers 554 are preferably implemented as a relatively large unified register file, such as a 128-entry register file. This allows for deeply pipelined high-frequency implementations without requiring register renaming to avoid
5 register starvation. Renaming hardware typically consumes a significant fraction of the area and power in a processing system. Consequently, advantageous operation may be achieved when latencies are covered by software loop unrolling or other interleaving techniques.

10 Preferably, the SPU core 510A is of a superscalar architecture, such that more than one instruction is issued per clock cycle. The SPU core 510A preferably operates as a superscalar to a degree corresponding to the number of simultaneous instruction dispatches from the instruction
15 buffer, such as between 2 and 3 (meaning that two or three instructions are issued each clock cycle). Depending upon the required processing power, a greater or lesser number of floating point execution stages 556 and fixed point execution stages 558 may be employed. In a preferred embodiment, the
20 floating point execution stages 556 operate at a speed of 32 billion floating point operations per second (32 GFLOPS), and the fixed point execution stages 558 operate at a speed of 32 billion operations per second (32 GOPS).

The MFC 510B preferably includes a bus interface unit (BIU) 564, a memory management unit (MMU) 562, and a direct memory access controller (DMAC) 560. With the exception of the DMAC 560, the MFC 510B preferably runs at half
5 frequency (half speed) as compared with the SPU core 510A and the bus 512 to meet low power dissipation design objectives. The MFC 510B is operable to handle data and instructions coming into the SPU 508 from the bus 512, provides address translation for the DMAC, and snoop-operations for data
10 coherency. The BIU 564 provides an interface between the bus 512 and the MMU 562 and DMAC 560. Thus, the SPU 508 (including the SPU core 510A and the MFC 510B) and the DMAC 560 are connected physically and/or logically to the bus 512.

The MMU 562 is preferably operable to translate
15 effective addresses (taken from DMA commands) into real addresses for memory access. For example, the MMU 562 may translate the higher order bits of the effective address into real address bits. The lower-order address bits, however, are preferably untranslatable and are considered both logical and
20 physical for use to form the real address and request access to memory. In one or more embodiments, the MMU 562 may be implemented based on a 64-bit memory management model, and may provide 2^{64} bytes of effective address space with 4K-, 64K-, 1M-, and 16M- byte page sizes and 256MB segment sizes.

Preferably, the MMU 562 is operable to support up to 2^{65} bytes of virtual memory, and 2^{42} bytes (4 TeraBytes) of physical memory for DMA commands. The hardware of the MMU 562 may include an 8-entry, fully associative SLB, a 256-entry, 4way set associative TLB, and a 4x4 Replacement Management Table (RMT) for the TLB - used for hardware TLB miss handling.

The DMAC 560 is preferably operable to manage DMA commands from the SPU core 510A and one or more other devices such as the PU 504 and/or the other SPUs. There may be three categories of DMA commands: Put commands, which operate to move data from the local memory 550 to the shared memory 514; Get commands, which operate to move data into the local memory 550 from the shared memory 514; and Storage Control commands, which include SLI commands and synchronization commands. The synchronization commands may include atomic commands, send signal commands, and dedicated barrier commands. In response to DMA commands, the MMU 562 translates the effective address into a real address and the real address is forwarded to the BIU 564.

The SPU core 510A preferably uses a channel interface and data interface to communicate (send DMA commands, status, etc.) with an interface within the DMAC 560. The SPU core 510A dispatches DMA commands through the channel

interface to a DMA queue in the DMAC 560. Once a DMA command is in the DMA queue, it is handled by issue and completion logic within the DMAC 560. When all bus transactions for a DMA command are finished, a completion signal is sent back to the SPU core 510A over the channel interface.

FIG. 7 illustrates the preferred structure and function of the PU 504. The PU 504 includes two basic functional units, the PU core 504A and the memory flow controller (MFC) 504B. The PU core 504A performs program execution, data manipulation, multi-processor management functions, etc., while the MFC 504B performs functions related to data transfers between the PU core 504A and the memory space of the system 100.

The PU core 504A may include an L1 cache 570, an instruction unit 572, registers 574, one or more floating point execution stages 576 and one or more fixed point execution stages 578. The L1 cache provides data caching functionality for data received from the shared memory 106, the processors 102, or other portions of the memory space through the MFC 504B. As the PU core 504A is preferably implemented as a superpipeline, the instruction unit 572 is preferably implemented as an instruction pipeline with many stages, including fetching, decoding, dependency checking,

issuing, etc. The PU core 504A is also preferably of a superscalar configuration, whereby more than one instruction is issued from the instruction unit 572 per clock cycle. To achieve a high processing power, the floating point execution stages 576 and the fixed point execution stages 578 include a plurality of stages in a pipeline configuration. Depending upon the required processing power, a greater or lesser number of floating point execution stages 576 and fixed point execution stages 578 may be employed.

10 The MFC 504B includes a bus interface unit (BIU) 580, an L2 cache memory, a non-cachable unit (NCU) 584, a core interface unit (CIU) 586, and a memory management unit (MMU) 588. Most of the MFC 504B runs at half frequency (half speed) as compared with the PU core 504A and the bus 512 to meet low
15 power dissipation design objectives.

