

(19) World Intellectual Property Organization
International Bureau



(43) International Publication Date
5 June 2008 (05.06.2008)

PCT

(10) International Publication Number
WO 2008/065477 A1

(51) International Patent Classification:
G06F 12/00 (2006.01) **G06F 13/18** (2006.01)

(74) Agents: **DERRY, Paul** et al.; 20 Little Britain, London,
London EC1A 7DH (GB).

(21) International Application Number:
PCT/IB2006/054491

(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, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LV, LY, MA, MD, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, SV, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(22) International Filing Date:
28 November 2006 (28.11.2006)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant (*for all designated States except US*): **Nokia Corporation** [FI/FI]; Keilalahdentie 4, FIN-02150 Espoo (FI).

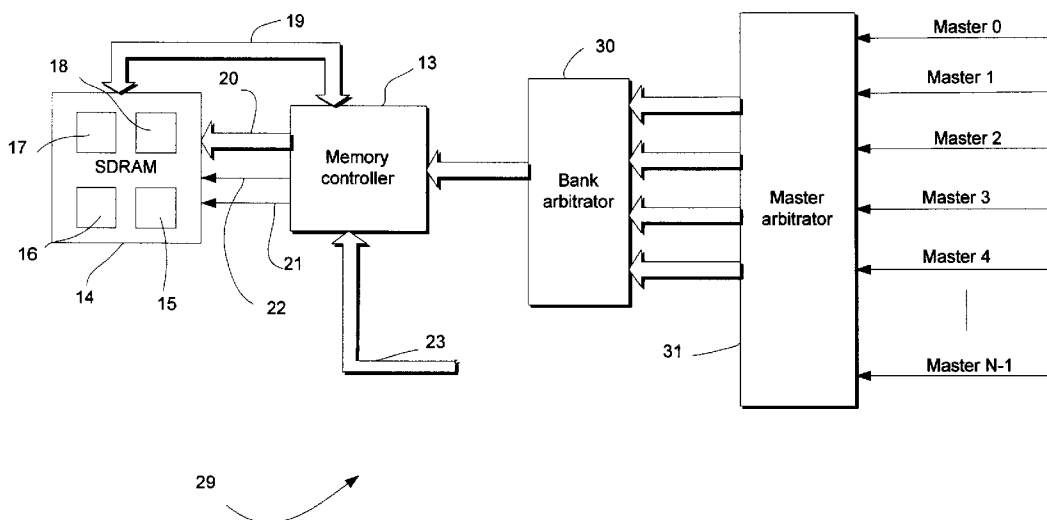
(84) Designated States (*unless otherwise indicated, for every kind of regional protection available*): ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT,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

(72) Inventors; and

(75) Inventors/Applicants (*for US only*): **KOIKKALAINEN, Mika** [FI/FI]; Vilppulanpolku 2 B 20, FIN-33720 Tampere (FI). **KOLINUMMI, Pasi** [FI/FI]; Saarenmaantie 111 A 1, FIN-36200 Kangasala (FI). **VEHVILANEN, Juhani** [FI/FI]; c/o Nokia Corporation, Keilalahdentie 4, FIN-02150 Espoo (FI).

Published:
— with international search report

(54) Title: MEMORY ARBITRATION



(57) Abstract: A memory arbitrator comprises plural ports, each for receiving commands from a respective master, and connectable to plural banks of memory via a memory interface, wherein the memory arbitrator is operable to arbitrate memory access requests received at the plural ports based at least in part on a comparison of an operation currently being performed and an operation requested in a received access request. Arbitration is performed also on the basis of a comparison of a row of memory in which an operation is currently being performed and a row of memory associated with a received access request. When moving to a different row, priority is given to access requests relating to different banks of memory.

WO 2008/065477 A1

Memory Arbitration

Description

This invention relates to memory arbitrators and to methods of arbitrating memory
5 access requests. The invention relates particularly, but not exclusively to SDRAM
memory arbitration.

It is relatively common for computing devices, including mains-powered home or
business machines such as personal computers (PCs), portable devices such as
10 laptop or notebook computers, and handheld devices such as mobile phones,
smartphones and personal digital assistants (PDAs) to utilise synchronous dynamic
RAM (SDRAM) as execution or working memory. Software applications which
utilise functions for the processing of still or moving images tend to require the
movement of significant amounts of data to and from SDRAM memory. There are
15 numerous other software applications which move significant amounts of data to
and from SDRAM memory. Especially in view of continuing advances in the
performance of microprocessors, it is increasingly the speed at which data can be
moved to and from SDRAM which limits software applications. Many efforts have
been made to increase the maximum clock speed of SDRAM memories, and the
20 operating speed of SDRAM continues to increase. Although it is technically
feasible to improve SDRAM performance by including additional SDRAM
interfaces, by providing ASIC internal memories or by increasing memory data
width, such measures are costly so are avoided wherever possible.

25 Some efforts have been made also to increase the efficiency of utilisation of the
interface to the SDRAM. For instance, US-B-5,987,574 discloses a memory
controller connected between plural devices and a memory comprising two banks of
SDRAM. As is conventional when a SDRAM memory having plural banks of
SDRAM is connected to a memory controller via an interface, only one command at
30 a time may be passed to the SDRAM over the interface. In this document, it is
described that a first bank can be precharged whilst a second bank is being read
from in burst mode. Performance improvements are obtained by arbitrating
sequential memory accesses to alternating memory banks.

The invention was made in this context.

5 According to a first aspect of the present invention, there is provided a memory arbitrator comprising plural ports, each for receiving commands from a respective master, and connectable to plural banks of memory via a memory interface, wherein the memory arbitrator is operable to arbitrate memory access requests received at the plural ports based at least in part on a comparison of an operation currently being performed and an operation requested in a received access request.

10

This aspect of the invention can provide more efficient utilisation of the interface between SDRAM memory and the masters, leading to an increase in the amount of data that can be passed over the interface in a given time period. The invention thus can give rise to improved overall system performance.

15

The memory arbitrator may be operable to arbitrate memory access requests so as to maintain a consecutive sequence of a predetermined length of operations of the same operation type, and, in response to a determination that the sequence of the predetermined length has been reached, to prioritise access requests relating to operations of a different operation type. This helps to avoid starving masters. Optionally, the memory arbitrator is operable to arbitrate memory access requests so as to maintain a consecutive sequence of a predetermined length of operations of the different operation type.

