

(19) World Intellectual Property Organization
International Bureau



(43) International Publication Date
18 December 2003 (18.12.2003)

PCT

(10) International Publication Number
WO 03/104968 A2

(51) International Patent Classification⁷: **G06F 5/06**

(21) International Application Number: PCT/IB03/02282

(22) International Filing Date: 21 May 2003 (21.05.2003)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
02077223.2 7 June 2002 (07.06.2002) EP

(71) Applicant (for all designated States except US): **KONINKLIJKE PHILIPS ELECTRONICS N.V.** [NL/NL];
Groenewoudseweg 1, NL-5621 BA Eindhoven (NL).

(72) Inventors; and

(75) Inventors/Applicants (for US only): **HOOGERBRUGGE, Jan** [NL/NL]; c/o Prof. Holstlaan 6, NL-5656 AA Eindhoven (NL). **STRAVERS, Paul** [NL/NL]; c/o Prof. Holstlaan 6, NL-5656 AA Eindhoven (NL).

(74) Agent: **DE JONG, Durk, J.**; Internationaal Octrooibureau B.V., Prof. Holstlaan 6, NL-5656 AA Eindhoven (NL).

(81) Designated States (national): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, 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, NI, NO, NZ, OM, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

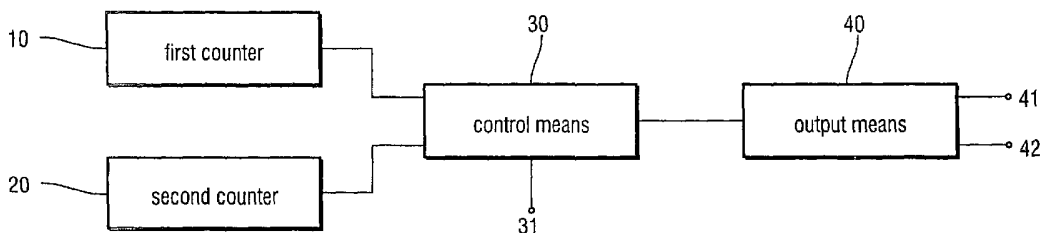
(84) Designated States (regional): ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IT, LU, MC, NL, PT, RO, SE, SI, SK, TR), OAPI patent (BF, BJ, CF,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

Published:

— without international search report and to be republished upon receipt of that report

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

(54) Title: SPACECAKE COPROCESSOR COMMUNICATION



(57) Abstract: The invention is based on the idea to maintain two counters for an input or output port of a FIFO. A device for writing data elements from a coprocessor into a FIFO memory is provided. Said device is embedded in a multiprocessing environment comprising at least one coprocessor, a FIFO memory and a controller. Said device comprises a first counter for counting the available room in said FIFO memory, and a second counter for counting the number of data elements written into said FIFO memory. Said device further comprises a control means for checking said first counter for available room in said FIFO memory, and for checking said second counter whether a predetermined number N of data elements have been written into said FIFO memory. Said control means decrements the count of said first counter and increments the count of said second counter, after a data element has been written into said FIFO memory. Said device finally comprises an output means for outputting data elements to said FIFO memory. Said control means issues a first message when the count of said second counter has reached said predetermined number N and issues a first call for available room in said FIFO memory to said controller. Said output means forwards said first message and/or said first call to said controller.



WO 03/104968 A2

Spacecake coprocessor communication

The invention relates to a device and a method for writing data elements from a coprocessor into a FIFO, a device and method for reading data elements from a FIFO memory into a coprocessor as well as a multiprocessing computer system.

5 New applications and services like High Definition TV (HDTV), Broadcast Satellite Services, Video-conferencing, Interactive Storage Media etc. require a huge amount of data processing in the video domain as well as in the audio domain. Thus, the data rates involved in those applications make computation very time consuming.

When coping with the computing requirements in signal processing, designers usually follow two architectural approaches, namely the dedicated and the programmable
10 approach. The dedicated architecture aims to fully exploit the computational features of algorithms. VLSI implementations of dedicated architecture are optimised with regard to area, power and performance. The advantage of a dedicated architecture is that they provide a good performance, while they lack the flexibility to further extend an algorithm set. On the other hand the programmable approach can satisfy the requirement for a greater flexibility,
15 since algorithms can be further extended simply by applying the necessary software modifications. Due to the fact, that a number of applications can run on the hardware, the cost per application hardware is reduced. The disadvantage of a programmable architecture is that the architecture as well as application models must be faster, since the computational properties of algorithms are not fully exploited. This problem can be solved by implementing
20 multiprocessing strategies. It is therefore essential to provide an application model which has sufficient parallelism in order to fully exploit the computational resources of a multiprocessor architecture. One such application model which provides parallelism for signal processing applications is the Kahn process network model. An significant increase in processing speed can be achieved as soon as the application are executed in a multiprocessor architecture.

