Description

Field of the invention

The present invention relates to a multiple data path central processing unit (CPU) architecture for use in a data processing system. More particularly, the present invention relates to the organization of the various functional units which comprise a CPU and enable a main arithmetic logic unit (ALU) and various special function units to access data registers, literal constants and data from a memory cache.

Background of the invention

The central processing unit (CPU) of a data processing system is the portion of the system where data manipulation, logical and arithmetic operations and other data alteration take place. The physical and logical interconnection of the elements of the CPU, the arithmetic logic unit and various registers and multiplexers in a CPU is known as a data path.

Most prior art CPU's have a single data path architecture; that is, they are organized such that the flow of data through the CPU occurs along a single path.

Some prior art CPU's are designed with multiple data paths. These CPU's have used numerous multiported registers in their implementation. However, such arrangements have required the use of large multiplexers. For instance, in a CPU having thirty registers, each multiplexer would be constructed having a thirty-wide input set so as to be capable of servicing all of the registers.

In addition, prior art CPU's may or may not be arranged in a pipelined configuration, i.e., in two or more stages from a timing standpoint such that the execution of a single microinstruction line proceeds in several steps through the data path. Pipelined architecture enables a faster throughput of data through a CPU by allowing portions of different consecutive instructions to be executed simultaneously. For example, while a later phase or rank of a first instruction is being executed, an earlier phase or rank of a second instruction may also be executed.

The data path implementations of the prior art known to the inventors as described above have proved adequate to the task; however, each suffers from one or more of the following drawbacks. Multiple data path CPU's which use multiported registers and multiplexers are usually characterized by a relatively large parts count due to the use of wide multiplexers. In addition, the use of such multiplexers has a limiting effect on the cycle time of the CPU since it is generally true that the wider the multiplexer the longer its associated selection time must be.

Prior art CPU's which are not organized in a pipelined manner are generally not capable of the same instruction throughput rate which pipelined machines are capable.

Another feature of multiple data path prior art CPU's which utilize wide multiplexers is the total flexibility to access the contents of more than two data registers simultaneously. This flexibility, however, may often prove to provide only an illusory performance advantage since statistically the most used operand is the result from the previous operation.

In Computer, Vol. 14, No. 9, September 1981, A. E. Charlesworth: "An approach to scientific array processing: The architectural design of the AP—120B/FPS-164 family", there is disclosed a CPU architecture which has parallel addition and multiplication pipelines and memories and registers from which operands are selectively supplied to and to which results are returned from the adder and multiplier as appropriate.

Therefore, despite the availability of the prior art data path implementation schemes there is still room for improvement from both performance and performance versus cost standpoints.

In most prior art systems portions of the CPU go unchecked or, alternatively, expensive redundant checking hardware is required.

Furthermore, prior art CPU's generally require more interconnections between the test logic and the various functional units to perform microbranching.

Accordingly, it is an object of the present invention to provide a CPU data path architecture which overcomes one or more drawbacks of prior art data path architectures.

Another object of the present invention is to provide a CPU data path architecture having multiple data paths.

A further object of the present invention is to provide a multiple data path CPU architecture which minimises the multiporting of registers and the use of wide multiplexers.

Another object of the present invention is to provide a multiple data path CPU architecture which allows more than one operation to take place in the CPU per instruction cycle.

These and other objects and advantages of the present invention will become apparent to those of ordinary skill in the art from an examination of the specification, accompanying drawings and appended claims.

According to the present invention, there is provided a CPU architecture for use in a data processing system wherein an instruction causes operands to be processed during execution of a microcode word, including:

arithmetic logic means in a first data path for performing arithmetic logic operations on operands,

at least one selectable function means in a second data path for performing operations on operands simultaneously with the arithmetic logic operations,

a plurality of first register means for holding operands for processing by said arithmetic logic means and said selectable function means,

a plurality of first select logic means, for selectively connecting one of said first register means to said arithmetic logic means and one of said first register means to said selectable function means,

first return bus means for returning results from

an operation of said arithmetic logic means to at least one of said first register means,

second return bus meacs for returning results from an operation of said selectable function means to at least one of said first register means,

at least one second register means connected to at least one of said first register means, for holding operands for processing by said arithmetic logic means and said selectable function means and selectively supplying said operands to said first register means, and

at least one second select logic means connected to at least one of said first register means for selectively connecting one of a plurality of data sources to said first register means.