20

The memory arbitrator may be operable to arbitrate memory access requests based also at least in part on priorities associated with the masters from which the access requests are received.

25 The memory may be an SDRAM memory.

30

The operation types may be read and write.

The invention also provides a system comprising a memory arbitrator as recited above and a memory controller. The system may be operable to cause refreshing of a bank of memory whilst a read or write operation involving a different bank of memory is taking place. The memory arbitrator and the memory controller may be
5 integrated as a single device.

A second aspect of the invention provides a method of arbitrating memory access requests, the method comprising:

- receiving memory access requests from plural masters;
- 10 performing a comparison of an operation currently being performed with an operation requested in a received access request;
- arbitrating the received memory access requests based at least in part on the result of the comparison; and
- ordering the memory access requests for execution on the basis of the
15 arbitration.

According to a third aspect of the present invention there is provided a memory arbitrator comprising plural ports, each for receiving commands from a respective master, and connectable to plural banks of memory via a memory interface, wherein
20 the memory arbitrator is operable to arbitrate memory access requests received at the plural ports based at least in part on a comparison of a row of memory in which an operation is currently being performed and a row of memory associated with a received access request.

25 This aspect of the invention can provide more efficient utilisation of the interface between SDRAM memory and the masters, leading to an increase in the amount of data that can be passed over the interface in a given time period. The invention thus can give rise to improved overall system performance.

30 The memory arbitrator may be responsive to a determination that no pending access request relates to the row of memory in which an operation is currently being performed to select as a next access request to be performed an access request which relates to a different bank of memory as the operation currently being

performed. This can improve efficiency by avoiding idle cycles which would be incurred when moving between rows in one bank.

5 The memory arbitrator may be operable to arbitrate memory access requests so as to maintain a consecutive sequence of a predetermined length of operations of the same operation type to the same row and to the same bank, and, in response to a determination that the sequence of the predetermined length has been reached, to prioritise access requests relating to different banks. This can help to avoid starving masters.

10

The memory arbitrator may be operable to arbitrate access requests so as to implement a round robin sequence between banks of memory.

15 The memory arbitrator may be operable to arbitrate memory access requests received at the plural ports based at least in part on a comparison of the type of operation currently being performed and the type of operation requested in a received access request. This can improve efficiency by avoiding idle cycles incurred when changing operation type. In this case, the memory arbitrator may be operable to arbitrate memory access requests so as to maintain a consecutive
20 sequence of a predetermined length of operations of the same operation type, and, in response to a determination that the sequence of the predetermined length has been reached, to prioritise access requests relating to operations of a different operation type. Here, the memory arbitrator may be operable to arbitrate memory access requests so as to maintain a consecutive sequence of a predetermined length
25 of operations of the different operation type. These latter two features help to avoid starving masters.

The memory arbitrator may be operable to arbitrate memory access requests based also at least in part on priorities associated with the masters from which the access
30 requests are received.

The memory may be an SDRAM memory.

The operation types may be read and write.

The invention also provides a system comprising a memory arbitrator according to the third aspect of the invention and a memory controller.

5

This system may be operable to cause refreshing of a bank of memory whilst a read or write operation involving a different bank of memory is taking place. This can improve memory efficiency.

10 The memory arbitrator and the memory controller may be integrated as a single device.

According to a fourth aspect of the invention, there is provided a method of arbitrating memory access requests, the method comprising:

15 receiving memory access requests from plural masters;

performing a comparison of a row of memory in which an operation is currently being performed and a row of memory associated with a received access request;

20 arbitrating the received memory access requests based at least in part on the result of the comparison; and

ordering the memory access requests for execution on the basis of the arbitration.

Embodiments of the present invention will now be described by way of example only with reference to the accompanying drawings, of which:

25

Figure 1 is a schematic drawing of a first embodiment of a system-on-a-chip embodying the present invention;

Figure 2 is a flow chart illustrating operation of an arbitrator of the Figure 1 system; and

30 Figure 3 is a schematic drawing of a further embodiment of a system-on-a-chip according to the present invention.

Figure 1 is a schematic diagram of an embodiment of a system-on-a-chip. Referring to Figure 1, a system-on-a-chip 10 is shown schematically. The system 10 includes an arbitrator 11, a memory controller 13 and an SDRAM memory 14. The SDRAM memory includes first to fourth memory banks 15 to 18. The memory controller 13 is connected to the SDRAM by a data bus 19, which is 32 bits wide. The memory controller 13 also is connected to the SDRAM 14 by a 32 bit wide address bus 20, a row address strobe (RAS) line 21 and a column address strobe (CAS) line 22. The interface between the memory controller 13 and the SDRAM may conform to a JEDEC standard. A second data bus 23 is connected to the SDRAM controller 13 and to each of plural masters (not shown).

Instead of a 32 bit wide address bus 20, a narrower address bus could be used. In SDRAM, it is usual to use two phase addressing, in which row address and column addresses are provided separately, and in such systems a narrower bus is suitable.

15

The system includes numerous other components, which are omitted from the drawing and from this explanation since they are not crucial to the invention. The particular hardware implementation is not crucial to the invention, so is not described in detail here. The hardware implementation may for example be like the architecture shown in US 5,987,574.

20

The arbitrator 11 receives access requests from plural masters and includes a single output port connected to the memory controller 13.

The arbitrator 11 is connected to receive requests for SDRAM accesses from plural masters (not shown) at respective plural ports, which are labelled Master 0 to Master N-1 in Figure 1. The master devices may take any suitable form. In a mobile device, the master devices may include multiple processors, one or more accelerators, one or more direct memory controllers, a display controller, one or more graphics accelerators, etc.

30

The connection between the memory controller 13 and the SDRAM 14 is such that only one command can be passed in a clock cycle. However, the memory banks 15 to 18 are able to function independently of one another.

5 Each bank 15 to 18 has several rows and each row has several columns. Each column stores one word of data. The SDRAM 14 is controlled with commands that are given with control signals. SDRAM is a synchronous memory and all the commands are synchronous to the memory clock. Every clock cycle, only one command can be sent to the SDRAM 14. The possible commands, in this
10 implementation, are idle, read, write, active (open a row), precharge (close a row) and refresh (refresh content of a row). The data, address and control signals on the data bus 19, the address bus 20 and RAS and CAS lines 21, 22 are shared between the banks 15 to 18, but the commands can be given for each bank 15 to 18 separately. The banks 15 to 18 can process commands in parallel.

