(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 582 980 B1

(12) **EUROPEAN PATENT SPECIFICATION**

(45) Date of publication and mention of the grant of the patent:
21.03.2012 Bulletin 2012/12

(21) Application number: **03780720.3**

(22) Date of filing: **11.12.2003**

(51) Int Cl.:
G06F 9/46 *(2006.01)*

(86) International application number:
PCT/JP2003/015838

(87) International publication number:
WO 2004/063925 (29.07.2004 Gazette 2004/31)

(54) **CONTEXT SWITCHING METHOD, DEVICE, PROGRAM, RECORDING MEDIUM, AND CENTRAL PROCESSING UNIT**

KONTEXTWECHSELVERFAHREN, VORRICHTUNG, PROGRAMM, AUFZEICHNUNGSMEDIUM UND ZENTRALEINHEIT

PROCEDE DE COMMUTATION DE CONTEXTE, DISPOSITIF, PROGRAMME, SUPPORT D'ENREGISTREMENT, ET UNITE CENTRALE

(84) Designated Contracting States:
DE FR GB NL

(30) Priority: **09.01.2003 JP 2003003038**

(43) Date of publication of application:
05.10.2005 Bulletin 2005/40

(73) Proprietor: **Japan Science and Technology Agency**
Kawaguchi-shi,
Saitama 332-0012 (JP)

(72) Inventor: **YAMASAKI, Nobuyuki**
Yokohama, Kanagawa 221-0802 (JP)

(74) Representative: **Vinsome, Rex Martin**
Urquhart-Dykes & Lord LLP
12th Floor, Cale Cross House
Pilgrim Street 156
Newcastle-upon-Tyne NE1 6SU (GB)

(56) References cited:
EP-A- 0 942 365
EP-A- 1 233 340
JP-A- 3 009 431
JP-A- 10 011 301
JP-A- 2000 076 079
JP-A- 2000 242 505
JP-A- 2002 513 182

- **GUANGZUO C, ET AL: 'PARALLEL REPLACEMENT MECHANISM FOR MULTITHREAD' ADVANCES IN PARALLEL AND DISTRIBUTED COMPUTING 1997 PROCEEDINGS 1997, pages 338 - 344, XP010215695**
- **WILLS D S, ET AL: 'AN ULTRA-LIGHT PROCESSOR FOR HIGH-THROUGHPUT APPLICATIONS' COMPUTER DESIGN VLSI IN COMPUTERS AND PROCESSORS ICCD '93 PROCEEDINGS 1993, pages 410 - 414, XP000463439**
- **KWAK H, ET AL: 'EFFECT OF MULTITHREADING ON CACHE PERFORMANCE' COMPUTERS IEEE TRANSACTIONS ON 1999, pages 176 - 184, XP000832007**

EP 1 582 980 B1

Description

Technical Field

[0001] The present invention relates to context switchihg methods, context switching units, central processing units, context switching programs, and computer-readable storage media having stored the programs, and more specifically, to a context switching method, a context switching unit, a central processing unit, a context switching program, and a computer-readable storage medium having stored the program that allow overhead caused by context switching in an operating system (OS) such as a real-time operating system (RT-OS) to be reduced. Context here means current execution statuses or information for executing each thread stored in a storage portion (such as a register file), including a general-purpose register, a floating-point register, a program counter, a status register, and others.

Background Art

[0002] Fig. 8 is a diagram showing a context switch operation.

[0003] This figure shows an example in which a plurality of contexts (threads) are switched and executed by a single central processing unit. When a context is switched, the statuses of the current context (a general-purpose register, a floating-point register, a program counter, a status register, and others, and hereafter simply referred to as a context) must be saved, and the statuses of a new context must be read out. The time required for the switch operation is referred to as overhead, and the overhead occurs each time a context is switched.

[0004] Some of the conventional techniques for reducing the context switching time are described in the following documents.

[0005] Patent Document 1 (Japanese Unexamined Patent Application Publication No. Hei-07-141208) describes a technique for reducing a dispatch time in a multitasking apparatus using a real-time operating system, by providing a plurality of register banks occupied by tasks and by switching the register bank to save and restore the context and others.

[0006] Patent Document 2 (Japanese Unexamined Patent Application Publication No. Hei-09-212371) describes a register save and restore system for reducing the overhead of an OS by providing a bit indicating whether the contents of each corresponding register have changed or not in a multitasking microprocessor and, when a task switch occurs, executing a save instruction if the contents of the register have been changed or not executing a save instruction if the contents of the register have not changed, in accordance with the bit.

[0007] Patent Document US 6,401,155 describes a method of operating a single processor to achieve rapid thread processing by transferring complete thread contexts between a memory and a context register set. Here, each thread context is read from a respective memory location in response to either a designated interrupt or an instruction.

Disclosure of Invention

[0008] In the conventional methods, when a context is switched, each context status held in the central processing unit is stored in a storage unit outside the central processing unit by means of a store instruction of software such as an OS. Then, the software, such as an OS, reads a new context from the storage unit by means of a load instruction. Each time a context is switched, large overhead occurs because several hundreds to one thousand and several hundreds of memory access cycles occur to store the context and read another context. The conventional methods read and store context by means of a load instruction and a store instruction of software, so that just one data item can be handled each time. Therefore, as the number of statuses to be stored increases, the context switching time increases.

[0009] Guangzuo C, et al: "Parallel replacement mechanism for multithread", Advances in Parallel and Distributed Computing 1997. Proceedings 1997, pages 338 - 344, XP010215695 discloses a context switching apparatus according to the preamble of claim 1.

[0010] Other known apparatus are disclosed in Wills D S, et al: "An ultra-light processor for high-throughput applications", Computer design: VLSI in computers and processors, 1993. ICCD 93 Proceedings, 1993, pages 410 - 414, XP000463439 and WO99/56209.

[0011] An object of the present invention is to reduce overhead caused by context switching significantly, especially in an application, such as a real-time OS, involving frequent context switching. Another object of the present invention is to provide such a context switching method and a context switching unit, a central processing unit, a context switching program, and a computer-readable storage medium having stored the program that enable a context to be stored and another context to be read by one or several memory access cycles each time a context switch occurs.

[0012] A further object of the present invention is to keep the context switching time constant and to minimize the time quantum of the real-time operation, especially in a system involving frequent context switching such as a real-time processing system.

[0013] The present invention reduces overhead resulting from context switching by

1. Providing a special storage unit (context cache) for holding a context, and
2. Connecting the special storage unit (context cache) and the central processing unit (CPU) by a special bus wider in bit width than a register.

[0014] According to a first aspect of the present invention, there is provided a context switching unit for switch-

ing a plurality of contexts, the context switching unit comprising:

a register file having stored a context related to a thread to be executed by an arithmetic logic unit or a memory access unit, the register file comprising a register read port, a register write port, a context-switching read port, and a context-switching write port;
characterised in that the context switching unit further comprises:
a context cache used exclusively for saving and restoring contexts, the context cache comprising a read port and a write port, connected directly to the register file and integrated in a central processing unit on a chip, the context cache being not connected to a memory through a bus, which connects the memory, an instruction cache and a data cache to each other, and being independent from a memory system including the memory, the instruction cache and the data cache, to realize context switching at a high processing speed without interference from the bus;
a context switching bus for connecting the register file and the context cache, the context switching bus comprising a restore bus and a save bus for connecting the read port and the write port of the context cache to the context-switching write port and the context-switching read port of the register file respectively; and
a thread control unit for controlling data transfer between the context cache and the register file, the thread control unit comprising a thread identifier table for storing a thread identifier for identifying the context of a thread stored in the context cache and being connected in parallel with the arithmetic logic unit and the memory access unit,
wherein, in case of a context switch operation which executes both a save operation for saving a context from the register file to the context cache and a restore operation for restoring a context from the context cache to the register file in parallel at the same time, the thread control unit receives a context switch instruction for executing the context switch operation with the identifier of a new thread to be interchanged;
the thread control unit obtains a restore address where a new context to be interchanged is stored in the context cache and the save register identifier indicating the location where the current context is stored in the register file, by searching through the thread identifier table in accordance with the thread identifier;
the thread control unit sends the obtained address to the context cache and the register identifier to the register file in parallel at the same time;
the register file, in accordance with the register identifier given by the thread control unit, outputs the data of the context to be saved from the context-switching read port to the save bus for the save operation and holds the data of the context to be restored in parallel at the same time, sent from the read port of the context cache to the context-switching write port through the restore bus for the restore operation;
the context cache, in accordance with the address given by the thread control unit, outputs the data of the context to be restored from the read port to the restore bus for the restore operation and holds the data of the context to be saved in parallel at the same time, sent from the context-switching read port of the register file to the write port via the save bus for the save operation; and
whereby the context switching unit switches contexts by the context switch operation which executes both the restore operation and the save operation in parallel at the same time.

[0015] According to a second aspect of the present invention, there is provided a central processing unit, the central processing unit including the above-described context switching unit,
an instruction cache for caching an instruction and a data cache for caching data;
an instruction fetch unit for fetching an instruction from the instruction cache and decoding the instruction;
an arithmetic logic unit for performing an operation in accordance with an instruction;
a memory access unit for, accessing the data cache and memory, and executing a load or store operation; and
an arithmetic bus for connecting the register file, the arithmetic logic unit, the memory access unit, and the thread control unit in parallel.

[0016] According to a third aspect of the present invention, there is provided a context switching method for switching a plurality of contexts by using a context switching unit comprising:

a register file having stored a context related to a thread to be executed by an arithmetic logic unit or a memory access unit, the register file comprising a register read port, a register write port, a context-switching read port, and a context-switching write port;
a context cache used exclusively for saving and restoring contexts, the context cache comprising a read port and a write port, connected directly to the register file and integrated in a central processing unit on a chip, the context cache being not connected to a memory through a bus, which connects the memory, an instruction cache and a data cache to each other, and being independent from a memory system including the memory, the instruction cache and the data cache, to realize context switching at a high processing speed without interference from the bus;
a context switching bus for connecting the register file and the context cache, the context switching bus

comprising a restore bus and a save bus for connecting the read port and the write port of the context cache to the context-switching write port and the context-switching read port of the register file respectively; and

a thread control unit for controlling data transfer between the context cache and the register file, the thread control unit comprising a thread identifier table for storing a thread identifier for identifying the context of a thread stored in the context cache and being connected in parallel with the arithmetic logic unit and the memory access unit,

wherein, in case of a context switch operation which executes both a save operation for saving a context from the register file to the context cache and a restore operation for restoring a context from the context cache to the register file in parallel at the same time, the thread control unit receives a context switch instruction for executing the context switch operation and the identifier of a new thread to be interchanged, the thread control unit obtains a restore address where a new context to be interchanged is stored in the context cache and the save register identifier indicating the location where the current context is stored in the register file, by searching through the thread identifier table in accordance with the thread identifier;

the thread control unit sends the obtained address to the context cache and the register identifier to the register file in parallel at the same time;

the register file, in accordance with the register identifier given by the thread control unit, outputs the data of the context to be saved from the context-switching read port to the save bus for the save operation and holds the data of the context to be restored in parallel at the same time, sent from the read port of the context cache to the context-switching write port through the restore bus for the restore operation;

the context cache, in accordance with the address given by the thread control unit, outputs the data of the context to be restored from the read port to the restore bus for the restore operation and holds the data of the context to be saved in parallel at the same time, sent from the context-switching read port of the register file to the write port via the save bus for the save operation; and

whereby the context switching unit switches contexts by the context switch operation which executes both the restore operation and the save operation in parallel at the same time.

[0017] According to a fourth aspect of the present invention, there is provided a context switching program for switching a plurality of contexts on a computer by using a context switching unit, and a computer-readable recording medium having recorded the program comprising:

a register file having stored a context related to a thread to be executed by an arithmetic logic unit or a memory access unit, the register file comprising a register read port, a register write port, a context-switching read port, and a context-switching write port;

a context cache used exclusively for saving and restoring contexts, the context cache comprising a read port and a write port, connected directly to the register file and integrated in a central processing unit on a chip, the context cache being not connected to a memory through a bus, which connects the memory, an instruction cache and a data cache to each other, and being independent from a memory system including the memory, the instruction cache and the data cache, to realize context switching at a high processing speed without interference from the bus; and

a context switching bus for connecting the register file and the context cache, the context switching bus comprising a restore bus and a save bus for connecting the read port and the write port of the context cache to the context-switching write port and the context-switching read port of the register file respectively; and

a thread control unit for controlling data transfer between the context cache and the register file, the thread control unit comprising a thread identifier table for storing a thread identifier for identifying the context of a thread stored in the context cache and being connected in parallel with the arithmetic logic unit and the memory access unit; characterised in that the context switching program causes the computer to execute:

a step in which, in case of a context switch operation which executes both a save operation for saving a context from the register file to the context cache and a restore operation for restoring a context from the context cache to the register file in parallel at the same time, the thread control unit receives a context switch instruction for executing the context switch operation with the identifier of a new thread to be interchanged;

a step in which the thread control unit obtains a restore address where a new context to be interchanged is stored in the context cache and a register identifier indicating the location where the current context is stored in the register file, by searching through the thread identifier table in accordance with the thread identifier;

a step in which the thread control unit sends the obtained address to the context cache and sends the register identifier to the register file in parallel at the same time;

a step in which the register file, in accordance with the register identifier given by the thread control unit, outputs the data of the context to be saved from the context-switching read port to the save bus for the

save operation and holds the data of the context to be restored in parallel at the same time, sent from the read port of the context cache to the context-switching write port through the restore bus for the restore operation;
a step in which the context cache, in accordance with the address given by the thread control unit, outputs the data of the context to be restored from the read port to the restore bus for the restore operation and holds the data of the context to be saved in parallel at the same time, sent from the context-switching read port of the register file to the write port via the save bus for the save operation; and
whereby the context switching unit switches contexts by the context switch operation which executes both the restore operation and the save operation in parallel at the same time.