In the following embodiment of the invention, a multiple data path CPU is disclosed, wherein data sources such as a multiple register file, a memory cache, literal constant register, ALU output and special function output register are organised into two sets of operand buses, one set feeding an ALU and one set feeding a set of special function units whose functions are microcode controlled.

Each operand bus is sourced by a multiplexer which draws from a subset of the possible data sources, including holding registers which store the results of the previous operations of the ALU and special function units.

While prior art multiple data path schemes generally multiplex the data sources in a single stage using one or more large multiplexers, the present invention utilizes two stages in a pipelined arrangement where a first rank accomplishes the decoding of the multiple register file and a miscellaneous register multiplexer and a second rank addresses main multiplexers and controls ALU and special function unit functions. Control of special functions includes the selection of one of N special functions units and the definition of its function.

The special function arrangement is a closely coupled scheme wherein an auxiliary data path and a set of N special function units, each capable of executing a plurality of functions, interfaces with the main ALU data path. Virtually any special function may be implemented, i.e., an auxiliary ALU; access to a set of scratchpad registers; barrel shifter; interface to other processors in a multiprocessor system via an interprocessor bus; interface to main memory; interrupt implementation; read and write access to writable control store; interval timer; I/O channel interface.

The face that there are multiple data paths allows for implementation of checking of the various CPU subsystems in a manner transparent to the system user. During certain operations, unused microcode options may be coded to halt the CPU on a miscompare of two buses.

Brief description of the drawings

Figure 1 is a block diagram of a CPU embodying the present invention.

Figure 2 is a block diagram of an auxiliary ALU implemented as a special function.

Figure 3 is a block diagram of a scratchpad register array implemented as a special function unit.

Figure 4 is a block diagram of a special function parity generation hardware.

Figure 5 is a block diagram of the conditional test hardware capabilities of the special function modules.

Detailed description of the invention

Referring first to Figure 1, the overall data path architecture block diagram is depicted. In this preferred embodiment, there are two unit resources being sourced. ALU 10 and special functions units 12a, 12b, 12c ... 12n are serviced by the available data resources. The output of ALU 10 is made available on main bus 14 and the outputs of the special function units 12 are shared (i.e., only a single one is enabled at a time) by special main bus 16.

The sources for these two data resources are deliverd on two analogous sets of buses, main buses J and K, 18 and 20 respectively, sourced by 4:1 multiplexers 22 and 24 respectively, and special buses SJ and SK, 26 and 28 respectively, sourced by 4:1 multiplexers 30 and 32 respectively. Multiplexer selection is made via microcode fields as is well understood by those of ordinary skill in the art. While in this preferred embodiment 4:1 multiplexers are used, those skilled in the art will readily observe that a system of richer input interconnection could be supplied, but at the expense of greater hardware complexity with the uncertain benefit of enhanced flexibility.

Not all data resources are available to each multiplexer simultaneously. The 4:1 multiplexers 22 and 30, sourcing the J bus 18 and SJ bus 26, are sourced by several sources; J register 34, CD register 36, M register 38 and SM register 40. The 4:1 multiplexers 24 and 32, sourcing the K bus 20 and SK bus 28, are sourced by M register 38, SM register 40, K register 42 and LIT register 44. Note that the data path is enriched by the commonality of data source register M 38 and SM 40 to both the M and SM data paths through either side of ALU 10, via J register 34 and K register 42, and thus across J bus 18 and K bus 20, and also through either side of any of the special function modules 12a through 12n, via SJ 4:1 multiplexer 30 and SK 4:1 multiplexer 32, and through SJ bus 26 and SK bus 28.

