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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
20 September 2012 (20.09.2012)

WIPO | PCT

(10) International Publication Number
WO 2012/125201 A1

(51) International Patent Classification:
G06F 9/06 (2006.01) *G06F 12/08* (2006.01)
G06F 9/22 (2006.01) *G06F 11/07* (2006.01)

(21) International Application Number:
PCT/US2011/067963

(22) International Filing Date:
29 December 2011 (29.12.2011)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
13/048,053 15 March 2011 (15.03.2011) US

(71) Applicant *(for all designated States except US)*: **INTEL CORPORATION** [US/US]; 2200 Mission College Boulevard, Santa Clara, California 95052 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **GINZBURG, Boris** [IL/IL]; Beilis 6/3, 34814 Haifa (IL). **NATANZON, Esfirush** [IL/IL]; Lea 50/6, 33405 Haifa (IL). **OSADCHIY, Ilya** [IL/IL]; Galil 39/5, 32000 Haifa (IL). **RONEN, Ronny** [IL/IL]; 156 Aba Hushi Avenue, 34988 Haifa (IL). **WEISSMANN, Eliezer** [IL/IL]; Albert Switcher 10, 34995 Haifa (IL). **ZACH, Yoav** [IL/IL]; 11 Mechora Street, 37091 Pardes Hana Karkur (IL). **FARRELL, Robert, L.** [US/US]; 6275 Barcelona Court, Granite Bay, California 95746 (US).

(74) Agent: **TROP, Timothy, N.**; Trop, Pruner & Hu, P.C., 1616 S. Voss Rd., Ste. 750, Houston, Texas 77057-2631 (US).

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

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

[Continued on next page]

(54) Title: PAGE FAULT HANDLING MECHANISM

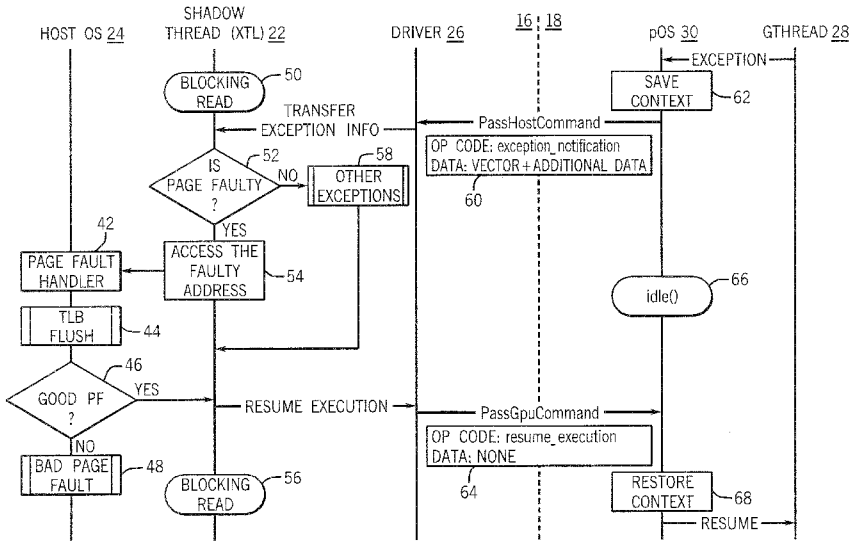

FIG. 3

(57) Abstract: Page faults arising in a graphics processing unit may be handled by an operating system running on the central processing unit. In some embodiments, this means that unpinned memory can be used for the graphics processing unit. Using unpinned memory in the graphics processing unit may expand the capabilities of the graphics processing unit in some cases.

WO 2012/125201 A1

Declarations under Rule 4.17:

— *as to the identity of the inventor (Rule 4.17(i))*

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

— *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii))*

Published:

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

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

Page Fault Handling Mechanism

Background

[0001] This relates generally to processing units to handle page faults that arise in specialized devices, such as graphics processing units.

[0002] A page fault is an interrupt that occurs when software attempts to read from or to write to a virtual memory location that is marked as "not present" or when a page permission attribute prohibits corresponding access. Virtual memory systems maintain such status information about every page in a virtual memory address space. These pages are mapped onto physical addresses or are "not present" in physical memory. For example, when a read or write is detected to an unmapped virtual address or when page access permissions are violated, the device “page walker” generates a page fault interrupt. The operating system (OS) page fault handler responds to this page fault by swapping in data from disk to system memory, or by allocating new page (“copy on write”) and updating the status information in page table.