[0018] According to a fifth aspect of the present invention, there is provided a computer-readable recording medium having recorded a context switching program as defined above.

Brief Description of the Drawings

[0019]

Fig. 1 is a view showing the configuration of a computer including a general CPU.
Fig. 2 is a flow chart of context switching by the general CPU.
Fig. 3 is a view showing the configuration of a computer including a CPU which uses a context cache.
Fig. 4 is a flow chart of context switching by the CPU which uses the context cache.
Fig. 5 is a view showing a detailed configuration of a context switching unit.
Fig. 6 is a flow chart of the processing of a swap instruction.
Fig. 7 is a view showing the implementation of the context switching unit.
Fig. 8 is a diagram showing a context switch operation.

Description of the Preferred Embodiments

1. RELATED ART

[0020] The present embodiment is used, for instance, as a context cache on the central processing unit of a responsive multi-threaded (RMT) processor for distributed real-time parallel control. A computer including the CPU related to the present embodiment will be described first.

[0021] Fig. 1 is a view showing the configuration of a computer including a general CPU.

[0022] This computer includes a CPU 10, a memory 20, an input/output unit (I/O) 30, and a bus 40. The CPU 10 is connected to the memory 20, the I/O 30, and others, via the bus 40. The CPU 10 includes a register file 1, an instruction fetch unit 2, an instruction cache 3, an arithmetic logic unit (ALU) 4, a memory access unit 5, a data cache 6, a bus interface unit 7, and an arithmetic bus 11.

[0023] The register file 1 includes a general-purpose register (GPR), a floating-point register (FPR), a program counter (PC), a status register (SR), and other registers. The register file 1 stores a current context. The instruction cache 3 and the data cache 6 uses an SRAM, a flip-flop (FF), and other elements, and can process an access, a read, a write, and others at a high speed although the storage capacity is small. The memory 20 outside the CPU 10 uses an SDRAM, a DRAM, and other elements, and can process an access, a read, a write, and others at a lower speed than the cache, although the storage capacity is large.

[0024] The ALU 4 uses necessary data and a necessary instruction from the instruction cache 3, the data cache 6, or the memory 20, in accordance with the principle of locality. If the instruction cache 3 or the data cache 6 has an instruction and data needed for processing, the ALU 4 uses the instruction and the data. If not, the ALU 4 accesses the memory 20 to obtain a necessary instruction or data. The bus interface unit 7 is a unit for connecting the instruction cache 3, the memory access unit 5, and the data cache 6 with the memory 20, the I/O 30, and others outside the CPU via the bus 40 and performing data input and output between the inside and the outside of the CPU. The arithmetic bus 11 is a bus, such as an arithmetic pipeline, for connecting the register file 1, the ALU 4, and the memory access unit 5 in parallel.

[0025] The instruction fetch unit 2 outputs an address to the instruction cache 3, fetches an instruction from the instruction cache 3, and decodes the instruction. The ALU 4 reads a necessary operand from the register file 1 in accordance with the decoded instruction. The ALU 4 performs an arithmetic operation accordingly and writes the result of the arithmetic operation back into the register file 1. If a memory access instruction such as load and store is given, the memory access unit 5 reads an operand in accordance with the decoded instruction.

[0026] The memory access unit 5 accesses the data cache 6 and performs a load operation or a store operation. When a store instruction is given, the memory access unit 5 sends an address and data to the data cache 6 and stores the data in the data cache 6. When a load instruction is given, the memory access unit 5 sends an address to the data cache 6 and reads data from the data cache 6. The read data is written back into the register file 1. If necessary data cannot be found in the data cache 6, the data is read from the memory 20.

[0027] In this general configuration, a context stored in the register file 1 is saved by means of a store instruction.

[0028] Fig. 2 is a flow chart of context switching by the general CPU.

[0029] Software, such as an OS, issues a store instruc-

tion is starts context switching. When the store instruction is issued, data is read from the register file 1 and sent to the memory access unit 5. The memory access unit 5 calculates a data storage address and accesses the data cache 6 (S101). If a data cache error occurs (S103), the data cache 6 reads a cache line from the memory 20 (S105). If no data cache error occurs (S103), the processing proceeds to step S107. The data cache 6 stores the data sent from the memory access unit 5 at an appropriate address in it (S107). The software, such as an OS, repeats the processing of the store instruction as many times as the number of registers to be saved. Until the contents of all the registers are saved, the processing is repeated back from step S101. When the contents of all the registers are saved, the processing proceeds to step S111 (S109)

[0030] Then, the software, such as an OS, uses a load instruction to restore a new context to be executed. After the current context is saved, the software, such as an OS, issues a load instruction. When the load instruction is issued, the memory access unit 5 calculates a data read address and accesses the data cache 6 (S111). If a data cache error occurs (S113), the data cache 6 reads a cache line from the memory 20 (S115). If no data cache error occurs (S113), the processing proceeds to step S117. When data is returned from the data cache 6 (S117), the memory access unit 5 writes the data back into the register file 1. The load instruction is processed as many times as the number of registers to be restored. When the contents of all the registers are not read, the processing is repeated back from step S111. When the contents of all the registers are read, context switching ends (S119).

2. CPU PROVIDED WITH CONTEXT SWITCHING UNIT

[0031] Fig. 3 is a view showing the configuration of a computer including a CPU which uses a context cache.

[0032] The CPU 100 includes a register file 1, an instruction fetch unit 2, an instruction cache 3, an ALU 4, a memory access unit 5, a data cache 6, a bus interface unit 7, a context cache 8, a thread control unit 9, an arithmetic bus 11, and a context bus 12. The arithmetic bus 11 is a bus, such as an arithmetic pipeline, for connecting the register file 1, the ALU 4, the memory access unit 5, and the thread control unit 9 in parallel. The configuration and operation of each block denoted by the same reference numeral as in the CPU 10 shown in Fig. 1 are as described earlier.

[0033] The context cache 8 uses a SRAM, a FF, and other elements, and can process an access, a read, a write, and others at a high processing speed. The context cache 8 is connected to the register file 1 via the context switching bus 12, and is used to cache a context. The thread control unit 9 is a unit for controlling the context cache 8 and is connected in parallel with the ALU 4 and the memory access unit 5. A thread generally means a processing unit or the smallest unit into which a process or a task is divided when the OS performs parallel processing of processes or tasks. Some processes or tasks may not be divided, and one process or one task may become one thread. When a context switch occurs, the context (a general-purpose register, a floating-point register, a program counter, a status register, and others) of the current thread must be saved, and the context of a new thread to be executed must be restored. When the context cache 8 of the present embodiment is used for context switching, contexts are saved and restored by means of a swap instruction for interchanging the data of the register file 1 and the context cache 8 via the context switching bus 12.

[0034] Fig. 4 is a flow chart of context switching by the CPU which uses the context cache.

[0035] Software, such as an OS, issues a swap instruction and starts context switching. The swap instruction is given to the thread control unit 9, together with the identifier (ID) of the thread to be interchanged (S201). The thread ID is used to identify a thread stored in the context cache 8. The thread control unit 9 saves data from the register file 1 to the context cache 8 via the context switching bus 12 while sending the data of the new thread from the context cache 8 to the register file 1 concurrently. The thread control unit 9 automatically interchanges the data of the register file 1 and the data of the context cache 8 as much as needed in accordance with the sent thread ID (S203). Once the software, such as an OS, issues a swap instruction, the special hardware performs and finishes the context switching.

3. DETAILS OF CONTEXT SWITCHING UNIT

[0036] Fig. 5 is a view showing a detailed configuration of a context switching unit.

[0037] The register file 1 includes general-purpose registers 111, floating-point registers 112, a program counter 113, and a status register 114. The context cache 8 includes a given number of context storage areas 8-1, 8-2, ... 8-n for storing a given number of contexts. The thread control unit 9 is a controller for controlling the context cache 8 and the register file 1 formed on an identical chip. The thread control unit 9 contains a thread ID table 91 for holding a given number of thread IDs for identifying the contexts stored in a on-chip memory. A multi-thread processor has a plurality of register files 1 arranged in parallel.

[0038] Fig. 6 is a flow chart of the processing of a swap instruction.

[0039] Context switching can be performed by using a special context switch instruction in a context switch handler, for instance. When software, such as an OS, issues a swap instruction, the thread control unit 9, which is special hardware, receives the swap instruction and a thread ID (S300). The thread control unit 9 searches through the thread ID table 91 in accordance with the thread ID, calculates an address where the data (context) of the thread to be interchanged is stored, as an access location

of the context cache, and calculates a register ID as an access location of the register file 1 (S301). A loop of steps S302 to S304 is repeated for the data of all contexts (the general-purpose registers, the status register, and the others). The thread control unit 9 accesses the context cache 8 in accordance with the calculated address, reads the data (context) of the thread to be interchanged, and writes the data in the register file 1 (S303). At the same time or in parallel, the thread control unit 9 accesses the register file 1, reads the data (context) of the current thread, and writes the data in the context cache 8 (S303). The data is now interchanged between the register file 1 and the context cache 8. Until all the data is interchanged (S304), the thread control unit 9 increments the address of the context cache 8 to be accessed and the register ID of the register file 1 to be accessed by one and repeats the processing back from the step S303 (loop of steps S302 to S304). When all the data is interchanged (S304), the processing of the swap instruction ends.

[0040] The thread control unit 9 interchanges as much data as needed by incrementing the addresses of the context cache 8 and the register file 1 successively. The context switch operation, which requires several hundreds to one thousand and several hundreds of clock cycles if a load instruction and a store instruction are used, can be completed just in one to several clock cycles according to the present embodiment because the context switching bus 12 between the register file 1 and the context cache 8 has a data transfer width much greater than the bit width of the register. To be more specific, a group of registers is handled as a single large register, and each large register is given a register ID. If all registers are handled as one large register, a context switch operation can be completed in a single clock cycle.

[0041] The thread control unit 9 processes a backup instruction for saving a context and a restore instruction for restoring a context, as well as a swap instruction for interchanging contexts. When a backup instruction is given, the context data is not transferred from the context cache 8 to the register file 1, and the context data is transferred just from the register file 1 to the context cache 8. When a restore instruction is given, the context data is not transferred from the register file 1 to the context cache 8, and the context data is transferred just from the context cache 8 to the register file 1.

[0042] Fig. 7 is a view showing the implementation of the context switching unit.

[0043] The context cache 8 is an on-chip memory provided for context backup, and the context cache 8 of the shown embodiment has two ports and is contained in the CPU. The shown context cache 8 has a write port 82 and a read port 83. The context cache 8 can contain storage areas for a given number of contexts (such as 32 contexts).

[0044] The register file 1 has a normal read port, a normal write port, and also special ports for context switching, which are a context-switching read port 17 and a context-switching write port 18, and a storage unit for holding a context is connected to these ports. In the shown embodiment, the register file 1 has a register read port 15, a register write port 16, the context-switching read port 17, and the context-switching write port 18. The register read port 15 is a port for reading a register from the register file 1 to a unit in the CPU; the register write port 16 is a port for writing a register from a unit in the CPU to the register file 1; the context-switching read port 17 is a port for reading a register from the register file 1 to the context cache 8; and the context-switching write port 18 is a port for writing a register from the context cache 8 to the register file 1.

[0045] When a context switch occurs, the software, such as an OS, issues a swap instruction, and the thread control unit saves data from the context cache 8 in the CPU to the register file 1 and fetches a new context from the register file 1, through the context-switching read port 17 and the context-switching write port 18. Context buses 12-1 and 12-2 connecting the register file 1 and the context cache 8 have a greater width than the bit width of the register file 1, so that a greater amount of data can be interchanged at one time. In the shown embodiment, the register file 1 and the on-chip context cache 8 are connected by the context switching buses 12-1 and 12-2 with a width of 256 bits each. A two-port on-chip memory can be used as the context cache 8, so that a read and a write can be carried out simultaneously. The context of 32 general-purpose registers each having 32 bits can be interchanged in four clock cycles.

[0046] The number of bits, the storage capacity, the number of ports, and other parameters given above are just a few examples, and can be specified appropriately.

4. OTHERS

[0047] The context switching method or the context switching unit or system of the present invention can be implemented by a context switching program for executing each step by a computer, a computer-readable recording medium having recorded the context switching program, a program product which includes the context switching program and can be loaded into an internal memory of a computer, a computer, such as a server, including the program, and others.

Industrial Applicability

[0048] According to the present invention, overhead caused by context switching can be substantially reduced, especially in an application involving frequent context switching, such as a real-time OS. The present invention can also provide a context switching method, a context switching unit, a central processing unit, and a computer-readable recording medium having recorded a context switching program which allow a memory access operation for storing a context and reading another context to be completed in one to several cycles at each context switch, for instance,.

[0049] According to the present invention, the context switching time can be kept constant, and the time quantum of a real-time operation can be minimized, especially in a system involving frequent context switching, such as a real-time processing system.

