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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
26 August 2004 (26.08.2004)

PCT

(10) International Publication Number
WO 2004/072852 A1

(51) International Patent Classification[7]: G06F 9/46

(21) International Application Number:
PCT/US2004/003701

(22) International Filing Date: 9 February 2004 (09.02.2004)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/446,524 11 February 2003 (11.02.2003) US
60/454,734 14 March 2003 (14.03.2003) US

(71) Applicant *(for all designated States except US)*: **THOMSON LICENSING S.A.** [FR/FR]; 46, Quai A. Le Gallo, F-92648 Boulogne (FR).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **LEWIS, Jody, Western** [US/US]; 262 W 1000 N, Farmington, UT 84025 (US). **RIGGS, Kerry, Lynn** [US/US]; 3467 Magic View Drive, Salt Lake, UT 84121 (US).

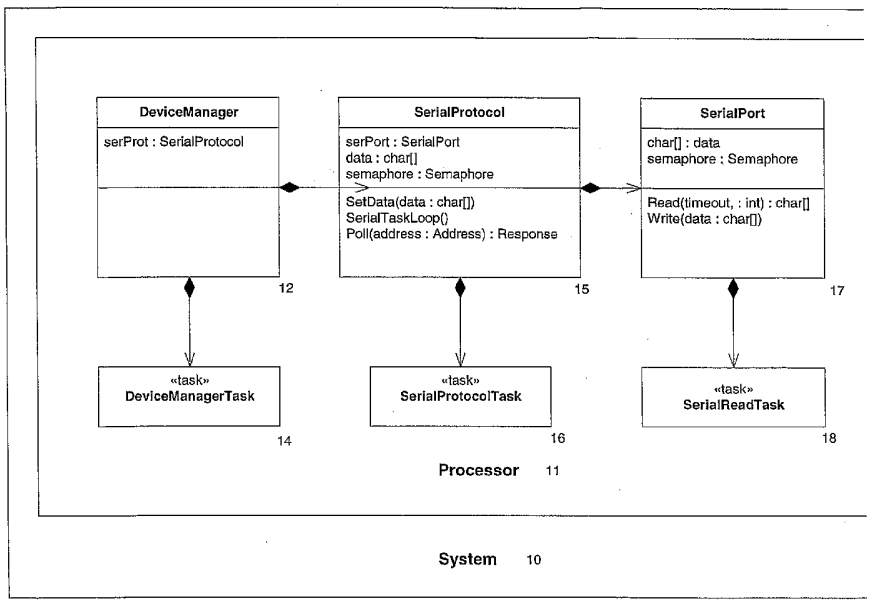
(74) Agents: **TRIPOLI, Joseph, S.** et al.; Thomson Licensing Inc., Two Independence Way, Suite #200, Princeton, NJ 08540 (US).

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

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

[Continued on next page]

(54) Title: ASYNCHRONOUS COMMUNICATIONS TECHNIQUE



(57) Abstract: A main processor (11) manages serial communication with one or more external devices by establishing the requisite tasks needed for serial communications. For example, these tasks can include (1) serial device handling, (2) protocol encapsulation, and (3) low-level communication with external devices. A priority is assigned to each of the tasks so that timing requirements are met, while maximizing processor efficiency of the main processor. Upon its completion, each lower priority task initiates execution of a next higher priority task to synchronize data processing with data communication.

WO 2004/072852 A1

Published:

— *with international search report*

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments*

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

ASYNCHRONOUS COMMUNICATIONS TECHNIQUE

This invention relates to a technique for achieving asynchronous serial communication between devices.

CROSS-REFERENCE TO RELATED APPLICATIONS

This application claims priority under 35 U.S.C. 119(e) to U.S. Provisional Patent Applications Serial No 60/446,524 filed February 11, 2003, and 60/454,734, filed March 14, 2003, the teachings of which are incorporated herein.