At this point it will be instructive to point out the functions of the various registers and buses as used in a preferred embodiment of the present invention, although those of ordinary skill in the art will readily observe that these selections are to a degree arbitrary and that other functions or priorities might well be assigned to the various buses and registers and arrangements thereof without departing from the scope of this disclosure.

The M register 38 holds the results of the previous main ALU result, delivered to its input from the output of ALU 10 via M bus 14. The SM register 40 holds the result of the previous

selected special function result from any one of special function modules 12a through 12n.

The J register 34 and the K register 42 each hold the contents of a selected one of a plurality of registers, principally including a selected one of the multiple register file 46. The register file may be loaded from the M register 38. Those skilled in the art will recognize that the J or the K registers may take data from other registers, such as a program counter or memory address registers through other multiplexers as is known.

The CD register 36 holds the output contents of an addressable cache 48 which is loaded from SM register 40 via line 49. The last principal register in this rank of registers is the LIT register 44 which, as its name implies, holds a literal constant supplied by a microcode bit field 50.

The multiple register file 46 is controlled by several microcode fields which specify; the register, if any, to drive to J register 34, and K rgister 42; the register in the file to be loaded from the M register 38. A pointer register (not shown) acts in conjunction with microcode to select registers to be loaded and read. This register may be loaded from M bus 14 or incremented or decremented by use of microcode fields. All fields used to load these file registers are in rank two, all fields used to read them are in rank one.

In a preferred embodiment of the present invention, the rank one microcode bit field of a given microinstruction decodes the microcode addressing functions for the multiple register file 46, the multiplexer 52 which places the results of one of a number of miscellaneous registers into the input of J register 34, and multiplexer 54 which performs a similar function to the input of the K register 42. Multiplexers 52 and 54 form ports to less frequently accessed registers, such as a program counter, memory address registers, or status registers. Those skilled in the art will recognise that this feature allows access to many registers while leaving the main register paths unencumbered.

The rank two portion microcode bit field of any given microinstruction decodes the 4:1 multiplexers 22, 24, 30 and 32 and controls and defines the operation of ALU 10 and special function units 12a through 12n. With respect to the special function units, a portion of the rank two microcode bit field controls both the selection of one of n special function units 12a through 12n, and also defines its function. Those skilled in the art will recognize that the particular portion of the available microcode field used is unimportant and that any bit positions in the appropriate rank of microcode field may be used. All registers are clocked simultaneously except that CD register 36 gets clocked only when microcode specifies a load from the cache. A microcode field provides an inhibit signal (not shown) to CD register 36 to perform. The loading of certain registers (M, SM, CD and multiple register file) may be optionally inhibited during microbranching.

It will be apparent to those of ordinary skill in the art that the arrangement of the data paths described in the foregoing description of a preferred embodiment of the present invention provides a degree of flexibility from a cost and performance standpoint which affords a successful compromise between hardware complexity and versatility.

Each special function unit 12a through 12n may be dedicated to a related group of tasks. For example, one may be an auxiliary ALU capable of performing logical or arithmetic functions and returning the result to the SM bus.

Referring now to Figure 2, an embodiment of a special function unit as an ALU. ALU unit 100 having input 102 connected to SJ bus 26 and input 104 connected to SK bus 28 also has its output 106 connected to buffer 108. Buffer 108 returns the ALU result to the SM bus 16. Since more than one special function unit is attached to the SM bus, buffer 108 has means to prevent bus contention by assuring that this special function unit has control of SM bus 16 only when it is selected. Thus, buffer 108 has an OE (output enable) input which is driven by decode unit 110. A microcode field 112 in rank two drives decode unit 110 and drives similar decode units for all special function units attached to SM bus 16. Each decode unit is conditioned to recognize only one of all possible combinations of microcode appearing in microcode field 112. Thus, only one decode unit will enable a buffer to attach a special function unit to the SM bus 16.