Claims

1. A context switching unit for switching a plurality of contexts, the context switching unit comprising:

a register file (1) having stored a context related to a thread to be executed by an arithmetic logic unit (4) or a memory access unit (5), the register file comprising a register read port (15), a register write port (16), a context-switching read port (17), and a context-switching write port (18);
characterised in that the context switching unit further comprises:
a context cache (8) used exclusively for saving and restoring contexts, the context cache comprising a read port (83) and a write port (82), connected directly to the register file (1) and integrated in a central processing unit on a chip, the context cache (8) being not connected to a memory (20) through a bus (40), which connects the memory (20), an instruction cache (3) and a data cache (6) to each other, and being independent from a memory system including the memory (20), the instruction cache (3) and the data cache (6), to realize context switching at a high processing speed without interference from the bus (40);
a context switching bus (12) for connecting the register file and the context cache, the context switching bus comprising a restore bus (12-1) and a save bus (12-2) for connecting the read port and the write port of the context cache to the context-switching write port and the context-switching read port of the register file respectively; and
a thread control unit (9) for controlling data transfer between the context cache and the register file, the thread control unit comprising a thread identifier table (91) for storing a thread identifier for identifying the context of a thread stored in the context cache and being connected in parallel with the arithmetic logic unit (4) and the memory access unit (5),
wherein, in case of a context switch operation which executes both a save operation for saving a context from the register file to the context cache and a restore operation for restoring a context from the context cache to the register file in parallel at the same time, the thread control unit (9) is configured to receive a context switch instruction for executing the context switch operation with the identifier of a new thread to be interchanged;
the thread control unit (S301) is configured to obtain a restore address where a new context to be interchanged is stored in the context cache and the save register identifier indicating the location where the current context is stored in the register file, by searching through the thread identifier table in accordance with the thread identifier;
the thread control unit (S303) is configured to send the obtained address to the context cache and the register identifier to the register file in parallel at the same time;
the register file (S303) being configured to, in accordance with the register identifier given by the thread control unit, output the data of the context to be saved from the context-switching read port to the save bus for the save operation and holds the data of the context to be restored in parallel at the same time, sent from the read port of the context cache to the context-switching write port through the restore bus for the restored operation;
the context cache (S303) being configured to, in accordance with the address given by the thread control unit, output the data of the context to be restored from the read port to the restore bus for the restore operation and holds the data of the context to be saved in parallel at the same time, sent from the context-switching read port of the register file to the write port via the save bus for the save operation; and
whereby the context switching unit (S303) switches contexts by the context switch operation which executes both the restore operation and the save operation in parallel at the same time.

2. A context switching unit according to Claim 1, wherein the context switching bus has a bus width greater than the bit width of the register file.

3. A context switching unit according to Claim 1, wherein the thread control unit comprises as many thread identifier tables as required to identify contexts cached in the context cache.

4. A context switching unit according to any of Claims 1 to 3, wherein the thread control unit saves the context of the current thread from the register file to the context cache and sends the context of a new thread from the context cache to the register file in parallel to automatically interchange a required number of data items between the register file and the context cache, when software, such as an operating system, issues a swap instruction for interchanging contexts, including a thread identifier as an operand, if the

swap instruction is executed.

5. A context switching unit according to any of Claims 1 to 4, wherein the thread control unit transfers the data of a context from the register file to the context cache and does not transfer the data of a context from the context cache to the register file, when software, such as an operating system, issues a backup instruction for saving a context, including a thread identifier as an operand, if the backup instruction is executed.

6. A context switching unit according to any of Claims 1 to 4, wherein the thread control unit transfers the data of a context from the context cache to the register file and does not transfer the data of a context from the register file to the context cache, when software, such as an operating system, issues a restore instruction for restoring a context, including a thread identifier as an operand, if the restore instruction is executed.

7. A central processing unit comprising:

a context switching unit according to any of Claims 1 to 6;
an instruction cache (3) for caching an instruction and a data cache (6) for caching data;
an instruction fetch unit (2) for fetching an instruction from the instruction cache and decoding the instruction;
an arithmetic logic unit (4) for performing an operation in accordance with an instruction;
a memory access unit (5) for, accessing the data cache and memory, and executing a load or store operation; and
an arithmetic bus (11) for connecting the register file, the arithmetic logic unit, the memory access unit, and the thread control unit in parallel.

8. A context switching method for switching a plurality of contexts by using a context switching unit comprising:

a register file (1) having stored a context related to a thread to be executed by an arithmetic logic unit or a memory access unit, the register file comprising a register read port (15), a register write port (16), a context-switching read port (17), and a context-switching write port (18);
a context cache (8) used exclusively for saving and restoring contexts, the context cache comprising a read port (83) and a write port (82), connected directly to the register file (1) and integrated in a central processing unit on a chip, the context cache (8) being not connected to a memory (20) through a bus (40), which connects the memory (20), an instruction cache (3) and a data cache (6) to each other, and being independent from a memory system including the memory (20), the instruction cache (3) and the data cache (6), to realize context switching at a high processing speed without interference from the bus (40);
a context switching bus (12) for connecting the register file and the context cache, the context switching bus comprising a restore bus (12-1) and a save bus (12-2) for connecting the read port and the write port of the context cache to the context-switching write port and the context-switching read port of the register file respectively; and
a thread control unit (S300) for controlling data transfer between the context cache and the register file, the thread control unit comprising a thread identifier table (91) for storing a thread identifier for identifying the context of a thread stored in the context cache and being connected in parallel with the arithmetic logic unit and the memory access unit, said context switching method comprising the following steps:
in case of a context switch operation which executes both a save operation for saving a context from the register file to the context cache and a restore operation for restoring a context from the context cache to the register file in parallel at the same time, the thread control unit(9) receives a context switch instruction for executing the context switch operation and the identifier of a new thread to be interchanged,
the thread control unit (S301) obtains a restore address where a new context to be interchanged is stored in the context cache and the save register identifier indicating the location where the current context is stored in the register file, by searching through the thread identifier table in accordance with the thread identifier;
the thread control unit (S303) sends the obtained address to the context cache and the register identifier to the register file in parallel at the same time;
the register file (S303), in accordance with the register identifier given by the thread control unit, outputs the data of the context to be saved from the context-switching read port to the save bus for the save operation and holds the data of the context to be restored in parallel at the same time, sent from the read port of the context cache to the context-switching write port through the restore bus for the restore operation;
the context cache (S303), in accordance with the address given by the thread control unit, outputs the data of the context to be restored from the read port to the restore bus for the restore operation and holds the data of the context to be saved in parallel at the same time, sent from

the context-switching read port of the register file to the write port via the save bus for the save operation; and

whereby the context switching unit (S303) switches contexts by the context switch operation which executes both the restore operation and the save operation in parallel at the same time.

9. A context switching method according to Claim 8, comprising saving the context of the current thread from the register file to the context cache and sending the context of a new thread from the context cache to the register file in parallel to automatically interchange a required number of data items between the register file and the context cache, when software, such as an operating system, issues a swap instruction for interchanging contexts, including a thread identifier as an operand, if the swap instruction is executed.

10. A context switching method according to Claim 8, comprising transferring the data of a context from the register file to the context cache and not transferring the data of a context from the context cache to the register file, when software, such as an operating system, issues a backup instruction for saving a context, including a thread identifier as an operand, if the backup instruction is executed.

11. A context switching method according to Claim 8, comprising transferring the data of a context from the context cache to the register file and not transferring the data of a context from the register file to the context cache, when software, such as an operating system, issues a restore instruction for restoring a context, including a thread identifier as an operand, if the restore instruction is executed.

12. A context switching program for switching a plurality of contexts on a computer by using a context switching unit comprising:

a register file (1) having stored a context related to a thread to be executed by an arithmetic logic unit or a memory access unit, the register file comprising a register read port (15), a register write port (16), a context-switching read port (17), and a context-switching write port (18);

a context cache (8) used exclusively for saving and restoring contexts, the context cache comprising a read port (83) and a write port (82), connected directly to the register file (1) and integrated in a central processing unit on a chip, the context cache (8) being not connected to a memory (20) through a bus (40), which connects the memory (20), an instruction cache (3) and a data cache (6) to each other, and being independent from a memory system including the memory (20), the instruction cache (3) and the data cache (6), to realize context switching at a high processing speed without interference from the bus (40); and

a context switching bus (12) for connecting the register file and the context cache, the context switching bus comprising a restore bus (12-1) and a save bus (12-2) for connecting the read port and the write port of the context cache to the context-switching write port and the context-switching read port of the register file respectively; and

a thread control unit (9) for controlling data transfer between the context cache and the register file, the thread control unit comprising a thread identifier table (91) for storing a thread identifier for identifying the context of a thread stored in the context cache and being connected in parallel with the arithmetic logic unit and the memory access unit; wherein the context switching program causes the computer to execute:

a step in which, in case of a context switch operation which executes both a save operation for saving a context from the register file to the context cache and a restore operation for restoring a context from the context cache to the register file in parallel at the same time, the thread control unit(9) receives a context switch instruction for executing the context switch operation with the identifier of a new thread to be interchanged;

a step in which the thread control unit obtains a restore address where a new context to be interchanged is stored in the context cache and a register identifier indicating the location where the current context is stored in the register file, by searching through the thread identifier table in accordance with the thread identifier;

a step in which the thread control unit (S303) sends the obtained address to the context cache and sends the register identifier to the register file in parallel at the same time;

a step in which the register file (S303), in accordance with the register identifier given by the thread control unit, outputs the data of the context to be saved from the context-switching read port to the save bus for the save operation and holds the data of the context to be restored in parallel at the same time, sent from the read port of the context cache to the context-switching write port through the restore bus for the restore operation;

a step in which the context cache (S303), in accordance with the address given by the thread control unit, outputs the data of the context to be restored from the read port to the restore bus for the restore operation and holds the data of

the context to be saved in parallel at the same time, sent from the context-switching read port of the register file to the write port via the save bus for the save operation; and
whereby the context switching unit (S303) switches contexts by the context switch operation which executes both the restore operation and the save operation in parallel at the same time.

13. A computer-readable recording medium having recorded a context switching program according to claim 12.

Patentansprüche

1. Kontextwechselvorrichtung zum Umschalten einer Mehrzahl von Kontexten, die Kontextvorrichtung umfassend:

einen Registerspeicher (1), in dem ein Kontext gespeichert ist, der auf einen Prozess bezogen ist, und in einer arithmetischen Logikeinheit (4) oder einer Speicherzugriffseinheit (5) ausgeführt wird, wobei der Registerspeicher eine Register-Leseschnittstelle (1), eine Register-Schreibschnittstelle (16), eine Kontextwechsel-Leseschnittstelle (1) und eine Kontextwechsel-Schreibschnittstelle (18) umfasst;

dadurch gekennzeichnet, dass die Kontextwechselvorrichtung ferner umfasst:

einen Kontext-Pufferspeicher (8), der ausschließlich zur Speicherung und Wiederherstellung von Kontexten dient, wobei der Kontext-Pufferspeicher eine Leseschnittstelle (83) und eine Schreibschnittstelle (82) umfasst, und direkt mit dem Registerspeicher (1) verbunden ist und in eine zentrale Verarbeitungseinheit (CPU) auf einem Chip integriert ist, und der Kontext-Pufferspeicher (8) nicht mit einem Speicher (20) mittels eines Bus (40) verbunden ist, der der Speicher (20), einen Befehls-Pufferspeicher (3) und einen Daten-Pufferspeicher (6) untereinander verbindet, wobei der Kontextpufferspeicher (8) unabhängig von einem Speichersystem ist, das den Speicher (20), den Befehls-Pufferspeicher (3) und den Daten-Pufferspeicher (6) umfasst, und einen Kontextwechseln mit hoher Verarbeitungsgeschwindigkeit ohne Zusammenwirken mit dem Bus (40) gewährleistet;
einen Kontextwechsel-Bus (12) zur Verbindung des Registerspeichers und des Kontext-Pufferspeichers, wobei der Kontextwechsel-Bus einen Wiederherstellungs-Bus (12-1) und einen Speicher-Bus (12-2) zum Verbinden der Leseschnittstelle und der Schreibschnittstelle des Kontext-Pufferspeichers jeweils mit der Kontextwechsel-Schreibschnittstelle und der Kontextwechsel-Leseschnittstelle des Registerspeichers umfasst; und
eine Prozess-Steuereinheit (9) zum Steuern des Datendurchsatzes zwischen dem Kontext-Pufferspeicher und dem Registerspeicher, die Prozess-Steuereinheit umfassend eine Prozess-Identifikationstabelle (91) zum Speichern eines Prozess-Bezeichners zum Identifizieren des Kontexts eines Prozesses, der im Kontext-Pufferspeicher gespeichert ist und parallel mit der arithmetischen Logikeinheit (4) und der Speicherzugriffseinheit (5) verbunden ist,
wobei im Falle einer Kontextwechsel-Operation, die sowohl eine Speicher-Operation zum Speichern eines Kontexts aus dem Registerspeicher in den Kontext-Pufferspeicher als auch eine Wiederherstellen-Operation zum Wiederherstellen eines Kontexts aus dem Kontext-Pufferspeicher in den Registerspeicher parallel gleichzeitig ausführt, die Prozess-Steuereinheit (9) ausgelegt ist, einen Kontextwechsel-Befehl zur Ausführung der Kontextwechsel-Operation mit dem Bezeichner eines neuen Prozesses, der auszutauschen ist, zu empfangen;
die Prozess-Steuereinheit (S301) dazu ausgelegt ist, eine Wiederherstellungsadresse, wo ein neuer auszutauschender Kontext im Kontext-Pufferspeicher gespeichert ist und den Speicher-Register-Bezeichner, der den Ort anzeigt, wo der aktuelle Kontext im RegisterSpeicher gespeichert ist, mittels Durchsuchen der Prozess-Identifikationstabelle entsprechend dem Prozess-Bezeichner zu erhalten; und die Prozess-Steuereinheit dazu ausgelegt ist, die erhaltene Adresse an den Kontext-Puffspeicher und den Register-Bezeichner und den Registerspeicher parallel gleichzeitig zu senden;
die Prozess-Steuereinheit (S303) dazu ausgelegt ist, entsprechend dem Register-Bezeichner, der durch die Prozess-Steuereinheit vorgegeben ist, die Daten des zu speichernden Kontexts von der Kontextwechsel-Schnittstelle an den Speicher-Bus für die Speicher-Operation auszugeben und die Daten des wiederherzustellenden Kontexts parallel gleichzeitig behält, die von der Lese-Schnittstelle des Kontext-Pufferspeichers für eine Wiederherstellen-Operation durch den Wiederherstellungs-Bus zur Kontextwechsel-Schreibschnittstelle geschickt werden;
und der Kontext-Pufferspeicher (S303) dazu ausgelegt ist, entsprechend einer Adresse, die durch die Prozess-Steuereinheit vorgegeben ist, zu einer Wiederherstellungs-Operation die Daten des wiederherzustellenden Kontexts von

der Lese-Schnittstelle zum Wiederherstellungs-Bus auszugeben und die Dates des zu speichernden Kontexts parallel gleichzeitig zu behalten, die für eine Speicher-Operation von der Kontextwechsel-Leseschnittstelle der Registerspeichers über den Speicher-Bus zum Schreib-Schnittstelle geschickt werden; und
wobei die Kontextwechselvorrichtung (S303) die Kontexte mittels der Kontextwechseloperation austauscht, die sowohl die Wiederherstellungs-Operation als auch die Speicher-Operation parallel gleichzeitig ausführt.