 The BIU 580 provides an interface between the bus 512 and the L2 cache 582 and NCU 584 logic blocks. To this end, the BIU 580 may act as a Master as well as a Slave device on the bus 512 in order to perform fully coherent memory
20 operations. As a Master device it may source load/store requests to the bus 512 for service on behalf of the L2 cache 582 and the NCU 584. The BIU 580 may also implement a flow control mechanism for commands which limits the total number

of commands that can be sent to the bus 512. The data operations on the bus 512 may be designed to take eight beats and, therefore, the BIU 580 is preferably designed around 128 byte cache-lines and the coherency and synchronization
5 granularity is 128KB.

The L2 cache memory 582 (and supporting hardware logic) is preferably designed to cache 512KB of data. For example, the L2 cache 582 may handle cacheable loads/stores, data pre-fetches, instruction fetches, instruction pre-
10 fetches, cache operations, and barrier operations. The L2 cache 582 is preferably an 8-way set associative system. The L2 cache 582 may include six reload queues matching six (6) castout queues (e.g., six RC machines), and eight (64-byte wide) store queues. The L2 cache 582 may operate to provide a
15 backup copy of some or all of the data in the L1 cache 570. Advantageously, this is useful in restoring state(s) when processing nodes are hot-swapped. This configuration also permits the L1 cache 570 to operate more quickly with fewer ports, and permits faster cache-to-cache transfers (because
20 the requests may stop at the L2 cache 582). This configuration also provides a mechanism for passing cache coherency management to the L2 cache memory 582.

The NCU 584 interfaces with the CIU 586, the L2

cache memory 582, and the BIU 580 and generally functions as a queueing/buffering circuit for non-cacheable operations between the PU core 504A and the memory system. The NCU 584 preferably handles all communications with the PU core 504A that are not handled by the L2 cache 582, such as cache-inhibited load/stores, barrier operations, and cache coherency operations. The NCU 584 is preferably run at half speed to meet the aforementioned power dissipation objectives.

The CIU 586 is disposed on the boundary of the MFC 504B and the PU core 504A and acts as a routing, arbitration, and flow control point for requests coming from the execution stages 576, 578, the instruction unit 572, and the MMU unit 588 and going to the L2 cache 582 and the NCU 584. The PU core 504A and the MMU 588 preferably run at full speed, while the L2 cache 582 and the NCU 584 are operable for a 2:1 speed ratio. Thus, a frequency boundary exists in the CIU 586 and one of its functions is to properly handle the frequency crossing as it forwards requests and reloads data between the two frequency domains.

The CIU 586 is comprised of three functional blocks: a load unit, a store unit, and reload unit. In addition, a data pre-fetch function is performed by the CIU 586 and is preferably a functional part of the load unit. The CIU 586 is

preferably operable to: (i) accept load and store requests from the PU core 504A and the MMU 588; (ii) convert the requests from full speed clock frequency to half speed (a 2:1 clock frequency conversion); (iii) route cachable requests to the L2 cache 582, and route non-cachable requests to the NCU 584; (iv) arbitrate fairly between the requests to the L2 cache 582 and the NCU 584; (v) provide flow control over the dispatch to the L2 cache 582 and the NCU 584 so that the requests are received in a target window and overflow is avoided; (vi) accept load return data and route it to the execution stages 576, 578, the instruction unit 572, or the MMU 588; (vii) pass snoop requests to the execution stages 576, 578, the instruction unit 572, or the MMU 588; and (viii) convert load return data and snoop traffic from half speed to full speed.

The MMU 588 preferably provides address translation for the PU core 540A, such as by way of a second level address translation facility. A first level of translation is preferably provided in the PU core 504A by separate instruction and data ERAT (effective to real address translation) arrays that may be much smaller and faster than the MMU 588.

In a preferred embodiment, the PU 504 operates at 4

- 6 GHz, 10F04, with a 64-bit implementation. The registers are preferably 64 bits long (although one or more special purpose registers may be smaller) and effective addresses are 64 bits long. The instruction unit 570, registers 572 and
5 execution stages 574 and 576 are preferably implemented using PowerPC technology to achieve the (RISC) computing technique.

Additional details regarding the modular structure of this computer system may be found in U.S. Patent No. 6,526,491, the entire disclosure of which is hereby
10 incorporated by reference.

The processor element or a basic processing module has the modular structure and may have any number of sub-processing units therein. In accordance with processing performance demanded from various types of information
15 equipment with the processor element installed, such as a PC, a server, a portable device, a game console, a household electrical appliance, or the like, the number of sub-processing units may be configured. In addition, one or more processor elements may be provided in the information
20 equipment and an appropriate number of the processor elements may be chosen depending on the performance demanded from the information equipment. Furthermore, a distributed processing system comprising network-connected nodes, each of which is

equipped with one or more processor elements, may be configured. The number of the processor elements installed in each node and the number of the sub-processing units in the processor element may be designed in accordance with the performance demanded from the distributed processing system. Since the processor element has the modular structure, a system can be implemented with any number of the processor elements and any number of sub-processing units in each processor element. Therefore, the system has flexibility and scalability.

