(12) UK Patent Application (19) GB (11) 2 328 299 (13) A

(43) Date of A Publication 17.02.1999

(21) Application No 9814645.9

(22) Date of Filing 06.07.1998

(30) Priority Data
(31) 19728971 (32) 07.07.1997 (33) DE

(51) INT CL^6
G06F 9/46 11/32

(52) UK CL (Edition Q)
G4A AFI APX

(56) Documents Cited
US 5513349 A US 4875159 A

(58) Field of Search
UK CL (Edition P) G4A AFI APX
INT CL^6 G06F 9/46 11/32

(71) Applicant(s)
Robert Bosch GmbH
(Incorporated in the Federal Republic of Germany)
Wernerstrasse 1, Stuttgart-Feuerbach,
D-70442 Stuttgart 30, Federal Republic of Germany

(72) Inventor(s)
Ole See

(74) Agent and/or Address for Service
A A Thornton & Co
Northumberland House, 303-306 High Holborn,
LONDON, WC1V 7LE, United Kingdom

(54) Abstract Title
Data processing system for ensuring data consistency

(57) The system provides a data-processing device having a first facility for executing a first data processing using global variables V_1, V_2, by means of a processor facility, a second facility for executing a second data processing by means of the processor facility which is configured in such a way that, at the start, it creates local copies GLK1, GLK2 of the global variables used by the first facility, and can be interrupted during the copying process by the first facility, a status indication flag F set by the second facility during the copying process and readable by the first facility for indicating when the copying process is active, the first facility being configured in such a way that, before the end of a respective interruption of the second facility, it checks whether the status indication flag is set and, if so, overwrites the local copy in the form of the global variable with its current value.

