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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
28 February 2008 (28.02.2008)

PCT

(10) International Publication Number
WO 2008/024670 A2

(51) International Patent Classification: Not classified

(21) International Application Number:
PCT/US2007/076103

(22) International Filing Date: 16 August 2007 (16.08.2007)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
11/466,915 24 August 2006 (24.08.2006) US

(71) Applicant *(for all designated States except US)*: **MICROCHIP TECHNOLOGY INCORPORATED** [US/US]; 2355 West Chandler Boulevard, Chandler, AZ 85224-6199 (US).

(72) Inventor; and

(75) Inventor/Applicant *(for US only)*: **RAJBHARTI, Nilesh** [US/US]; 12628 W. Ocotillo Road, Glendale, AZ 85307 (US).

(74) Agent: **SLAYDEN, Bruce, W., II**; Baker Botts L.L.P., 1500 San Jacinto Center, 98 San Jacinto Blvd., Austin, TX 78701 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, 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.

(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, MT, 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).

Declarations under Rule 4.17:

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

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

Published:

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

(54) Title: DIRECT MEMORY ACCESS CONTROLLER

(57) Abstract: A direct memory access controller may comprise a data register for transferring data from a source to a destination address, a pattern register, a data comparator coupled with the data register and the pattern register, and a control unit coupled with the comparator operable to stop a data transfer if the comparator detects a match of the data register and the pattern register.

WO 2008/024670 A2

DIRECT MEMORY ACCESS CONTROLLER

TECHNICAL FIELD

The technical field of the present application relates to a direct memory access controller.

BACKGROUND

Direct memory access controller (DMA) are typically used in microprocessor systems, integrated microcontrollers, etc. DMA controllers are used to perform a data transfer from and to memory to and from a peripheral independently from the central processing unit of the computer system. To this end, a DMA controller can be seen as a second programmable processing unit with limited capabilities. Generally, a DMA controller is instructed to transfer a specific amount of data from a source location to a destination location. The source can be within a memory, for example, a data memory of a microcontroller, memory of a peripheral, or data generated by or accessible within a peripheral, such as an analog to digital converter, a port, a capture compare unit, etc. The destination can also be within a memory, thus, allowing high speed transfers within a memory device of a computer system or microcontroller. However, the destination can also be a peripheral, such as a digital to analog converter, a port, etc. To transfer data from a source to a destination the DMA controller must receive the respective source and destination addresses. In addition, each transfer length needs to be specified. To this end, the DMA controller needs to receive either the length of the data transfer or the start and end address of the data to be transferred. There, thus, exists a need for a more flexible DMA controller.

SUMMARY

In one embodiment, a direct memory access controller may comprise a data register for transferring data from a source to a destination address, a pattern register, a data comparator coupled with the data register and the pattern register, and a control unit coupled with the comparator operable to stop a data transfer if the comparator detects a match of the data register and the pattern register.

In another embodiment, a microcontroller may comprise a central processing unit, a bus coupled with the central processing unit, a memory coupled with the bus, a

peripheral unit coupled with the bus, and a direct memory access controller coupled with the bus comprising a data register for transferring data from a source to a destination address, a pattern register, a data comparator coupled with the data register and the pattern register, and a control unit coupled with the comparator operable to stop a data transfer if the comparator detects a match of the data register and the pattern register.

In yet another embodiment, a microcontroller may comprise a central processing unit, a first bus coupled with the central processing unit, a peripheral unit coupled with the first bus, a second bus coupled with the central processing unit, a memory coupled with the second bus, a direct memory access controller coupled with the first and second bus comprising a data register for transferring data from a source to a destination address, a pattern register, a data comparator coupled with the data register and the pattern register, and a control unit coupled with the comparator operable to stop a data transfer if the comparator detects a match of the data register and the pattern register.

In a further embodiment, a method of performing a transmission of a plurality of data from a source to a destination by a direct memory access controller may comprise the steps of a) determining whether a pattern detection is used; b) storing start addresses for a source address and a destination address; c) if a pattern detection is used storing a pattern; otherwise storing a transmission length; d) reading data from the source address; e) storing data in the destination address; f) if a pattern detection is used, comparing the data with the pattern and terminating the transmission upon a match; g) incrementing the source and destination addresses; h) if no pattern detection is used, terminating the transmission if the transmission length has been reached; and i) repeating steps d)-h).

Other technical advantages of the present disclosure will be readily apparent to one skilled in the art from the following figures, descriptions, and claims. Various embodiments of the present application obtain only a subset of the advantages set forth. No one advantage is critical to the embodiments.

BRIEF DESCRIPTION OF THE DRAWINGS

A more complete understanding of the present disclosure and advantages thereof may be acquired by referring to the following description taken in conjunction with the accompanying drawings, in which like reference numbers indicate like features, and wherein:

Figure 1 is a block diagram showing typical DMA controller within a computer system, such as for example, a microcontroller;

Figure 2 is a block diagram showing a details of a first embodiment of a DMA controller;

Figure 3 is a block diagram showing a details of a second embodiment of a DMA controller;

Figure 4 is a block diagram showing a details of a third embodiment of a DMA controller;

Figure 5 is a flowchart of a initialization routine for a DMA controller; and

Figure 6 is a flowchart of a transfer routine of a DMA controller.

DETAILED DESCRIPTION

As stated above, in one embodiment, a direct memory access controller may comprise a data register for transferring data from a source to a destination address, a pattern register, a data comparator coupled with the data register and the pattern register, and a control unit coupled with the comparator operable to stop a data transfer if the comparator detects a match of the data register and the pattern register.

A direct memory access controller may further comprise an address comparator for comparing a source or destination address with a predefined end address. A direct memory access controller may further comprise logic means for generating an end of transmission signal from output signals of the data and address comparators. The pattern register may comprise a plurality of data segments and the data comparator can be controlled to compare a selected segment with the data register. A direct memory access controller may further comprise a plurality of comparators coupled with the data registers. A direct memory access controller may further comprise a multiplexer coupled with the pattern register and the data register.