[0003] In order to avoid the possibility of page faults in graphics processing units, graphics processing units are generally constrained to using pinned memory. This means that in the last case, the page which is in graphics processor use, is pre-allocated and cannot be swapped to disk or remapped to new location in system memory.

[0004] In conventional systems, separate page tables are used by the central processing unit and the graphics processing unit. The operating system manages the host page table used by the central processing unit and a graphics processing unit driver manages the page table used by the graphics processing unit. The graphics processing unit driver copies data from user space into the driver memory for processing on the graphics processing unit. Complex data structures must be repacked into an array when pointers are replaced by offsets. The overhead related to copying and repacking limits graphics processing unit applications where data is represented as arrays. Thus, graphics processing units may be of limited value in

some applications, including those that involve complex data structures such as databases.

Brief Description Of The Drawings

[0005] Figure 1 is a schematic depiction of one embodiment of the present invention;

[0006] Figure 2 is extended thread and memory model in accordance with one embodiment of the present invention;

[0007] Figure 3 is a flow chart for page fault handling in accordance with one embodiment of the present invention; and

[0008] Figure 4 is a system depiction for one embodiment.

Detailed Description

[0009] In some embodiments, graphics processing applications may use complex data structures, such as databases, by using a shared virtual memory model that does not require pinning of shared memory. Pinning of shared virtual memory reduces an operating system's ability to manage system memory. In some embodiments, unpinned shared virtual memory may be used on the graphics processing unit when there is no guarantee that the page used by the graphics processing unit is present in system memory.

[0010] The graphics processing unit driver propagates page faults on the graphics processing unit to a shadow thread on the host/central processing unit. The host then emulates the page faults as if they occurred on the central processing unit to trigger the operating system to resolve the fault for the benefit of the graphics processing unit.

[0011] While the term graphics processing unit is used in the present application, it should be understood that the graphics processing unit may or may not be a separate integrated circuit. The present invention is applicable to situations where the graphics processing unit and the central processing unit are integrated into one integrated circuit.

[0012] In addition, while an example relating to graphics processing is given herein, in other embodiments, the same page fault handling techniques may be used in other specialized processing units, such as video processing, cards and input/output devices. In general, the page fault handling techniques may be used with any device that may experience page faults and which is accompanied by a processor that may act as a proxy to resolve those page faults. As used herein, a processor or processing unit may be a processor, controller, or coprocessor.

[0013] Referring to Figure 1, a host/central processing unit 16 communicates with the graphics processing unit 18. The host central processing unit 16 includes user applications 20 which provide control information to a shadow thread 22. The shadow thread 22 then communicates exceptions and control information to the graphics processing unit driver 26. A shadow thread also communicates with the host operating system 24.

[0014] As shown in Figure 1, the user level 12 includes a shadow thread 22 and the user applications 20, while the kernel level 14 includes a host operating system 24, and the graphics processing unit driver 26. The graphics processing unit driver 26 is a driver for the graphics processing unit even though that driver is resident in the central processing unit 16.

[0015] The graphics processing unit 18 includes, in user level 12, the gthread 28 which sends and receives control and exceptions messages to the operating system 30. A gthread is user code that runs on the graphics processing unit, sharing virtual memory with the parent thread running on the central processing unit. The operating system 30 may be a relatively small operating system, running on the graphics processing unit, that is responsible for graphics processing unit exceptions. It is a small relative to the host operating system 24, as one example.

[0016] User applications 20 are any user process that runs on the central processing unit 16. The user applications 20 spawn threads on the graphics processing unit 18.

[0017] An eXtended Threaded Library or XTL is an extension to create and manage user threads on the graphics processing unit. This library creates the shadow thread for each gthread.

[0018] User applications offload computations to the graphics processing unit using an extension of a traditional multithreaded model such as:

xthread_create (thread, attr, gpu_worker,arg).

[0019] The gthread or worker thread created on the graphics processing unit shares virtual memory with the parent thread. It behaves in the same way as a regular thread in that all standard inter-process synchronization mechanisms, such as Mutex and semaphore, can be used. At the same time, a new shadow thread is created on the host central processing unit 16. This shadow thread works as a proxy for exception handling units and synchronization between threads on the central processing unit and the graphics processing unit.