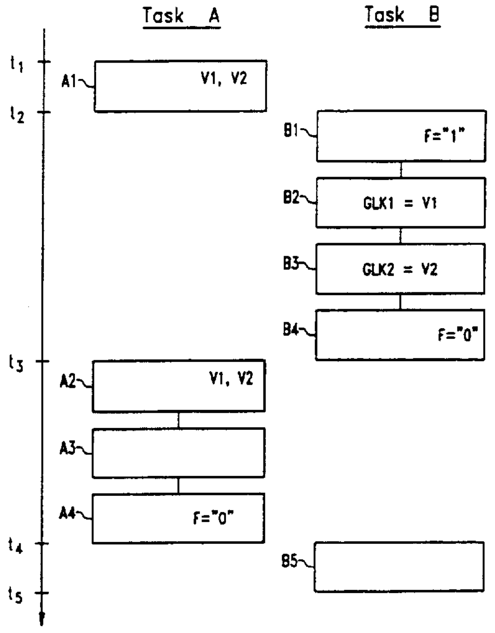

Fig.1

GB 2 328 299 A

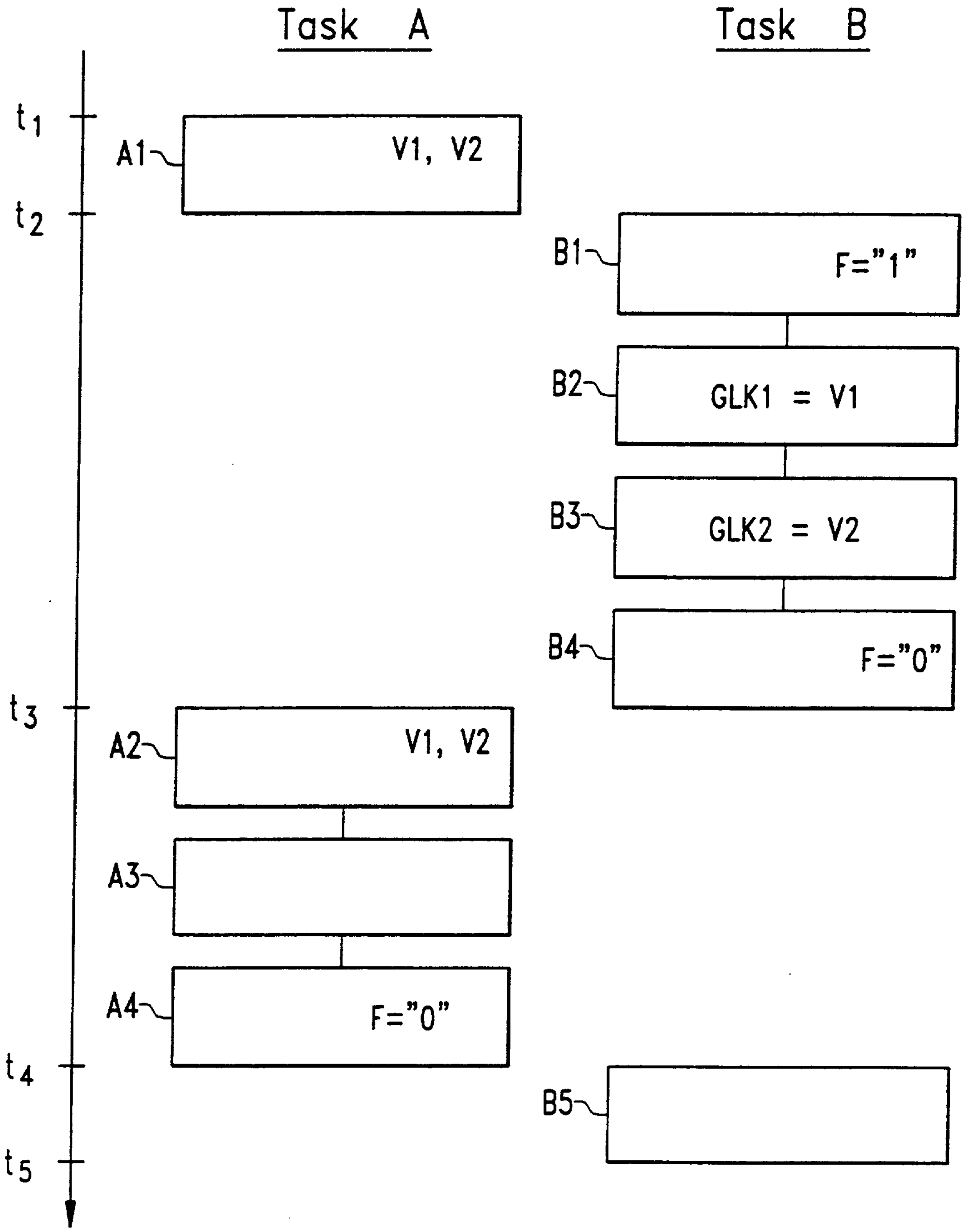
Task A
Task B
t1
t2
t3
t4
t5
A1
V1, V2
B1
F="1"
B2
GLK1 = V1
B3
GLK2 = V2
B4
F="0"
A2
V1, V2
A3
A4
F="0"
B5