25 A typical Kahn process network is composed of a number of process networks, processes and FIFO memories (First In – First Out). In such a process network a process is implemented by a processor or a coprocessor and a FIFO represents a communication channel, i.e. the processors communicate with each other via FIFO buffers. The FIFOs have a single reader and single writer. A process usually defines the functionality

of the application. It interacts with its environment through input and output ports. A process can be in one of three states, namely running, blocked or ready. For example, a process will be blocked if it tries to read from an empty FIFO or if it tries to write into a full FIFO.

5 In the Kahn process network the processes communicate with each other in a pipelined manner, i.e. the slowest process in the pipeline determines its throughput. If the computation is properly balanced among processes it will reduce the effective parallelism within the application model.

Usually, processes communicate with each other via FIFOs. Now when these processes are implemented as software processes, an implementation of a FIFO includes a
10 buffer for storing the data, a read pointer, a write pointer as well as room and data semaphores for synchronising data and room in the buffer. Therefore, writing into a FIFO is implemented firstly by waiting for free room, i.e. space, in said FIFO. When there is room available in said FIFO, data is input into the FIFO and the write pointer is incremented. Finally, it is signalled that valid data has been added into the FIFO. Reading from a FIFO is
15 symmetrical to writing, i.e. the roles of room and data as well as the writer and reader pointer have changed.

In a multiprocessing environment there is usually a conflict for the available communication resources, like buses etc. . The communication expenses increase as the number of processors in the architecture increases. Therefore optimising the application
20 model with regard to a reduced communication will result in an improved speed. A reduced communication is achievable by carefully choosing the token structure and controlling the number of tokens transferred over the communication channels.

It is therefore an object of the invention to improve the communication between a coprocessor and a controller in a multiprocessing environment.

25 This object is solved by a device for writing data elements from a coprocessor into a FIFO memory according to claim 1, a method for writing data elements from a coprocessor into a FIFO memory according to claim 7, a device for reading data elements from a FIFO memory into a coprocessor according to claim 12, a method for reading data elements from a FIFO memory into a coprocessor according to claim 18, and a
30 multiprocessing computer system according to claim 23.

The invention is based on the idea to maintain two counters for an input or output port of a FIFO.

According to the invention, a device for writing data elements from a coprocessor into a FIFO memory is provided. Said device is embedded in a multiprocessing

environment comprising at least one coprocessor, a FIFO memory and a controller. Said device comprises a first counter for counting the available room in said FIFO memory, and a second counter for counting the number of data elements written into said FIFO memory. Said device further comprises a control means for checking said first counter for available room in said FIFO memory, and for checking said second counter whether a predetermined number N of data elements have been written into said FIFO memory. Said control means decrements the count of said first counter and increments the count of said second counter, after a data element has been written into said FIFO memory. Said device finally comprises an output means for outputting data elements to said FIFO memory. Said control means issues a first message when the count of said second counter has reached said predetermined number N and issues a first call for available room in said FIFO memory to said controller. Said output means forwards said first message and/or said first call to said controller.

The advantage of a writing device according to the invention is that by using said first and second counter for writing data into the FIFO memory there is no need for a coprocessor performing a write operation to request the controller for two semaphore operations (a P and a V operation) when writing to said FIFO, thereby reducing the communication traffic between coprocessor and controller.

According to an embodiment of the present invention said first message indicates that sufficient data elements have been written into said FIFO memory. Thus advantageously the communication traffic between coprocessor and controller is further reduced since said message is only transmitted after N data elements have been written into said FIFO.

According to a further embodiment of the present invention, said control means increments a write pointer, when data elements were output to said FIFO memory.

According to still a further embodiment of the present invention, said control means performs a wrap-around test after said write pointer was incremented.

According to an embodiment of the present invention, said control means resets said second counter after issuing said first message, i.e. the counter is reset after its count has reached N and said message has been issued so that the count can start again.