15 The SDRAM 14 is operable in to carry out burst read and write operations. In a burst operation, a read or write command is given on one clock cycle and a read or write operation is performed over a number of successive clock cycles and from successive memory locations. No further read or write command is given after the
20 initial command. During the successive clock cycles, commands can be given to any of the banks 15 to 18 which are not participating in the read or write operation.

Only read and write commands actually result in the transfer of data over the data bus 19; the other commands are a requirement of the physical implementation of
25 the SDRAM memory 14.

Operation is as follows. Briefly, the arbitrator 11 receives access requests from the masters, and the arbitrator 11 determines the manner in which the requests are to be fulfilled by the memory controller 13 and the SDRAM 14. This is particular
30 includes determining the order in which requests are fulfilled, and providing signals to the memory controller 13 accordingly. The memory controller 13 converts received internal ASIC bus operations into SDRAM control signals, which are fed to the SDRAM 14 so as to cause it to provide the required operations. The

memory controller 13 does not effect any reordering of the operations - it issues control signals in respect of access requests in the order in which the access requests were received from the arbitrator 12. As is described below, the memory controller 13 sends signals in respect of the opening of rows in advance of access requests relating thereto

In more detail, the arbitrator 11 performs arbitration of the different masters and divides the operations between the SDRAM banks 15 to 18 based on the operation addresses, which are received from the masters with their access requests. The number N of input ports can be configured.

The arbitrator 11 may perform pure round-robin arbitration, as is discussed in a little detail below, or it may perform arbitration based on priorities.

When arbitration is based partly on priorities, the master arbitrator 11 may be programmed with priorities that have been assigned to the masters. Alternatively, a master may send a signal which the arbitrator 11 receives and uses to assign a particular priority to that master. An assigned priority may be temporary. The capability to use priorities can allow latency requirements of the masters to be met.

In either case, where two or more masters provide access requests in respect of a given bank 15 to 18, the arbitrator 11 ensures that all of the relevant masters are granted access to the memory. This is achieved by the arbitrator 11 allowing only a predetermined, programmable, number of memory accesses before processing an access request from other masters. This helps to avoid starving of any particular master.

Access requests are held at the arbitrator 11 until they are passed onto the memory controller 13.

The arbitrator 11 performs operation order optimisation, as is described below with reference to Figure 2. Briefly, the arbitrator 11 operates in a round-robin fashion but gives priority to access requests which relate to the same row and are of the

same type, i.e. read or write, as an operation currently being performed, and also gives priority to access requests which of the same type, i.e. read or write, as an operation currently being performed, but relate to a different row.

5 Operation of the arbitrator 11 will now be described with reference to Figure 2.

Referring to Figure 2, the operation begins at step S1. In the Figure, the arbitrator 11 is already causing, through the memory controller 13, a read or write operation to occur between the SDRAM 14 and one of the masters. At step S1, the arbitrator
10 11 initialises a same operation type count and a same row count. At step S2, the arbitrator 11 determines the type, row and bank that the current operation relates to. In particular, at step S2, the arbitrator 11 determines whether the current operation is a read or a write operation, determines which of the four SDRAM banks 15 to 18 the operation relates to and determines which row in that bank the
15 operation relates to.

At step S3, the arbitrator 11 determines whether the same row count exceeds a threshold. For instance, the same row threshold might be sixteen. The threshold may alternatively be a significantly higher number. If the threshold is exceeded, this
20 is an indication that a large number of consecutive operations to the same row have been performed. In the event of a negative result, the operation proceeds to step S4.

At step S4, the arbitrator 11 determines whether any of the input ports has present
25 thereon an access request which relates to the same operation type, bank and row as the current operation. Any such access request could come from the same master that issued the access request relating to the operation currently being performed. Indeed, this is fairly likely. In particular, it is not uncommon for a master to require a number of consecutive operations each relating to the same row in a bank of
30 memory. However, occasionally one master will issue an access request relating to the same operation type, bank and row as an operation currently being performed in respect of an access request issued by another master. In the event of a negative

determination at step S4, or in the event of a positive determination at step S3, the operation proceeds to step S5. At step S5, the same row count is re-set.

In the event of a positive determination from step S4, the operation proceeds to
5 step S6. At this step, the next access request relating to the same operation type,
bank and row is assigned as the next access request to perform. Here, arbitrator 11
gives preference to the input port relating to the master which issued the access
request relating to the operation currently being performed. Thus, the arbitrator 11
arbitrates in favour of the same master if that master issues an access request
10 relating to the same operation type, bank and row, but assigns an access request
from another master as the next access request to perform if there is no further
access request from the master which issued the access request relating to the
operation currently being performed or if a next request from that master relates to
a different operation type, bank or row.

15

Following step S6, the same operation type count is incremented at step S7. At step
S8, following step S7, the same row count is incremented. Following step S8, the
operation proceeds to step S9, at which the next access request is performed.
Following step S9, the operation proceeds again to step S2.

20

Following step S5, it is determined at step S10 whether the same operation
threshold is exceeded. The same operation threshold may, for example, be sixteen.
Alternatively, the threshold may be significantly higher. In the event of step S10
yielding positive determination, operation proceeds to step S11. Here, the arbitrator
25 11 determines whether any access request received at any of the input ports relates
to a different operation type. In particular, if the current operation is a read
operation, the arbitrator 11 determines whether any of the access requests relates to
a write operation. If the current operation is a write operation, the arbitrator 11
determines whether any of the access requests relates to a read operation. In the
30 event of a positive determination, the operation proceeds to step S12. Here, the
next input port with an access request relating to a different operation type is
selected as the next access request to perform. At this step, priority is given to
access requests which relate to a different bank. Following step S12, the same

operation type count is re-set at step S13, and the operation then proceeds to perform the next access request at step S9.

5 The effects of steps S10, S11, S12 and S13 is to cause the arbitrator 11 to change to a different operation type if ten or more operations of the same type are performed consecutively, if there is an access request relating to a different operation type.

If step S11 yields a negative determination, at step S14 the arbitrator 11 determines whether any access request relates to a different bank. In particular, this requires
10 determining whether any of the access requests received at the input ports relates to one of the banks 15 to 18 which differs to the bank relating to the current operation. In the event of a negative determination, the next input port at which an access request is received is selected as the next access request to perform at step S16. In the event of a positive determination at step S14, the next input port with
15 an access request relating to a different bank is selected at step S15. As discussed below, an access request relating to the next bank in a sequence of banks is given priority.