Fig.1

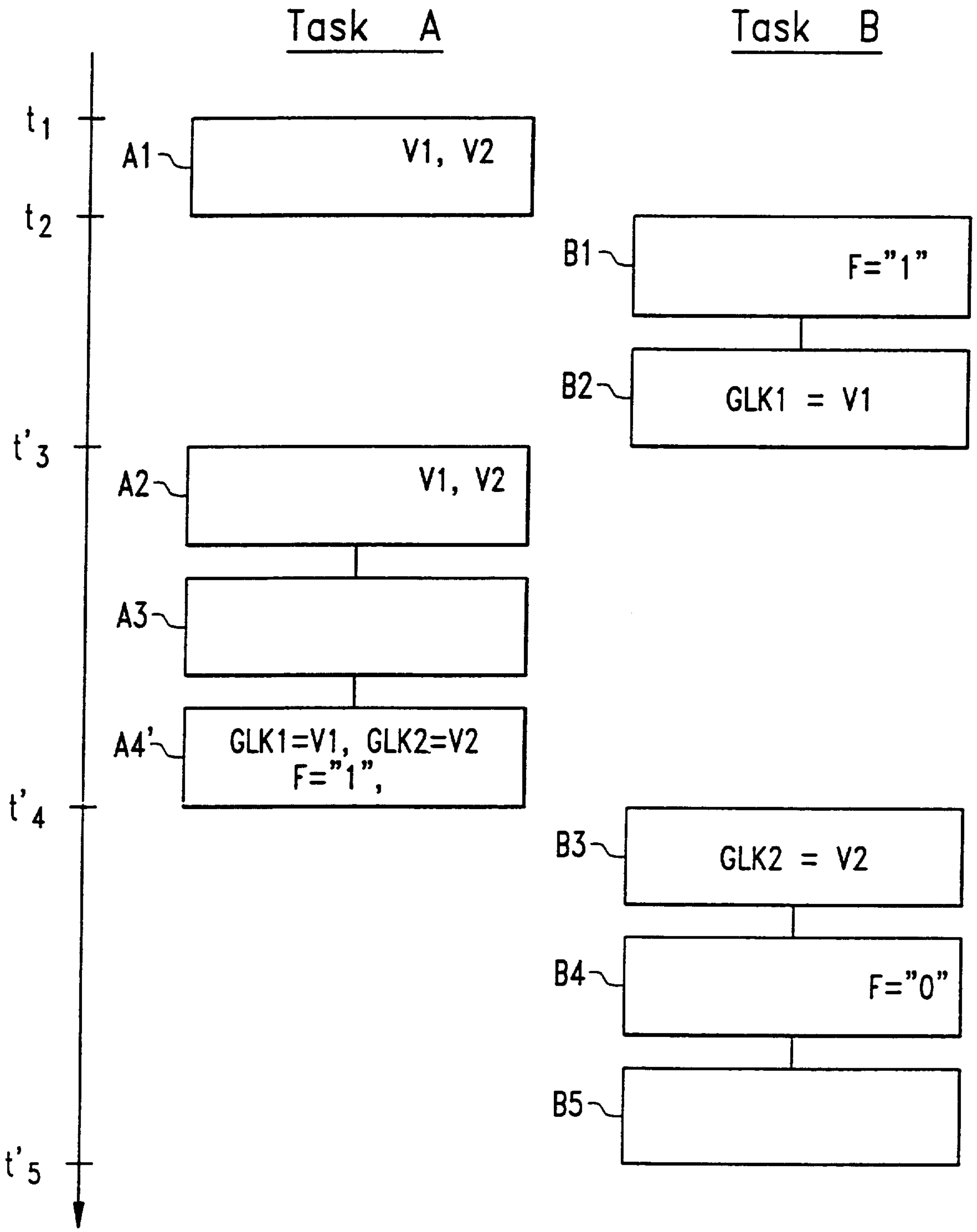
Task A
Task B
t_1
t_2
t'_3
t'_4
t'_5
A1
V1, V2
B1
F="1"
B2
GLK1 = V1
A2
V1, V2
A3
A4'
GLK1=V1, GLK2=V2
F="1",
B3
GLK2 = V2
B4
F="0"
B5

Fig.2

```
     Task   A     (10ms             )
     :
S1:  V1  =  _   _   _
S2:  V2  =  _   _   _
     :
S4:  IF   (FLAG  =  1)   THEN
S5:  GLK1  =  V1
S6:  GLK2  =  V2
S7:  END    IF
     END
```

Fig.3

```
      Task   B     (50ms             )
      :
S1':  FLAG  =  1
S2':  GLK1  =  V1
S3':  GLK2  =  V2
S4':  FLAG  =  0
      :
S5':  _  _  _  =  FUNKTION  (GLK1,  GLK2)
      :
      END
```

Fig.4

Data-processing device and method

PRIOR ART

The present invention relates to a data-processing device and a corresponding data-processing method.

Although it is applicable to any data-processing devices, the present invention and also the problems underlying it are explained in relation to a data-processing device which is situated on board a motor vehicle and which is used to network various control appliances comprising sensor devices and actuator devices, for example the real-time-capable serial bus system referred to as "controller area network" (CAN).

A control appliance may be considered for setting an engine parameter, such as, for example, the fuel injection quantity. In said control appliance, various tasks or application programs which use the time-slicing method of a processor run in the operating system. In this connection, the computing time of the processor is divided up among all the tasks. Each task is allotted a share of the computer output cyclically or according to certain priorities. The switching back and forth between the tasks takes place, as a rule, at such high speed that the interruptions and restarts are not perceptible externally. All the data which are utilized by a plurality of tasks are defined as global variables.