[0020] In some embodiments, the parent thread, the host shadow thread and the graphics processing unit worker threads may share unpinned virtual memory as shown in Figure 2. Host/central processing unit 16 includes the parent thread 32 that generates the xthread_create() for the shadow thread 22. The shadow thread 22 accesses the shadow stack which is a private address space in the process address space 36. The parent thread 32 also accesses the memory descriptors 34 and the main stack, which is a private address space within the process address space 36. The memory descriptors 34 may also communicate with the gthread worker 28. The gthread worker 28 can access the gthread code within the process space 36 as well as the shared data section and the private gthread stack. The material in the upper blocks corresponds to the process model 38 and the lower blocks correspond to the memory model 40.

[0021] Referring to Figure 3, the page fault handling algorithms may be implemented in hardware, software and/or firmware. In software embodiments, the algorithms may be implemented as computer executable instructions stored on a non-transitory computer readable medium such as an optical, semiconductor or magnetic memory. In Figure 3, the flows for the host operating system 24, the

shadow thread 22, driver 26 of the central processing unit 16, and the operating system 30, gthread 28 in the graphics processing unit 18 are shown as parallel vertical flow paths with interactions between them indicated by a generally horizontal arrows.

[0022] The graphics processing unit operating system 30 initially receives a page fault as indicated by the word "exception" and the corresponding arrow in Figure 3, from the gthread 28. The operating system 30 saves the context (block 62) and sends a message 60 with the page fault information to the driver 26. The message may include an opcode "exception_notification" and data including the vector and additional information. Then the operating system 30 marks the thread as idle(), as indicated in block 66, so the thread is considered "not ready, waiting for page fault resolution" and switches to another thread.

[0023] The driver 26 wakes up the shadow thread 22 and transfers the page fault data to the shadow thread as indicated by the arrow labeled "transfer exception info."

[0024] At 50, the shadow thread performs a blocking read to stop other activities until the page fault is resolved. Then the shadow thread 22 receives the page fault data. After checking to see if the page is faulty (diamond 52), the shadow thread reproduces the same access to the faulty address, as indicated a block 54, if the page is faulty. If the page is not faulty, the flow goes to block 58 to check for other exceptions, bypassing block 54. Then the block read is released at 56.

[0025] The host operating system 24 handles the page fault in the page fault handler 42. Effectively, the host operating system is tricked into handling the exception for the graphics processing unit. Then the translation lookaside buffer (TLB) may be flushed at 44. A check at diamond 46 determines if the page fault is good, i.e. fixed, in which case it advises the shadow thread 22. Otherwise, a bad page fault is indicated at 48, which may, for example, result in an error.

[0026] The shadow thread 22 sends the page fault resolved message (i.e. RESUME EXECUTION) to the driver 26. Then the shadow thread goes to a sleep state waiting for the next message from the driver using another blocking read 56.

[0027] The driver 26 receives the resume execution message from the shadow thread and sends a PassGPUCommand to the operating system 30 as indicated by the block 64. The message may include the opcode to resume execution with no data. The operating system 30 marks the thread as ready for execution, as indicated at 68, and returns from the exception by sending a resume message to the gthread 28.

[0028] The computer system 130, shown in Figure 4, may include a hard drive 134 and a removable medium 136, coupled by a bus 104 to a chipset core logic 110. A keyboard and mouse 120, or other conventional components, may be coupled to the chipset core logic via bus 108. The core logic may couple to the graphics processor 112, via a bus 105, and the central processor 100 in one embodiment. The graphics processor 112 may also be coupled by a bus 106 to a frame buffer 114. The frame buffer 114 may be coupled by a bus 107 to a display screen 118. In one embodiment, a graphics processor 112 may be a multi-threaded, multi-core parallel processor using single instruction multiple data (SIMD) architecture.

[0029] In the case of a software implementation, the pertinent code may be stored in any suitable semiconductor, magnetic, or optical memory, including the main memory 132 (as indicated at 139) or any available memory within the graphics processor. Thus, in one embodiment, the code to perform the sequences of Figure 3 may be stored in a non-transitory machine or computer readable medium, such as the memory 132, and/or the graphics processor 112, and/or the central processor 100 and may be executed by the processor 100 and/or the graphics processor 112 in one embodiment.

[0030] Figure 3 is a flow chart. In some embodiments, the sequences depicted in this flow chart may be implemented in hardware, software, or firmware. In a software embodiment, a non-transitory computer readable medium, such as a semiconductor memory, a magnetic memory, or an optical memory may be used to store instructions and may be executed by a processor to implement the sequences shown in Figure 3.