The effect of steps S14 to S16 is to cause the arbitrator 11 to select as the next
20 access request to perform an access request which relates to a different one of the banks 15 to 18 of memory.

In the event of step S10 yielding a negative determination, the operation proceeds to step S17. Here, the arbitrator 11 determines whether any access request relates
25 to the same operation type and a different bank as the operation currently being performed. In the event of a negative determination, the operation proceeds to step S14, described above. In the event of step S17 yielding a positive determination, the bank arbitrator proceeds to step S18. Here, the next access request relating to the same operation type and a different bank as the current operation is assigned as
30 the next access request to perform. Following step S18, the same operation type count is incremented at step S19.

The effect of steps S17 and S18 is to cause the arbitrator 11 to give preference to access requests relating to the same operation type as the operation currently being performed, as long as the same operation threshold is not exceeded. In this way, the number of transitions from a read operation to a write operation and from a write operation to a read operation are reduced. This results in a performance improvement compared to prior SDRAM memory arbitrators. The performance improvement results from fewer idle cycles. Since a change in direction of data in the data bus requires an idle cycle, reducing the number of transitions reduces the number of idle cycles. Performance is improved also because read operations typically involve a delay between the command being issued and the data being provided by the SDRAM 14. The delay typically is three clock cycles. However, by following a read operation with another read operation, the second read command can be issued while the first read operation is in progress, so the data for the second read operation can be provided immediately after the data from the first read operation.

Following step S15, S16 or S19, the next access request is performed at step S9 before the operation again returns to step S2. The step of performing comprises the feeding by the arbitrator 11 of the access request to the memory controller 13, where it is pipelined. The number of access requests that the memory controller 13 can pipeline depends on its implementation. A pipeline of nine access requests might be typical. In due course after receiving the access request, the memory controller 13, for a read operation, activates (opens) a row of the relevant one of the banks 15 to 18 if that row is not already open. Where the next access request relates to write command, the memory controller 13 activates (opens) a row of the relevant one of the banks 15 to 18 if that row is not already open. In both cases, if another row in the relevant bank is already open, the memory controller 13 closes the open row before opening the new row. The memory controller 13 carries out these actions without further involvement of the arbitrator 11. The memory controller 13 is operable to prepare banks for operations two or three banks ahead of an operation currently being performed.

The action of the Figure 2 flowchart is advantageous since it reduces, potentially to zero, the number of idle cycles required between completing an operation and commencing the read or write part (not the opening or closing) of the succeeding operation. In particular, operations of the same type to the same row are performed
5 in sequence wherever possible (to a limit), and otherwise the arbitrator 11 switches to another bank wherever possible. Thus, the number of idle cycles are reduces compared to the corresponding prior art system.

In some implementations, it can be important that every master has access to the
10 SDRAM 14 in a given time period. In this case, the operation of steps S6, S12 and S18 gives priority to masters which have not had access to the SDRAM 14 for the greatest length of time.

Although not shown in the Figure, the arbitrator 11 is operable to provide a list of
15 access requests in the order in which they are to be performed. Thus, the arbitrator 11 'looks ahead' by more than one access request. The list of access requests is fed to the memory controller 13 such that the memory controller 13 typically has two or more pending access requests in a pipeline. This provides greater opportunity to prepare a row of memory for an operation before the immediately preceding
20 operation has completed, thus reducing the number of idle cycles. In this connection, it will be appreciated that the term "current operation" refers to the access request which was the last to be fed to the memory controller 13. The current operation may remain pipelined at the time.

25 The operation of the arbitrator 11 to prioritise access requests relating to the same operation type, row and bank provides certain advantages. In particular, this increases the probability that the row of memory that a future operation relates to is prepared (opened) by the memory controller 13 before the immediately preceding operation is completed. In this connection, it will be appreciated that closing a row
30 typically takes 3 or 4 clock cycles, and opening a row typically takes 3 or 4 clock cycles. In cases where the row of memory that the future operation relates to is not opened by the memory controller before the immediately preceding operation is completed, the operation of the arbitrator 11 can reduce the number of idle cycles

needed before the row is opened. Additionally, because opening and closing rows consumes power, completing two operations in the same row of a bank of memory typically consumes less power than completing the same operations in different banks of memory.

5

In Figure 1, when the arbitrator 11 decides to perform an access request received in respect of a different input port, the arbitrator 11 attempts to process access requests relating to different banks in a predetermined sequence. Thus, following an operation relating to the first bank 15, the arbitrator 11 prioritises access requests relating to the second bank 16 over access requests meeting the same criteria but relating to other ones of the banks 17, 18 of the SDRAM 14. Thus, the arbitrator 11 addresses the banks 15 to 18 in a round-robin manner wherever possible. This relates in particular to the steps S6, S12, S16 and S18 in Figure 1. This is advantageous since it maximises the time between successive and non-consecutive accesses to a given bank 15 to 18 of SDRAM 14. This improves efficiency of utilisation of the SDRAM 14 since it maximises the possibility of the bank having been prepared for access following the previous access.

10

15

The presence of the steps S3 and S8 ensure that the bank arbitrator 12 does not become stuck performing operations into one particular row, since the step ensures that the bank is changed periodically if access requests relating to the other banks are present. This helps to avoid starving of masters.

20

It will be appreciated that various other schemes for prioritising access requests relating to the same row as a current operation are possible.

25

The presence of the steps S10 and S19 ensure that the bank arbitrator 12 does not become stuck performing one particular type of operation, since the step ensures that the operation type is changed periodically if access requests relating to the other operation type are present. In this connection, it will be appreciated that successful operation can be achieved if the count of consecutive operations of the same type is reset prior to step S12 or following a negative result from step S12. Indeed, such operation may in some circumstances provide better performance.

30

It will be appreciated that various other schemes for prioritising access requests relating to an operation of the same type as a current operation are possible.

5 In the above, the arbitrator 11 causes a move to another bank if no access request relates to the same bank, row and operation type. In other embodiments, the arbitrator 11 is operable to arbitrate to remain in the same row if no access requests relating to the same row, bank and operation type are received if an access request relating to the same bank and row but different operation type is received. This
10 provides performance advantages in particular when the next row that would otherwise be moved onto would not be opened in time to perform the next operation immediately on completion of preceding operations. Although changing between read and write operations necessarily results in some inefficiency, in so far as data is not able to be communicated over the interface on every clock cycle, the
15 number of idle cycles may nevertheless be less than the number of idle cycles needed to wait to a row in another bank to be opened.