A second microcode field 114 in rank two is shown connected to ALU unit 100. This microcode field defines the operation to be performed by ALU unit 100.

In operation, any special unit, such as ALU unit 100, is always attached to the SJ and SK buses, thus always prepared to perform its function. Only one function has control of the SM bus at any one time. This selection is determined by microcode field 112 and decoded by decoder 110 thus enabling buffer 108 to drive the SM bus 16.

ALU 10 may have an output ALU=0 (shown at numeral 116) which is useful for performing one of several tests. As previously mentioned, unused microcode options in previous instructions may be used to create operations through the CPU shown generally in Figure 1 which result in quantities being placed on the SJ and SK buses. If these microcode test routines are designed such that the results appearing simultaneously on the SJ and SK buses are supposed to be equal, output 116 of ALU 100 may be used as the final step in the test to determine whether the hardware tested by the microcode is correctly functioning. Any result other than ALU=0 at this point in the program can be used to halt the processor because of a miscompare on SJ and SK buses 26 and 28, respectively. This provides a method for verifying the correct operation of various parts of tqe CPU which are not protected by other means (such as parity) without large amounts of redundant circuitry. The checking proceeds concurrently with normal system operation without impacting performance. For example, the simple

ADD macroinstruction does not require the use of the special function modules 12a through 12n.

The operands to be added are presented to ALU 10 on J bus 18 and K bus 20. In addition, due to the flexibility of the data paths, the same operands may be simultaneously presented to special function ALU 100. The following clock, the results of the main ALU and special ALU are available in M register 38 and SM register 40. These results may then be passed through multiplexers 30 and 32 and presented to inputs 102 and 104 of the special function ALU. The microcode field 114 performs the check function which instructs special ALU 100 to perform a subtract and to halt the CPU if the result is nonzero as indicated by line 116. Thus, the correct operation of both ALU's as well as many of the control paths has been verified.

Another example of a special function unit which may be implemented in the CPU of the present invention is that of a multiple address scratchpad. Referring to Figure 3, scratchpad 120 which may be configured of RAM or other suitable memory or register devices, is shown connected to the SJ, SK and SM buses, 26, 28, and 16, respectively. The SJ bus is used as a data input port for scratchpad 120, the SK bus is used to hold a portion of the address used to access scratchpad 120. Another portion of the address used to address scratchpad 120 is derived from microcode field 114.

As in the ALU example, a decode unit 122 examines microcode field 112 to determine whether a particular bit pattern has selected scratchpad 120 for enabling on to SM bus 16. However, unlike decode unit 110 associated with ALU 100, decode unit 122 has two outputs, One of these outputs, line 124, provides a single bit to the output enable of scratchpad 120. This output enable input is used as is known in the art to disable the output drivers of scratchpad 120 so that it may selectively be connected to SM bus 16. The second output, line 126, is used to drive read/write input of scratchpad 120 and defines whether scratchpad 120 will be performing a read or a write operation.

In a more general sense, it can be seen from the above examples, that microcode field 112 is generally usable to decode the selection of a particular special function unit and place its output on SM bus 16. Microcode field 114 is useful for performing special addressing functions or function definitions for whatever special function unit is embodied.

While parity is normally sent along with the data it serves, the special function units 12a through 12n send parity a clock cycle later. Otherwise, in cases where a special function unit 12a through 12n would generate parity in series with output data from a special function unit, the generation of parity would slow down the cycle time of the machine.

To prevent cycle time slow downs, special function units 12a through 12n send parity information a clock cycle later than data information. The parity generation time therefore does not affect the minimum cycle time of the CPU.

Referring now to Figure 4, the parity generating mechanism for the special function units is described.