A direct memory access controller may further comprise a control unit for controlling the multiplexer.

As stated above, in another embodiment, a microcontroller may comprise a central processing unit, a bus coupled with the central processing unit, a memory coupled with the bus, a peripheral unit coupled with the bus, and a direct memory access controller coupled with the bus comprising a data register for transferring data from a source to a destination address, a pattern register, a data comparator coupled with the data register and the pattern register, and a control unit coupled with the comparator operable to stop a data transfer if the comparator detects a match of the data register and the pattern register.

The above mentioned direct memory access controller may further comprise an address comparator for comparing a source or destination address with a predefined end address. The direct memory access controller may further comprise logic means for generating an end of transmission signal from output signals of the data and address comparators. The pattern register may comprise a plurality of data segments and the data comparator can be controlled to compare a selected segment with the data register. The direct memory access controller may further comprise a plurality of comparators coupled with the data registers. The direct memory access controller may further comprise a multiplexer coupled with the pattern register and the data register. The direct memory access controller may further comprise a control unit for controlling the multiplexer.

As stated above, in a further embodiment, a method of performing a transmission of a plurality of data from a source to a destination by a direct memory access controller may comprise the steps of a) determining whether a pattern detection is used; b) storing start addresses for a source address and a destination address; c) if a pattern detection is used storing a pattern; otherwise storing a transmission length; d) reading data from the source address; e) storing data in the destination address; f) if a pattern detection is used, comparing the data with the pattern and terminating the transmission upon a match; g) incrementing the source and destination addresses; h) if no pattern detection is used, terminating the transmission if the transmission length has been reached; and i) repeating steps d)-h).

Such a method may further comprise using step h) even if a pattern detection is used. A plurality of data can be compared with a plurality of data pattern and only if the plurality of data matches the plurality of data pattern the transmission is terminated. The plurality of data pattern can be a sequential data pattern. The sequential data pattern may comprise a sequence of continuous data.

Turning to the drawings, exemplary embodiments of the present application will now be described. Figure 1 depicts a typical embodiment of a DMA controller within a computer system, such as a microcontroller. A computer system 100, such as for example a microcontroller, generally comprises a central processing unit (CPU) 110 which is coupled to one or more bus systems. In Fig. 1 an exemplary main bus 140 is shown. Through this bus 140, the CPU 110 can communicate with a plurality of peripheral devices 180a...180n, such as I/O ports, memories; A/D and D/A converters, timers; pulse width modulators, etc. Furthermore, a dedicated memory bus 160 can be provided to couple the CPU 110 with a main memory 120. A direct memory access controller (DMA) is shown with numeral 130. this DMA controller 130 is also coupled with the main bus 140 to allow for data transfer between the devices coupled with this bus 140. In addition, the DMA unit may also be coupled through an additional bus 170 with main memory 120. Furthermore, DMA controller 130 may receive a plurality of control signals 150 from CPU 110. Such a system allows for data transfer by the DMA controller without CPU 110 between any devices 180a...180n coupled with bus 140 as well as between those devices 180a...180n and memory 120 or within memory 120. CPU 110 is usually only needed to initialize DMA controller 130. Once DMA controller 130 is programmed, the data transfer proper is conducted without the aid of CPU 110. CPU 110 is then free to perform other tasks.

Fig. 2 shows more details of a DMA controller. According to one embodiment, a DMA controller 130 comprises at least a data register or latch 210 and an associated address register or latch 240 both coupled a receiving bus 270 and a transmitting bus 280. Fig. 2 indicates by the dotted line that receiving bus 270 and transmitting bus 280 can be one and the same, such as bus 140 shown in Fig. 1. However, in another embodiment, DMA controller 130 can also be configured to communicate between two different buses. In such an embodiment, a receiving bus

270 can be, for example, bus 140 as shown in Fig. 1 and a transmitting bus 280 can be bus 170 as shown in Fig. 1. Bother registers or latches for address and data 210 and 240 are coupled with these buses. Alternatively separate registers for transmitting and receiving can be provided. Separate address registers 240a and 240b as shown in Fig. 2 are useful as source and destination address usually differ. The data register 210 and the transmitting and/or receiving register 240a, b may further be coupled with a comparator to determine a match in the data or address.

During a DMA data transfer, the DMA controller 130 can be programmed to transfer a specific amount of data beginning at a source start address and ending at a source end address to a destination which also begins at a destination start address and ends at a destination end address. To this end, for example, register 240a is loaded with the source start address and register 240b is loaded with the destination start address. DMA controller 130 further receives either the length of the data block to transferred or an source end address. If the source end address is used, then this address is loaded in a respective register 240c coupled with a comparator 250. Alternatively, the destination end address or as stated above, the length of the data block can be used to determine the end of the transfer. Once such an exemplary initialization has been conducted, the data transfer proper takes place.

In the following a transfer using a single bus 140 is explained. DMA controller 130 puts the first address, the source start address stored in register 240a on bus 140. This address can be within a memory or any device coupled with bus 140. An associated data is then transferred into data register 210. In a following step, register 240a is decoupled from bus 140 and register 240b is coupled with bus 140. Alternatively, if only one address register is used, this address register is loaded with the destination start address. Thus, the destination address is now put on bus 140 addressing the respective destination such as a memory or any device coupled with bus 140. Then, the data stored in register 210 is transferred to this destination address. Source address register 240a and destination address register 240b are then incremented and the process is repeated until the source address register 240a contains the source end address. This end of transmission can, for example, be detected by comparator 250 which compares the contents of address register 240a and register 240c.

In addition, DMA controller 130 allows for a conditional transfer that stops the transfer once a specific pattern such as a predefined data byte has been received. To this end, DMA controller 130 comprises a further pattern register 210b. Upon initialization, this register 210b is loaded with a pattern such as a specific byte. Once comparator 220 detects a match between the loaded data in register 210a and the pattern register 210b a respective signal 230 is generated which indicates to DMA controller 130 that the end of transmission has been reached. To prevent an endless transmission, the address comparator 250 can be used in addition defining a maximum transfer.