BACKGROUND ART

Embedded systems often employ some form of asynchronous serial communication with other devices. Usually a coprocessor will manage the intricate details of the serial protocol(s) used for communication. In the absence of a coprocessor, the task of servicing serial devices will fall on the main processor, thus consuming valuable resources and imposing burdensome priority and timing requirements. Most serial protocols typically have stringent timing requirements. Without a coprocessor, the main processor handling the serial communication task will need a detailed knowledge of the system hardware to exploit the processor's capabilities. Knowledge of the system hardware entails knowing about low-level device driver routines, which are often prone to errors. In that regard, the main processor must verify a serial device's compliance as well as ensuring its own compliance. Taking a generic approach with regard to processor priorities can waste processor resources. Setting task priorities too low can cause missed deadlines due to task pre-emption resulting from other tasks running at higher priorities Setting task priorities too high can block critical tasks and waste processor time.

Thus, there is need for a technique for achieving asynchronous serial communication in the absence of a co-processor, which overcomes the aforementioned disadvantages.

BRIEF SUMMARY OF THE INVENTION

Briefly, in accordance with a preferred embodiment of the present principles, there is provided a method for managing serial communication by a main processor. The method commences by establishing the requisite tasks needed for serial communications. For example, these tasks can include (1) serial device handling, (2) protocol encapsulation, and (3) low-level communication with external devices. A priority is assigned to each of the tasks so that timing requirements are met, while maximizing processor efficiency of the main processor. Upon its completion, each lower priority task initiates execution of a next higher priority task to synchronize data processing with data communication.

BRIEF SUMMARY OF THE DRAWINGS

FIGURE 1 depicts a schematic diagram depicting each of a set of logical blocks (objects) associated with different tasks related to serial communication; and

FIGURE 2 depicts a timing diagram that depicts the sequence of events performed by the tasks of FIG. 1.

DETAILED DESCRIPTION

FIGURE 1 shows a block diagram of a system 10 in accordance with the present principles for efficiently managing serial communications with one or more peripheral devices (not shown) without the need for a dedicated co-processor. The system 10 comprises a main processor 11 having a first block 12 that takes the form of a set of instructions and associated data files. The first block 12, herein after referred to as "DeviceManager" performs a first set of operations in connection with serial communications, including the polling of an address space that includes serial device(s) (not shown) with which the processor 10 seeks to communicate through a serial port 13. When present, each serial device connected to the main processor 11 through the serial port 13 will respond, thereby allowing the DeviceManager 12 to request data from such devices required for communications purposes. For the identified serial device(s), the DeviceManager block 12 creates a task, depicted in FIG. 1 as DeviceManagerTask 14 that initiates polling of the identified device(s) for data updating purposes. The DeviceManagerTask

14 runs at a lower priority, at least as compared to other system tasks. As discussed hereinafter, a task (e.g., DeviceManagerTask 14) is defined as the performance of an action, whereas, an object (e.g., DeviceManager 12) provides functions and/or data for use by a task.

Within the processor 11, a second block 15, hereinafter referred to as the SerialProtocol block, encapsulates the details of the serial protocol(s) employed to communicate with external device(s) through the serial port 13. An example of such a serial protocol is the esTributary protocol, although others exist and could easily be used. The SerialProtocol block 15 provides functions used to format outbound messages according to the protocol specification and ensure outbound messages comply with protocol's timing requirements. The SerialProtocol block 15 also contains functions to verify inbound messages' compliance with the protocol specifications and the protocol timing requirements. The SerialProtocol block 15 creates a task 16, hereinafter, referred to as the SerialProtocolTask, which controls the writing of data to, and the reading of data from the external device(s) through the serial port 13 according to this particular protocol's requirements. The SerialProtocolTask 15 runs at a sufficiently high priority to ensure that the task meets assigned timing deadlines.