If data are calculated in a task A which are further processed in another task B, data consistency must be ensured even if the task B can be interrupted at any time by the task A. In other words, before the termination of task B, its current data set must not change although the task A already provides fresh data after the interruption.

For the purpose of more-detailed illustration of the problems, the following case of rotational speed control in an engine control appliance may be considered.

The engine control appliance calculates the manipulated variable of the rotational speed controller. The engine control appliance receives the rotational speed set-point value to be controlled and also the operating mode, i.e. the controller response (e.g. proportional component, differential component, integral component etc.) via the CAN bus of the vehicle control appliance every 10 ms. The control parameters (control parameter set), i.e. the parameter set for the control module, are, however, calculated every 50 ms in the engine control appliance. Consequently it is clear that the engine control appliance must not accept any fresh data from the vehicle control appliance before the end of its respective processing time of 50 ms although the vehicle control appliance provides in principle such fresh data every 10 ms.

At present there are three approaches to ensuring data consistency in the prior art.

The first approach envisages the creation of local copies of the data calculated in the task having the higher priority at the start of the task having the lower priority.

A disadvantage of this procedure is that the data consistency is no longer ensured if the task of lower priority is interrupted during the actual copying process of the task of higher priority. After all, in this case, when the task of lower priority is continued after terminating the interruption, the copies already created are no longer refreshed, but the copying process is continued at the point of the interruption. After termination of the copying process, there is therefore inconsistency between the data copied first and last.

The second approach envisages the creation of local copies of the data as in the case of the first approach. During the copying process, however, any interruption or interrupt is blocked so that the task having higher priority cannot "disturb" the task having lower priority.

However, this approach has the disadvantage that the interrupt blocking time can be exceeded during copying relatively large data sets, for example structures, which may result in a restart on the part of the operating system. This response is undesirable since it may paralyse the entire system, for example, in the case of engine control appliances, it may result in failures in the fuel injection or even in engine stoppage.

The third approach provides for the application of a "shared memory" or jointly used memory. This is a region in the working memory which is managed by the operating system and into which all the tasks can in principle write data or out of which all the tasks can in principle read data if the operating system allows it. To protect the data consistency in the case mentioned, the task of higher priority can write into the shared memory again only if the task of lower priority has produced its local copies. That is to say, during the production of the local copies by the task having lower priority, access of the task having higher priority to the "shared memory" is forbidden.

This approach has the disadvantage that the shared memory has to be managed by the operating system, which results in an increased complexity, that is to say a greater intervention into the operating system.

In the case of the above, known approaches, the fact that they either have no adequate data protection and/or require a high complexity has been found to be disadvantageous.

ADVANTAGES OF THE INVENTION

The data processing device according to the invention having the features of Claim 1 and the corresponding data processing method in accordance with Claim 6 have the advantage over the known approaches to a solution that they ensure a good data consistency using simple means, i.e. without high equipment complexity or programming complexity.

The data consistency is ensured even in the case of data which are closely related to one another and have therefore to be further processed under identical conditions, i.e. must not originate from two different cycles of the first data processing.

The problems associated with the interrupt blocking are eliminated and the intervention into the system is also low and takes place locally and not at operating system level.

The idea underlying the present invention is that a first task or application program having higher priority sets certain global variables which are used by a second task having lower priority. In the second task, a global flag, or a status indicator, which can be read by the first task is set during a data copying process. In the event of interruption of this data copying process by the first task, the state of the flag in the first task is checked before the end of the interruption and if it is set, this is viewed as an indication of a possible alteration of data already read. Accordingly, the first task either overwrites with their current values only the altered variables or all the variables which are used by the second task locally and which are likewise defined globally for this purpose. The local copies are created at the beginning of the second task having lower priority, that is to say before any further processing of the data to be copied. The local copies are overwritten, if necessary, at the end of the first task having higher priority, that is to say, at the earliest after all the modifications of the global variables used by the second task have been concluded.

The subclaims contain advantageous developments and improvements of the data-processing device specified in Claim 1 or of the data-processing method specified in Claim 6.