Thus, the DMA controller according to such an embodiment generally allows two different types of transaction. A first transaction is defined as a fixed length transmission allowing for a defined block of data to be transferred by he DMA controller. The second transaction has an open length and its length is defined by a specific pattern. For example, the end of transmission can be defined by a specific symbol. Moreover, in another embodiment, a sequence of symbols may define the end of transmission. Thus, a more flexible definition of an end of transmission is possible. For example, instead of a single byte a sequence of two bytes can be used such as a carriage return (CR) line feed (LF) sequence. To this end, a plurality of registers can be provided which are programmed with the respective sequence. The second transaction type can be enhanced with a maximum transfer length to avoid an endless transfer loop and thus provide for additional security.

Fig. 3 shows specific registers of another exemplary embodiment of a DMA controller 130. In this embodiment, a source register 310 and a destination address register 320 are provided. Furthermore, a length/maximum length register 330 is coupled with a comparator 380 which can generate an end address detection EndAddr. Comparator 380 is furthermore coupled with a counter 370. A pattern register 340 is coupled with another comparator 350 which is also coupled with a data register 360 receiving and transmitting the data proper D_{in} and D_{out}. Comparator 350 generates a signal Match which can then produce an end of transmission signal EofTrans. Multiple pattern registers 340 and associated comparators can be provided as indicated by respective elements using dotted lines. Respective logic between the registers and the comparators can be provided to allow for the different types of

transmission. As an example, Fig 3 shows a first controllable driver 385 receiving the signal EndAddr from comparator 380. The output of driver 385 is coupled with a first input of an OR gate 390. A second controllable inverter 355 receives signal Match from comparator 350. The output of driver 355 is coupled with the second input of OR gate 390. Thus, the output of OR gate 90 provides for an end of transmission signal EofTrans.

In a first mode, driver 355 is controlled to decouple comparator 350 from OR gate 390 generating a constant logic low signal at the second input of OR gate 390 and driver 385 is controlled to couple comparator 380 with OR gate 390. Register 310 and 320 are loaded with the respective source start address and destination start address. Register 330 is loaded with the length of the data block to be transmitted and counter 370 is reset to zero. Then the data transfer can be started. To this end, the data addressed by register 310 is loaded into register 360 and written to the address contained in register 320. Then, registers 310 and 320 are incremented by the size of the transmitted data. For example, if data register is a byte wide register, then registers 310 and 320 are incremented by 1. If data register is a 16 bit register, then registers 310 and 320 are incremented by 2, and so on. Counter 370 is incremented accordingly. Comparator 380 is comparing the counter value of counter 370 with register 330. The transfer of data is repeated until the counter value matches the content of register 330. If such a match is reached, the EndAddr signal goes logic high and the output of Or gate 390 will go high indicating an end of transmission EofTrans. As a result, the DMA controller is stopped.

In a second mode, driver 385 is controlled to decouple comparator 380 from OR gate 390 generating a constant logic low signal at the first input of OR gate 390 and driver 355 is controlled to couple comparator 350 with OR gate 390. In this mode, again source and destination registers 310 and 320 are loaded with the respective start addresses. In addition pattern register 340 is loaded with predefined pattern. Again, the DMA transfer is started and the first data associated with the source register is loaded into data register 360 and compared with pattern register 340. If a match occurs, comparator 350 generates a high signal at its output which causes a high signal at the output of OR gate 390 and indicates the end of the transmission.

Otherwise, registers 310 and 3320 are incremented and the transfer continues until a match between the transmitted data and the pattern register 340 occurs.

In a third mode, both drivers 355 and 385 are activated coupling both comparators 350 and 380 with OR gate 390. In this mode, registers 310 and 320 are again loaded with the respective start addresses. Furthermore, a maximum length value is loaded into register 330 and a pattern is loaded into register 340. The data transfer takes place as described above. However, the end of transmission is either generated by comparator 350 or 380. During a normal transmission, comparator 350 will detect the end of transmission by a match between the transmitted data and the pattern register 340. If, however, due to an error or a malfunction this data never occurs, then comparator 380 will terminate the transmission once the maximum length of data transmission has been reached.

Fig. 4 shows another embodiment of a pattern register. In this embodiment a pattern register 410 has a width of 32 bits and, thus, consists of four bytes 410a...410d. Each byte 410a...410d is coupled with a 4:1 multiplexer 420 having 32 inputs and 8 outputs. The output lines of multiplexer 420 are coupled with a comparator 430 which compares the 8 bits with the content of the data register (not shown). A control unit 440 is provided to control multiplexer 420. This embodiment allows for the comparison of a data sequence. In a first mode, pattern register 410 is loaded with a predefined 32 bit pattern. At the beginning of a data transfer, the multiplexer is controlled to couple the first byte 410a of register 410 with comparator 430. Once a match with a transmitted data byte occurs, the comparator signals such a match to control unit 440. Control unit 440 then switches multiplexer 420 to couple the second byte 410b with comparator 430. If the next data byte transferred byte the DMA controller matches the second byte 410b, then control unit switches the multiplexer again to allow for a comparison of the next byte with byte 410c. Only if four sequential bytes match the content of register 410 an end of transmission signal is generated. If a miss occurs before the last byte 410d has been compared, then multiplexer 420 is reset to the first byte 410a to start a new sequence comparison. The unit can be programmed to allow for any length of the pattern sequence. In this embodiment, for example, a one byte sequence, a two byte sequence, a three byte

sequence or a four byte sequence can be used. Depending on the size of the pattern register any sequence length can be implemented.