2. Kontextwechselvorrichtung gemäß Anspruch 1, wobei der Kontextwechsel-Bus eine Busbreite aufweist, die größer ist als die Bit-Breite des Registerspeichers.

3. Kontextwechselvorrichtung gemäß Anspruch 1, wobei die Prozess-Steuereinheit so viele Prozess-Identifikationstabellen aufweist, wie zu einer Identifikation von im Kontext-Pufferspeicher zwischengespeicherten Kontexten erforderlich sind.

4. Kontextwechselvorrichtung gemäß einem der Ansprüche 1 bis 3, wobei die Prozess-Steuereinheit den Kontext des aktuellen Prozesses aus dem Registerspeicher in den Kontext-Pufferspeicher speichert und parallel den Kontext des neuen Prozesses aus dem Kontext-Pufferspeicher an den Registerspeicher schickt um automatisch eine erforderliche Anzahl von Datenwörtern zwischen dem Registerspeicher und dem Kontext-Pufferspeicher auszutauschen, wenn eine Software, wie ein Betriebssystem, eine Tausch-Anweisung zum Wechsel von Kontexten ausgibt, die einen Prozess-Bezeichner als Operanden umfasst, wenn die Tausch-Anweisung ausgeführt wird.

5. Kontextwechselvorrichtung gemäß einem der Ansprüche 1 bis 4, wobei die Prozess-Steuereinheit die Daten eines Kontexts vom Registerspeicher zum Kontext-Pufferspeicher transferiert und die Daten eines Kontexts aus dem Kontext-Pufferspeicher nicht in den Registerspeicher transferiert werden, wenn eine Software, wie ein Betriebssystem, eine Backup-Anweisung zum Speichern eines Kontexts, der einen Prozess-Bezeichner als Operanden umfasst, ausgibt, wenn die Backup-Anweisung ausgeführt wird.

6. Kontextwechselvorrichtung nach einem der Ansprüche 1 bis 4, wobei die Prozess-Steuereinheit die Daten eines Kontexts aus dem Kontext-Pufferspeicher in den Registerspeicher transferiert und die Daten eines Kontexts aus dem Registerspeicher nicht in den Kontext-Pufferspeicher transferiert, wenn Software, wie ein Betriebssystem, eine Wiederherstellungs-Anweisung zur Wiederherstellung eines Kontexts, der einen Prozess-Bezeichner als Operanden umfasst, ausgibt, wenn die Wiederherstellungs-Anweisung ausgeführt wird.

7. CPU, umfassend:

eine Kontextwechselvorrichtung gemäß einem der Ansprüche 1 bis 6;
einen Befehls-Pufferspeicher (3) zum Zwischenspeichern einer Anweisung und einen Daten-Pufferspeicher (6) zum Zwischenspeichern von Daten;
eine Befehlsabhol-Vorrichtung (2) zum Abholen eines Befehls aus dem Befehls-Pufferspeicher und Entschlüsseln des Befehls;
eine arithmetische Logikeinheit (4) zur Durchführung einer Operation gemäß einem Befehl;
eine Speicherzugriffseinheit (5) zum Zugriff auf den Daten-Pufferspeicher und den Speicher und zum Ausführen einer Lade- oder Speicher-Operation; und
einen arithmetischen Bus (11) zur parallelen Verbindung des Registerspeichers, der arithmetischen Logikeinheit, der Speicherzugriffseinheit und der Prozess-Steuereinheit.

8. Kontextwechselverfahren zum Umschalten einer Mehrzahl von Kontexten mittels einer Kontextwechselvorrichtung, umfassend:

einen Registerspeicher (1), in dem ein Kontext, der auf einen Prozess bezogen ist, der mit einer arithmetischen Logikeinheit oder einer Speicherzugriffseinheit auszuführen ist, wobei der Registerspeicher eine Register-Leseschnittstelle (15), eine Register-Schreibschnittstelle (16), eine Kontextwechsel-Leseschnittstelle (17) und eine Kontextwechsel-Schreibschnittstelle (18) aufweist;
einen Kontext-Pufferspeicher (8) der ausschließlich zum Speichern und Wiederherstellen von Kontexten vorgesehen ist, der Kontext-Pufferspeicher umfassend eine Lese-Schnittstelle (83) und eine Schreib-Schnittstelle (82), der direkt mit dem Registerspeicher (1) verbunden ist und in eine CPU auf einem Chip integriert ist, wobei der Kontext-Pufferspeicher (8) nicht mit dem Speicher (20) über einen Bus (40) verbunden ist, der untereinander den Speicher (20), den Befehls-Speicher (3) und den Daten-Pufferspeicher (6) verbindet, und einen Kontextwechsel mit hoher Verarbeitungsgeschwindigkeit ohne Zusammenwirken mit dem Bus (40) gewährleistet;
einen Kontextwechsel-Bus (12) zur Verbindung des Registerspeichers und des Kontext-Pufferspeichers, wobei der Kontextwechsel-Bus ei-

nen Wiederherstellungs-Bus (12-1) und einen Speicher-Bus (12-2) zum Verbinden der Leseschnittstelle und der Schreibschnittstelle des Kontext-Pufferspeichers jeweils mit der Kontextwechsel-Schreibschnittstelle und der Kontextwechsel-Leseschnittstelle des Registerspeichers umfasst; und

eine Prozess-Steuereinheit (S300) zum Steuern eines Datentransfers zwischen dem Kontext-Pufferspeicher und dem Registerspeicher vorgesehen ist, wobei die Prozess-Steuereinheit eine Prozess-Identifikationstabelle (91) zum Speichern eines Prozess-Bezeichners zum Identifizieren des Kontexts eines Prozesses, der im Kontext-Pufferspeicher gespeichert ist, umfasst, und parallel mit der arithmetischen Logikeinheit und der Speicherzugriffseinheit verbunden ist, wobei das Kontextwechsel-Verfahren die folgenden Schritte umfasst:

im Falle einer Kontextwechsel-Operation durchgeführt wird, die sowohl eine Speicher-Operation zum Speichern eines Kontexts aus dem Registerspeicher in den Kontext-Pufferspeicher als auch eine Wiederherstellungs-Operation zum Wiederherstellen eines Kontexts aus dem Kontext-Pufferspeicher in den Registerspeicher parallel gleichzeitig durchführt, erhält die Prozess-Steuereinheit (9) einen Kontextwechsel-Befehl zur Durchführung der Kontextwechsel-Operation und den Bezeichner des neuen auszutauschenden Prozesses,

die Prozess-Steuereinheit (S301) erhält eine Wiederherstellungsadresse, wo ein neuer auszutauschender Kontext im Kontext-Pufferspeicher gespeichert ist und den Speicher-Register-Bezeichner, der den Ort anzeigt, wo der aktuelle Kontext im Registerspeicher gespeichert ist, indem die Prozess-Identifikationstabelle gemäß dem Prozess-Bezeichner durchsucht wird;

die Prozess-Steuereinheit (S303) schickt die erhaltene Adresse an den Kontext-Pufferspeicher und den Speicher- Register-Bezeichner an den Registerspeicher parallel gleichzeitig;

der Registerspeicher (S303) gibt, gemäß dem Speicher-Register-Bezeichner, der durch die Prozess-Steuereinheit vorgegeben ist, die Daten des Kontexts aus, der von der Kontextwechsel-Leseschnittstelle an den Speicher-Bus für die Speicher-Operation zu speichern ist, und behält die Daten des parallel gleichzeitig wiederherzustellenden Kontexts, der von der Leseschnittstelle des Kontext-Pufferspeichers für die Wiederherstellungs-Operation durch den Wiederherstellungs-Bus zur Kontextwechsel-Schreibschnittstelle geschickt wird;

der Kontext-Pufferspeicher (S303), entsprechend einer Adresse, die durch die Prozess-Steuereinheit vorgegeben ist, zu einer Wiederherstellungs-Operation die Daten des wiederherzustellenden Kontexts von der Lese-Schnittstelle zum Wiederherstellungs-Bus ausgibt und parallel die Dates des zu speichernden Kontexts gleichzeitig behält, die für eine Speicher-Operation von der Kontextwechsel-Leseschnittstelle des Registerspeichers über den Speicher-Bus zur Schreib-Schnittstelle geschickt werden; und

wobei die Kontextwechselvorrichtung (S303) die Kontexte umschaltet, mittels der Kontextwechseloperation austauscht, die sowohl die Wiederherstellungs-Operation als auch die Speicher-Operation parallel gleichzeitig ausführt.

9. Kontextwechselverfahren nach Anspruch 8, umfassend ein Speichern des Kontexts des aktuellen Prozesses aus dem Register in den Kontext-Pufferspeicher und ein paralleles Senden des Kontexts einen neuen Prozesses vom Kontext-Pufferspeicher in den Registerspeicher zum automatischen Austausch einer erforderlichen Anzahl von Datenwörtern zwischen dem Registerspeicher und dem Kontext-Pufferspeicher, wenn Software, wie ein Betriebssystem, einen Tausch-Befehl zum Austausch von Kontexten ausgibt, der einen Prozess-Bezeichner als Operanden umfasst, wenn der Tausch-Befehl ausgeführt wird.

10. Kontextwechselverfahren nach Anspruch 8, umfassend ein Transferieren der Daten eines Kontexts aus dem Registerspeicher in den Kontext-Pufferspeicher und kein Transferieren der Daten des Kontexts aus dem Kontext-Pufferspeicher in den Registerspeicher, wenn Software, wie ein Betriebssystem, einen Backup-Befehl zum Speichern eines Kontexts ausgibt, der einen Prozess-Bezeichner als Operanden umfasst, wenn der Backup-Befehl ausgeführt wird.

11. Kontextwechselverfahren nach Anspruch 8, umfassend ein Transferieren der Daten eines Kontexts aus dem Kontext-Pufferspeicher in den Registerspeicher und kein Transferieren der Daten des Kontexts aus dem Registerspeicher in den Kontext-Pufferspeicher, wenn Software, wie ein Betriebssystem, einen Backup-Befehl zum Speichern eines Kontexts ausgibt, der einen Prozess-Bezeichner als Operanden umfasst, wenn der Backup-Befehl ausgeführt wird.

12. Kontextwechselprogramm zum Umschalten einer Mehrzahl von Kontexten auf einem Computer mittels einer Kontextwechselvorrichtung, umfassend:

einen Registerspeicher (1), in dem ein Kontext gespeichert ist, der auf einen Prozess bezogen ist, der in einer arithmetischen Logikeinheit (4) oder einer Speicherzugriffseinheit (5) ausgeführt wird, wobei der Registerspeicher eine Register-Leseschnittstelle (15), eine Register-Schreibschnittstelle (16), eine Kontextwechsel-Leseschnittstelle (17) und eine Kontextwechsel-Schreibschnittstelle (18) umfasst;
einen Kontext-Pufferspeicher (8), der ausschließlich zur Speicherung und Wiederherstellung von Kontexten dient, wobei der Kontext-Speicher eine Leseschnittstelle (83) und eine Schreibschnittstelle (82) umfasst, und direkt mit dem Registerspeicher (1) verbunden ist und in eine CPU auf einem Chip integriert ist, und der Kontext-Pufferspeicher (8) nicht mit einem Speicher (20) mittels eines Bus (40) verbunden ist, der den Speicher (20), einen Befehls-Pufferspeicher (3) und einen Daten-Pufferspeicher (6) untereinander verbindet, wobei der Kontextpufferspeicher (8) unabhängig von einem Speichersystem ist, das den Speicher (20), den Befehls-Pufferspeicher (3) und den Daten-Pufferspeicher (6) umfasst, und einen Kontextwechseln mit hoher Verarbeitungsgeschwindigkeit ohne Zusammenwirken mit dem Bus (40) gewährleistet;
einen Kontextwechsel-Bus (12) zur Verbindung des Registerspeichers und des Kontext-Pufferspeichers, wobei der Kontextwechsel-Bus einen Wiederherstellungs-Bus (12-1) und einen Speicher-Bus (12-2) zum Verbinden der Leseschnittstelle und der Schreibschnittstelle des Kontext-Pufferspeichers jeweils mit der Kontextwechsel-Schreibschnittstelle und der Kontextwechsel-Leseschnittstelle des Registerspeichers umfasst; und
eine Prozess-Steuereinheit (9) zum Steuern des Datendurchsatzes zwischen dem Kontext-Pufferspeicher und dem Registerspeicher, die Prozess-Steuereinheit umfassend eine Prozess-Identifikationstabelle (91) zum Speichern eines Prozess-Bezeichners zum Identifizieren des Kontexts eines Prozesses, der im Kontext-Pufferspeicher gespeichert ist und parallel mit der arithmetischen Logikeinheit (4) und der Speicherzugriffseinheit (5) verbunden ist, wobei das Kontextwechselprogramm den Computer durchführen lässt:
einen Schritt, in dem im Falle einer Kontextwechsel-Operation, die sowohl eine Speicher-Operation zum Speichern eines Kontexts aus dem Registerspeicher in den Kontext-Pufferspeicher als auch eine Wiederherstellen-Operation zum Wiederherstellen eines Kontexts aus dem Kontext-Pufferspeicher in den Registerspeicher parallel gleichzeitig ausführt, die Prozess-Steuereinheit (9) einen Kontextwechsel-Befehl zur Ausführung der Kontextwechsel-Operation mit dem Bezeichner eines neuen Prozesses, der auszutauschen ist, empfängt;
einen Schritt, in dem die Prozess-Steuereinheit eine Wiederherstellungsadresse erhält, wo in Kontext-Pufferspeicher ein neuer auszutauschender Kontext gespeichert ist, und einen Speicher-Register-Bezeichner, der den Ort anzeigt, wo der aktuelle Kontext im Registerspeicher gespeichert ist, indem die Prozess-Identifikationstabelle nach dem Prozess-Bezeichner durchsucht wird;
einen Schritt, in dem die Prozess-Steuereinheit (S303) die empfangene Adresse an den Kontext-Pufferspeicher sendet und parallel den Speicher-Register-Bezeichner gleichzeitig an den Registerspeicher sendet;
einen Schritt, in dem der Registerspeicher (S303) gemäß dem Speicher-Register-Bezeichner, der durch die Prozess-Steuereinheit vorgegeben ist, die Daten des zu speichernden Kontexts aus der Kontextwechsel-Leseschnittstelle an den Speicher-Bus ausgibt, und parallel die Daten des wiederherzustellenden Kontexts gleichzeitig behält, die von der Leseschnittstelle des Kontext-Pufferspeichers an die Kontextwechsel-Schreibschnittstelle durch den Wiederherstellungs-Bus für die Wiederherstellungsoperation gesendet wird;
einen Schritt, in dem der Kontext-Zwischenspeicher (S303) gemäß der Adresse, die durch die Prozess-Steuereinheit vorgegeben ist, für die Wiederherstellungs-Operation Daten des wiederherzustellenden Kontexts von der Leseschnittstelle an den Wiederherstellungs-Bus ausgibt, und parallel die Daten des zu speichernden Kontexts gleichzeitig behält, die für die Speicher-Operation von der Kontextwechsel-Leseschnittstelle des Registerspeichers durch den Speicher-Bus an die Schreibschnittstelle gesendet wird; und
wobei die Kontextwechselvorrichtung (S303) mittels der Kontextwechsel-Operation, die sowohl die Wiederherstellungs-Operation und gleichzeitig parallel die Speicher-Operation ausführt, zwischen Kontexten umschaltet.