In accordance with a preferred development, the first facility is assigned a first, higher priority and the second facility is assigned a second, lower priority in regard to the use of the processor facility.

In accordance with a further preferred development, the first facility is assigned a first, shorter time period and the second facility is assigned a second, longer time period in regard to the use of the processor facility.

In accordance with a further preferred development, the second facility is configured so that, at the end of a respective interruption, it continues at the point of the interruption. This is advantageous if as little time as possible is to be lost in performing the second data processing.

In accordance with a further preferred development, the first facility comprises a determining facility for determining which local copy or local copies the second facility uses. This has the advantage that only certain altered variables which are determined by the first data processing and are used in the second data processing have to be refreshed.

DRAWINGS

Exemplary embodiments of the invention are shown in the drawings and described in greater detail in the description below.

In the drawings:

Figure 1 shows a time flowchart for a first task A and a second task B when the first task A interrupts the second task B after the copying process, in accordance with a first exemplary embodiment of the invention;

Figure 2 shows a time flowchart for a first task A and a second task B when the first task A interrupts the second task B during the copying process, in accordance with the first exemplary embodiment of the invention;

Figure 3 shows a diagrammatic representation of data-processing steps in a first task A in accordance with an exemplary embodiment of the invention; and

Figure 4 shows a diagrammatic representation of data-processing steps in a second task B in accordance with the second exemplary embodiment of the invention.

DESCRIPTION OF THE EXEMPLARY EMBODIMENTS

Figure 1 shows a time flowchart for a first task A and a second task B when the first task A interrupts the second task B after the copying process, in accordance with a first exemplary embodiment of the invention.

In Figure 1, t_1, t_2, t_3, t_4 and t_5 denote a respective consecutive first to fifth time instant on the vertical time axis. A1, A2, A3 and A4 denote respective program blocks of a task or user program A having a first higher priority. B1, B2, B3, B4 and B5 denote respective program blocks of a task or user program B having a second lower priority. In this connection, program blocks denote a part of the entire program.

Tasks A and B are both processed according to the time-slicing method making allowance for their priority on the same processor facility. In particular, the task A calculates variables V1 and V2 which the task B needs for its part. In this connection, it should be noted that, for reasons of data consistency during a cycle, the task B is allowed to use only a related data pair for V1 and V2.

The processing starts at time instant t_1 with the programming block A1 of the task A which, inter alia, comprises the calculation of the data of the global variables V1 and V2. In this connection, global variable means that at least the task B also has an access to said variables.

At the time instant t_2, the time slice of the task A has expired and the task B starts the program block B1 in which a flag F is set to logic "1", said flag F embodying the status indication facility according to the invention. The flag F thus indicates the start and the progress of the copying process.

In task B, local copies of the variables V1 and V2, namely GLK1 as a copy of V1 in B2 and GLK2 as a copy of V2 in B3, are then produced in the program blocks B2 and B3.

It should be expressly pointed out that the variables GLK1, GLK2 and F are likewise global variables, that is to say at least the task A also has access to them.

In the program block B4, the flag F is set to logic "0" in order to indicate the end of the copying operation.

At the time instant t_3, the time slice of the task B has expired and the task A is restarted with the program block A2 which, inter alia, comprises the recalculation of the data of the global variables V1 and V2.

In the subsequent program block A3, which comes in sequence after all the recalculations of the variables V1, V2 needed by the task B, the task A checks whether the flag F is set or not. Since, in the present case, the task B has been interrupted after the copying process, the flag F is at logic "0".

Since the task A has consequently been informed that the task B possesses the required related copies GLK1 and GLK2 and these were not altered by task A, no further steps are necessary and the task A terminates with the program block A4 at the time instant t_4.

The task B then continues with the program block B5, which terminates the task B or runs until the next interruption by the task A. However, neither the first interruption described nor any further interruption endangers the data consistency in task B.

Figure 2 shows a time flowchart for a first task A and a second task B when the first task A interrupts the second task B during the copying process, in accordance with the first exemplary embodiment of the invention.