In a second mode, the sequence must not occur in a continuous order. For example, any number of intermediary bytes can be allowed. Thus, multiplexer 420 would not be reset to the first byte 410a if a mismatch occurs in a following byte. Thus, if pattern register is programmed, for example, with "0x64A8" a sequence of "0x68840AABC8" (underlined bytes show match) would trigger the end of transmission signal. If yet another embodiment, a plurality of comparators and a multiple byte register can be used to even allow for a non sequential match. Thus, an end of transmission signal would be generated if all bytes within the pattern occurred even if they are out of order. Instead of a plurality of comparators, the configuration shown in Fig. 4 can also be used to compare each byte 410a...410d with each transmitted data by sequentially switching through all four data bytes 410a...410d per data cycle. The control unit can set respective bits for each data segment 410a...410d. Once all bits have been set, an end of transmission signal can be generated. Any other combination of possible comparisons can be implemented.

Fig. 5 shows a flow chart of an embodiment of an initialization routine for a DMA controller. In a first step 510 a decision is made whether a pattern mode is used or not. If no, the routine branches to step 520 in which the source and destination address are stored. in a following step 530 the length of the transmission is determined either by an end address or a block length. then the routine ends. If the pattern mode is used, the routine goes to step 540 in which the source and destination are set. Then in step 550, optionally the maximum length of the transmission is set and in step 560 the data termination pattern value is stored. In step 570 the pattern mode is set and the initialization routine ends.

Fig. 6 shows a flow chart of an embodiment of a DMA transfer routine following the initialization routine shown in Fig. 5. The routine starts in step 610 in which the source data under the start address is read. In step 620 this data written to the destination address. In step 630 it is checked whether the pattern matching mode has been set. if not, the source address and destination address are incremented and in step 660 the data length value is decremented. In step 670 it is checked whether the

data length value is 0. If yes, the transfer ends, if not the routine returns to step 610. If the pattern mode has been set, the routine branches from step 630 to step 650 in which it is checked whether the data matches the pattern. If a match occurs, then the transfer ends. Otherwise, the routine continues with step 640 as stated above.

The invention, therefore, is well adapted to carry out the objects and attain the ends and advantages mentioned, as well as others inherent therein. While the invention has been depicted, described, and is defined by reference to particular preferred embodiments of the invention, such references do not imply a limitation on the invention, and no such limitation is to be inferred. The invention is capable of considerable modification, alteration, and equivalents in form and function, as will occur to those ordinarily skilled in the pertinent arts. The depicted and described preferred embodiments of the invention are exemplary only, and are not exhaustive of the scope of the invention. Consequently, the invention is intended to be limited only by the spirit and scope of the appended claims, giving full cognizance to equivalents in all respects.

WHAT IS CLAIMED IS:

1. A direct memory access controller comprising:

a data register for transferring data from a source to a destination address;

a pattern register;

a data comparator coupled with the data register and the pattern register; and

a control unit coupled with the comparator operable to stop a data transfer if the comparator detects a match of the data register and the pattern register.

2. A direct memory access controller according to claim 1, further comprising an address comparator for comparing a source or destination address with a predefined end address.

3. A direct memory access controller according to claim 2, further comprising logic means for generating an end of transmission signal from output signals of the data and address comparators.

4. A direct memory access controller according to claim 1, wherein said pattern register comprises a plurality of data segments and said data comparator is controlled to compare a selected segment with the data register.

5. A direct memory access controller according to claim 1, further comprising a plurality of comparators coupled with the data registers.

6. A direct memory access controller according to claim 4, further comprising a multiplexer coupled with said pattern register and said data register.

7. A direct memory access controller according to claim 6, further comprising a control unit for controlling said multiplexer.

8. A microcontroller comprising:

a central processing unit;

a bus coupled with said central processing unit;

a memory coupled with said bus;

a peripheral unit coupled with said bus; and

a direct memory access controller coupled with said bus comprising:

a data register for transferring data from a source to a destination address;

a pattern register;

a data comparator coupled with the data register and the pattern register; and

a control unit coupled with the comparator operable to stop a data transfer if the comparator detects a match of the data register and the pattern register.

9. A microcontroller according to claim 8, further comprising an address comparator for comparing a source or destination address with a predefined end address.

10. A microcontroller according to claim 9, further comprising logic means for generating an end of transmission signal from output signals of the data and address comparators.

11. A microcontroller according to claim 8, wherein said pattern register comprises a plurality of data segments and said data comparator is controlled to compare a selected segment with the data register.

12. A microcontroller according to claim 8, further comprising a plurality of comparators coupled with the data registers.

13. A microcontroller according to claim 11, further comprising a multiplexer coupled with said pattern register and said data register.

14. A microcontroller according to claim 13, further comprising a control unit for controlling said multiplexer.

15. A microcontroller comprising:

a central processing unit;

a first bus coupled with said central processing unit;

a peripheral unit coupled with said first bus;

a second bus coupled with said central processing unit;

a memory coupled with said second bus; and

a direct memory access controller coupled with said first and second bus comprising:

a data register for transferring data from a source to a destination address;

a pattern register;

a data comparator coupled with the data register and the pattern register; and

a control unit coupled with the comparator operable to stop a data transfer if the comparator detects a match of the data register and the pattern register.

16. A method of performing a transmission of a plurality of data from a source to a destination by a direct memory access controller comprising the steps of:

a) determining whether a pattern detection is used;

b) storing start addresses for a source address and a destination address;

c) if a pattern detection is used storing a pattern; otherwise storing a transmission length;

d) reading data from said source address;

e) storing data in said destination address;

f) if a pattern detection is used, comparing said data with said pattern and terminating said transmission upon a match;

g) incrementing said source and destination addresses;

h) if no pattern detection is used, terminating said transmission if said transmission length has been reached; and

i) repeating steps d)-h).

17. A method according to claim 16, further comprising using step h) even if a pattern detection is used.

18. A method according to claim 16, wherein a plurality of data is compared with a plurality of data pattern and only if the plurality of data matches the plurality of data pattern the transmission is terminated.