The arbitrator 11 and the memory controller 13 are operable to coordinate to perform SDRAM refreshes in a novel way. In particular, the memory controller 13
20 is operable in response to signals from the arbitrator 11 to issue a refresh control signal to a SDRAM bank 15 to 18 whilst another of the banks is performing a burst mode read or write operation. Thus, a bank is refreshed whilst a read or write operation is being performed in another bank.

25 Alternatively, the same effect is achieved by the memory controller 13 being arranged to refresh a bank and to inform the arbitrator 11 which bank is being refreshed. In response to receiving such information, the arbitrator 11 is arranged to refrain from arbitrating access requests relating to that bank until the bank refresh is complete.

30

In a conventional SDRAM system, all the banks are refreshed at the same time. Whilst the banks are refreshing, no read or write operations can be performed. Thus, this feature of the invention reduces SDRAM idle time compared to the prior

art. With this feature of the invention, the number of refresh periods increases, but at any given time some part of the memory can be used. Depending on the SDRAM implementation, this refresh scheme can result in a performance of improvement of about 1-2%.

5

Although Figure 2 has been described with reference to the single arbitrator system of Figure 1, in another embodiment the same functionality is provided by a dual arbitrator system, shown in Figure 3.

10 Referring to Figure 3, a further embodiment of a system-on-a-chip 10 is shown schematically. Reference numerals are retained from Figure 1 for like elements. The primary difference is that the Figure 3 system 29 includes a bank arbitrator 30 and a master arbitrator 31, rather than a single arbitrator as in Figure 1.

15 In this embodiment, the master arbitrator 31 performs arbitration of the access requests received from the masters. The master arbitrator 31 passes to the bank arbitrator 30 access requests on four different ports, each port relating to a different one of the banks 15 to 18 of SDRAM 14. The bank arbitrator 30 then arbitrates these access requests and provides access requests on a single port to the memory
20 controller 13.

The master arbitrator 31 may perform pure round-robin arbitration, or it may perform arbitration based on priorities.

25 By round-robin, it is meant that masters are granted access to the bank arbitrator 30 strictly in a predetermined sequence. If a master does not have an access request pending, access to the bank arbitrator 30 is granted to the next master in the sequence which does have a pending access request that relates to the same bank 15 to 18. In this way, accesses to the bank arbitrator 30 are not granted to idle
30 masters.

Master arbitration may instead be based partly on priorities. For instance, the master arbitrator 31 may be programmed with priorities that have been assigned to

the masters. Alternatively, a master may send a signal which the master arbitrator 31 receives and uses to assign a particular priority to that master. An assigned priority may be temporary. The capability to use priorities can allow latency requirements of the masters to be met.

5

In either case, where two or more masters provide access requests in respect of a given bank 15 to 18, the master arbitrator 31 ensures that all of the relevant masters are granted access to the bank arbitrator 30. This is achieved by the master arbitrator 31 allowing only a predetermined, programmable, number of memory
10 accesses before processing an access request from other masters. This helps to avoid starving of masters.

Access requests are held at the bank arbitrator 30 until they are passed onto the memory controller 13. A master does not receive any acceptance of an access
15 request until the memory controller 13 accepts the request, which necessarily is after arbitration by the arbitrator 11.

The master arbitrator 31 and the bank arbitrator 30 together may perform exactly the same function as the arbitrator 11 of Figure 1. This requires separate processes
20 to be performed in the master arbitrator 31 and the bank arbitrator 30, so the flowchart of Figure 2 cannot be used in that form.

The use of two separate arbitrators 31 and 30 provides certain advantages. In particular, the master arbitrator 31 can provide improved arbitration of the masters,
25 in particular by ensuring that no master monopolises the memory interface and by ensuring that each master gets access to the SDRAM 14 in a given time period.

Various modifications to the above can be made whilst remaining within the scope of the inventions.

30

For instance, the bank arbitrator 30 and the memory controller 13 can be combined into a single device, and the master arbitrator 31 connected to the combined bank

arbitrator and memory controller. Alternatively, the bank arbitrator 30, the memory controller 13 and the master arbitrator 31 can be combined into a single device.

5 The memory controller 13 and the Figure 1 arbitrator 11 may be combined into a single device.

Also, although the above has been described with reference to a single SDRAM, it will be appreciated that the teachings can be applied also to a system including plural SDRAMs, each with plural banks. Furthermore, the teachings can be applied
10 to memory systems other than SDRAM memory systems.

The scope of the invention is not limited by the described embodiments, but by the scope and spirit of the appended claims and their equivalents.

Claims

1. A memory arbitrator comprising plural ports, each for receiving commands from a respective master, and connectable to plural banks of memory via a memory interface, wherein the memory arbitrator is operable to arbitrate memory access requests received at the plural ports based at least in part on a comparison of an operation currently being performed and an operation requested in a received access request.
2. A memory arbitrator as claimed in claim 1, wherein the memory arbitrator is operable to arbitrate memory access requests so as to maintain a consecutive sequence of a predetermined length of operations of the same operation type, and, in response to a determination that the sequence of the predetermined length has been reached, to prioritise access requests relating to operations of a different operation type.
3. A memory arbitrator as claimed in claim 2, wherein the memory arbitrator is operable to arbitrate memory access requests so as to maintain a consecutive sequence of a predetermined length of operations of the different operation type.
4. A memory arbitrator as claimed in any preceding claim, wherein the memory arbitrator is operable to arbitrate memory access requests based also at least in part on priorities associated with the masters from which the access requests are received.
5. A memory arbitrator as claimed in any preceding claim, wherein the memory is an SDRAM memory.
6. A memory arbitrator as claimed in any preceding claim, wherein the operation types are read and write.
7. A system comprising a memory arbitrator according to any preceding claim and a memory controller.

8. A system as claimed in claim 7, wherein the system is operable to cause refreshing of a bank of memory whilst a read or write operation involving a different bank of memory is taking place.

5

9. A system as claimed in claim 7 or claim 8, wherein the memory arbitrator and the memory controller are integrated as a single device.

10. A method of arbitrating memory access requests, the method comprising:
10 receiving memory access requests from plural masters;
performing a comparison of an operation currently being performed with an operation requested in a received access request;
arbitrating the received memory access requests based at least in part on the result of the comparison; and
15 ordering the memory access requests for execution on the basis of the arbitration.