13. Ein computer-lesbares Speichermedium, in dem ein Kontextwechselprogramm nach Anspruch 12 gespeichert ist

Revendications

1. Unité de commutation de contextes destinée à commuter une pluralité de contextes, l'unité de commutation de contextes comprenant :

un fichier de registre (1) c'est de stocké un contexte connexe à un fil d'exécution destiné à être exécuté par une unité logique arithmétique (4) ou une unité d'accès en mémoire (5), le fichier de registre comprenant un port de lecture de registre (15), un port d'écriture de registre (16), un port de lecture de commutation de contextes (17), et un port d'écriture de commutation de contextes (18) ;

caractérisé en ce que l'unité de commutation de contextes comprend en outre :

une mémoire cache de contextes (8) utilisée exclusivement pour sauvegarder et restaurer des contextes, la mémoire cache de contextes comprenant un port de lecture (83) et un port d'écriture (82), connectés directement au fichier de registre (1) et intégrés dans une unité de traitement centrale sur une puce, la mémoire cache de contextes (8) n'étant pas connectée à une mémoire (20) par un bus (40), lequel connecte mutuellement la mémoire (20), une mémoire cache d'instructions (3) et une mémoire cache de données (6), et étant indépendante d'un système de mémoire incluant la mémoire (20), la mémoire cache d'instructions (3) et la mémoire cache de données (6), en vue de mettre en oeuvre une commutation de contextes à une vitesse de traitement élevée sans occasionner de brouillage à partir du bus (40) ;

un bus de commutation de contextes (12) destiné à connecter le fichier de registre et la mémoire cache de contextes, le bus de commutation de contextes comprenant un bus de restauration (12 - 1) et un bus de sauvegarde (12 - 2) pour connecter le port de lecture et le port d'écriture de la mémoire cache de contextes au port d'écriture de commutation de contextes et au port de lecture de commutation de contextes du fichier de registre, respectivement ; et

une unité de commande de fil d'exécution (9) pour commander un transfert de données entre la mémoire cache de contextes et le fichier de registre, l'unité de commande de fil d'exécution comprenant une table d'identifiants de fils d'exécution (91) pour stocker un identifiant de fil d'exécution destiné à identifier le contexte d'un fil d'exécution stocké dans la mémoire cache de contextes et étant connectée en parallèle à l'unité logique arithmétique (4) et à l'unité d'accès en mémoire (5) ;

dans laquelle, dans le cas d'une opération de commutation de contextes qui exécute une opération de sauvegarde destinée à sauvegarder un contexte en provenance du fichier de registre vers la mémoire cache de contextes et une opération de restauration destinée à restaurer un contexte, de la mémoire cache de contextes vers le fichier de registre en parallèle simultanément, l'unité de commande de fil d'exécution (9) est configurée de manière à recevoir une instruction de commutation de contextes visant à exécuter l'opération de commutation de contextes avec l'identifiant d'un nouveau fil d'exécution à échanger;

l'unité de commande de fil d'exécution (S301) est configurée de manière à obtenir une adresse de restauration où un nouveau contexte à échanger est stocké dans la mémoire cache de contextes et l'identifiant de registre de sauvegarde indiquant l'endroit où le contexte en cours est stocké dans le fichier de registre, en recherchant dans la table d'identifiants de fils d'exécution selon l'identifiant de fil d'exécution ;

l'unité de commande de fil d'exécution (S303) est configurée de manière à envoyer l'adresse obtenue à la mémoire cache de contextes et l'identifiant de registre au fichier de registre en parallèle simultanément ;

le fichier de registre (S303) étant configuré de manière à, selon l'identifiant de registre donné par l'unité de commande de fil d'exécution, générer en sortie les données du contexte à enregistrer du port de lecture de commutation de contextes au bus de sauvegarde pour l'opération de sauvegarde et à conserver les données du contexte à restaurer en parallèle simultanément, envoyées du port de lecture de la mémoire cache de contextes au port d'écriture de commutation de contextes à travers le bus de restauration pour l'opération de restauration ;

la mémoire cache de contextes (S303) étant configurée de manière à, selon l'adresse donnée par l'unité de commande de fil d'exécution, générer en sortie les données du contexte à restaurer du port de lecture au bus de restauration pour l'opération de restauration et à conserver les données du contexte à enregistrer en parallèle simultanément, envoyées du port de lecture de commutation de contextes du fichier de registre au port d'écriture via le bus de sauvegarde en vue de l'opération de sauvegarde ; et

moyennant quoi l'unité de commutation de contextes (S303) commute des contextes par le biais de l'opération de commutation de contextes qui exécute l'opération de restauration et l'opération de sauvegarde en parallèle simultanément.

2. Unité de commutation de contextes selon la revendication 1, dans laquelle le bus de commutation de

contextes présente une largeur de bus supérieure à la largeur de bits du fichier de registre.

3. Unité de commutation de contextes selon la revendication 1, dans laquelle l'unité de commande de fil d'exécution comprend autant de tables d'identifiants de fils d'exécution que ce qui est nécessaire pour identifier les contextes mis en cache dans la mémoire cache de contextes.

4. Unité de commutation de contextes selon l'une quelconque des revendications 1 à 3, dans laquelle l'unité de commande de fil d'exécution sauvegarde le contexte du fil d'exécution en cours, du fichier de registre vers la mémoire cache de contextes, et envoie le contexte d'un nouveau fil d'exécution en provenance de la mémoire cache de contextes au fichier de registre, en parallèle, en vue d'échanger automatiquement un nombre requis d'éléments de données entre le fichier de registre et la mémoire cache de contextes, lorsqu'un logiciel, par exemple un système d'exploitation, émet une instruction d'échange visant à permuter des contextes, comportant un identifiant de fil d'exécution en qualité d'opérande, si l'instruction d'échange est exécutée.

5. Unité de commutation de contextes selon l'une quelconque des revendications 1 à 4, dans laquelle l'unité de commande de fil d'exécution transfère les données d'un contexte, du fichier de registre vers la mémoire cache de contextes, et ne transfère pas les données d'un contexte, de la mémoire cache de contextes au fichier de registre, lorsqu'un logiciel, par exemple un système d'exploitation, émet une instruction de sauvegarde visant à sauvegarder un contexte, comportant un identifiant de fil d'exécution en qualité d'opérande, si l'instruction de sauvegarde est exécutée.

6. Unité de commutation de contextes selon l'une quelconque des revendications 1 à 4, dans laquelle l'unité de commande de fil d'exécution transfère les données d'un contexte, de la mémoire cache de contextes vers le fichier de registre, et ne transfère pas les données d'un contexte, du fichier de registre vers la mémoire cache de contextes, lorsqu'un logiciel, par exemple un système d'exploitation, émet une instruction de restauration visant à restaurer un contexte, comportant un identifiant de fil d'exécution en qualité d'opérande, si l'instruction de restauration est exécutée.

7. Unité de traitement centrale comprenant :

une unité de commutation de contextes selon l'une quelconque des revendications 1 à 6 ;
une mémoire cache d'instructions (3) pour mettre en cache une instruction et une mémoire cache de données (6) pour mettre en cache des données ;
une unité de recherche d'instructions (2) pour récupérer une instruction à partir de la mémoire cache d'instructions et décoder l'instruction ;
une unité logique arithmétique (4) pour mettre en oeuvre une opération selon l'instruction ;
une unité d'accès en mémoire (5) pour accéder à la mémoire cache de données et à la mémoire, et pour exécuter une opération de chargement ou de stockage ;
un bus arithmétique (11) pour connecter le fichier de registre, l'unité logique arithmétique, l'unité d'accès en mémoire, et l'unité de commande de fil d'exécution, en parallèle.

8. Procédé de commutation de contextes destiné à commuter une pluralité de contextes en utilisant une unité de commutation de contextes comprenant :

un fichier de registre (1) dans lequel est stocké un contexte connexe à un fil d'exécution destiné à être exécuté par une unité logique arithmétique ou une unité d'accès en mémoire, le fichier de registre comprenant un port de lecture de registre (15), un port d'écriture de registre (16), un port de lecture de commutation de contextes (17), et un port d'écriture de commutation de contextes (18) ;
une mémoire cache de contextes (8) utilisée exclusivement pour sauvegarder et restaurer des contextes, la mémoire cache de contextes comprenant un port de lecture (83) et un port d'écriture (82), connectés directement au fichier de registre (1) et intégrés dans une unité de traitement centrale sur une puce, la mémoire cache de contextes (8) n'étant pas connectée à une mémoire (20) par un bus (40), lequel connecte mutuellement la mémoire (20), une mémoire cache d'instructions (3) et une mémoire cache de données (6), et étant indépendante d'un système de mémoire incluant la mémoire (20), la mémoire cache d'instructions (3) et la mémoire cache de données (6), en vue de mettre en oeuvre une commutation de contextes à une vitesse de traitement élevée sans occasionner de brouillage à partir du bus (40) ;
un bus de commutation de contextes (12) destiné à connecter le fichier de registre et la mémoire cache de contextes, le bus de commutation de contextes comprenant un bus de restauration (12 - 1) et un bus de sauvegarde (12 - 2) pour connecter le port de lecture et le port d'écriture de la mémoire cache de contextes au port d'écriture de commutation de contextes et au port de lecture de commutation de contextes du fichier de registre, respectivement ;
une unité de commande de fil d'exécution

(S300) pour commander un transfert de données entre la mémoire cache de contextes et le fichier de registre, l'unité de commande de fil d'exécution comprenant une table d'identifiants de fils d'exécution (91) pour stocker un identifiant de fil d'exécution destiné à identifier le contexte d'un fil d'exécution stocké dans la mémoire cache de contextes et étant connectée en parallèle à l'unité logique arithmétique et à l'unité d'accès en mémoire ; ledit procédé de commutation de contextes comportant les étapes ci-dessous, dans lesquelles :

dans le cas d'une opération de commutation de contextes qui exécute une opération de sauvegarde destinée à sauvegarder un contexte en provenance du fichier de registre vers la mémoire cache de contextes et une opération de restauration destinée à restaurer un contexte, de la mémoire cache de contextes vers le fichier de registre en parallèle simultanément, l'unité de commande de fil d'exécution (9) reçoit une instruction de commutation de contextes visant à exécuter l'opération de commutation de contextes et l'identifiant d'un nouveau fil d'exécution à échanger ;

l'unité de commande de fil d'exécution (S301) obtient une adresse de restauration où un nouveau contexte à échanger est stocké dans la mémoire cache de contextes et l'identifiant de registre de sauvegarde indiquant l'endroit où le contexte en cours est stocké dans le fichier de registre, en recherchant dans la table d'identifiants de fils d'exécution selon l'identifiant de fil d'exécution ;

l'unité de commande de fil d'exécution (S303) envoie l'adresse obtenue à la mémoire cache de contextes et l'identifiant de registre au fichier de registre en parallèle simultanément ;

le fichier de registre (S303), selon l'identifiant de registre donné par l'unité de commande de fil d'exécution, génère en sortie les données du contexte à sauvegarder du port de lecture de commutation de contextes au bus de sauvegarde pour l'opération de sauvegarde, et conserve les données du contexte à restaurer en parallèle simultanément, envoyées du port de lecture de la mémoire cache de contextes au port d'écriture de commutation de contextes à travers le bus de restauration pour l'opération de restauration ;

la mémoire cache de contextes (S303), selon l'adresse donnée par l'unité de commande de fil d'exécution, génère en sortie les données du contexte à restaurer du port de lecture au bus de restauration pour l'opération de restauration, et conserve les données du contexte à enregistrer en parallèle simultanément, envoyées du port de lecture de commutation de contextes du fichier de registre au port d'écriture via le bus de sauvegarde en vue de l'opération de sauvegarde ; et

moyennant quoi l'unité de commutation de contextes (S303) commute des contextes par le biais de l'opération de commutation de contextes qui exécute l'opération de restauration et l'opération de sauvegarde en parallèle simultanément.