19. A method according to claim 18, wherein said plurality of data pattern is a sequential data pattern.

20. A method according to claim 19, wherein said sequential data pattern comprises a sequence of continuous data.

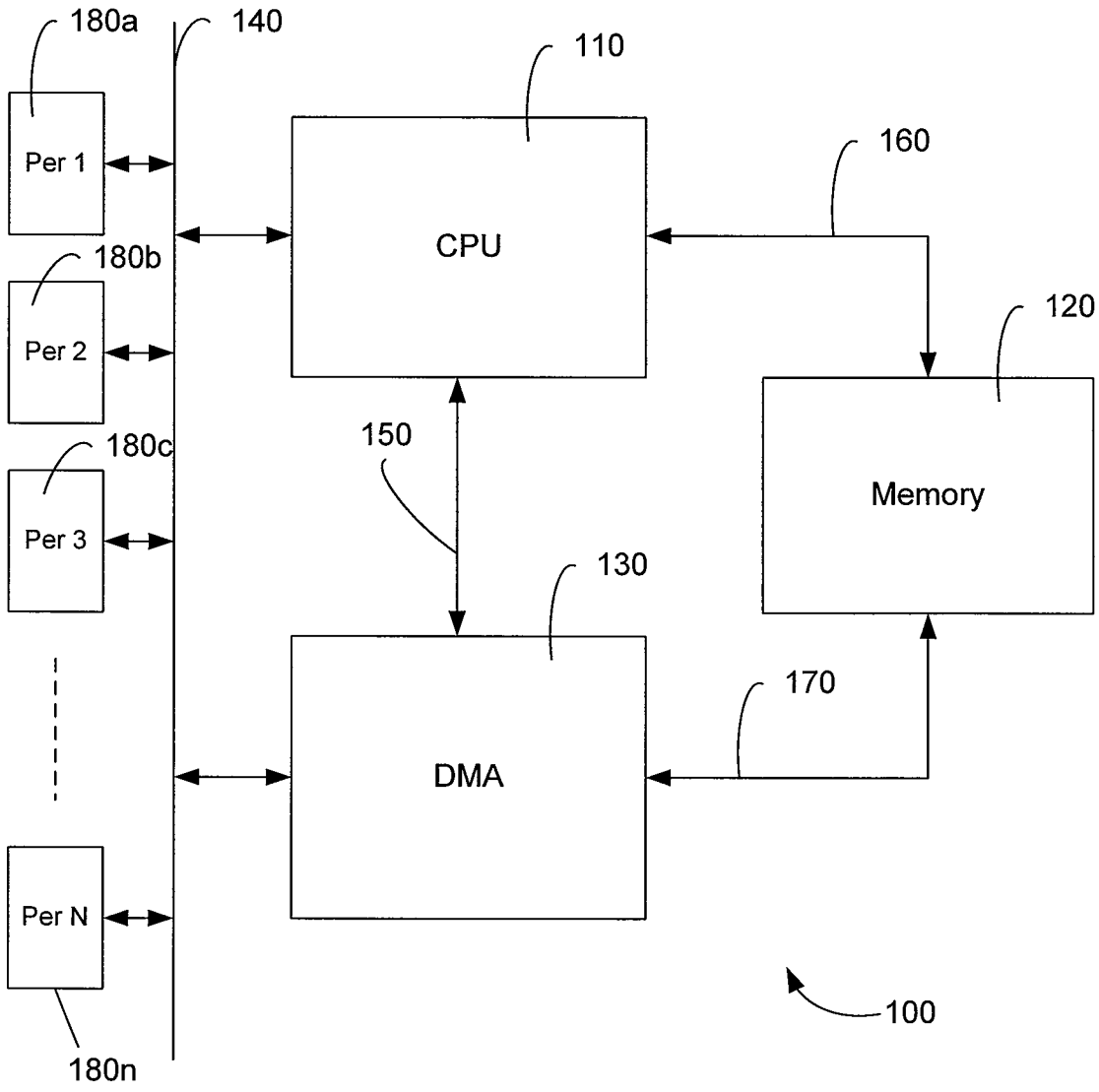

180a
140
110
Per 1
160
CPU
180b
Per 2
120
150
180c
Memory
Per 3
130
170
DMA
Per N
100
180n