In accordance with at least one further aspect of the present invention, the methods and apparatus described above may be achieved utilizing suitable hardware, such as that illustrated in the figures. Such hardware may be implemented utilizing any of the known technologies, such as standard digital circuitry, any of the known processors that are operable to execute software and/or firmware programs, one or more programmable digital devices or systems, such as programmable read only memories (PROMs), programmable array logic devices (PALs), etc. Furthermore, although the apparatus illustrated in the figures are shown as being partitioned into certain functional blocks, such blocks may be implemented by way of separate circuitry and/or combined into one or more functional units. Still further, the various

aspects of the invention may be implemented by way of software and/or firmware program(s) that may be stored on suitable storage medium or media (such as floppy disk(s), memory chip(s), etc.) for transportability and/or distribution.

5 Although the invention herein has been described with reference to particular embodiments, it is to be understood that these embodiments are merely illustrative of the principles and applications of the present invention. It is therefore to be understood that numerous modifications may
10 be made to the illustrative embodiments and that other arrangements may be devised without departing from the spirit and scope of the present invention as defined by the appended claims.

15 INDUSTRIAL APPLICABILITY

 The present invention is applicable to a technology for intra-processing system data transfers in multi-processing system.

CLAIMS

1. An apparatus, comprising:

a first bus operatively coupled to one or more processors of a multi-processor system, the first bus being adapted to transfer data to and from the processors at a first frequency;

a second bus operatively coupled to one or more interface circuits, the second bus being adapted to transfer data to and from the interface circuits at a second frequency; and

an asynchronous boundary circuit operable to transfer data between the first and second busses at the respective first and second frequencies.

2. The apparatus of claim 1, wherein the first frequency is lower than the second frequency.

3. The apparatus of claim 1 or 2, wherein:

the asynchronous boundary circuit includes at least a first asynchronous FIFO and a second asynchronous FIFO;

the first asynchronous FIFO being operable to receive data from the second bus at the second frequency and to transfer the data to the first bus at the first frequency; and

the second asynchronous FIFO being operable to receive

data from the first bus at the first frequency and to transfer the data to the second bus at the second frequency.

4. The apparatus of any one of claims 1 to 3, wherein the one or more interface circuits includes a memory interface circuit operable to facilitate data transfers between the one or more processors and a shared memory.

5. The apparatus of any one of claims 1 to 4, wherein the one or more interface circuits includes an external interface circuit operable to facilitate data transfers between the multi-processor system and an external device.

6. The apparatus of any one of claims 1 to 5, wherein the one or more interface circuits includes a peripheral component interconnect circuit operable to facilitate data transfers between the multi-processor system and a peripheral device.

7. An apparatus, comprising:
a plurality of processors capable of being coupled to a memory that is operable to store data;
a first bus operable to couple the processors together and

to transfer data to and from the processors at a first frequency;

a second bus operatively coupled to at least a memory interface circuit, the memory interface circuit being adapted to facilitate data transfers between the processors and the memory, and the second bus being adapted to transfer data to and from the memory interface circuit at a second frequency; and

an asynchronous boundary circuit operable to transfer data between the first and second busses at the respective first and second frequencies.

8. The apparatus of claim 7, wherein the first frequency is lower than the second frequency.

9. The apparatus of claim 7 or 8, further comprising a respective local memory that is not a hardware cache memory coupled to each of the processors, each local memory being adapted to execute instructions therein.

10. The apparatus of claim 9, wherein the processors and the local memories are disposed on a common semiconductor substrate.

11. The apparatus of any one of claims 7 to 9, further comprising the memory.

12. The apparatus of claim 11, wherein the processors, the associated local memories, and the memory are disposed on a common semiconductor substrate.

13. The apparatus of claim 9 or 10, further comprising a main processor operatively coupled to the processors over the first bus, which transfers data to and from the main processor at the first frequency.

14. The apparatus of claim 13, further comprising a hardware cache memory associated with the main processor and operable cache data obtained from at least one of the memory and one or more of the local memories of the processors.

15. The apparatus of claim 14, wherein the first bus is adapted to transfer data to and from the hardware cache memory at the first frequency.

16. The apparatus of claim 14 or 15, wherein the

processors, the local memories, and the main processor are disposed on a common semiconductor substrate.

17. A system, comprising:

a plurality of processors;

a first bus operable to couple the processors together and to transfer data to and from the processors at a first frequency;

a second bus operatively coupled to at least an external interface circuit, the external interface circuit being operable to facilitate data transfers between the processors and an external device, and the second bus being adapted to transfer data to and from the external interface circuit at a second frequency; and

an asynchronous boundary circuit operable to transfer data between the first and second busses at the respective first and second frequencies.

18. The system of claim 17, wherein the first frequency is lower than the second frequency.

19. The system of claim 17 or 18, wherein the external device is a multi-processor system operating together with the

processors to execute one or more application programs.

20. The system of any one of claims 17 to 19, further comprising:

a memory shared by the processors and operable to store data; and

a memory interface circuit, the memory interface circuit being adapted to facilitate data transfers between the processors and the memory,

wherein the second bus is adapted to transfer data to and from the memory interface circuit at the second frequency.

21. A method, comprising:

transferring data to and from one or more processors of a multi-processor system over a first bus at a first frequency;

transferring data to and from one or more interface circuits over a second bus at a second frequency; and

isolating the first and second busses by transferring data between the first and second busses at the respective first and second frequencies.

22. The method of claim 21, wherein the first frequency is lower than the second frequency.

23. The method of claim 21 or 22, wherein the isolating step includes:

receiving data from the second bus at the second frequency and transferring the data to the first bus at the first frequency; and

receiving data from the first bus at the first frequency and transferring the data to the second bus at the second frequency.