The main processor 11 of FIG. 1 also includes a third block 17, referred to as a SerialPort block that encapsulates low-level communication with the serial port 13. The SerialPort block 17 serves to create an abstract (i.e., a model) of the serial port 13 making the architecture of this block more portable and reusable. The SerialPort block 17 provides functions to read data from and write data to the serial port 13 and also has responsibility to implement timeouts on read operations. The SerialPort block 17 creates a task 18, hereinafter referred to as SerialReadTask that reads all serial data sent to the processor 11 through the serial port 13. For that reason, the Serial ReadTask must run at a priority high enough to ensure that the task meets timing deadlines as required by any protocol using the serial port.

FIGURE 2 depicts a timing diagram that illustrates the sequence of events associated with initiating serial communication illustrating the advantage of the serial communication technique of the present principles. The process of serial communication commences when the DeviceManagerTask 14 of FIG. 1 initiates polling of an address corresponding to the serial port 13 of FIG. 1 by calling the Poll 90 function of DeviceManager 12. Poll 90 in turn calls the Poll 100 function of SerialProtocol 15. Poll 100 in turn calls the SetData 110 function which copies the destination address to SerialProtocol block 15 and then calls the semGive 120 function of SerialPort 17.

At initialization or subsequent to some previous poll event SerialProtocolTask 16 blocked (stopped running) in semTake 130 waiting for its semaphore. The semGive 120 functions triggers SerialProtocolTask 16 that it has valid data and may now run. This results in the Write 140 function being called which writes the polling data to serial port 13 of FIG. 1. Subsequently the Read 150 function is called. Read 150 in turn calls the semTake 160 function which causes SerialProtocolTask 16 to block (stop running) until it is triggered in 170.

SerialReadTask 18 continually looks for incoming data from serial port 13 of FIG. 1 in its read 180 function. When any data are available they are copied into SerialPort block 17. The semTake 160 function returns in 170 when the data requested in Read 150 is available or the specified time has elapsed. This triggers SerialProtocolTask 16 to run which returns the poll data (if the read was successful) or an error indication (if the time limit was exceeded) to DeviceManagerTask 14.

Since SerialReadTask 18 runs at a high system priority it can be guaranteed to meet its timing deadlines, but since it only runs when data are available from serial port 13 of FIG. 1 it will never consume system resources unnecessarily.

Similarly, since SerialProtocolTask 16 runs at a high system priority it too can be guaranteed to meet its timing deadlines. It is only triggered to run when a poll function is required so it will not consume system resources unnecessarily when it is not needed. When it is active and waiting for a response from a serial device it also blocks (stops and waits) on the SerialPort block 17 to trigger it to run again, thus not consuming unnecessary system resources during this phase of the communication cycle. Polling is one method for communicating with a group of serial devices. Another option would be to have the serial port interrupt the processor when data is available. If there were only one serial device attached the processor could communicate to it without any polling or addressing. The present technique is applicable to all these mechanisms. The entire serial communication cycle is gated by the low priority DeviceManagerTask 14. This allows the system to meet timing requirements of the serial protocol during a serial communication cycle. But these serial communication cycles are only allowed to run when other, higher-priority tasks in the system allow the DeviceManagerTask 14 to run.

The foregoing describes a technique for achieving serial communication without the need for dedicated co-processor for managing communications tasks.

CLAIMS

1. A method for facilitating asynchronous serial communication by a main processor with at least one external communications device, comprising the steps of:

establishing at the main processor a set of blocks that each including at least one instruction for execution by the main processor for accomplishing serial communication, each block having at least one task that runs at a prescribed priority, such that the tasks of different blocks have a hierarchical priority;

executing the blocks so that at least one task within each block runs at its respective priority level; and

triggering each higher priority task upon completion of a corresponding lower priority task.

2. The method according to claim 1 where the set of blocks includes a Device Manager block that establishes communication with an external device.

3. The method according to claim 2 wherein the Device Manager Block initiates a Device Manager Task that initiates address polling for updating.

4. The method according to claim 1 wherein a set of blocks includes a Serial Protocol Block that incorporates details of each serial protocol employed to communicate with an external device.

5. The method according to claim 4 wherein the Serial Protocol block initiates a Serial Protocol task that controls the writing of data to, and the reading of data from an external device.