Figure 1

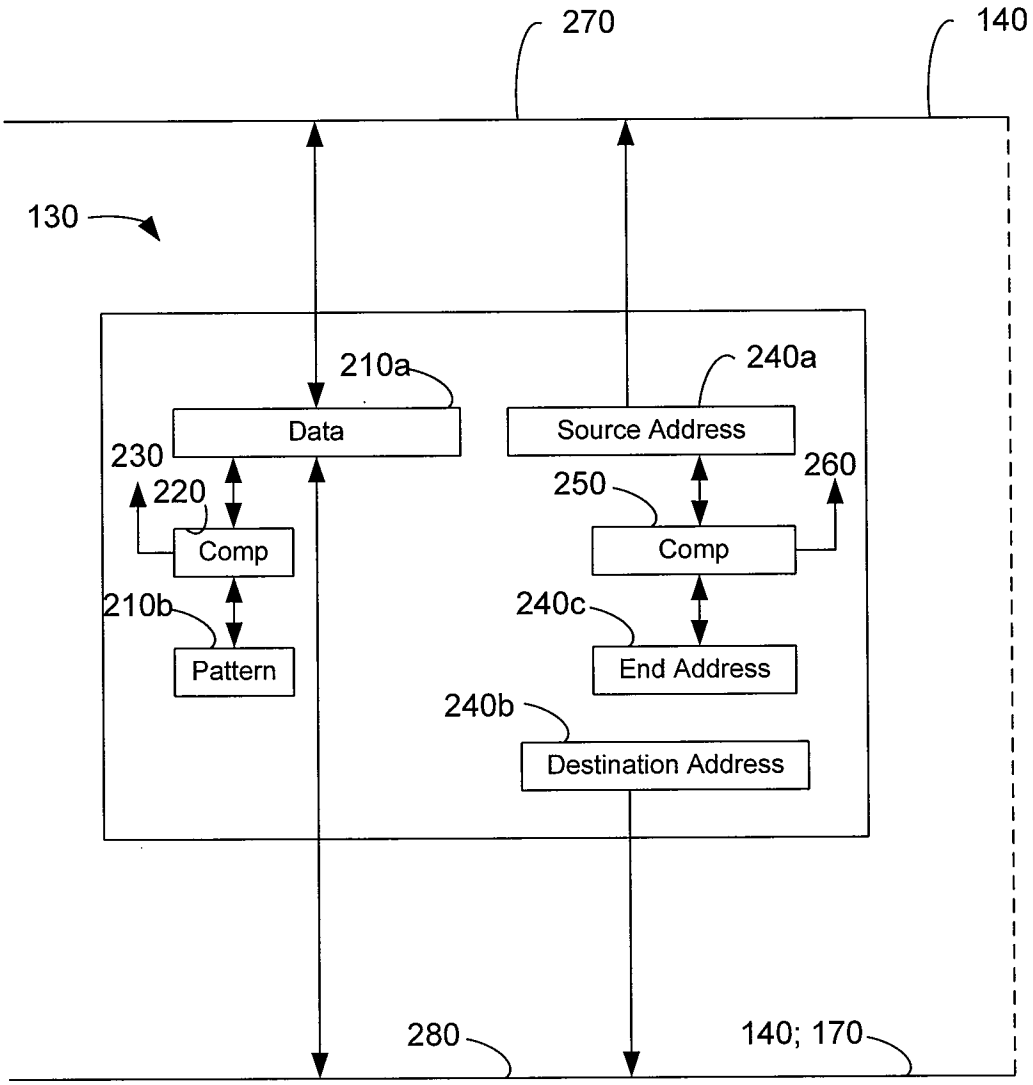

270
140
130
210a
240a
Data
Source Address
230
220
260
250
Comp
Comp
210b
240c
Pattern
End Address
240b
Destination Address
280
140; 170

Figure 2

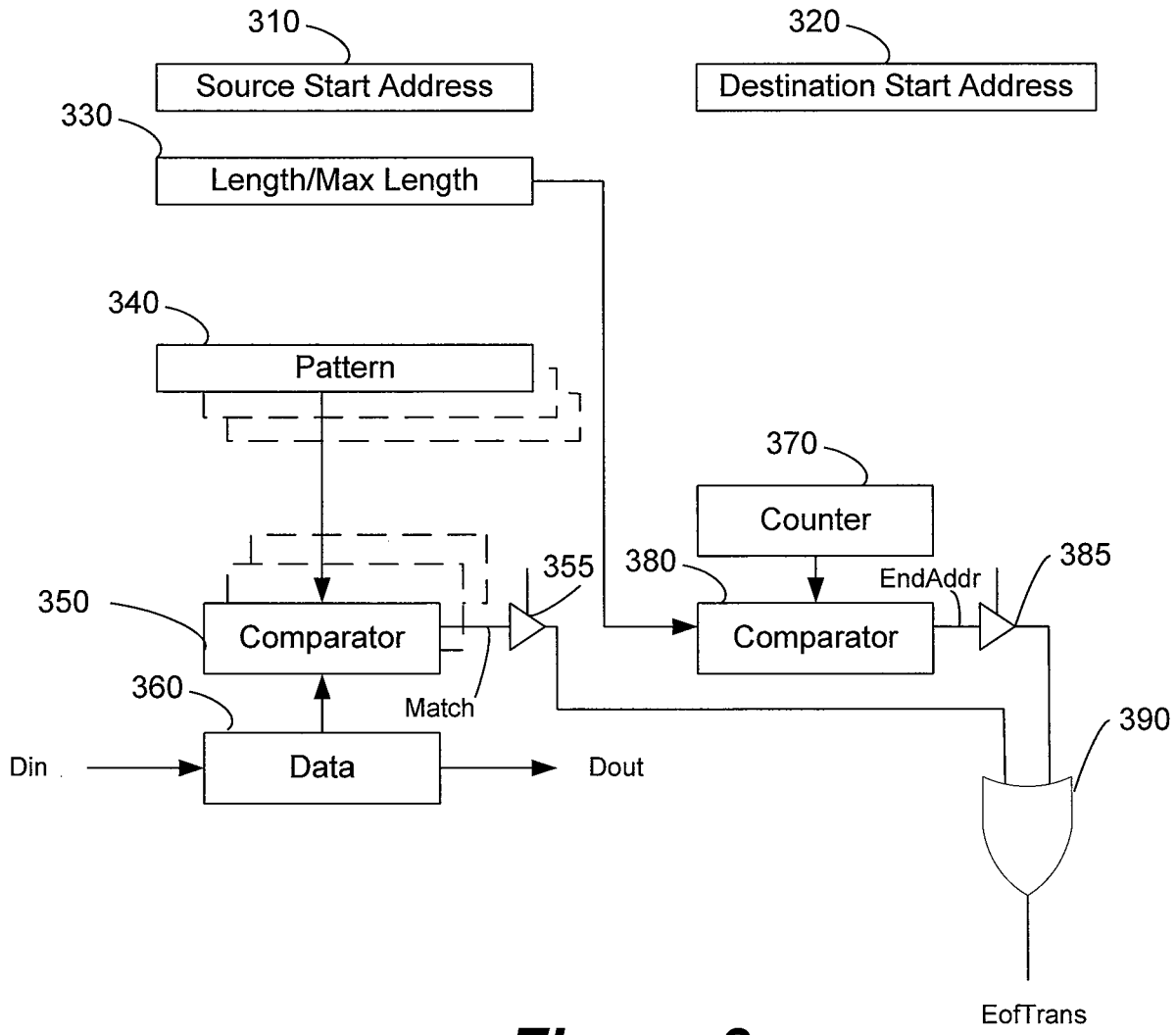

310
Source Start Address
320
Destination Start Address
330
Length/Max Length
340
Pattern
370
Counter
380
355
385
EndAddr
350
Comparator
Comparator
360
Match
390
Din
Data
Dout
EofTrans