[0031] The graphics processing techniques described herein may be implemented in various hardware architectures. For example, graphics functionality may be integrated within a chipset. Alternatively, a discrete graphics processor may be used. As still another embodiment, the graphics functions may be implemented by a general purpose processor, including a multicore processor.

[0032] References throughout this specification to "one embodiment" or "an embodiment" mean that a particular feature, structure, or characteristic described in connection with the embodiment is included in at least one implementation encompassed within the present invention. Thus, appearances of the phrase "one embodiment" or "in an embodiment" are not necessarily referring to the same embodiment. Furthermore, the particular features, structures, or characteristics may be instituted in other suitable forms other than the particular embodiment illustrated and all such forms may be encompassed within the claims of the present application.

[0033] While the present invention has been described with respect to a limited number of embodiments, those skilled in the art will appreciate numerous modifications and variations therefrom. It is intended that the appended claims cover all such modifications and variations as fall within the true spirit and scope of this present invention.

What is claimed is:

1. A method comprising:

handling page faults, arising in a first processing unit, by an operating system running on a second processing unit.

2. The method of claim 1 including handling page faults arising in a graphics processing unit using an operating system running on a central processing unit.

3. The method of claim 2 including using a thread running on the central processing unit to reproduce and handle a page fault on the graphics processing unit.

4. The method of claim 2 including using a graphics processing unit operating system to pass a page fault to a driver on the central processing unit.

5. The method of claim 1 including using unpinned shared virtual memory.

6. The method of claim 5 including sharing said unpinned virtual memory between the first and second processing units.

7. A non-transitory computer readable medium storing instructions to enable a first processor to:

handle page faults, arising in a second processor, using an operating system running on said first processor.

8. The medium of claim 7 further storing instructions to handle page faults arising in a graphics processing unit using an operating system running on a central processing unit.

9. The medium of claim 8 further storing instructions to use a thread running on the central processing unit to reproduce and handle a page fault on the graphics processing unit.

10. The medium of claim 8 further storing instructions to use a graphics processing unit operating system to pass a page fault to a driver on the central processing unit.

11. The medium of claim 7 further storing instructions to use unpinned shared virtual memory.

12. The medium of claim 11 further storing instructions to share said unpinned virtual memory between said processors.

13. An apparatus comprising:
a processor to handle page faults arising on another processor; and
a memory coupled to said processor.

14. The apparatus of claim 13 wherein said processor is a central processing unit.

15. The apparatus of claim 13 including another processor which incurs page faults and which transfers said page faults to said processor for handling.

16. The apparatus of claim 13 wherein said another processor is a graphics processing unit.

17. The apparatus of claim 13 wherein said processor uses a thread to reproduce and handle a page fault on said another processor.

18. The apparatus of claim 13 including said processor and said another processor, wherein said another processor to pass a page fault to a driver on said central processing unit.