Figure 4 illustrates special function units 12a through 12n with their inputs connected to SJ bus 26, SK bus 28, and their outputs connected to SM bus 16. In addition to these buses, Figure 4 illustrates SJ parity bus 26a and SK parity bus 28a driving the inputs of both special function modules as well as SM parity bus 16a being selectively driven by special function units 12a through 12n. Special function unit 12a, as illustrative of all special function units, is seen to contain parity pass or generate unit 202 as well as the special function operation 200, which may be, for instance, ALU 100 of Figure 2, or scratchpad 120 of Figure 3. Depending on the nature of the particular special function unit involved, parity for this unit may either be passed through or generated. For instance, if the special function unit is a scratchpad, parity is merely passed along with the data to be stored in the scratchpad. If special function operation 200 is an ALU, parity will be generated from the ALU result.

In other cases, such as where a special function unit involves a counting operation, if the input states and the input parity is known, an algorithm may be used to predict what the parity will be on the function result. As is known in the art, there are many hardware schemes which may be used to predict parity based on the input operands and the operation being performed. Parity pass or generate unit 202 can also employ these forms of parity prediction.

The resulting parity bit generated by parity pass or generate unit 202 is placed in SM parity bus delay register unit 204 on the first effective clock following its generation.

It will be noted that microcode field 112, which, as previously discussed is used to enable a particular special function unit 12a through 12n is shown driving the decode unit 206. Decode unit 206 is similar in function to decode unit 110 and 122 from Figures 2 and 3, however it is shown here possessing another function relating to parity. A portion of the output of decode unit 206 is shown performing the SM bus output function via buffer 208 which is shown schematically as one having a disable function as is known in the art. The output of decode unit 206 also supplies the data input to SM parity drive delay register 210. The output of SM parity drive delay register 210 drives the enable/disable line of output buffer 212 which delivers the SM parity bus delay register result to the SM parity bus 16a.

Those skilled in the art will realize that the result of the special function operation shown in box 200 is passed through buffer 208 to SM bus 16 one clock cycle prior to the parity bit being available on SM parity bus 16a because of the clock delay through register 204 and 210. However, to compensate for this, parity bus 16a drives parity check 216. The output of SM register 214 is

delivered to parity checker 216 as is the unregistered parity bit from SM paritybus 16a. Since the parity bit on SM bus parity bus 16a has been delayed by one clock cycle, the addition of SM register 214 to the parity data bus delays the data output on the SM bus 16 by that one clock cycle so that parity checking unit 216 receives both data and the parity bit at the same time.

While this disclosure has generally illustrated the concept of parity as a single bit per word, those skilled in the art will understand that other schemes may also be used. For instance, a number of parity bits may be used to cover various subfields of the entire word. Two parity bits, each covering one half (lower or upper) of a word might be used.

Referring now to Figure 5, a block diagram of the conditional test hardware capabilities of the special function modules, the hardware architecture allowing the CPU sequencer to branch on numerous test conditions will be disclosed.

Located within each of special function module units 12a through 12n, as previously described, are various special function units which are capable of performing various tests on the parameters with which they operate. For instance, if the special function unit is an ALU, the various tests which can be performed may include the setting of a carry bit, ALU=0, positive or negative ALU results.

Each special function unit 12 is equipped with two multiplexers 130 and 132, respectively which select from among the numerous test conditions available for that module via a portion of microcode field 114, discussed previously with respect to Figures 2, 3 and 4. Those skilled in the art will realize that the selection of two specific tests is somewhat arbitrary and that a greater or smaller number of tests could be selected from, thus altering the number of multiplexers like multiplexers 130 and 132. The outputs of multiplexers 130 and 132 are driven on two buses, STESTA bus 134 and STESTB bus 136, respectively, via buffers 138 and 140, respectively. In a manner similar to that depicted in Figures 2 through 4, buffers 138 and 140 may have their outputs disabled when special function unit 12 with which they are associated is not driving either the SM bus or STESTA line 134 or STESTB line 136. The disabling inputs of buffers 138 and 140 are driven by decode circuit 110 which decodes a microcode bit field 112 as previously discussed with respect to these special function units.