Figure 3

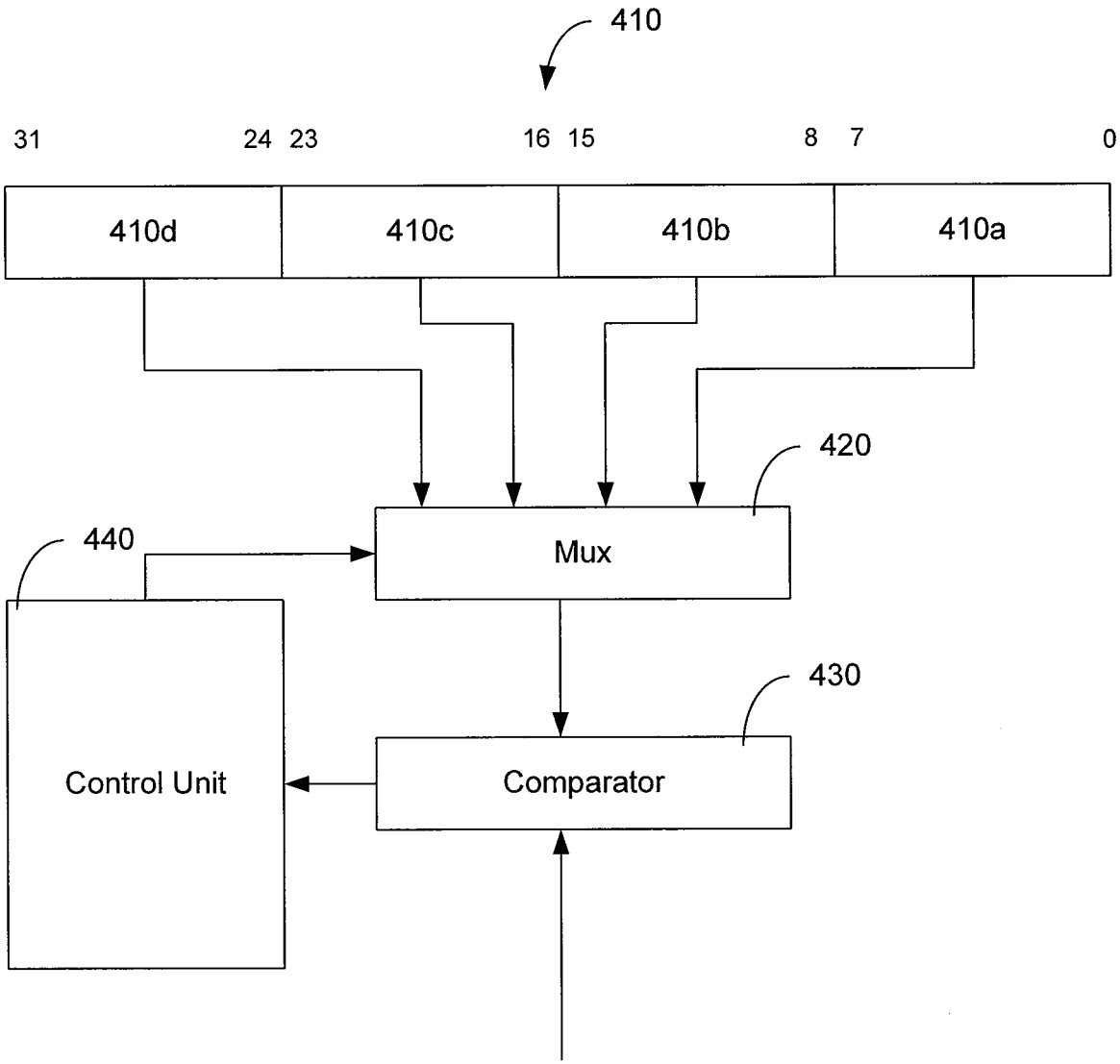

410
31
24 23
16 15
8 7
0
410d
410c
410b
410a
420
440
Mux
430
Control Unit
Comparator