According to a further embodiment of the present invention, said control means issues said first call for available room in said FIFO memory to said controller before said count of said first counter becomes zero. This is advantageous since the latency of such a call is hidden.

The invention also concerns a method for writing data elements from a coprocessor to a FIFO. Said method corresponds to the device for writing data elements as described above.

Moreover, the invention also provides a device and method for reading data
5 elements from a FIFO into a coprocessor. Said device and method are complementary to the writing device and writing method. Here, data is transferred from the FIFO to the coprocessor. Instead of the first message and first call, a second message indicating that sufficient data elements have been read from said FIFO, i.e. that room is available and a second call requesting available data elements in said FIFO, is issued.

10 According to the invention a multiprocessing computer system according to claim 23 is also provided. Furthermore, the invention also provides a computer program according to claim 24 for implementing said methods.

15 The invention will now be explained in more detail with reference to the drawing, in which:

Fig. 1 shows a block diagram of a multiprocessing computer system according to an embodiment of the invention,

20 Fig. 2 shows a block diagram of a writing device associated to coprocessors in Fig. 1, and

Fig. 3 shows a block diagram of a reading device associated to coprocessors in Fig. 1.

25 Fig. 1 shows a block diagram of a multiprocessing computer system according to an embodiment of the invention. Said computer system comprises memory banks 1, preferably in form of FIFO memories, three CPU's 3, three coprocessors 2, a controller 4 and an interconnect, like a bus, for connecting said FIFO memories with the CPU's 3, the coprocessors 2 and the controller 4. Said controller 4 is preferably embodied as a
30 programmable processor executing semaphore operations on request of the coprocessors 2 and preferably communicates with the coprocessors 2 by means of a dedicated ring network. Part of the processes are implemented in software and run on a programmable processor, i.e. a CPU 3, while the other part of the processes are implemented in dedicated hardware and

run on coprocessors 2. The memory 1 is shared between the processors 3 and the coprocessors 2 and is usually a random access memory.

For more details on this so called CAKE architecture please refer to “Exploring Design Space of Parallel Realizations: MPEG-2 Decoder Case Study”, by Basant et al. in Proc. of CODES 2001, Copenhagen, April 2001, pp. 92-97.

Fig. 2 shows a block diagram of a writing device associated to coprocessors 2 in Fig. 1. Said writing device comprises a first counter 10 and a second counter 20, a control means 30 and an output means 40, wherein said control means 30 is connected to said first and second counter 10, 20 and said output means 40, respectively. Said control means 30 has a terminal 31 to which a coprocessor 2 can be connected. Said output means 40 has a first terminal 41 connectable to said FIFO 1 and a second terminal 42 connectable to the controller 4, wherein said connection is preferably implemented via a dedicated ring network.

Said first counter 10 is used to count the available room in said FIFO 1, i.e. the count of said first counter 10 is room_available. It contains the number of free spaces in said FIFO 1, into which data can be written to before a call for free space or room need to be issued. Said second counter 20 is used to count the available data in said FIFO 1, i.e. the count of said second counter 20 is data_produced. It contains the number of data that has been written into the FIFO 1 but which have not been reported to the controller 4 yet.

First of all, the control means 30 checks the count of the first counter 10, i.e. the room_available counter 10. If the count of the first counter 10 is zero, the control means 30 issues a first call C1, i.e. a get_room call, to the controller 4 asking for room in the FIFO 1. If the controller 4 return a message that no room is currently available in the FIFO, the control means 30 repeats to issue a get_room call until the controller 4 returns that there is space available in the FIFO 1. When there is space available in said FIFO the output means 40 outputs data to the FIFO 1, where the data is stored. Next the control means 30 increments a write pointer and performs a pointer wrap-around test.

The wrap around test is performed by comparing the write/read pointer with a end-of-buffer address and if they equal each other, the write/read pointer is set to the begin-of-buffer address. Preferably the begin-of-buffer as well as the end-of-buffer addresses are stored in the coprocessor.

Thereafter, the control means 30 decrements the count of the first counter 10, i.e. the room_available counter, and increments the count of the second counter 20, i.e. the data_produced counter. The control means 30 determines whether the count of the second counter 20 has reached a predetermined number N, indicating that N data elements have been

written into the FIFO and are hence available to be read from said FIFO 1. As soon as the count of the second counter 20 has reached the predetermined number N, the control means 30 issues a first message M1, i.e. put_data, to the controller 4. Alternatively, the put_data message may have an argument indicating how many data have been produced.