Thus, when a special function unit has been selected, and a test or set of tests defined by a portion of the microcode field 114, the results of these tests through multiplexers 130 and 132 and buffers 138 and 140 are placed on STESTA line 134 and STESTB line 136.

STESTA line 134 and STESTB line 136 terminate in sequencer conditional branch hardware unit 139. The function of sequencer conditional branch hardware unit 139 is to use the tests results supplied to it by the STESTA and STESTB buses to perform microbranching and to selectively inhibit lines of microcode when certain microbranches have been taken.

The widths of the particular buses shown herein may be determined with reference to a particular application. While the inventors have employed a width of 16 bits plus two parity bits, the actual bus widths which may be used with the current invention are simply a matter of design choice.

The number and uses of the various registers is exemplary only and is not meant to indicate a limitation. Those skilled in the art will understand the invention could function with use of TTL, ECL, CMOS, MOS, NMOS or other technologies if the inherent technology dependent design rules are observed.

The registers, multiplexers, ALU's and special function logic may be implemented as discrete logic elements or as part of gate arrays or custom LSI.

Claim

A CPU architecture for use in a data processing system wherein an instruction causes operands to be processed during execution of a microcode word, including:

arithmetic logic means (10) in a first data path (18, 20) for performing arithmetic logic operations on operands,

at least one selectable function means (12a—12n) in a second data path (26, 28) for performing operations on operands simultaneously with the arithmetic logic operations,

a plurality of first register means (34, 36, 38, 40, 42, 44) for holding operands for processing by said arithmetic logic means (10) and said selectable function means (12a—12n),

a plurality of first select logic means (22, 24, 30, 32), for selectively connecting one of said first register means (34, 36, 38, 40, 42, 44) to said arithmetic logic means (10) and one of said first register means (34, 36, 38, 40, 42, 44) to said selectable function means (12a—12n),

first return bus means (14) for returning results from an operation of said arithmetic logic means (10) to at least one of said first register means (34, 36, 38, 40, 42, 44),

second return bus means (16) for returning results from an operation of said selectable function means (12a—12n) to at least one of said first register means (34, 36, 38, 40, 42, 44),

at least one second register means (46) connected to at least one of said first register means (34, 36, 38, 40, 42, 44), for holding operands for processing by said arithmetic logic means (10) and said selectable function means (12a—12n) and selectively supplying said operands to said first register means (34, 36, 38, 40, 42, 44), and

at least one second select logic means (52, 54) connected to at least one of said first register means for selectively connecting one of a plurality of data sources to said first register means.

delivered to parity checker 216 as is the unregistered parity bit from SM paritybus 16a. Since the parity bit on SM bus parity bus 16a has been delayed by one clock cycle, the addition of SM register 214 to the parity data bus delays the data output on the SM bus 16 by that one clock cycle so that parity checking unit 216 receives both data and the parity bit at the same time.

While this disclosure has generally illustrated the concept of parity as a single bit per word, those skilled in the art will understand that other schemes may also be used. For instance, a number of parity bits may be used to cover various subfields of the entire word. Two parity bits, each covering one half (lower or upper) of a word might be used.

Referring now to Figure 5, a block diagram of the conditional test hardware capabilities of the special function modules, the hardware architecture allowing the CPU sequencer to branch on numerous test conditions will be disclosed.

Located within each of special function module units 12a through 12n, as previously described, are various special function units which are capable of performing various tests on the parameters with which they operate. For instance, if the special function unit is an ALU, the various tests which can be performed may include the setting of a carry bit, ALU=0, positive or negative ALU results.