19. The apparatus of claim 15 wherein said another processor to use unpinned shared virtual memory.

20. The apparatus of claim 20 wherein said processor and said another processor share said unpinned virtual memory.

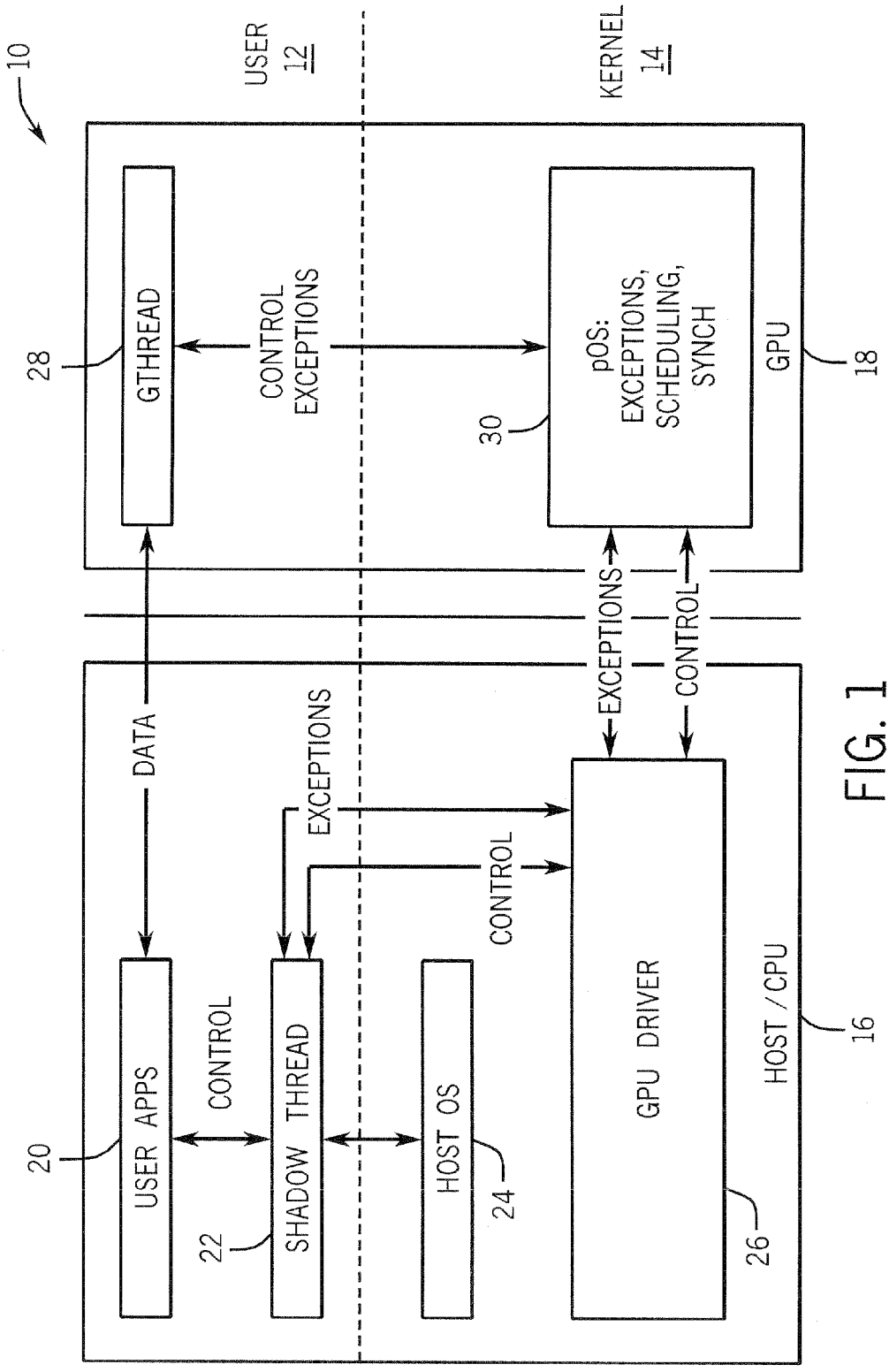
10
USER
12
KERNEL
14
28
GTHREAD
CONTROL
EXCEPTIONS
30
pOS:
EXCEPTIONS,
SCHEDULING,
SYNCH
GPU
18
EXCEPTIONS
CONTROL
DATA
EXCEPTIONS
CONTROL
20
USER APPS
CONTROL
22
SHADOW THREAD
HOST OS
24
GPU DRIVER
26
HOST / CPU
16