5 Fig. 3 shows a block diagram of a reading device associated to coprocessors 2 in Fig. 1. Said reading device comprises a third counter 50 and a fourth counter 60, a control means 70 and an input means 80, wherein said control means 70 is connected to said third and fourth counter 50, 60 and said input means 80, respectively. Said control means 70 has a terminal 71 to which a coprocessor 2 can be connected. Said input means 80 has a first
10 terminal 81 connectable to said FIFO 1 and a second terminal 82 connectable to the controller 4, wherein said connection is preferably implemented via a dedicated ring network.

Said third counter 50 is used to count the available data in said FIFO 1, i.e. the count of said first counter 10 is data_available. It contains the number of free data elements in said FIFO 1 which can be read before a call for data need to be issued. Said fourth counter
15 60 is used to count the available free space or room in said FIFO 1, i.e. the count of said second counter 20 is room_produced. It contains the number of data that have been read from the FIFO buffer 1 but which have not been reported to the controller 4 yet.

First of all, the control means 70 checks the count of the third counter 50, i.e. the data_available counter. If the count of the third counter 50 is zero, the control means 70
20 issues a second call C2, i.e. a get_data call, to the controller 4 asking for data in the FIFO 1. If the controller 4 return a message that no data is currently available in the FIFO 1, the control means 70 repeats to issue a get_data call until the controller 4 returns that there is data available in the FIFO 1. When there is data available in said FIFO 1 the input means 80 inputs or reads data from the FIFO 1. Next the control means 70 increments a read pointer
25 and performs a pointer wrap-around test.

Thereafter, the control means 70 decrements the count of the third counter 50, i.e. the data_available counter, and increments the count of the fourth counter 60, i.e. the room_produced counter. The control means 70 determines whether the count of the fourth counter 60 has reached a predetermined number N, indicating that N data elements have been
30 read from the FIFO 1 and that hence there are spaces available into which data can be written to. As soon as the count of the fourth counter 60 has reached the predetermined number N, the control means 70 issues a second message M2, i.e. put_room, to the controller 4. Alternatively, the put_room message may have an argument indicating how many data has been produced.

Accordingly, the controller 4 implemented as a programmable processor must be able to accept a get_room call and a put_data message from the control means 30 of the writing device as well as a get_data call and a put_room message from the control means 70 of the reading device. When the controller 4 receives a get_room call, it resets the value of a corresponding room semaphore to zero and returns the previous value of the room semaphore. This value is then returned to the coprocessor 2 or its associated writing device which issued the get_room call. When the controller 4 receives the put_data message from a writing device associated to a coprocessor 2, it increments the data semaphore by a specific value. This value may be the predetermined value N or alternatively the argument of the put_data message.

The implantation of the reaction of the controller 4 towards the get_data call and the put_room message is similar to the implementations of the get_room call and the put_data message. When the controller 4 receives a get_data call, it resets the value of a corresponding data semaphore to zero and returns the previous value of the data semaphore. This value is then returned to the coprocessor 2 or its associated reading device which issued the get_data call. When the controller 4 receives the put_room message from a reading device associated to a coprocessor 2, it increments the room semaphore by a specific value. This value may be the predetermined value N or alternatively the argument of the put_room message.

The above described communication scheme can be improved by issuing a get_data or get_room call requesting data or room in the FIFO 1 before the first counter 10, i.e. data_available counter, or the third counter 50, i.e. the room_available counter, have become zero. This improvement will hide the latency of the get_data/get_room calls.

As controller 4 a low-end MIPS PR 1910 processor can be used.

As mentioned above processes may be implemented either on a programmable processor or on a dedicated coprocessor. The choice between those two alternatives is the classical hardware/software co-design question. Preferably, in the FIFO communication protocol buffer access, pointer increments and pointer wrap-around tests are implemented in hardware, i.e. dedicated coprocessors, while semaphore operations are implemented in software. It is desirable to perform semaphore operations in software, i.e. the processes are executed on the programmable processors, because of their complexity, e.g. waking up sleeping processes.