24. The method of any one of claims 21 to 23, further comprising:

transferring data to and from a memory interface circuit over the second bus at the second frequency, the memory interface circuit being adapted to facilitate data transfers between the processors and a shared memory.

25. The method of any one of claims 21 to 24, further comprising transferring data to and from a main processor over the first bus at the first frequency, wherein the main processor is operable to manage the other processors.

26. The method of any one of claims 21 to 25, further

comprising:

transferring data to and from an external interface circuit over the second bus at the second frequency, the external interface circuit being operable to facilitate data transfers between the processors and an external device.

27. The method of claim 26, wherein the external device is a multi-processor system operating together with the processors to execute one or more application programs.

FIG. 1

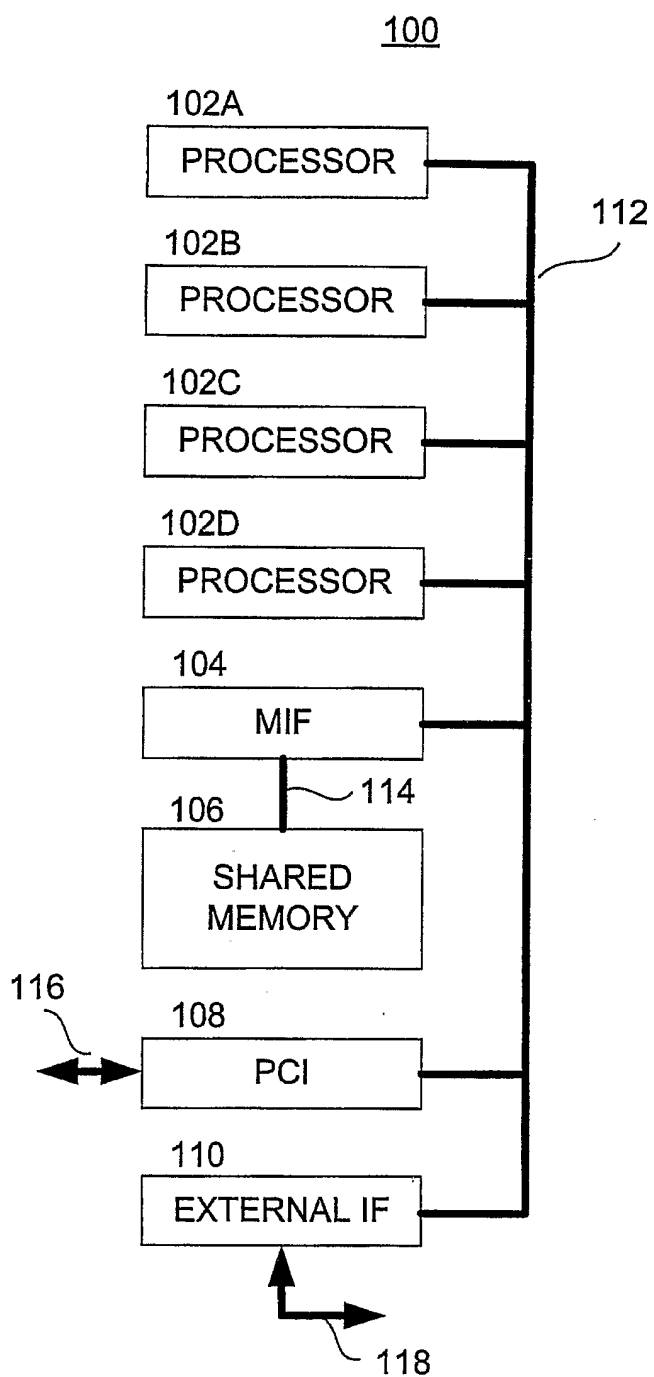


FIG. 2

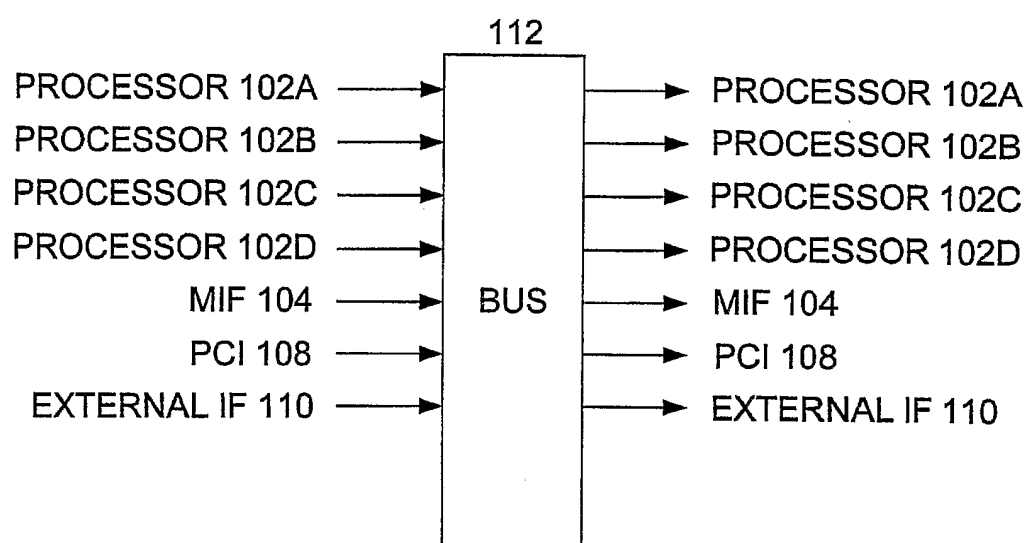


FIG. 3

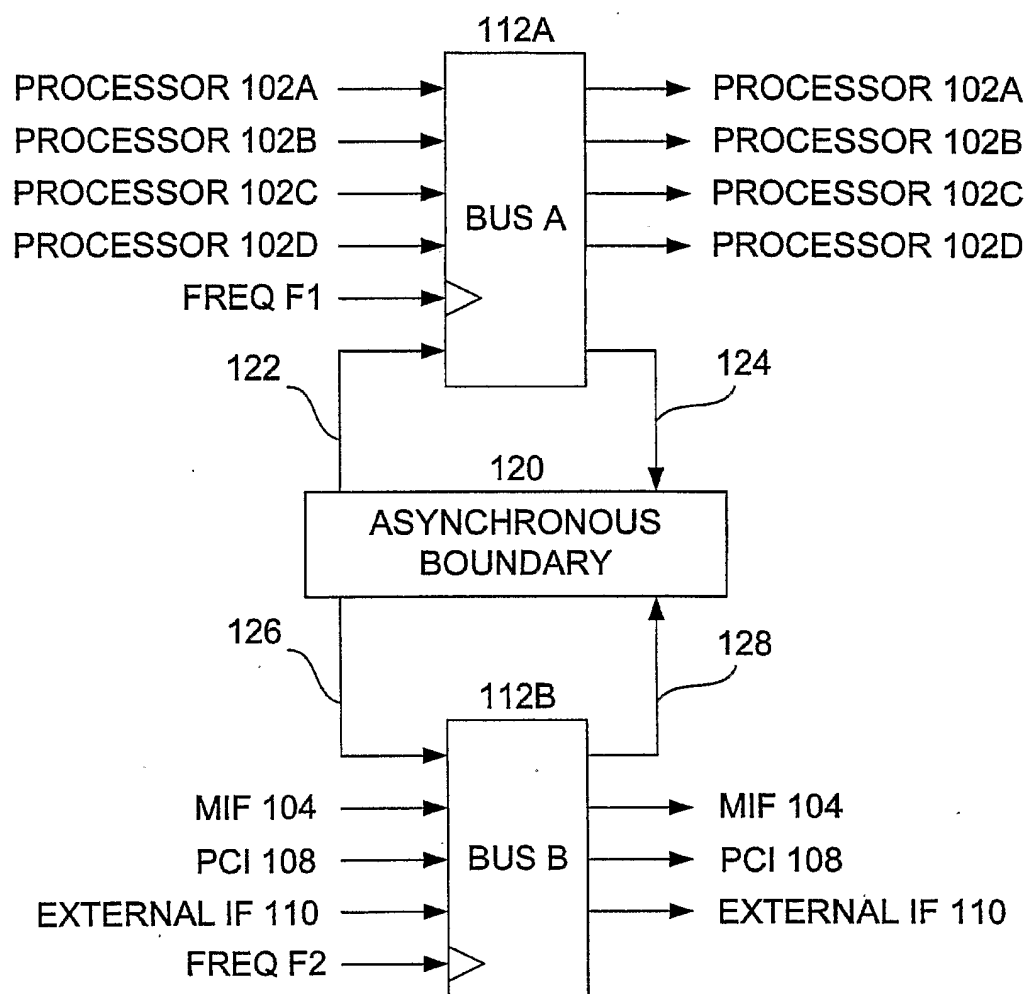


FIG. 4

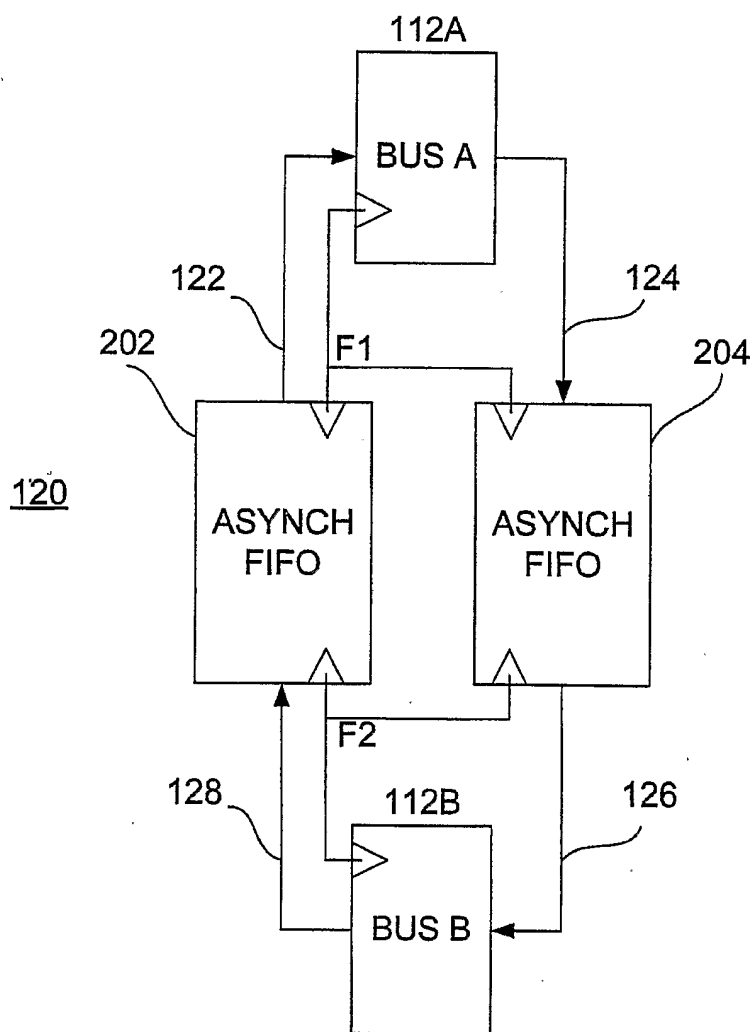


FIG. 5

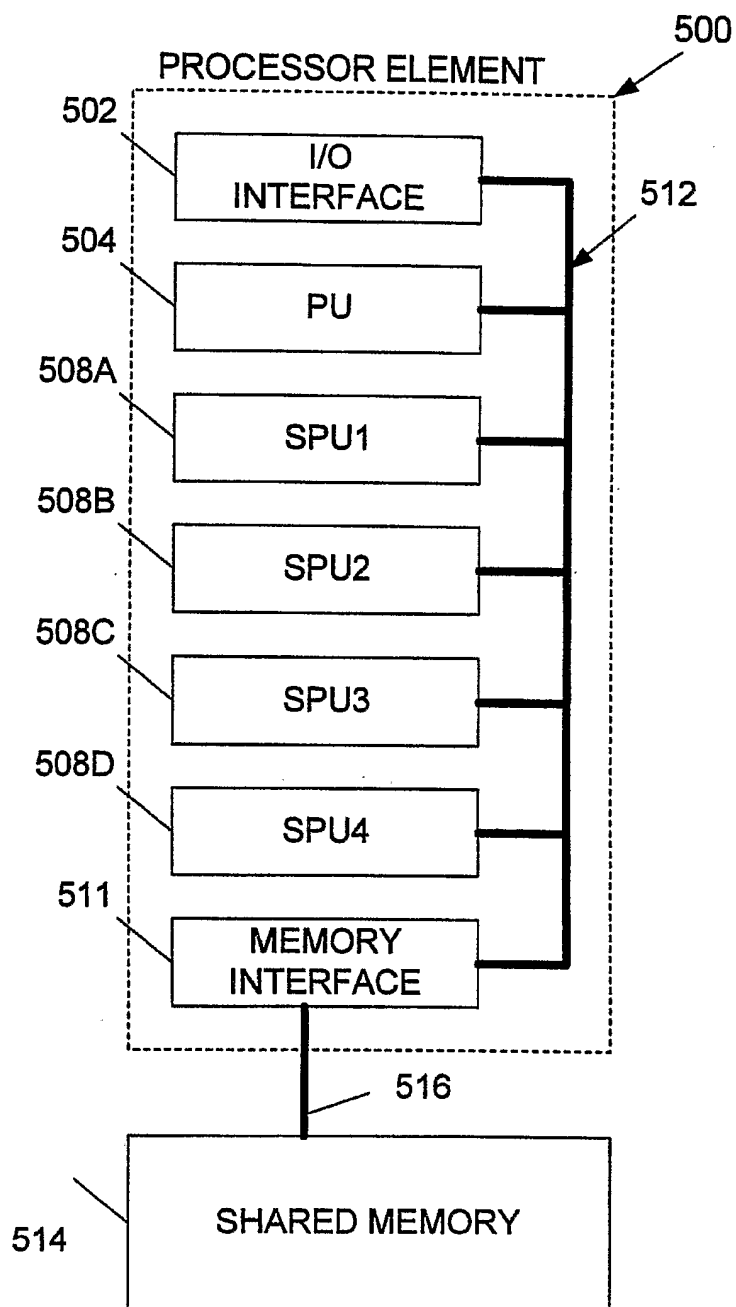


FIG. 6

508

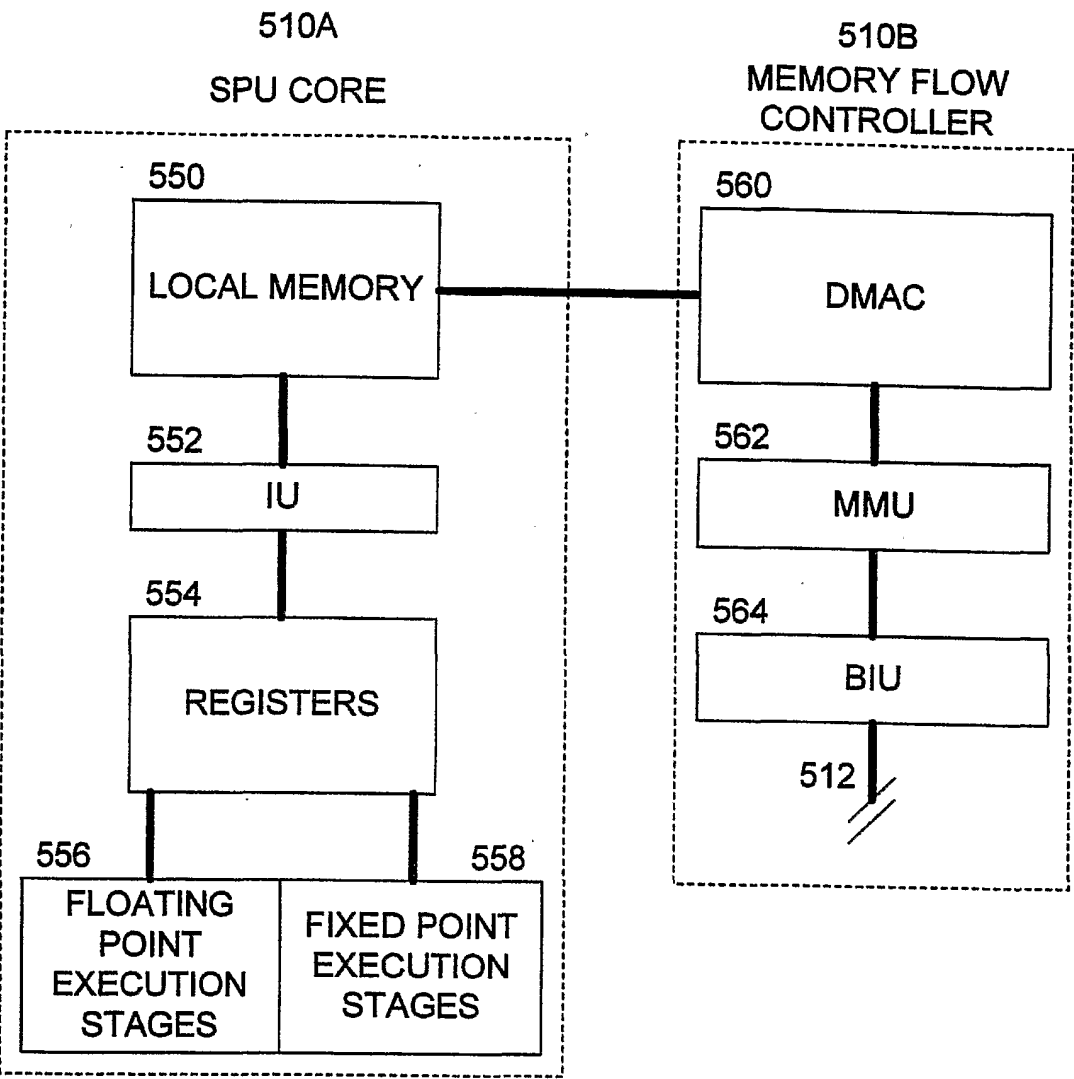
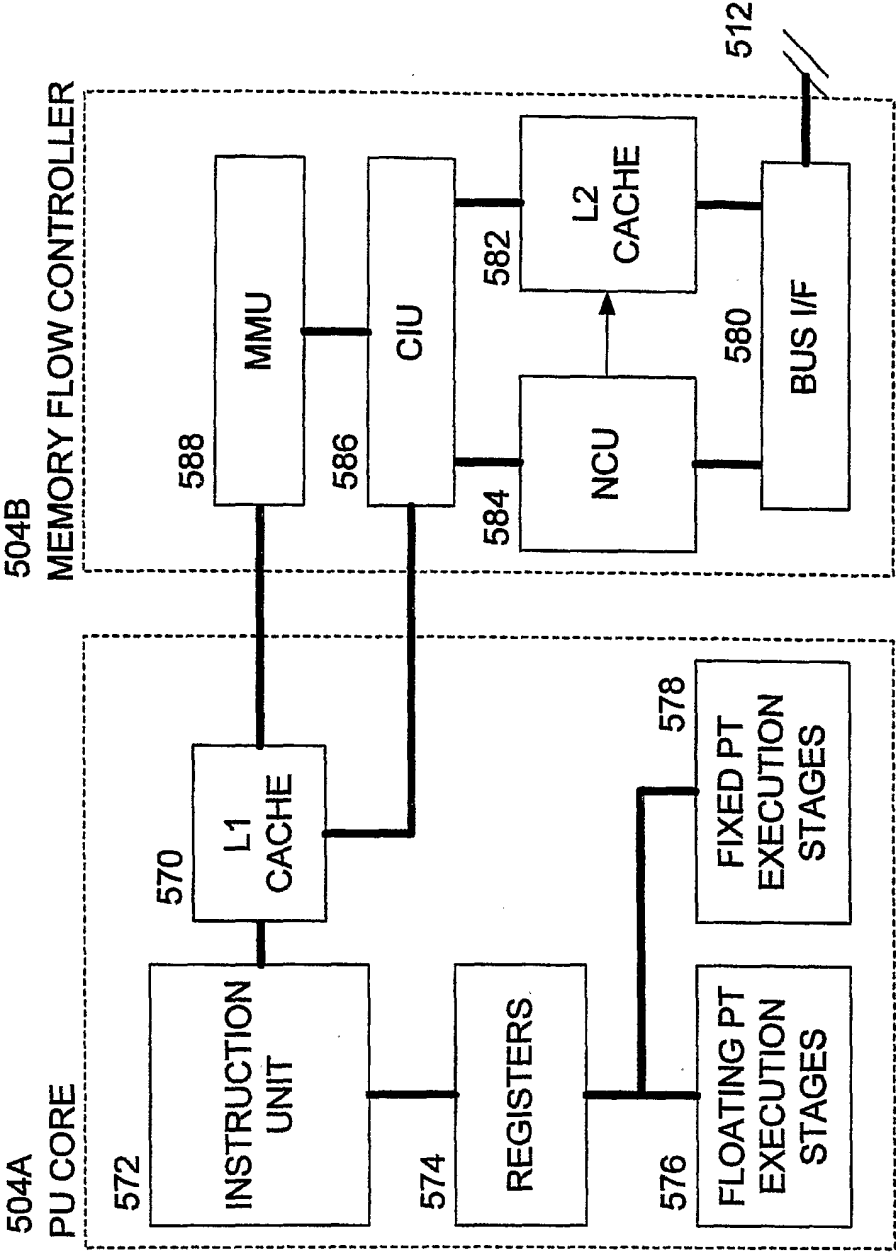