9. Procédé de commutation de contextes selon la revendication 8, comprenant l'étape consistant à sauvegarder le contexte du fil d'exécution en cours, du fichier de registre vers la mémoire cache de contextes, et l'étape consistant à envoyer le contexte d'un nouveau fil d'exécution, de la mémoire cache de contextes au fichier de registre en parallèle, en vue de permuter automatiquement un nombre requis d'éléments de données entre le fichier de registre et la mémoire cache de contextes, lorsqu'un logiciel, par exemple un système d'exploitation, émet une instruction d'échange visant à permuter des contextes, comportant un identifiant de fil d'exécution en qualité d'opérande, si l'instruction d'échange est exécutée.

10. Procédé de commutation de contextes selon la revendication 8, comprenant l'étape consistant à transférer les données d'un contexte, du fichier de registre vers la mémoire cache de contextes, et l'étape consistant à ne pas transférer les données d'un contexte, de la mémoire cache de contextes au fichier de registre, lorsqu'un logiciel, par exemple un système d'exploitation, émet une instruction de sauvegarde visant à sauvegarder un contexte, comportant un identifiant de fil d'exécution en qualité d'opérande, si l'instruction de sauvegarde est exécutée.

11. Procédé de commutation de contextes selon la revendication 8, comprenant l'étape consistant à transférer les données d'un contexte, de la mémoire cache de contextes vers le fichier de registre, et l'étape consistant à ne pas transférer les données d'un contexte, du fichier de registre vers la mémoire cache de contextes, lorsqu'un logiciel, par exemple un système d'exploitation, émet une instruction de restauration visant à restaurer un contexte, comportant un identifiant de fil d'exécution en qualité d'opérande, si l'instruction de restauration est exécutée.

12. Programme de commutation de contextes destiné à commuter une pluralité de contextes sur un ordinateur en utilisant une unité de commutation de contextes comprenant :

un fichier de registre (1) dans lequel est stocké un contexte connexe à un fil d'exécution destiné à être exécuté par une unité logique arithmétique ou une unité d'accès en mémoire, le fichier de registre comprenant un port de lecture de

registre (15), un port d'écriture de registre (16), un port de lecture de commutation de contextes (17), et un port d'écriture de commutation de contextes (18) ;

une mémoire cache de contextes (8) utilisée exclusivement pour sauvegarder et restaurer des contextes, la mémoire cache de contextes comprenant un port de lecture (83) et un port d'écriture (82), connectés directement au fichier de registre (1) et intégrés dans une unité de traitement centrale sur une puce, la mémoire cache de contextes (8) n'étant pas connectée à une mémoire (20) par un bus (40), lequel connecte mutuellement la mémoire (20), une mémoire cache d'instructions (3) et une mémoire cache de données (6), et étant indépendante d'un système de mémoire incluant la mémoire (20), la mémoire cache d'instructions (3) et la mémoire cache de données (6), en vue de mettre en oeuvre une commutation de contextes à une vitesse de traitement élevée sans occasionner de brouillage à partir du bus (40) ; et

un bus de commutation de contextes (12) destiné à connecter le fichier de registre et la mémoire cache de contextes, le bus de commutation de contextes comprenant un bus de restauration (12 - 1) et un bus de sauvegarde (12 - 2) pour connecter le port de lecture et le port d'écriture de la mémoire cache de contextes au port d'écriture de commutation de contextes et au port de lecture de commutation de contextes du fichier de registre, respectivement ;

une unité de commande de fil d'exécution (9) pour commander un transfert de données entre la mémoire cache de contextes et le fichier de registre, l'unité de commande de fil d'exécution comprenant une table d'identifiants de fils d'exécution (91) pour stocker un identifiant de fil d'exécution destiné à identifier le contexte d'un fil d'exécution stocké dans la mémoire cache de contextes et étant connectée en parallèle à l'unité logique arithmétique et à l'unité d'accès en mémoire ; dans lequel le programme de commutation de contextes amène l'ordinateur à mettre en oeuvre :

une étape dans laquelle, dans le cas d'une opération de commutation de contextes qui exécute une opération de sauvegarde destinée à sauvegarder un contexte en provenance du fichier de registre vers la mémoire cache de contextes, et une opération de restauration destinée à restaurer un contexte, de la mémoire cache de contextes vers le fichier de registre en parallèle simultanément, l'unité de commande de fil d'exécution (9) reçoit une instruction de commutation de contextes visant à exécuter l'opération de commutation de contextes, avec l'identifiant d'un nouveau fil d'exécution à échanger ;

une étape dans laquelle l'unité de commande de fil d'exécution obtient une adresse de restauration où un nouveau contexte à échanger est stocké dans la mémoire cache de contextes et un identifiant de registre de sauvegarde indiquant l'endroit où le contexte en cours est stocké dans le fichier de registre, en recherchant dans la table d'identifiants de fils d'exécution selon l'identifiant de fil d'exécution ;

une étape dans laquelle l'unité de commande de fil d'exécution (S303) envoie l'adresse obtenue à la mémoire cache de contextes et envoie l'identifiant de registre au fichier de registre en parallèle simultanément ;

une étape dans laquelle le fichier de registre (S303), selon l'identifiant de registre donné par l'unité de commande de fil d'exécution, génère en sortie les données du contexte à sauvegarder, du port de lecture de commutation de contextes au bus de sauvegarde pour l'opération de sauvegarde, et conserve les données du contexte à restaurer en parallèle simultanément, envoyées du port de lecture de la mémoire cache de contextes au port d'écriture de commutation de contextes à travers le bus de restauration pour l'opération de restauration ;

une étape dans laquelle la mémoire cache de contextes (S303), selon l'adresse donnée par l'unité de commande de fil d'exécution, génère en sortie les données du contexte à restaurer du port de lecture au bus de restauration pour l'opération de restauration, et conserve les données du contexte à enregistrer en parallèle simultanément, envoyées du port de lecture de commutation de contextes du fichier de registre au port d'écriture via le bus de sauvegarde en vue de l'opération de sauvegarde ; et

moyennant quoi l'unité de commutation de contextes (S303) commute des contextes par le biais de l'opération de commutation de contextes qui exécute l'opération de restauration et l'opération de sauvegarde en parallèle simultanément.

13. Support d'enregistrement lisible par un ordinateur, dans lequel est enregistré un programme de commutation de contextes selon la revendication 12.

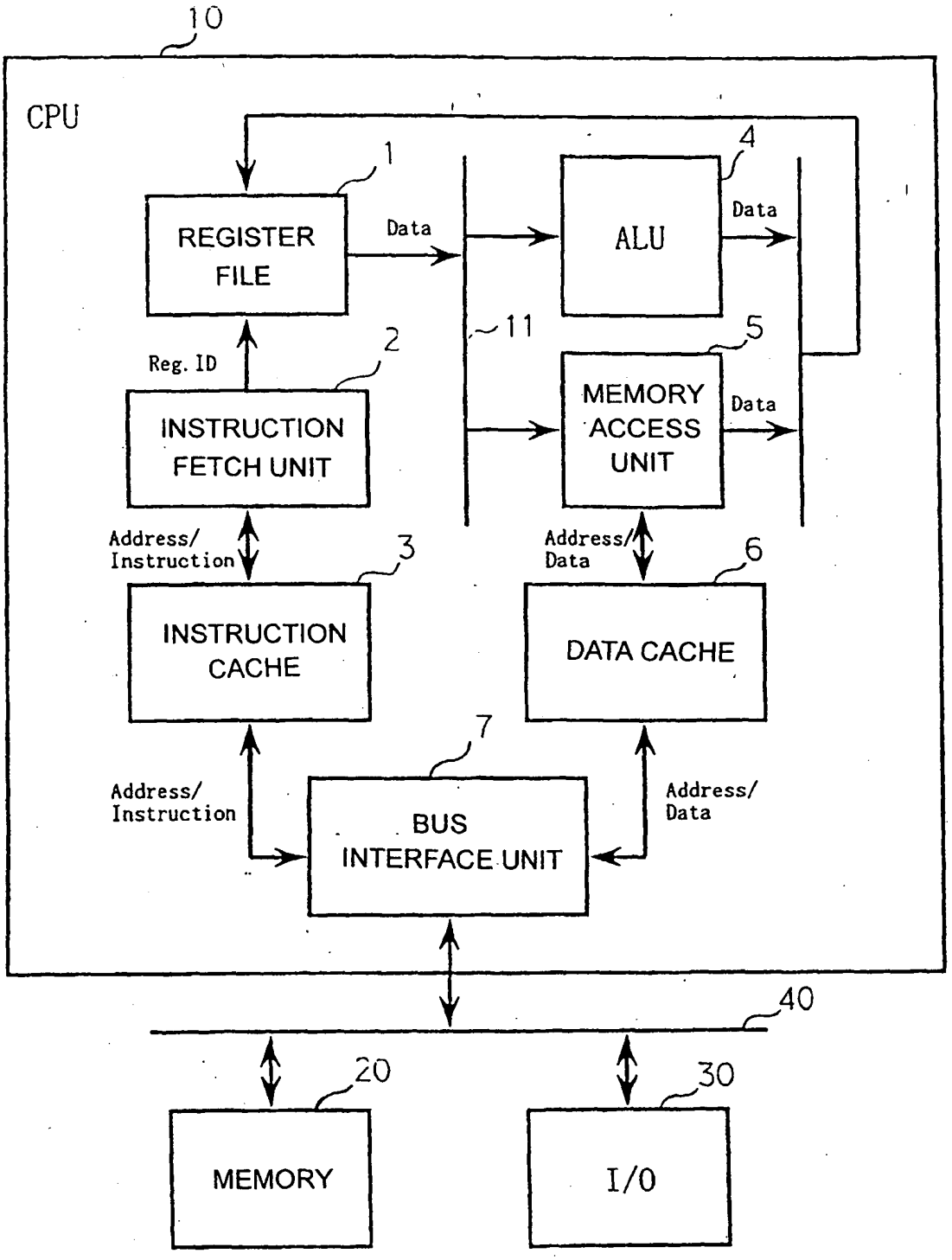
10
CPU
1
REGISTER FILE
Data
4
ALU
Data
11
2
Reg. ID
INSTRUCTION FETCH UNIT
5
MEMORY ACCESS UNIT
Data
Address/ Instruction
3
Address/ Data
6
INSTRUCTION CACHE
DATA CACHE
7
Address/ Instruction
BUS INTERFACE UNIT
Address/ Data
40
20
30
MEMORY
I/O

FIG.1

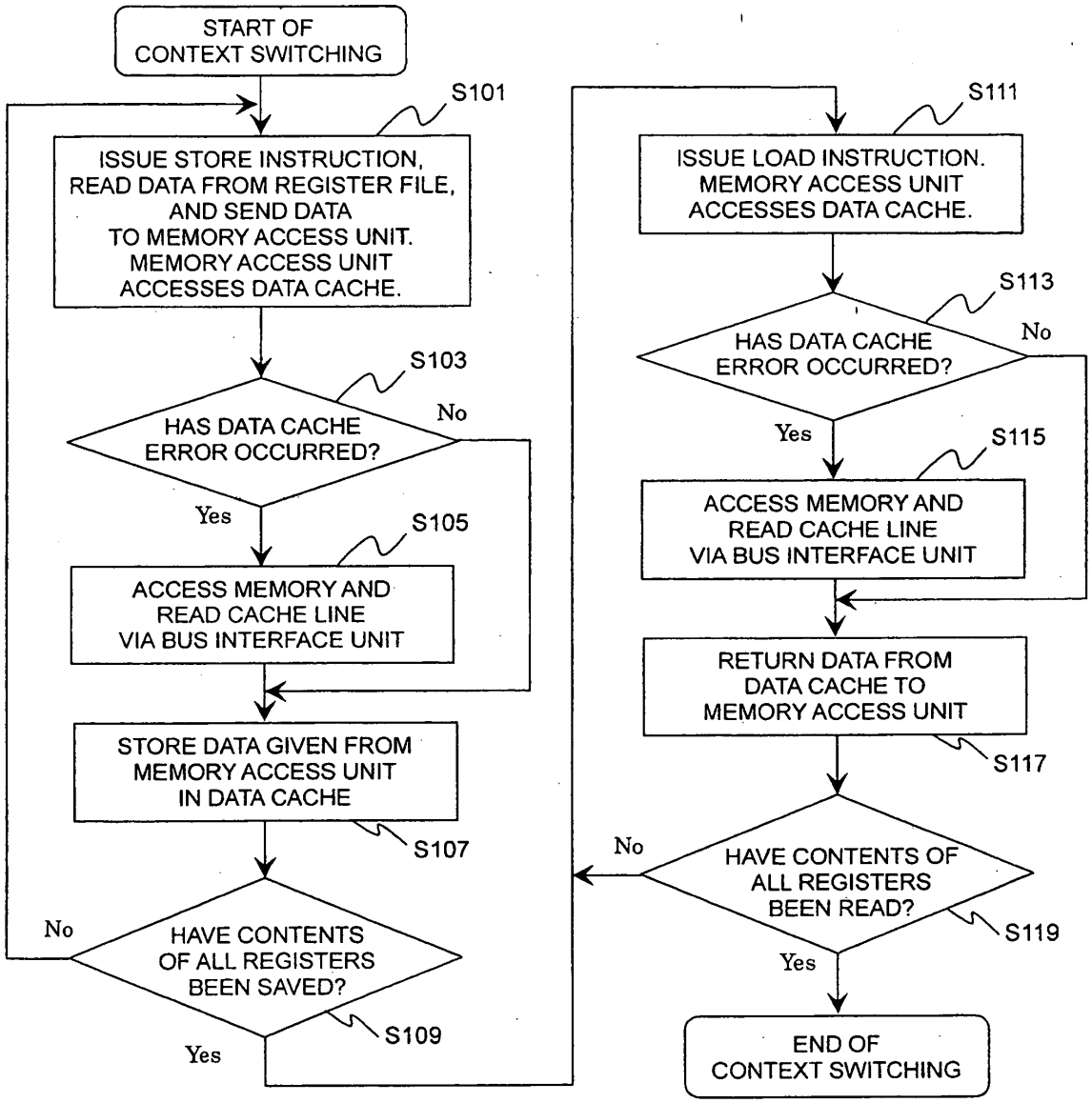
START OF CONTEXT SWITCHING
S101
ISSUE STORE INSTRUCTION, READ DATA FROM REGISTER FILE, AND SEND DATA TO MEMORY ACCESS UNIT. MEMORY ACCESS UNIT ACCESSES DATA CACHE.
S103
HAS DATA CACHE ERROR OCCURRED?
No
Yes
S105
ACCESS MEMORY AND READ CACHE LINE VIA BUS INTERFACE UNIT
STORE DATA GIVEN FROM MEMORY ACCESS UNIT IN DATA CACHE
S107
No
HAVE CONTENTS OF ALL REGISTERS BEEN SAVED?
S109
Yes
S111
ISSUE LOAD INSTRUCTION. MEMORY ACCESS UNIT ACCESSES DATA CACHE.
S113
HAS DATA CACHE ERROR OCCURRED?
No
Yes
S115
ACCESS MEMORY AND READ CACHE LINE VIA BUS INTERFACE UNIT
RETURN DATA FROM DATA CACHE TO MEMORY ACCESS UNIT
S117
No
HAVE CONTENTS OF ALL REGISTERS BEEN READ?
S119
Yes
END OF CONTEXT SWITCHING