CLAIMS:

1. Device for writing data elements from a coprocessor into a FIFO memory, in a multiprocessing environment comprising at least one coprocessor, a FIFO memory and a controller, said device comprising:
- a first counter for counting the available room in said FIFO memory;
 - 5 - a second counter for counting the number of data elements written into said FIFO memory;
 - control means for checking said first counter for available room in said FIFO memory, for checking said second counter whether a predetermined number N of data elements have been written into said FIFO memory, for decrementing the count of said first
 - 10 counter and for incrementing the count of said second counter after a data element has been written into said FIFO memory; and
 - output means for outputting data elements to said FIFO memory;
 - wherein said control means is adapted to issue a first message when the count of said second counter has reached said predetermined number N;
 - 15 wherein said control means is adapted to issue a first call for available room in said FIFO memory to said controller; and
 - wherein said output means is adapted to forward said first message and/or said first call to said controller.
- 20 2. Device according to claim 1, wherein said first message indicates that sufficient data elements have been written into said FIFO memory.
3. Device according to claim 2, wherein
- 25 said control means is further adapted to increment a write pointer, when data elements were output to said FIFO memory.
4. Device according to claim 3, wherein

said control means is further adapted to perform a wrap-around test after said write pointer was incremented.

5. Device according to claim 2, wherein
5 said control means is adapted to reset said second counter after issuing said first message.
6. Device according to claim 1 or 2, wherein
said control means is adapted to issue said first call for available room in said
10 FIFO memory to said controller before said count of said first counter becomes zero.
7. Method for writing data elements from a coprocessor into a FIFO memory
being in a multiprocessing environment comprising at least one coprocessor, a FIFO memory
and a controller, said method comprising the steps of:
15 - checking a first counter, indicating the available room in said FIFO memory,
in order to determine whether there is room available in said FIFO memory,
- issuing a first call for available room in said FIFO memory to said controller
until there is room in said FIFO memory;
- outputting data elements to said FIFO memory;
20 - decrementing the count of said first counter after a data element has been
written into said FIFO memory;
- incrementing a second counter for counting the number of data elements
written into said FIFO memory after a data element has been written into said FIFO memory;
- checking said second counter in order to determine whether a predetermined
25 number N of data elements have been written into said FIFO memory; and
- issuing a first message that sufficient data elements have been written into said
FIFO memory when the count of said second counter has reached said predetermined number
N.
- 30 8. Method according to claim 7, further comprising to step of:
- incrementing a write pointer, when data elements were written into said FIFO
memory.
9. Method according to claim 8, further comprising to step of:

- performing a wrap-around test after said write pointer was incremented.

10. Method according to claim 7, further comprising to step of:

- resetting said second counter after issuing said first message.

5

11. Method according to claim 7, further comprising to step of:

- issuing said first call for available room in said FIFO memory to said controller before said count of said first counter becomes zero.

10 12. Device for reading data elements from a FIFO memory into a coprocessor, in a multiprocessing environment comprising at least one coprocessor, a FIFO memory and a controller, said device comprising:

- a third counter for counting the available data elements in said FIFO memory;
- a fourth counter for counting the number of data elements read from said FIFO

15 memory;

- control means for checking said third counter for available data element in said FIFO memory, for checking said fourth counter in order to determine whether a predetermined number N of data elements have been read from said FIFO memory, for decrementing the count of said third counter and for incrementing the count of said fourth
- 20 counter after a data element has been read from said FIFO memory; and

- input means for inputting data elements from said FIFO memory; wherein said control means is adapted to issue a second message when the count of said fourth counter has reached said predetermined number N;

25 wherein said control means is adapted to issue a second call for available data elements in said FIFO memory to said controller; and

wherein said output means is adapted to forward said second message and/or said second call to said controller.

13. Device according to claim 12, wherein

30 said second message indicates that sufficient data elements have been read from said FIFO memory.

14. Device according to claim 13, wherein

said control means is further adapted to increment a read pointer, when data elements were input from said FIFO memory.

15. Device according to claim 14, wherein

5 said control means is further adapted to perform a wrap-around test after said read pointer was incremented.

16. Device according to claim 13, wherein

10 said control means is adapted to reset said fourth counter after issuing said second message.

17. Device according to claim 13, wherein

15 said control means is adapted to issue said second call for available data elements in said FIFO memory to said controller before said count of said third counter becomes zero.