FIG. 1

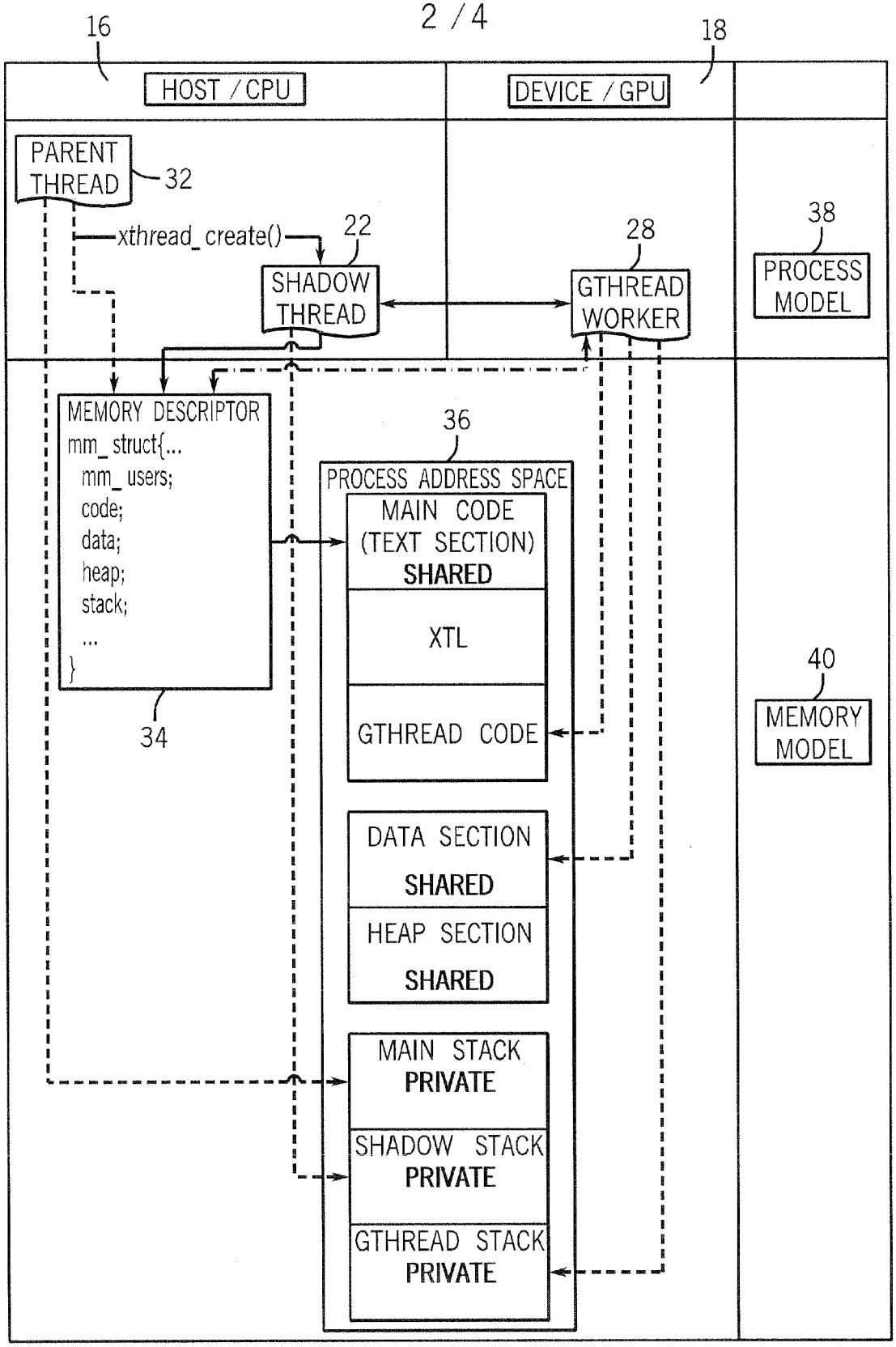
16
18
HOST / CPU
DEVICE / GPU
PARENT THREAD
32
xthread_create()
22
SHADOW THREAD
28
GTHREAD WORKER
38
PROCESS MODEL
MEMORY DESCRIPTOR
mm_struct{...
mm_users;
code;
data;
heap;
stack;
...
}
34
36
PROCESS ADDRESS SPACE
MAIN CODE (TEXT SECTION) SHARED
XTL
GTHREAD CODE
DATA SECTION SHARED
HEAP SECTION SHARED
MAIN STACK PRIVATE
SHADOW STACK PRIVATE
GTHREAD STACK PRIVATE
40
MEMORY MODEL