FIG.2

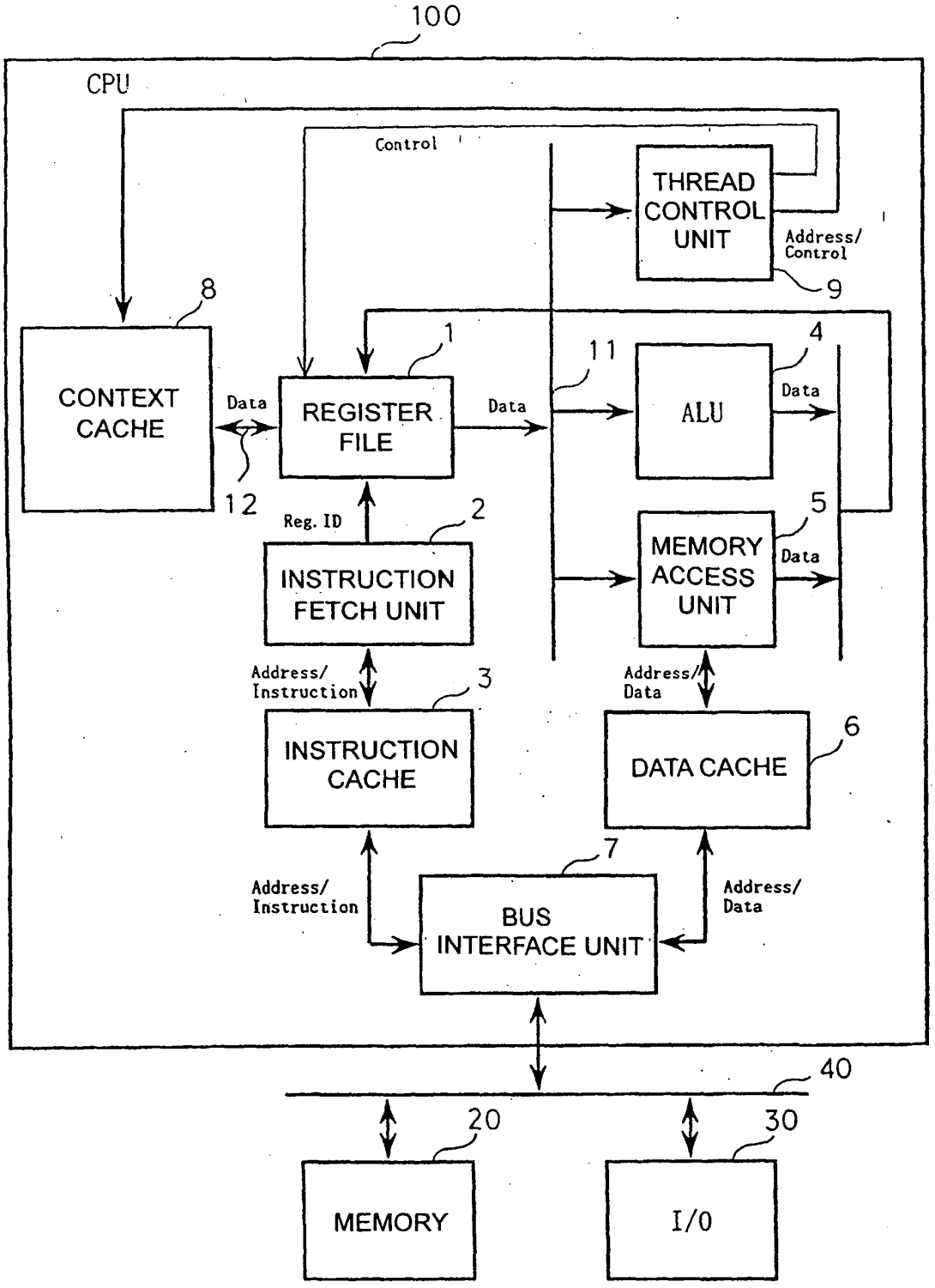
100
CPU
Control
THREAD CONTROL UNIT
Address/ Control
9
8
CONTEXT CACHE
Data
12
1
REGISTER FILE
Data
11
ALU
4
Data
Reg. ID
2
INSTRUCTION FETCH UNIT
MEMORY ACCESS UNIT
5
Data
Address/ Instruction
3
INSTRUCTION CACHE
Address/ Data
DATA CACHE
6
7
Address/ Instruction
BUS INTERFACE UNIT
Address/ Data
40
20
MEMORY
30
I/O

FIG.3

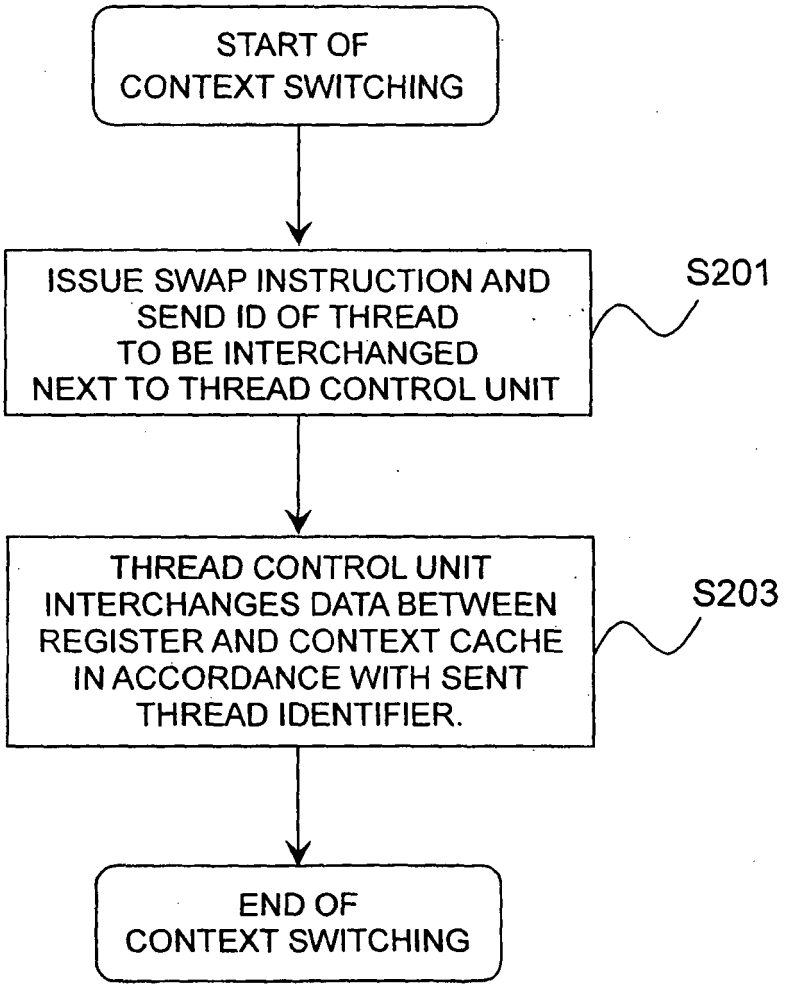
START OF
CONTEXT SWITCHING
ISSUE SWAP INSTRUCTION AND
SEND ID OF THREAD
TO BE INTERCHANGED
NEXT TO THREAD CONTROL UNIT
S201
THREAD CONTROL UNIT
INTERCHANGES DATA BETWEEN
REGISTER AND CONTEXT CACHE
IN ACCORDANCE WITH SENT
THREAD IDENTIFIER.
S203
END OF
CONTEXT SWITCHING

FIG.4

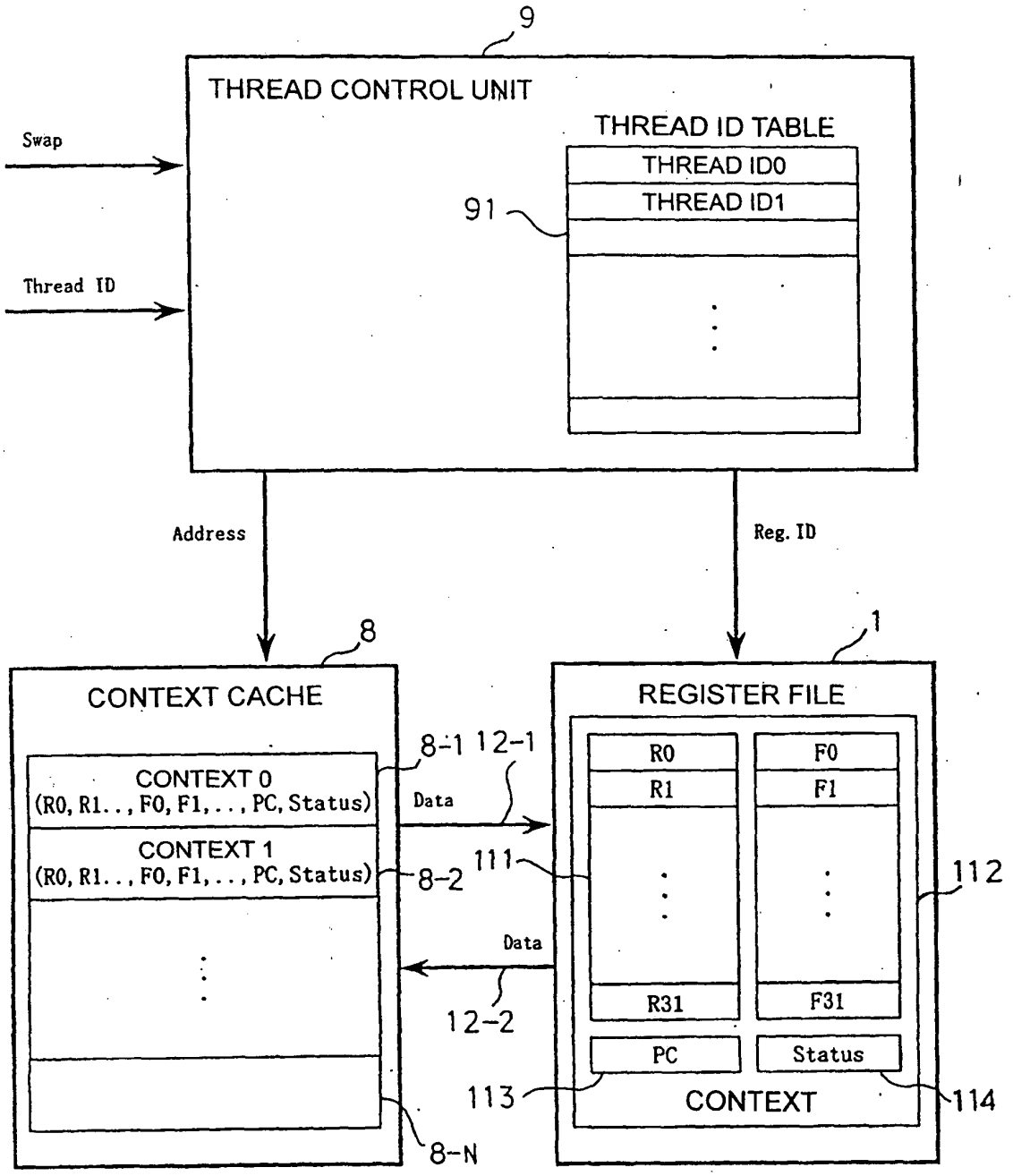
9
THREAD CONTROL UNIT
THREAD ID TABLE
THREAD ID0
THREAD ID1
91
Swap
Thread ID
Address
Reg. ID
8
CONTEXT CACHE
CONTEXT 0
(R0, R1. . , F0, F1, . . , PC, Status)
CONTEXT 1
(R0, R1. . , F0, F1, . . , PC, Status)
8-1
12-1
Data
8-2
111
Data
12-2
8-N
1
REGISTER FILE
R0
F0
R1
F1
R31
F31
PC
Status
CONTEXT
112
113
114