18. Method for reading data elements from a FIFO memory into a coprocessor being in a multiprocessing environment comprising at least one coprocessor, a FIFO memory and a controller, said method comprising the steps of:

- 20 - checking a third counter, indicating the available data elements in said FIFO memory, in order to determine whether there is data element available in said FIFO memory,
- issuing a second call for available data elements in said FIFO memory to said controller until there is data element in said FIFO memory;
- inputting data elements from said FIFO memory;
- 25 decrementing the count of said third counter after a data element has been read from said FIFO memory;
- incrementing a fourth counter for counting the number of data elements read from said FIFO memory after a data element has been read from said FIFO memory;
- checking said fourth counter in order to determine whether a predetermined
- 30 number N of data elements have been read from said FIFO memory; and
- issuing a second message that sufficient data elements have been read from said FIFO memory when the count of said fourth counter has reached said predetermined number N;

19. Method according to claim 18, further comprising to step of:
- incrementing a read pointer, when data elements were read from said FIFO memory.

5 20. Method according to claim 19, further comprising to step of:
- performing a wrap-around test after said read pointer was incremented.

21. Method according to claim 18, further comprising to step of:
- resetting said fourth counter after issuing said second message.

10

22. Method according to claim 18, further comprising to step of:
- issuing said second call for available data elements in said FIFO memory to said controller before said count of said third counter becomes zero.

15 23. Multiprocessing computer system, comprising:
- a FIFO memory;
- at least one coprocessor;
- a controller,
- a device for writing according to claims 1 to 6 and/or a device for reading
20 according to claim 12 to 17.

24. Computer program comprising computer program code means for causing a computer to perform the steps of the method as claimed in claims 7 to 11 as well as the method as claimed in claims 18 to 22 when said computer program is run on a computer.