In Figure 2, t_1, t_2,. t_3', t_4' and t_5 denote a respective consecutive first to fifth time instant on the vertical time axis. A1, A2, A3 and A4' denote respective program blocks of a task or user program A, and B1, B2, B3, B4 and B5 denote respective program blocks of a task or user program B. In this connection, the unprimed reference symbols correspond to the components explained in Figure 1.

As at time instant t_1 above, the processing starts at the programming block A1 of the task A which, inter alia, comprises the calculation of the data of the global variables V1 and V2.

At the time instant t_2, the time slice of the task A has expired and the task B starts the program block B1, in which the flag F is set to logic "1" and thus indicates the start and the progress of the copying operation.

In the task B, a local copy GLK1 of the variable V1 is produced in the program block B2. At the time instant t_3', the time slice of the task B is interrupted by the task A and the task A continues with the program block A2 which comprises, inter alia, the recalculation of the data of the global variables V1 and V2.

In the subsequent program block A3, which comes in sequence after all the recalculations of the variables V1, V2 needed by the task B, the task A checks whether the flag F is set or not. Since, in the present case, the task B has been interrupted during the copying process, the flag F is at logic "1".

The task A is consequently informed that the task B does not possess the required related copies GLK1 and GLK2 and further steps are necessary to ensure the data consistency.

For this purpose, the task A executes the program block A4', in which the copying process is executed for the task B, namely a copy of V1 is created in GLK1 and a copy of V2 is created in GLK2.

In this connection, it should be generally remarked that the task A has in principle to update only those copies whose corresponding variable it has altered. In the case of fairly large data sets this can be utilized for time saving purposes.

At the time instant t_4', the time slice of the task A has expired and the latter terminates.

The task B is then continued at the point at which it was previously interrupted and produces a local copy GLK2 of the variable V2 in the program block B3. In the program block B4, the flag F is then set to logic "0" in order to indicate the end of the copying process.

The task B is then continued with the program block B5 which concludes the task B or runs until the next interruption by the task A. However, neither the first interruption described nor any further interruption endangers the data consistency in the task B. In particular, any further interruption of the task B by the task A results in the situation described in connection with Figure 1.

Figure 3 shows a diagrammatic representation of data-processing steps in a first task A in accordance with the second exemplary embodiment of the invention.

The task A occupies a fixed time slice of 10 ms. In the steps S1 and S2, the data are calculated for the global variables V1 and V2, which are also used by the task B. In the broken-line region, any other data-processing steps take place.

At the end of the task A, the copying process of V1 into GLK1 and of V2 into GLK2 for the task B is repeated in the steps S4 to S7, if necessary, i.e. if the flag F is logic "1".

Figure 4 shows a diagrammatic representation of data-processing steps in a second task B in accordance with the second exemplary embodiment of the invention.

The task B occupies a fixed time slice of 50 ms. In the steps S1' to S4', global variables V1 and V2 which are used by the task B are copied into the global variables GLK1 and GLK2, respectively. It is only during this copying process that the flag F is logic "1", in order to indicate, in the event of an interruption by the task A, the need for measures to avoid data inconsistency. In the broken-line area, any other data-processing steps take place.

At the end of the task B in step S5', the local copies GLK1 and/or GLK2 are further processed, the present invention ensuring that, in a cycle of the task B, calculation is always carried out with one and the same consistent variable pair.

Although the present invention was described by reference to a preferred exemplary embodiment, it is not restricted thereto, but can be modified in a wide variety of ways.

In particular, the two facilities for executing the data processing and the status indication facility can be implemented not only in software, but, of course, also in hardware.

Furthermore, in the above exemplary embodiment, two variables V1 and V2 were used jointly by the task A and task B. The invention is, however, applicable to any desired number of jointly used variables. In this connection, the term variable is generally to be understood in the sense of variable quantity.

In addition, the present invention is applicable not only to the coordination of two tasks with one flag, but to a plurality of tasks which in each case use a flag pairwise.

The assignment of various time periods to different tasks is not obligatory. A 50 ms task is activated every 50 ms and a 10 ms task every 10 ms. That means only that a 50 ms task is allowed to last a maximum of 50 ms and a 10 ms task a maximum of 10 ms so that the respective task itself does not overreach itself and consequently trigger a restart of the operating system. This does not mean, however, that the time duration has to be 50 ms long or 10 ms long, respectively.