6. The method according to claim 1 wherein the set of blocks includes a Serial Port block that encapsulates low-level communication with a serial port.

7. The method according to claim 6 wherein the Serial Port block initiates a Serial Read Task that reads data from an external device.

8. A method for facilitating asynchronous serial communication by a main processor with at least one external communications device, comprising the steps of:

establishing at the main processor a first block for serial device handling, a second block for protocol encapsulation, and a third block for low-level communication with external devices, each block including at least one instruction for execution by the main processor and each block having at least one task that runs at a prescribed priority, such that the tasks of different blocks have a hierarchical priority;

executing the first, second and third blocks so that at least one task within each block runs at its respective priority level; and

triggering each higher priority task upon completion of a corresponding lower priority task.

9. The method according to claim 8 wherein the first block includes a first task that initiates address polling for updating.

10. The method according to claim 8 wherein the second block initiates a second task that controls the writing of data to, and the reading of data from an external device.

11. The method according to claim 8 wherein the third block initiates a third task that reads data from an external device.

12. A system which includes a main processor for facilitating asynchronous serial communication with at least one external communications device, comprising:

a first block of data and instructions executed by the processor for serial device handling,

a second block of data and instructions executed by the processor for protocol encapsulation, and

a third block of data and instructions executed by the processor for low-level communication with at least one external device,

wherein each block has at least one task that runs at a prescribed priority, such that the tasks of different blocks have a hierarchical priority; and wherein the processor executes the first, second and third blocks so that at least one task within each block runs at its respective priority level; and wherein each higher priority task is triggered upon completion of a corresponding lower priority task.

13. The system according to claim 12 wherein the first block includes a first task that initiates address polling for updating.

14. The system according to claim 12 wherein the second block initiates a second task that controls the writing of data to, and the reading of data from an external device.

15. The system according to claim 12 wherein the third block initiates a third task that reads data from an external device.