11. A method as claimed in claim 10, wherein the arbitrating step comprises arbitrating memory access requests so as to maintain a consecutive sequence of a
20 predetermined length of operations of the same operation type, and, in response to a determination that the sequence of the predetermined length has been reached, prioritising access requests relating to operations of a different operation type

12. A method as claimed in claim 11, wherein the arbitration step comprises
25 arbitrating memory access requests so as to maintain a consecutive sequence of a predetermined length of operations of the different operation type.

13. A method as claimed in any of claims 10 to 12, wherein the arbitration step comprises arbitrating memory access requests based also at least in part on priorities
30 associated with the masters from which the access requests are received.

14. A method as claimed in any of claims 10 to 13, wherein the memory is an SDRAM memory.

15. A method as claimed in any of claims 10 to 14, wherein the operation types are read and write.

5 16. A method as claimed in any of claims 10 to 15, comprising causing refreshing of a bank of memory whilst a read or write operation involving a different bank of memory is taking place.

10 17. A memory arbitrator comprising plural ports, each for receiving commands from a respective master, and connectable to plural banks of memory via a memory interface, wherein the memory arbitrator is operable to arbitrate memory access requests received at the plural ports based at least in part on a comparison of a row of memory in which an operation is currently being performed and a row of memory associated with a received access request.

15 18. A memory arbitrator as claimed in claim 17, wherein the memory arbitrator is responsive to a determination that no pending access request relates to the row of memory in which an operation is currently being performed to select as a next access request to be performed an access request which relates to a different bank
20 of memory as the operation currently being performed.

19. A memory arbitrator as claimed in claim 17 or claim 18, wherein the memory arbitrator is operable to arbitrate memory access requests so as to maintain a consecutive sequence of a predetermined length of operations of the same operation
25 type to the same row and to the same bank, and, in response to a determination that the sequence of the predetermined length has been reached, to prioritise access requests relating to different banks.

20. A memory arbitrator as claimed in any of claims 17 to 19, wherein the
30 memory arbitrator is operable to arbitrate access requests so as to implement a round robin sequence between banks of memory.

21. A memory arbitrator as claimed in any of claims 17 to 20, wherein the memory arbitrator is operable to arbitrate memory access requests received at the plural ports based at least in part on a comparison of the type of operation currently being performed and the type of operation requested in a received access request.

5

22. A memory arbitrator as claimed in claim 21, wherein the memory arbitrator is operable to arbitrate memory access requests so as to maintain a consecutive sequence of a predetermined length of operations of the same operation type, and, in response to a determination that the sequence of the predetermined length has
10 been reached, to prioritise access requests relating to operations of a different operation type

23. A memory arbitrator as claimed in claim 22, wherein the memory arbitrator is operable to arbitrate memory access requests so as to maintain a consecutive
15 sequence of a predetermined length of operations of the different operation type.

24. A memory arbitrator as claimed in any of claims 17 to 23, wherein the memory arbitrator is operable to arbitrate memory access requests based also at least in part on priorities associated with the masters from which the access requests
20 are received.

25. A memory arbitrator as claimed in any of claims 17 to 24, wherein the memory is an SDRAM memory.

25 26. A memory arbitrator as claimed in any of claims 17 to 25, wherein the operation types are read and write.

27. A system comprising a memory arbitrator according to any of claims 17 to 26 and a memory controller.

30

28. A system as claimed in claim 27, wherein the system is operable to cause refreshing of a bank of memory whilst a read or write operation involving a different bank of memory is taking place.

29. A system as claimed in claim 27 or claim 28, wherein the memory arbitrator and the memory controller are integrated as a single device.

5 30. A method of arbitrating memory access requests, the method comprising:
receiving memory access requests from plural masters;
performing a comparison of a row of memory in which an operation
is currently being performed and a row of memory associated with a received
access request;
10 arbitrating the received memory access requests based at least in part
on the result of the comparison; and
ordering the memory access requests for execution on the basis of the arbitration.

31. A method as claimed in claim 30, comprising, in response to a determination
15 that no pending access request relates to the row of memory in which an operation
is currently being performed, selecting as a next access request to be performed an
access request which relates to a different bank of memory as the operation
currently being performed.

20 32. A method as claimed in claim 30 or claim 31, comprising arbitrating memory
access requests so as to maintain a consecutive sequence of a predetermined length
of operations of the same operation type to the same row and to the same bank, and
responding to a determination that the sequence of the predetermined length has
been reached by prioritising access requests relating to different banks.

25 33. A method as claimed in any of claims 30 to 32, wherein the memory
arbitrator is operable to arbitrate access requests so as to implement a round robin
sequence between banks of memory.

30 34. A method as claimed in any of claims 30 to 33, comprising arbitrating
memory access requests received at the plural ports based at least in part on a
comparison of the type of operation currently being performed and the type of
operation requested in a received access request.

35. A method as claimed in claim 34, comprising arbitrating memory access requests so as to maintain a consecutive sequence of a predetermined length of operations of the same operation type, and responding to a determination that the
5 sequence of the predetermined length has been reached by prioritising access requests relating to operations of a different operation type

36. A method as claimed in claim 35, comprising arbitrating memory access requests so as to maintain a consecutive sequence of a predetermined length of
10 operations of the different operation type.

37. A method as claimed in any of claims 30 to 36, comprising arbitrating memory access requests based also at least in part on priorities associated with the masters from which the access requests are received.