Data-processing device and method

LIST OF REFERENCE SYMBOLS:

t_1, t_2, t_3, t_4, t_5 t_3', t_4'	time instants
A1 - A4, A4' B1 - B5	program blocks
V1, V2	global variables
GLK1, GLK2	global variables for local copies
F	flag
S1 - S7	program steps of task A
S1' - S5'	program steps of task B

CLAIMS

1. Data-processing device comprising:
a first facility for executing a first data processing using one or more global variables by means of a processor facility;

a second facility for executing a second data processing by means of the processor facility which is configured in such a way that it creates at the start at least one local copy in the form of a global variable of at least one of the global variables used by the first facility and can be interrupted during the copying operation by the first facility;

a status indication facility which is set by the second facility during the copying process and can be read by the first facility for indicating the active copying process;

wherein

the first facility is configured in such a way that, before the end of a respective interruption of the second facility, it checks whether the status indication facility is set and, if so, overwrites the at least one local copy in the form of the global variable with its current value.

2. Data-processing device according to Claim 1, characterized in that the first facility is assigned a first, higher priority and the second facility is assigned a second, lower priority in regard to the use of the processor facility.

3. Data-processing device according to Claim 1 or 2, characterized in that the first facility is assigned a first, shorter time period and the second facility is assigned a second, longer time period in regard to the use of the processor facility.

4. Data-processing device according to one of the preceding claims, characterized in that the second facility is configured so that, at the end of a respective interruption, it continues at the point of the interruption.

5. Data-processing device according to one of the preceding claims, characterized in that the first facility comprises a determining facility for determining which local copy or local copies the second facility uses.

6. Data-processing method comprising the following steps:

execution of a first data processing using one or more global variables by means of a processor facility;

execution of a second data processing by means of the processor facility which creates, at the start, at least one local copy in the form of a global variable of at least one of the global variables used by the first facility and which can be interrupted during the copying process by the first data processing;

indication of the active copying process of the second data processing in a manner readable by the first data processing; and

checking in the first data processing whether the indication is set before the end of a respective interruption of the second data processing and, if so, overwriting of the at least one local copy in the form of the global variable with its current value.

7. Data-processing method according to Claim 6, characterized by the following steps:

assignment of a first, higher priority to the first facility in regard to the use of the processor facility; and

assignment of a second, lower priority of the second facility in regard to the use of the processor facility.

8. Data-processing method according to Claim 6 or 7, characterized by the following steps:

assignment of a first, shorter time period to the first facility in regard to the use of the processor facility; and

assignment of a second, longer time period to the second facility in regard to the use of the processor facility.

9. Data-processing method according to Claim 6, 7 or 8, characterized by the following step:

continuation with the second data processing at the end of a respective interruption at the point of the interruption.

10. Data-processing method according to one of Claims 6 to 9, characterized by the following step:

determination in the first data processing of which local copy or local copies the second facility uses.

11. Any of the data processing devices substantially as hereinbefore described with reference to the accompanying drawings.

12. Any of the data processing methods substantially as hereinbefore described with reference to the accompanying drawings.

The Patent Office

17

Application No: GB 9814645.9 **Examiner:** Leslie Middleton
Claims searched: 1,6 **Date of search:** 8 December 1998

Patents Act 1977
Search Report under Section 17

Databases searched:

UK Patent Office collections, including GB, EP, WO & US patent specifications, in:

UK Cl (Ed.P): G4A (AFI, APX)

Int Cl (Ed.6): G06F 11/32

Other: Online: Computer Database, Inspec, WPI / DIALOG

Documents considered to be relevant:

Category	Identity of document and relevant passage	Relevant to claims
A	US 5513349 A (IBM) See cols. 1-3, especially	
A	US 4875159 A (Amdahl Corpn.) See cols. 1-3, especially	

X	Document indicating lack of novelty or inventive step	A	Document indicating technological background and/or state of the art.
Y	Document indicating lack of inventive step if combined with one or more other documents of same category.	P	Document published on or after the declared priority date but before the filing date of this invention.
		E	Patent document published on or after, but with priority date earlier than, the filing date of this application.
&	Member of the same patent family		