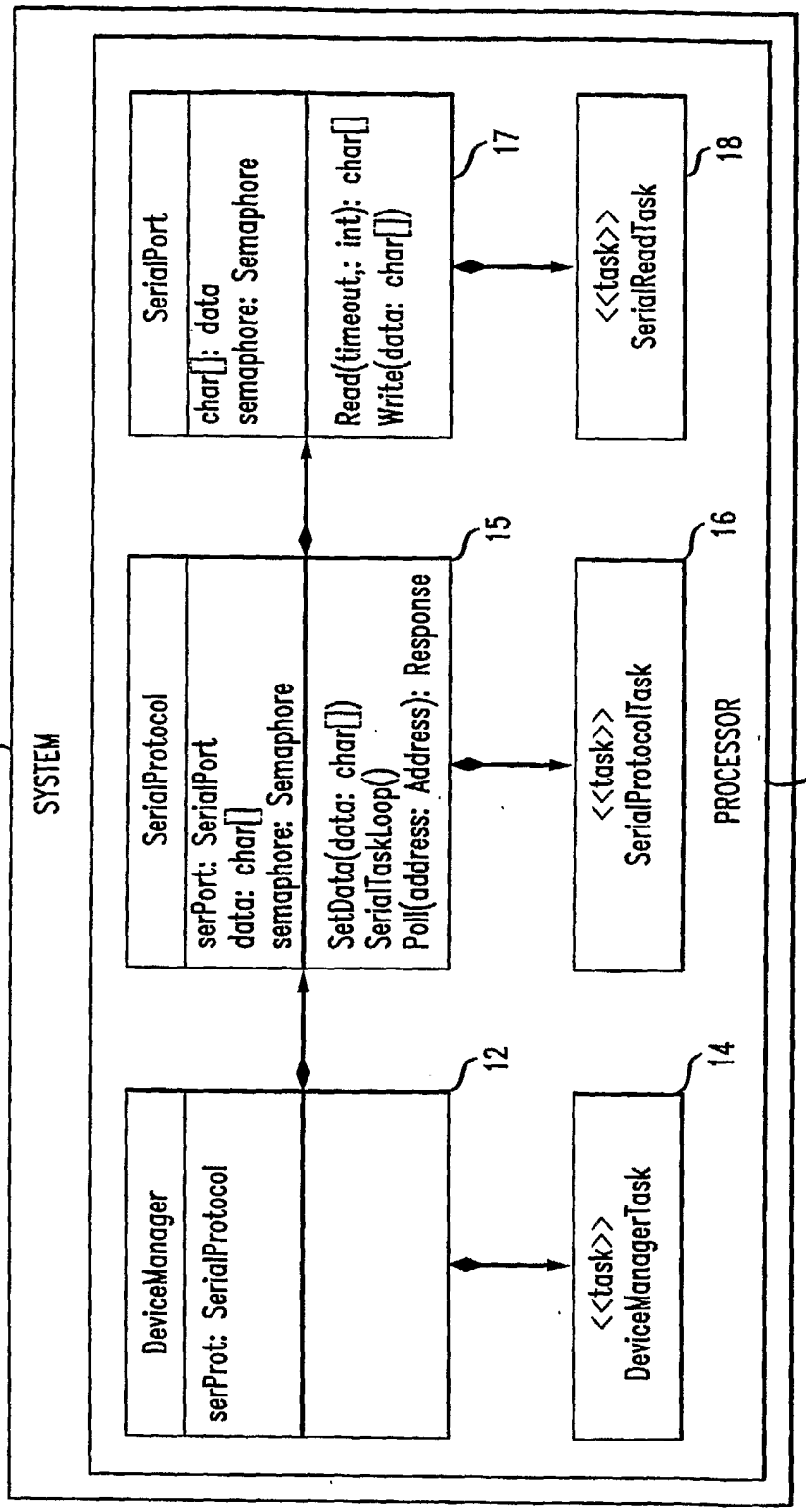

10
SYSTEM
DeviceManager
serProt: SerialProtocol
12
<<task>>
DeviceManagerTask
14
SerialProtocol
serPort: SerialPort
data: char[]
semaphore: Semaphore
SetData(data: char[])
SerialTaskLoop()
Poll(address: Address): Response
15
<<task>>
SerialProtocolTask
16
SerialPort
char[]: data
semaphore: Semaphore
Read(timeout,: int): char[]
Write(data: char[])
17
<<task>>
SerialReadTask
18
PROCESSOR
11