15

38. A method as claimed in any of claims 30 to 37, wherein the memory is an SDRAM memory.

39. A method as claimed in any of claims 30 to 38, wherein the operation types
20 are read and write.

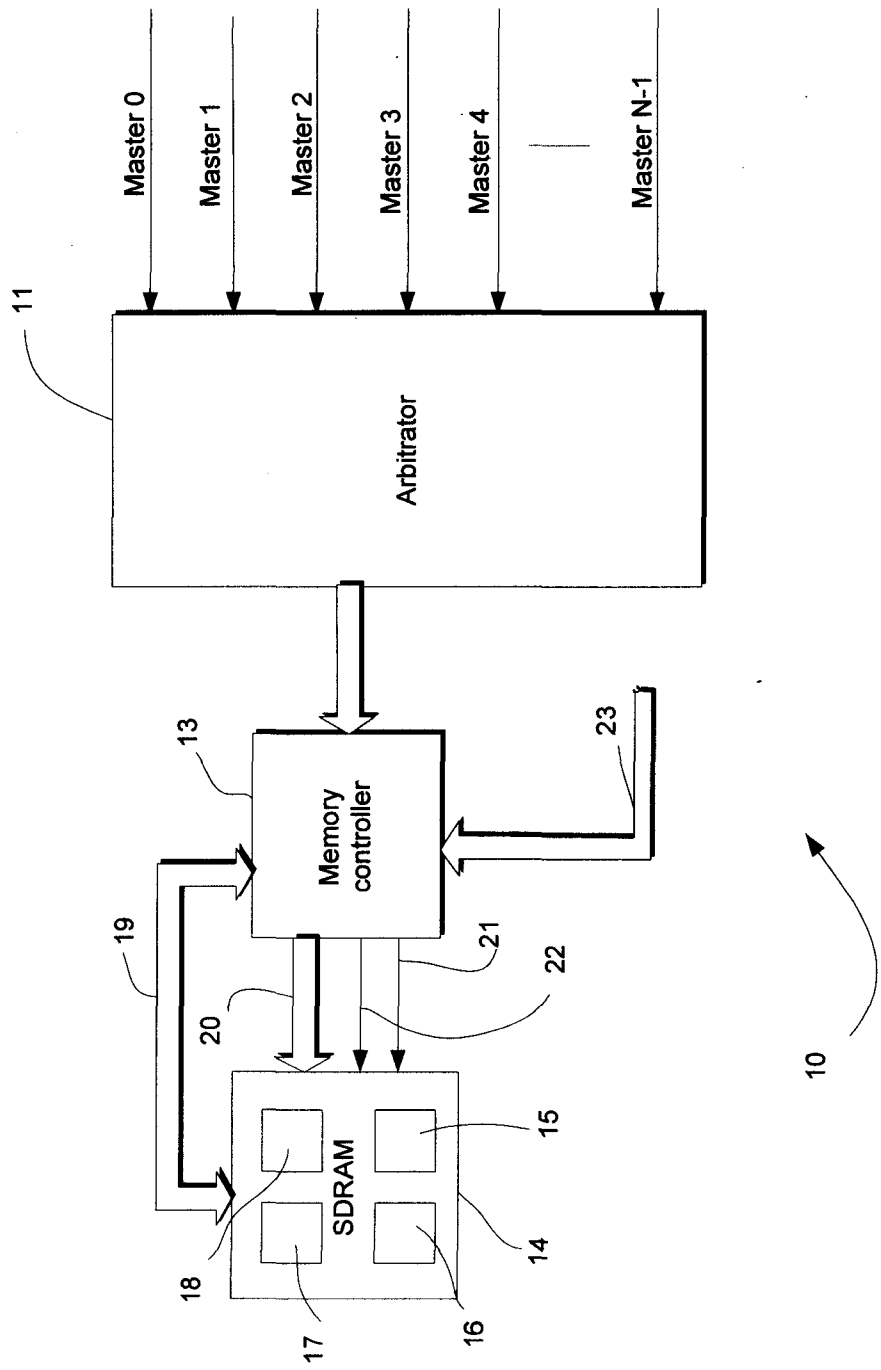


Figure 1

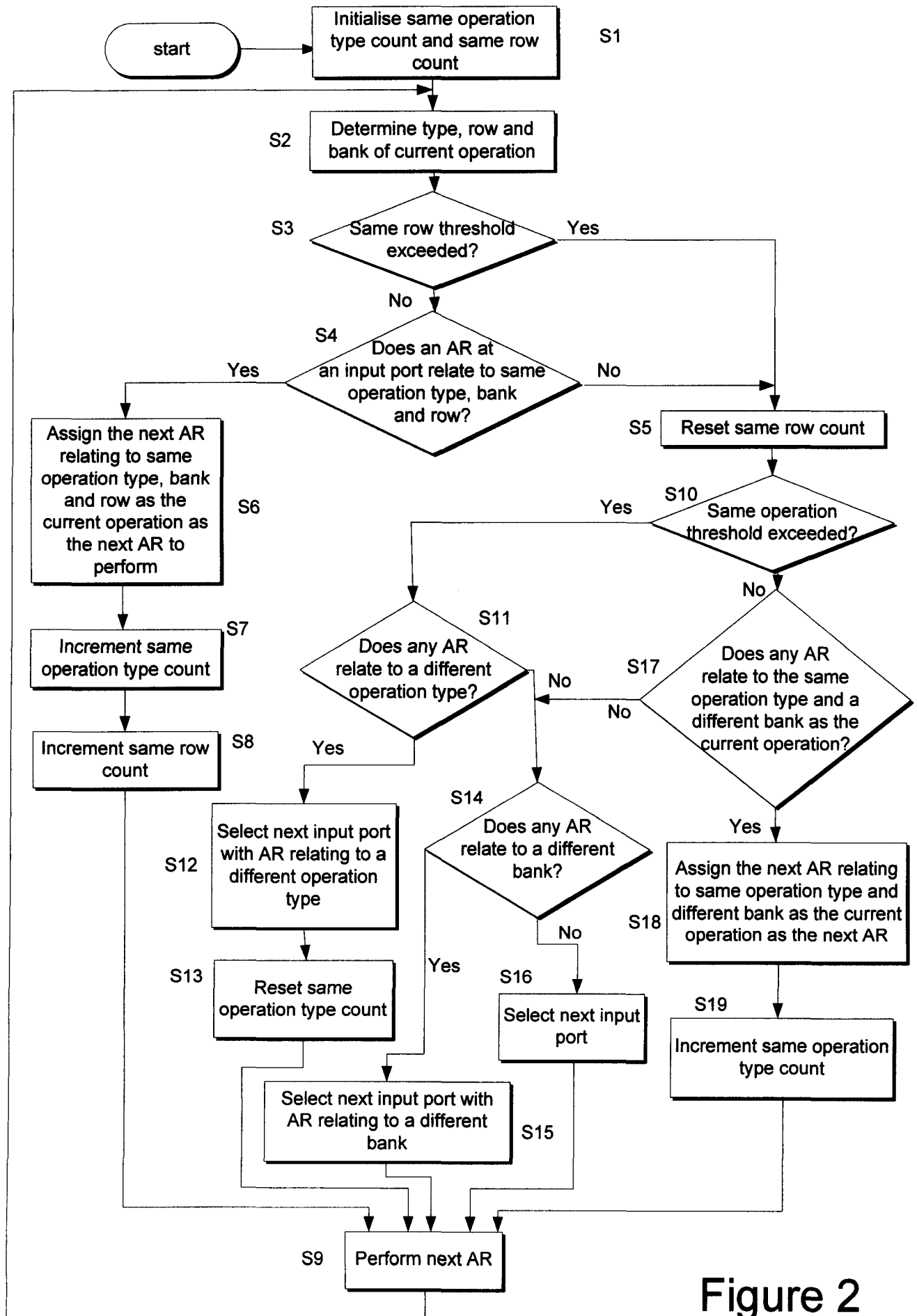


Figure 2

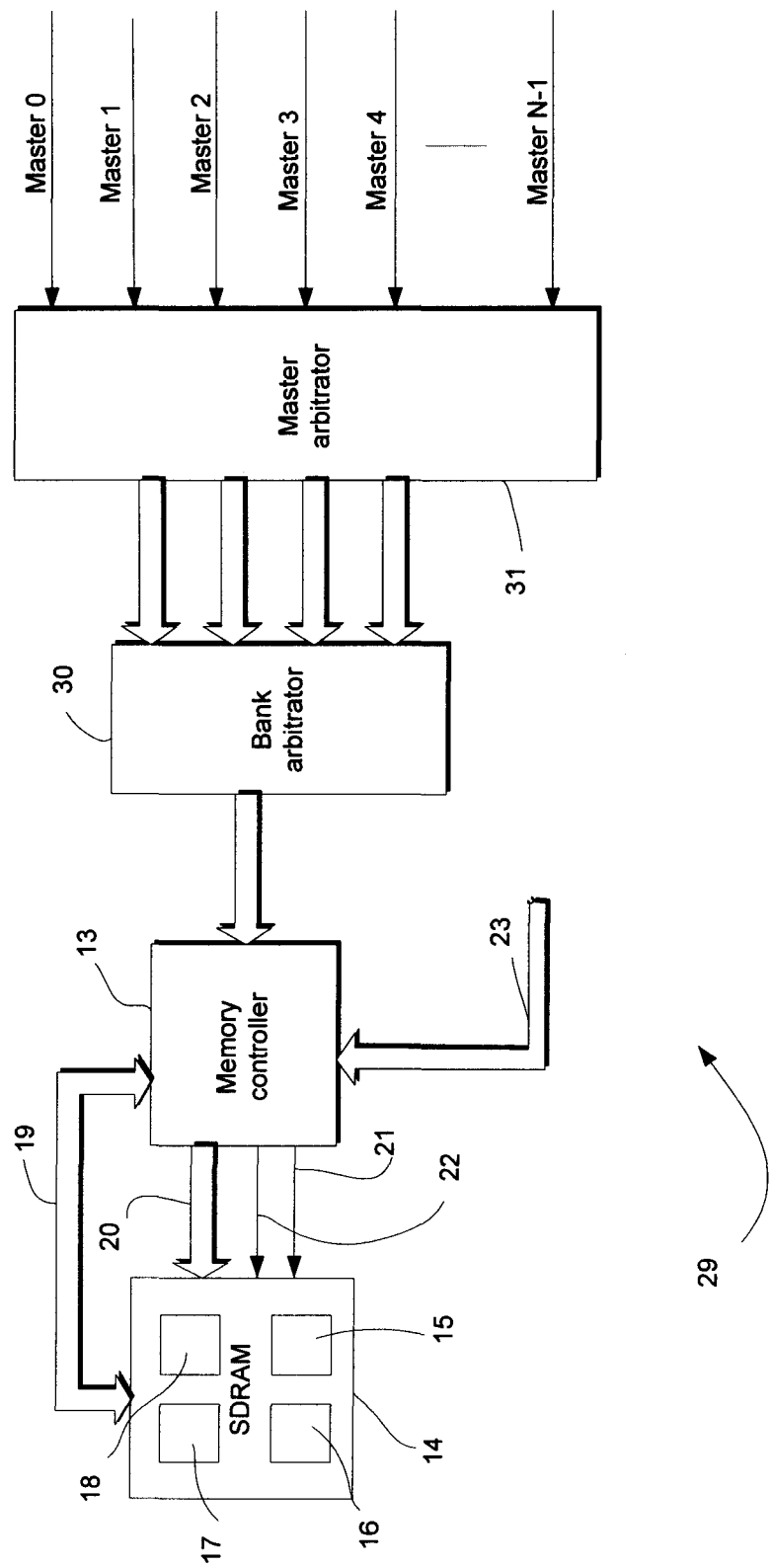


Figure 3

INTERNATIONAL SEARCH REPORT

International application No.

PCT/IB2006/054491

A. CLASSIFICATION OF SUBJECT MATTER

IPC: see extra sheet

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)

IPC: G06F

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

SE,DK,FI,NO classes as above

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

EPO-INTERNAL, WPI DATA, PAJ

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	EP 1313019 A1 (NEC CORPORATION), 21 May 2003 (21.05.2003), paragraphs 0002; 0003; 0005; 0009, figure 1, claim 1, abstract --	1-39
X	EP 1054382 A2 (ATI INTERNATIONAL SRL CHRIST CHURCH), 22 November 2000 (22.11.2000), paragraphs 0010; 0035; 0040; 0052, figure 9, claims 1; 2; 10, abstract -- -----	1-7,9-15, 17-27,29-39

☐ 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 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

9 July 2007