Figure 4

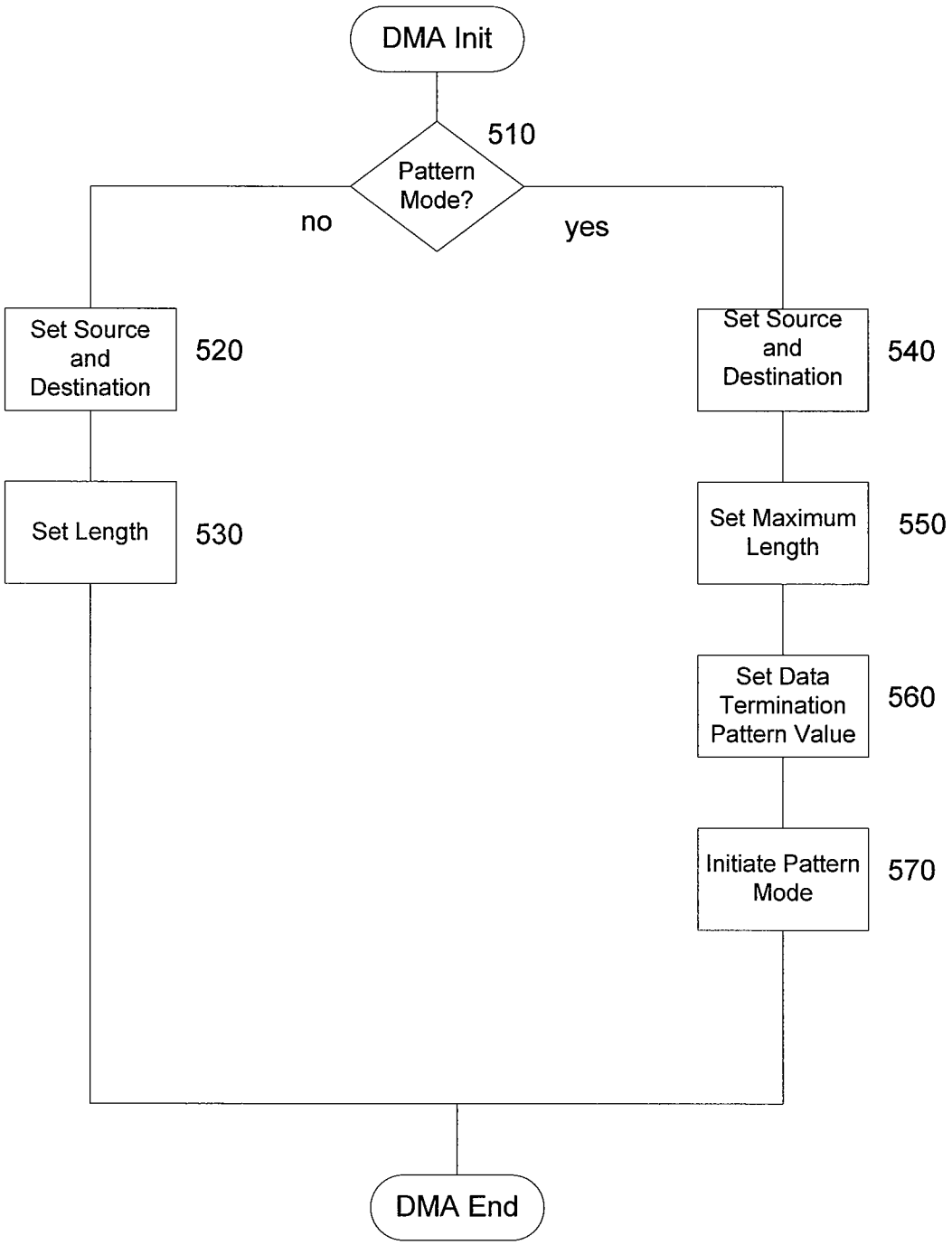

DMA Init
510
Pattern Mode?
no
yes
Set Source and Destination
520
Set Length
530
Set Source and Destination
540
Set Maximum Length
550
Set Data Termination Pattern Value
560
Initiate Pattern Mode
570
DMA End

Figure 5

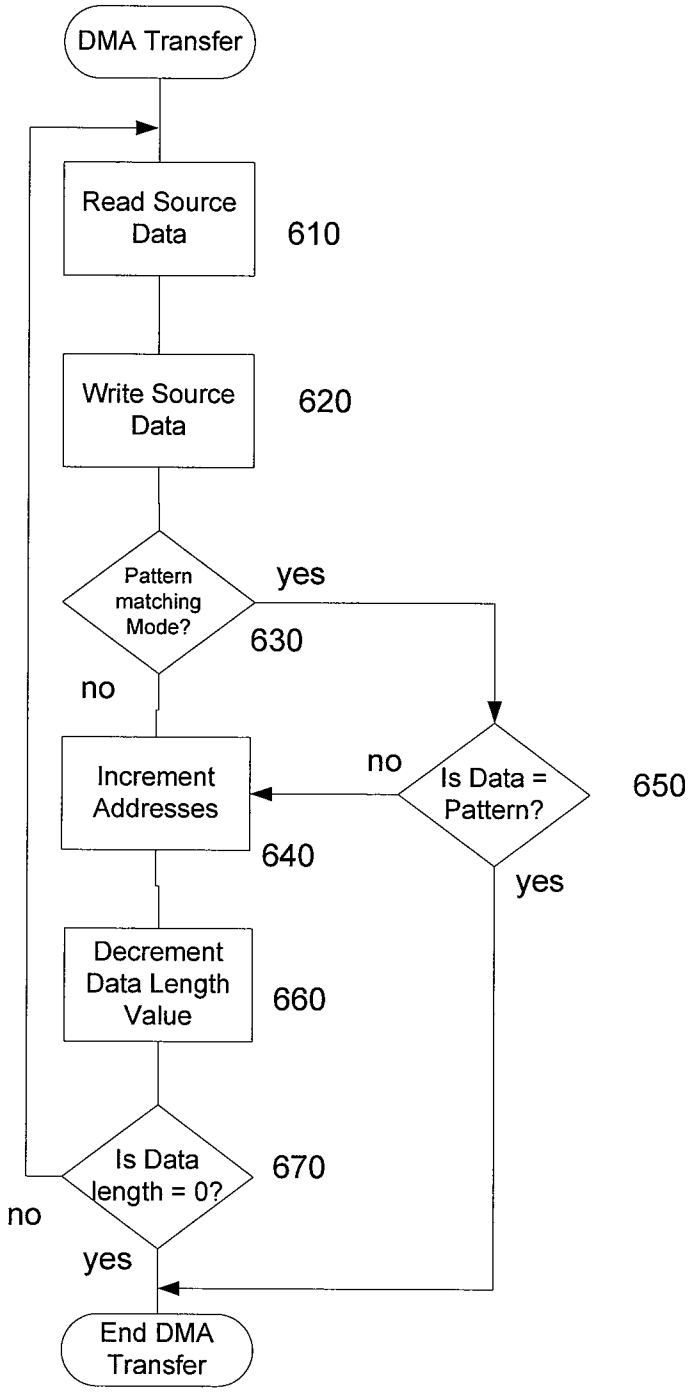

DMA Transfer
Read Source Data
610
Write Source Data
620
Pattern matching Mode?
630
yes
no
Increment Addresses
640
no
Is Data = Pattern?
650
yes
Decrement Data Length Value
660
Is Data length = 0?
670
no
yes
End DMA Transfer

Figure 6