FIG. 1

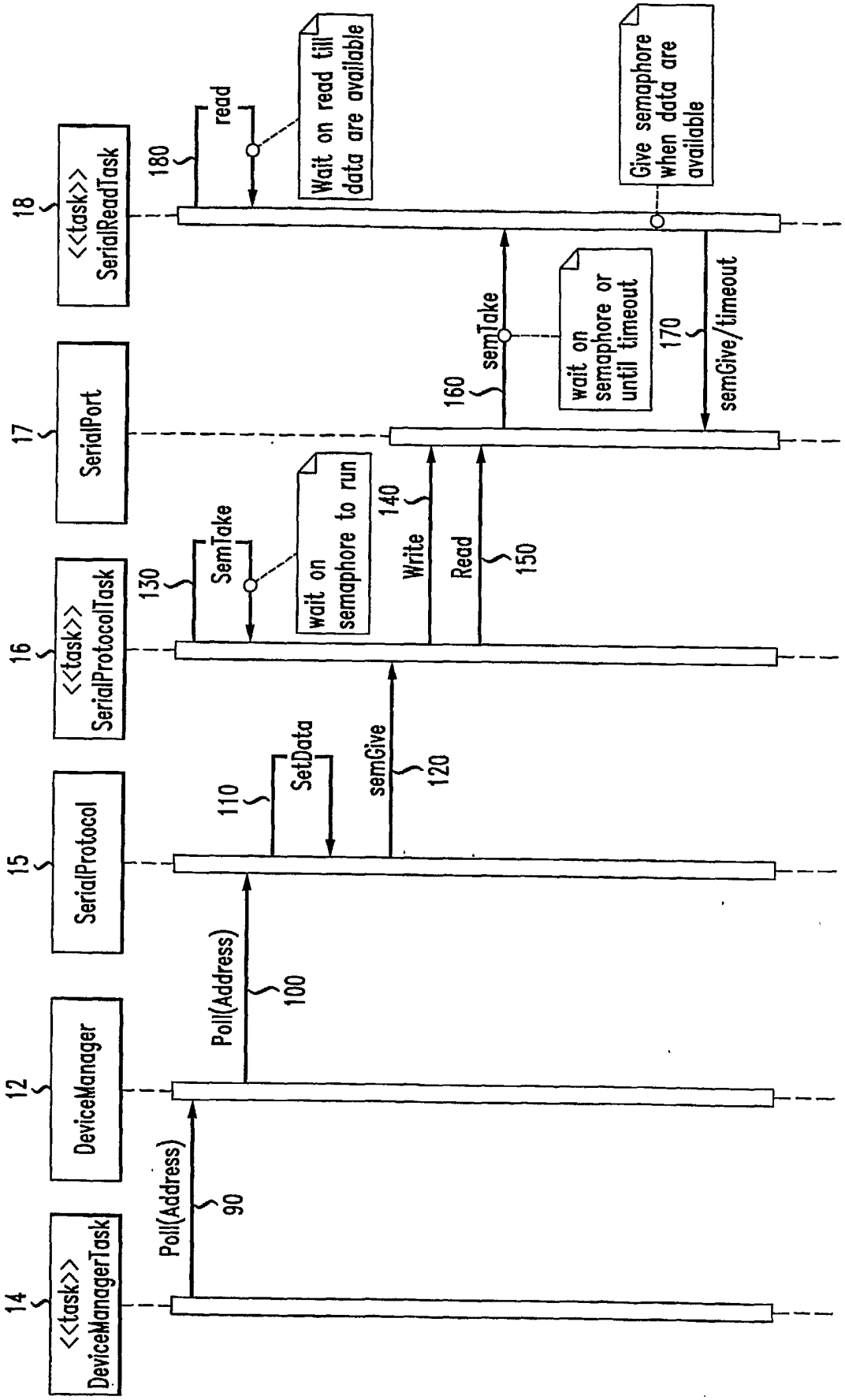

14
<<task>>
DeviceManagerTask
12
DeviceManager
15
SerialProtocol
16
<<task>>
SerialProtocolTask
17
SerialPort
18
<<task>>
SerialReadTask
Poll(Address)
90
Poll(Address)
100
SetData
110
semGive
120
SemTake
130
wait on
semaphore to run
140
Write
Read
150
semTake
160
wait on
semaphore or
until timeout
170
semGive/timeout
180
read
Wait on read till
data are available
Give semaphore
when data are
available

FIG. 2

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US04/03701

A. CLASSIFICATION OF SUBJECT MATTER

IPC(7) : G06F 9/46
US CL : 718/100,102,103

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)
U.S. : 718/100,102,103

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

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

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 6,275,864 B1 (MANCUSI et al.) 14 August 2001 (14.08.2001), column 1, lines 63-67 and column 2, lines 1-43.	1-15
A, E	US 2004/0054998 A1 (HAYASHI) 18 MARCH 2004 (18.03.2004), page 3, paragraphs 0058-0061.	1-15
A,P	US 6,577,635 B2 (NARAYANA et al.) 10 June 2003 (10.06.2003), column 1, lines 66-67 and column 2, lines 1-16.	1-15

☐ 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 published on or after the international filing date

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

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

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

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

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

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

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
09 July 2004 (09.07.2004)	03 AUG 2004
Name and mailing address of the ISA/US Mail Stop PCT, Attn: ISA/US Commissioner for Patents P.O. Box 1450 Alexandria, Virginia 22313-1450 Facsimile No. (703)305-3230	Authorized officer Jeffrey A. Gaffin for Michelle R. Eason Telephone No. (703) 305 3900

Form PCT/ISA/210 (second sheet) (July 1998)