FIG. 2

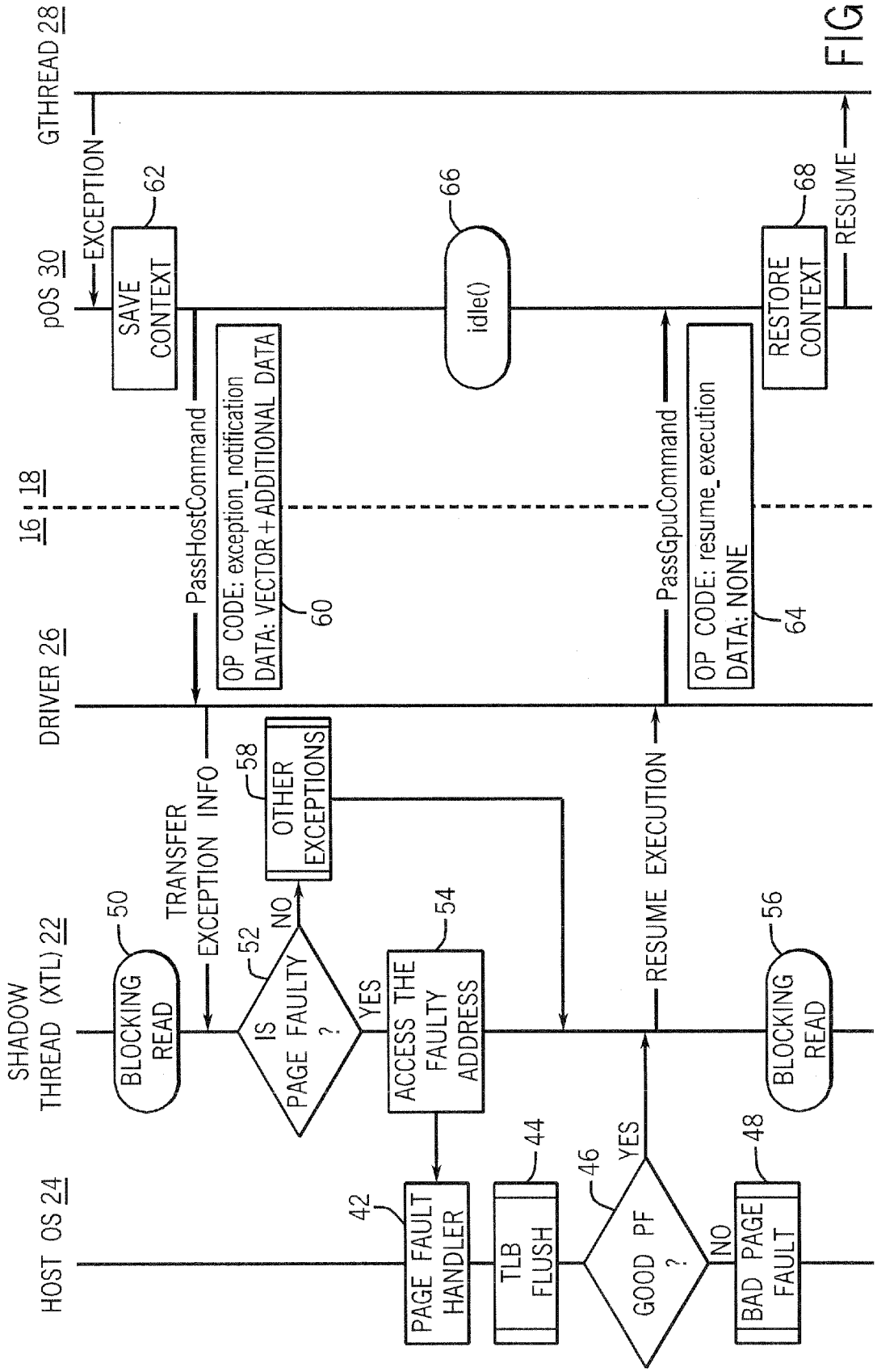
GTHREAD 28
pOS 30
EXCEPTION
62
SAVE CONTEXT
66
idle()
68
RESTORE CONTEXT
RESUME
16 : 18
PassHostCommand
OP CODE: exception_notification
DATA: VECTOR + ADDITIONAL DATA
60
PassGpuCommand
OP CODE: resume_execution
DATA: NONE
64
DRIVER 26
TRANSFER EXCEPTION INFO
58
OTHER EXCEPTIONS
RESUME EXECUTION
SHADOW THREAD (XTL) 22
50
BLOCKING READ
52
IS PAGE FAULTY ?
NO
YES
54
ACCESS THE FAULTY ADDRESS
56
BLOCKING READ
HOST OS 24
42
PAGE FAULT HANDLER
44
TLB FLUSH
46
GOOD PF ?
YES
NO
48
BAD PAGE FAULT