Each special function unit 12 is equipped with two multiplexers 130 and 132, respectively which select from among the numerous test conditions available for that module via a portion of microcode field 114, discussed previously with respect to Figures 2, 3 and 4. Those skilled in the art will realize that the selection of two specific tests is somewhat arbitrary and that a greater or smaller number of tests could be selected from, thus altering the number of multiplexers like multiplexers 130 and 132. The outputs of multiplexers 130 and 132 are driven on two buses, STESTA bus 134 and STESTB bus 136, respectively, via buffers 138 and 140, respectively. In a manner similar to that depicted in Figures 2 through 4, buffers 138 and 140 may have their outputs disabled when special function unit 12 with which they are associated is not driving either the SM bus or STESTA line 134 or STESTB line 136. The disabling inputs of buffers 138 and 140 are driven by decode circuit 110 which decodes a microcode bit field 112 as previously discussed with respect to these special function units.

Thus, when a special function unit has been selected, and a test or set of tests defined by a portion of the microcode field 114, the results of these tests through multiplexers 130 and 132 and buffers 138 and 140 are placed on STESTA line 134 and STESTB line 136.

STESTA line 134 and STESTB line 136 terminate in sequencer conditional branch hardware unit 139. The function of sequencer conditional branch hardware unit 139 is to use the tests results supplied to it by the STESTA and STESTB buses to perform microbranching and to selectively inhibit lines of microcode when certain microbranches have been taken.

The widths of the particular buses shown herein may be determined with reference to a particular application. While the inventors have employed a width of 16 bits plus two parity bits, the actual bus widths which may be used with the current invention are simply a matter of design choice.

The number and uses of the various registers is exemplary only and is not meant to indicate a limitation. Those skilled in the art will understand the invention could function with use of TTL, ECL, CMOS, MOS, NMOS or other technologies if the inherent technology dependent design rules are observed.

The registers, multiplexers, ALU's and special function logic may be implemented as discrete logic elements or as part of gate arrays or custom LSI.

Claim

A CPU architecture for use in a data processing system wherein an instruction causes operands to be processed during execution of a microcode word, including:

arithmetic logic means (10) in a first data path (18, 20) for performing arithmetic logic operations on operands,

at least one selectable function means (12a—12n) in a second data path (26, 28) for performing operations on operands simultaneously with the arithmetic logic operations,

a plurality of first register means (34, 36, 38, 40, 42, 44) for holding operands for processing by said arithmetic logic means (10) and said selectable function means (12a—12n),

a plurality of first select logic means (22, 24, 30, 32), for selectively connecting one of said first register means (34, 36, 38, 40, 42, 44) to said arithmetic logic means (10) and one of said first register means (34, 36, 38, 40, 42, 44) to said selectable function means (12a—12n),

first return bus means (14) for returning results from an operation of said arithmetic logic means (10) to at least one of said first register means (34, 36, 38, 40, 42, 44),

second return bus means (16) for returning results from an operation of said selectable function means (12a—12n) to at least one of said first register means (34, 36, 38, 40, 42, 44),

at least one second register means (46) connected to at least one of said first register means (34, 36, 38, 40, 42, 44), for holding operands for processing by said arithmetic logic means (10) and said selectable function means (12a—12n) and selectively supplying said operands to said first register means (34, 36, 38, 40, 42, 44), and

at least one second select logic means (52, 54) connected to at least one of said first register means for selectively connecting one of a plurality of data sources to said first register means.

* C L M S *

Patentanspruch

CPU-Architektur zur Benutzung in einem Datenverarbeitungssystem, in dem eine Anweisung bewirkt, daß Rechengrößen, während der Ausführung eines Mikrocode-Wortes verarbeitet werden, beinhaltend

—ein Arithmetik-Logikmittel (10) in einem ersten Datenweg (18, 20) zur Ausführung arithmetisch-logischer Operationen an Rechengrößen,

—mindestens ein selektives Funktionsmittel (12a bis 12n) in einem zweiten Datenweg (26, 28) zur gleichzeitig mit den arithmetisch-logischen Operationen erfolgenden Ausführung von Operationen an Rechengrößen,

—mehrere erste Speichermittel (34, 36, 38, 40, 42, 44) zur Speicherung von Rechengrößen für die Verarbeitung durch die Arithmetik-Logikmittel (10) und durch die selektiven Funktionsmittel (12a bis 12n),