FIG. 7

504



INTERNATIONAL SEARCH REPORT

International application No

PCT/JP2005/023430

A. CLASSIFICATION OF SUBJECT MATTER

INV. G06F13/40 G06F5/06 G06F15/167

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

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

EPO-Internal, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	SMITH, SCOTT F: "A MULTIPLE-CLOCK-DOMAIN BUS ARCHITECTURE USING ASYNCHRONOUS FIFOS AS ELASTIC ELEMENTS" INET, [Online] October 2003 (2003-10), XP002370956 University of Idaho, USA Retrieved from the Internet: URL: http://coen.boisestate.edu/SSmith/Dissertation.pdf [retrieved on 2006-03-02]	1, 3-7, 17, 20, 21, 23, 24, 26
Y	page 2, paragraph 2 page 23, paragraph 1 - page 24, paragraph 1; figure 3.2.1 page 50, paragraph 1 ----- -/--	2, 8-16, 18, 19, 22, 25, 27

☒ 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 document 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

23 March 2006

Date of mailing of the international search report

30/03/2006

Name and mailing address of the ISA/

European Patent Office, P.B. 5818 Patentlaan 2
NL - 2280 HV Rijswijk
Tel. (+31-70) 340-2040, Tx. 31 651 epo nl,
Fax: (+31-70) 340-3016

Authorized officer

Kamps, S

INTERNATIONAL SEARCH REPORT

International application No

PCT/JP2005/023430

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	HEMANI A ET AL: "Lowering power consumption in clock by using Globally Asynchronous Locally Synchronous design style" DESIGN AUTOMATION, 1999. 36TH ANNUAL CONFERENCE ON NEW ORLEANS, LA, USA 21-25 JUNE 1999, PISCATAWAY, NJ, USA, IEEE, 21 June 1999 (1999-06-21), pages 873-878, XP010762145 ISBN: 1-58113-109-7 page 1, right-hand column, paragraph 4 -----	2,8,18, 22
X	US 5 471 581 A (MUNIER ET AL) 28 November 1995 (1995-11-28) column 3, line 58 - column 4, line 11 -----	1,7,17, 21
X	NANDURI B V R: "SPECIALTY MEMORIES INCREASE PERFORMANCE AND REDUCE COST IN HIGH SPEED SYSTEM DESIGNS" WESCON TECHNICAL PAPERS, WESTERN PERIODICALS CO. NORTH HOLLYWOOD, US, vol. 35, 1 November 1991 (1991-11-01), pages 116-121, XP000320525 page 119; figure 5 -----	1,7,17, 21
X	BRENT G ET AL: "ASYNCHRONOUS MULTI-CLOCK BIDIRECTIONAL BUFFER CONTROLLER" IBM TECHNICAL DISCLOSURE BULLETIN, IBM CORP. NEW YORK, US, vol. 24, no. 8, January 1982 (1982-01), pages 4404-4406, XP002011019 ISSN: 0018-8689 the whole document -----	1,7,17, 21
Y	WO 02/071196 A (PACT INFORMATIONSTECHNOLOGIE GMBH; VORBACH, MARTIN; BAUMGARTE, VOLKER) 12 September 2002 (2002-09-12) page 3, line 12 - page 4, line 24 -----	2,8,18, 22
Y	US 2002/138701 A1 (SUZUOKI MASAKAZU ET AL) 26 September 2002 (2002-09-26) cited in the application the whole document -----	9-16,19, 25,27

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No

PCT/JP2005/023430

Patent document cited in search report		Publication date	Patent family member(s)	Publication date
US 5471581	A	28-11-1995	NONE	
WO 02071196	A	12-09-2002	WO 02071249 A2	12-09-2002
			JP 2004538675 T	24-12-2004
			JP 2004535613 T	25-11-2004
			JP 2004536373 T	02-12-2004
US 2002138701	A1	26-09-2002	CN 1496511 A	12-05-2004
			EP 1370948 A1	17-12-2003
			WO 02077826 A1	03-10-2002
			JP 3454808 B2	06-10-2003
			JP 2002342165 A	29-11-2002
			JP 2003303134 A	24-10-2003
			TW 556087 B	01-10-2003
			US 2003229765 A1	11-12-2003