FIG. 3

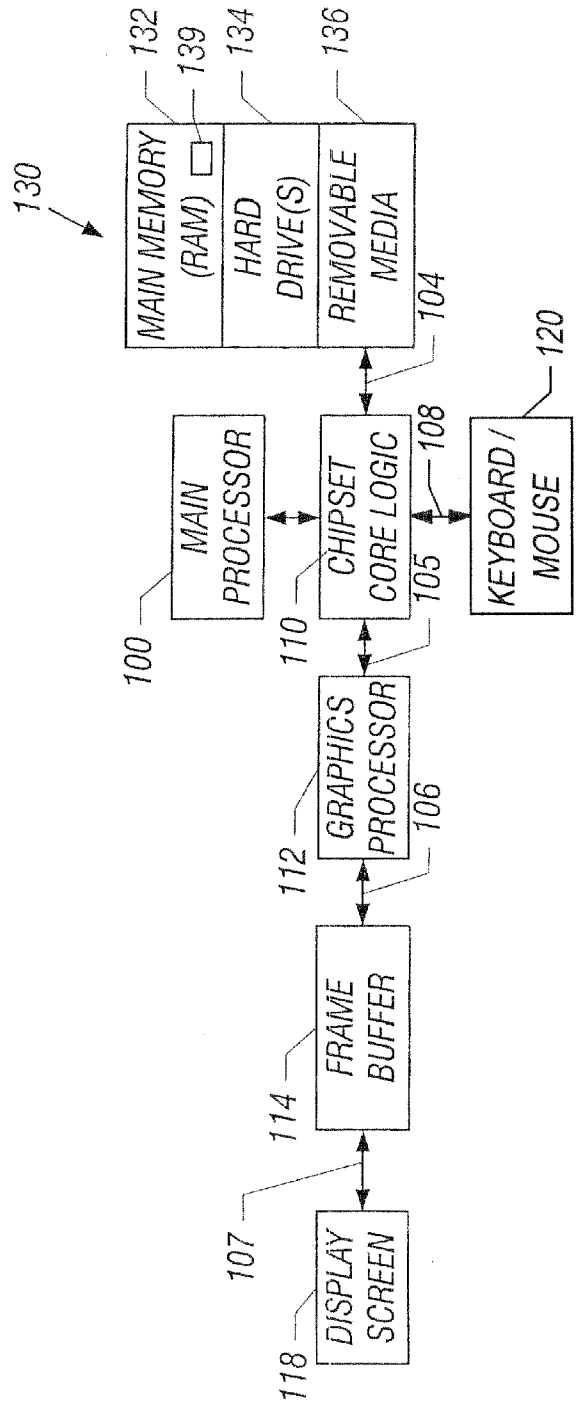
130
MAIN MEMORY (RAM)
HARD DRIVE(S)
REMOVABLE MEDIA
132
139
134
136
104
MAIN PROCESSOR
CHIPSET CORE LOGIC
KEYBOARD / MOUSE
120
108
100
110
105
GRAPHICS PROCESSOR
112
106
FRAME BUFFER
114
107
DISPLAY SCREEN
118

FIG. 4

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2011/067963

A. CLASSIFICATION OF SUBJECT MATTER

G06F 9/06(2006.01)i, G06F 9/22(2006.01)i, G06F 12/08(2006.01)i, G06F 11/07(2006.01)i

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

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

G06F 9/06; G06F 12/10; G06F 12/00; G06F 13/00

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

Korean utility models and applications for utility models
Japanese utility models and applications for utility models

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

eKOMPASS(KIPO internal) & Keywords: "page, fault, handle, processor, memory, operating system"

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 7623134 B1 (DANILAK RADOSLAV) 24 November 2009 See the abstract, column 4 line 17 - column 5 line 52, figures 1-2.	1-20
A	US 2005-0198444 A1 (MICHAEL YODER) 08 September 2005 See the abstract, paragraphs [0006] - [0010], [0026] - [0028], figures 1-5.	1-20
A	US 2007-0150695 A1 (SAMSUNG ELECTRONICS CO., LTD.) 28 June 2007 See the abstract, paragraphs [0050] - [0056], figures 5-6.	1-20

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

☒ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents,such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
23 JULY 2012 (23.07.2012)	**23 JULY 2012 (23.07.2012)**

Name and mailing address of the ISA/KR

Korean Intellectual Property Office
189 Cheongsa-ro, Seo-gu, Daejeon Metropolitan City, 302-701, Republic of Korea

Facsimile No. 82-42-472-7140

Authorized officer

AN, BYUNG IL

Telephone No. 82-42-481-8471

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2011/067963

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 7623134 B1	24.11.2009	None	
US 2005-0198444 A1	08.09.2005	US 7114040 B2	26.09.2006
US 2007-0150695 A1	28.06.2007	CN 100456266 C0	28.01.2009
		CN 1991796 A	04.07.2007
		DE 602006020015 D1	24.03.2011
		EP 1811384 A2	25.07.2007
		EP 1811384 A3	06.02.2008
		EP 1811384 B1	09.02.2011
		JP 2007-179537 A	12.07.2007
		KR 10-0755701 B1	05.09.2007
		KR20070068801A	02.07.2007
		US 7617381 B2	10.11.2009