—mehrere erste Auswahl-Logikmittel (22, 24, 30, 32) zur selektiven Verbindung von einem der ersten Speichermittel (34, 36, 38, 40, 42, 44) mit dem Arithmetik-Logikmittel (10) und von einem der ersten Speichermittel (34, 36, 38, 40, 42, 44) mit den selektiven Funktionsmitteln (12a bis 12n),

—eine erste Rückübertragungsleitung (14) zur Rückführung von Ergebnissen einer Operation des Arithmetik-Logikmittels (10) zu mindestens einem der Speichermittel (34, 36, 38, 40, 42, 44),

—eine zweite Rückübertragungsleitung (16) zur Rückführung von Ergebnissen einer Operation der selektiven Funktionsmittel (12a bis 12n) zu mindestens einem der ersten Speichermittel (34, 36, 38, 40, 42, 44),

—mindestens einem zweiten Speichermittel (46), das mit mindestens einem der ersten Speichermittel (34, 36, 38, 40, 42, 44) verbunden ist, um Rechengrößen für die Verarbeitung durch das arithmetik-Logikmittel (10) und durch die selektiven Funktionsmittel (12a bis 12n) zu speichern und um die Rechengrößen den ersten Speichermitteln (34, 36, 38, 40, 42, 44) zuzuführen, und

—mindestens einem zweiten Auswahl-Logikmittel (52, 54), das mit mindestens einem der ersten Speichermittel (34, 36, 38, 40, 42, 44) verbunden ist, um eine von mehreren Datenquellen mit den ersten Speichermitteln (34, 36, 38, 40, 42, 44) selektiv zu verbinden.

Revendication

Architecture d'une unité centrale de traitement CPU pour emploi dans un système de traitement des données, dans laquelle une instruction fait en sorte que des opérandes sont traités au cours de l'exécution d'un mot de microcode, comportant:

des moyens arithmétiques et logiques (10) sur un premier chemin de données (18, 20) pour effectuer des opérations arithmétiques et logiques sur les opérandes,

au moins un moyen de fonction sélectionnable (12a—12n), sur un second chemin de données (26, 28), pour effectuer des opérations sur les opérandes simultanément avec les opérations arithmétiques et logiques,

une pluralité de moyens formant premier registre (34, 36, 38, 40, 42, 44) pour contenir les opérandes à traiter par lesdits moyens arithmétiques et logiques (10) et par lesdits moyens de fonction sélectionnables (12a—12n),

une pluralité de premiers moyens logiques sélectifs (22, 24, 30, 32) pour relier sélectivement l'un desdits moyens formant premier registre (34, 36, 38, 40, 42, 44) auxdits moyens arithmétiques et logiques (10) et l'un desdits moyens formant premier registre (34, 36, 38, 40, 42, 44) auxdits moyens de fonction sélectionnables (12a—12n),

des moyens (14) formant premier bus de renvoi pour renvoyer les résultats d'une opération desdits moyens arithmétiques et logiques (10) à au moins l'un desdits moyens formant premier registre (34, 36, 38, 40, 42, 44),

des moyens (16) formant second bus de renvoi pour renvoyer les résultats d'une opération desdits moyens de fonction sélectionnables (12a—12n) à au moins l'un desdits moyens formant premier registre (34, 36, 38, 40, 42, 44),

au moins un moyen formant second registre (46) relié à au moins l'un desdits moyens formant premier registre (34, 36, 38, 40, 42, 44) pour contenir les opérandes à traiter par lesdits moyens arithmétiques et logiques (10) et par lesdits moyens de fonction sélectionnables (12a—12n) et pour renvoyer sélectivement lesdits opérandes auxdits moyens formant premier registre (34, 36, 38, 40, 42, 44), et

au moins un second moyen logique sélectif (52, 54) relié à au moins l'un desdits moyens formant premier registre pour relier sélectivement l'une d'une pluralité de sources de données auxdits moyens formant premier registre.