Date of mailing of the international search report

10 -07- 2007

Name and mailing address of the ISA/

Swedish Patent Office

Box 5055, S-102 42 STOCKHOLM

Facsimile No. +46 8 666 02 86

Authorized officer

Patrik Rydman/CM

Telephone No. +46 8 782 25 00

International patent classification (IPC)**G06F 12/00** (2006.01)**G06F 13/18** (2006.01)**Download your patent documents at www.prv.se**

The cited patent documents can be downloaded at www.prv.se by following the links:

- In English/Searches and advisory services/Cited documents (service in English) or
- e-tjänster/anförda dokument (service in Swedish).

Use the application number as username.

The password is **UBQJQCDEMH**.

Paper copies can be ordered at a cost of 50 SEK per copy from PRV InterPat (telephone number 08-782 28 85).

Cited literature, if any, will be enclosed in paper form.

INTERNATIONAL SEARCH REPORT

Information on patent family members

28/05/2007

International application No.

PCT/IB2006/054491

EP	1313019	A1	21/05/2003	US	7013368	B	14/03/2006
				US	20030140201	A	24/07/2003
				JP	2001356961	A	26/12/2001
				WO	0197040	A	20/12/2001

EP	1054382	A2	22/11/2000	US	6504549	B	07/01/2003
----	---------	----	------------	----	---------	---	------------