FIG.5

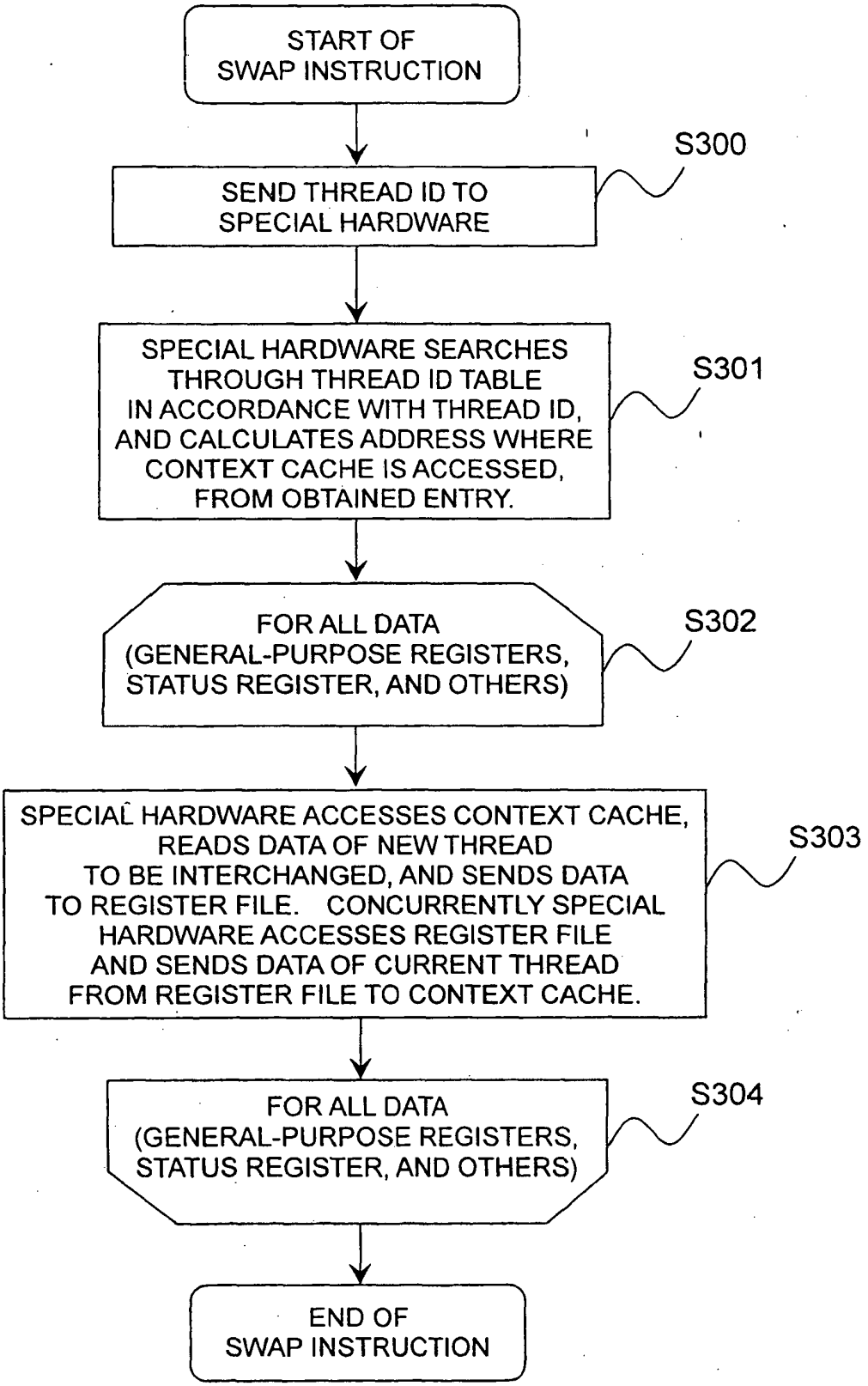
START OF SWAP INSTRUCTION
S300
SEND THREAD ID TO SPECIAL HARDWARE
SPECIAL HARDWARE SEARCHES THROUGH THREAD ID TABLE IN ACCORDANCE WITH THREAD ID, AND CALCULATES ADDRESS WHERE CONTEXT CACHE IS ACCESSED, FROM OBTAINED ENTRY.
S301
FOR ALL DATA (GENERAL-PURPOSE REGISTERS, STATUS REGISTER, AND OTHERS)
S302
SPECIAL HARDWARE ACCESSES CONTEXT CACHE, READS DATA OF NEW THREAD TO BE INTERCHANGED, AND SENDS DATA TO REGISTER FILE. CONCURRENTLY SPECIAL HARDWARE ACCESSES REGISTER FILE AND SENDS DATA OF CURRENT THREAD FROM REGISTER FILE TO CONTEXT CACHE.
S303
FOR ALL DATA (GENERAL-PURPOSE REGISTERS, STATUS REGISTER, AND OTHERS)
S304
END OF SWAP INSTRUCTION

FIG.6

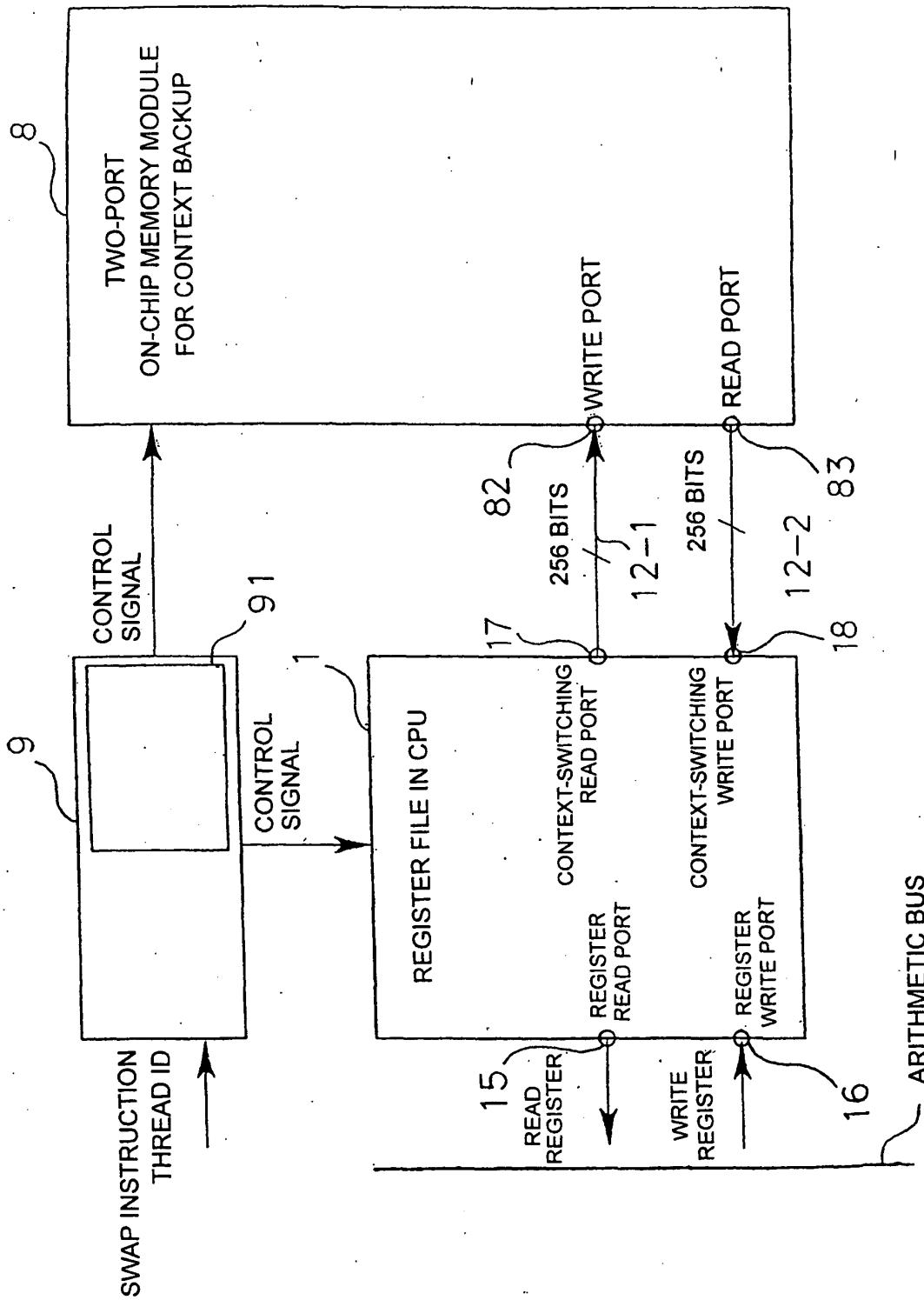
8
TWO-PORT
ON-CHIP MEMORY MODULE
FOR CONTEXT BACKUP
WRITE PORT
READ PORT
82
83
256 BITS
256 BITS
12-1
12-2
CONTROL
SIGNAL
91
17
18
9
1
CONTROL
SIGNAL
REGISTER FILE IN CPU
CONTEXT-SWITCHING
READ PORT
CONTEXT-SWITCHING
WRITE PORT
REGISTER
READ PORT
REGISTER
WRITE PORT
SWAP INSTRUCTION
THREAD ID
15
16
READ
REGISTER
WRITE
REGISTER
ARITHMETIC BUS

FIG.7

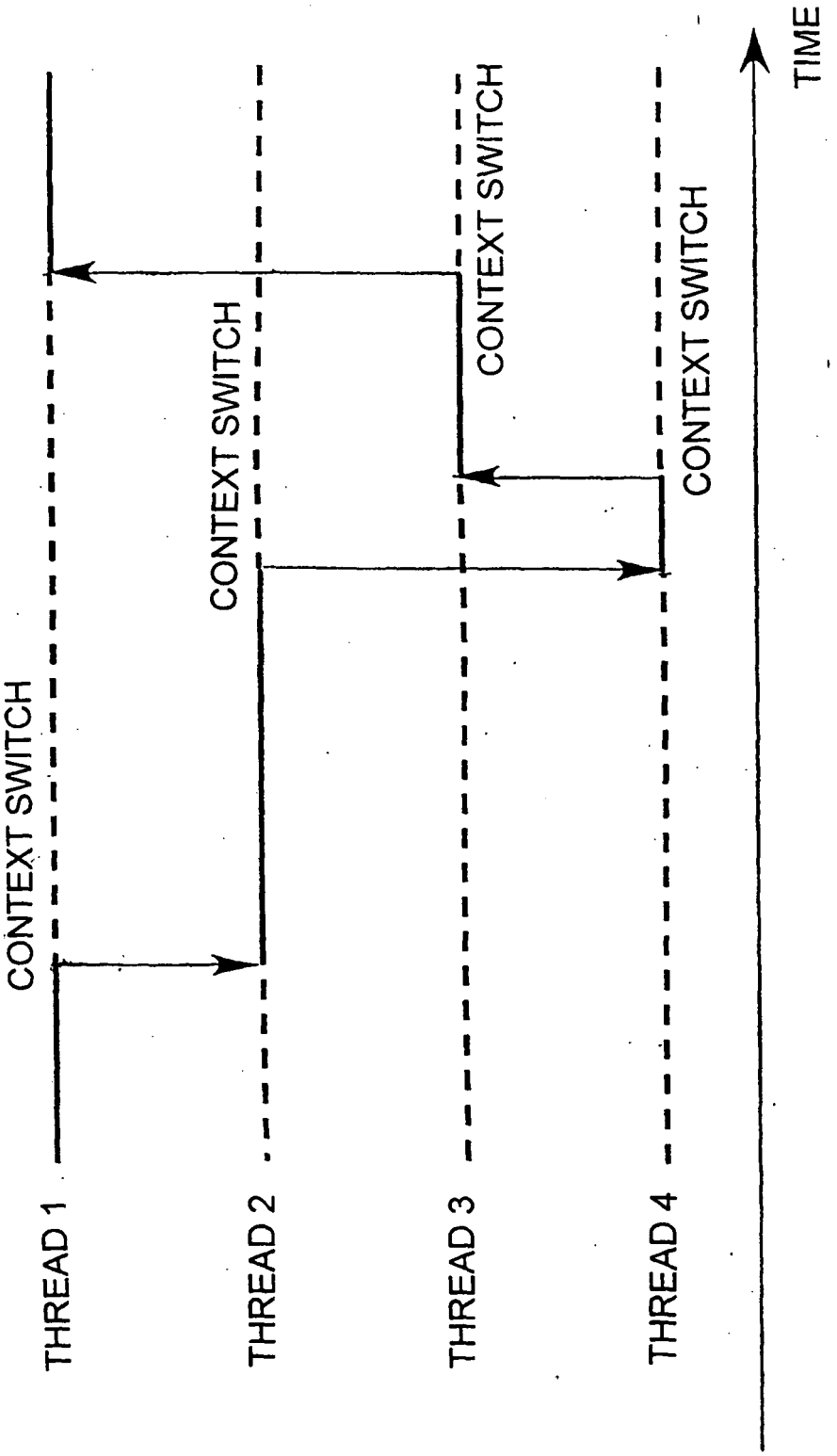
THREAD 1
THREAD 2
THREAD 3
THREAD 4
CONTEXT SWITCH
CONTEXT SWITCH
CONTEXT SWITCH
CONTEXT SWITCH
TIME

FIG.8

REFERENCES CITED IN THE DESCRIPTION

This list of references cited by the applicant is for the reader's convenience only. It does not form part of the European patent document. Even though great care has been taken in compiling the references, errors or omissions cannot be excluded and the EPO disclaims all liability in this regard.

Patent documents cited in the description

- JP HEI07141208 B **[0005]**
- JP HEI09212371 B **[0006]**
- US 6401155 B **[0007]**
- WO 9956209 A **[0010]**

Non-patent literature cited in the description

- **Guangzuo C et al.** Parallel replacement mechanism for multithread. *Advances in Parallel and Distributed Computing 1997. Proceedings,* 1997, 338-344 **[0009]**
- **Wills D S et al.** An ultra-light processor for high-throughput applications. *Computer design: VLSI in computers and processors, 1993. ICCD 93 Proceedings,* 1993, 410-414 **[0010]**