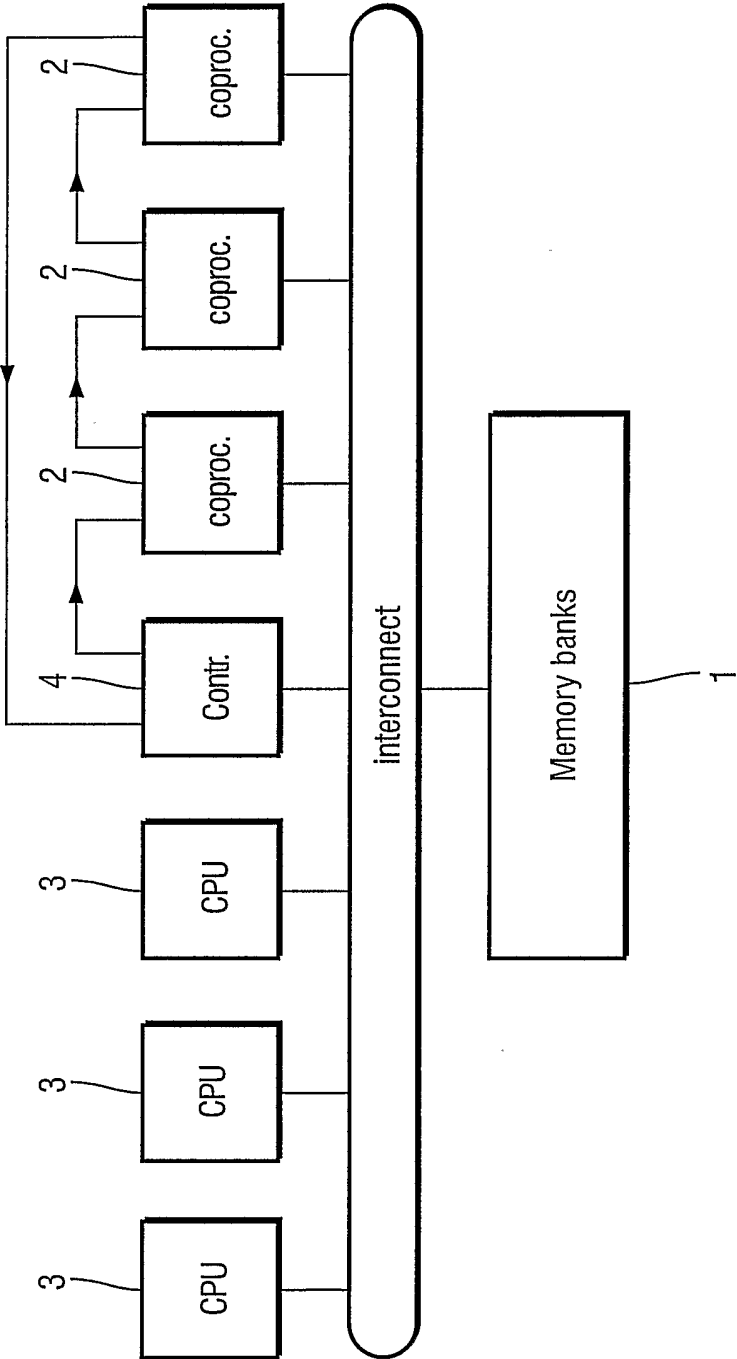


FIG.1

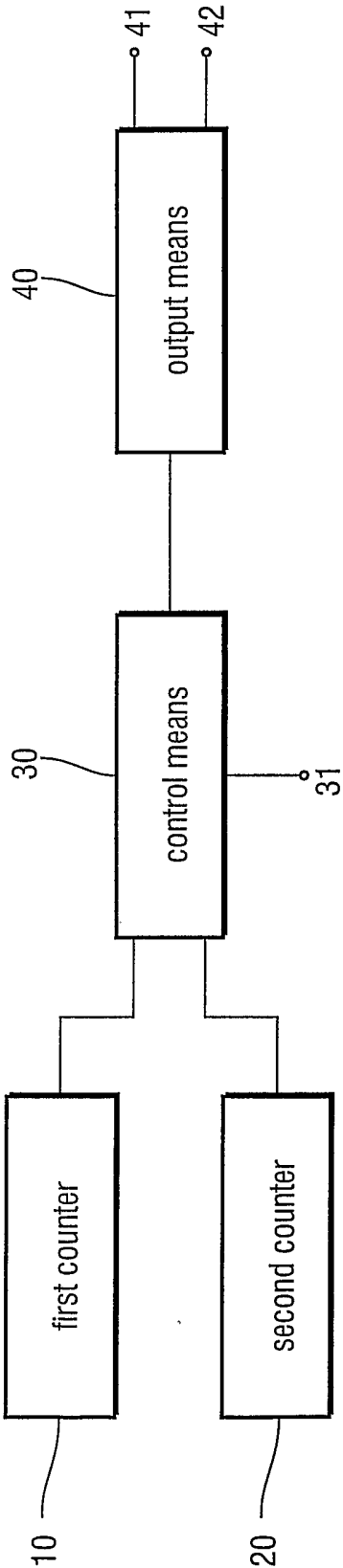


FIG. 2

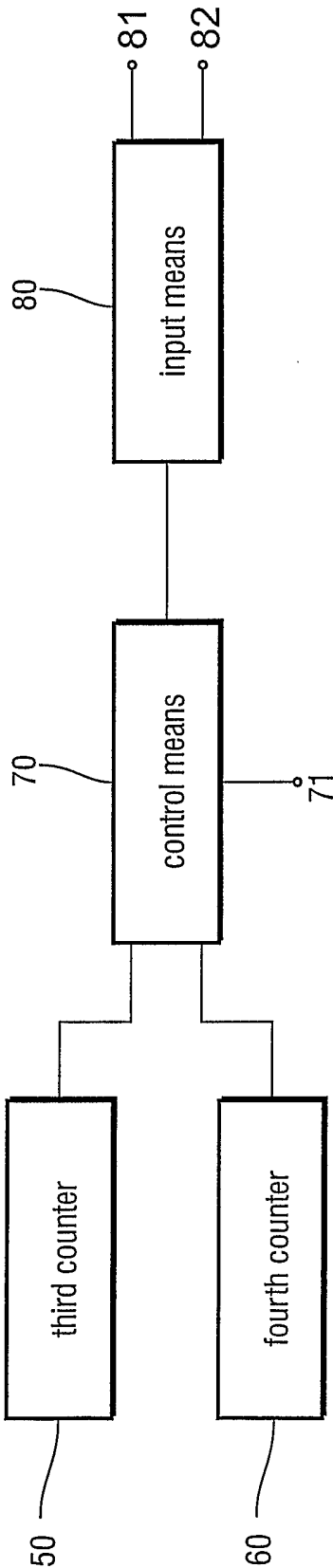


FIG. 3