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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
12 December 2002 (12.12.2002)

PCT

(10) International Publication Number
WO 02/099634 A1

(51) International Patent Classification[7]: G06F 9/38, 9/34, 9/345, 9/30, 12/10

(21) International Application Number: PCT/US02/16970

(22) International Filing Date: 30 May 2002 (30.05.2002)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/870,448 1 June 2001 (01.06.2001) US

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

(72) Inventors: **CATHERWOOD, Michael**; 13 Park Street, Pepperell, MA 01463 (US). **TRIECE, Joseph, W.**; 15610 S. 7th Place, Phoenix, AZ 85048 (US). **PYSKA, Michael**; 14601 South 14th Way, Phoenix, AZ 85048 (US). **CONNER, Joshua, M.**; 2470 West Jasper Avenue, Apache Junction, AZ 85220 (US).

(74) Agents: **BERTIN, Robert, C.** et al.; Swidler Berlin Shereff Friedman, LLP, 300 K Street, N.W., Washington, DC 20007 (US).

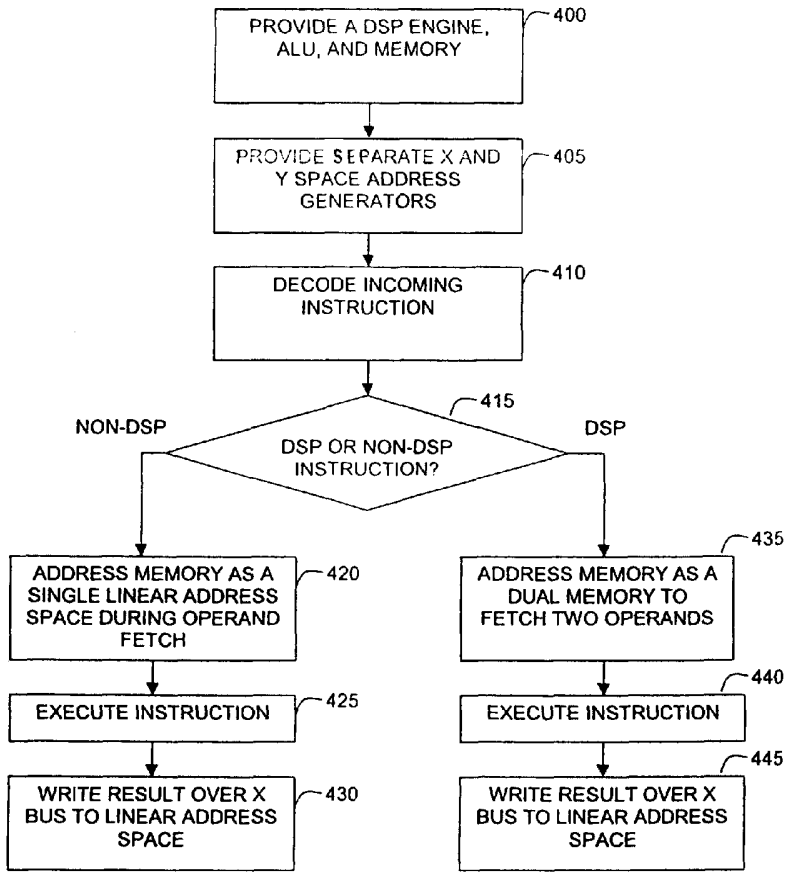
(81) Designated States *(national)*: CN, JP.

(84) Designated States *(regional)*: European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR).

Published:
— *with international search report*

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

(54) Title: DYNAMICALLY RECONFIGURABLE DATA SPACE



(57) Abstract: A processor is provided that has a data memory that may be addressed as a dual memory space in one mode and as a single linear memory space in another mode. The memory may permit dual concurrent operand fetches from the data memory when DSP instructions are processed. The memory may then dynamically permit the same memory to be accessed as a single linear memory address space for non-DSP instructions.

WO 02/099634 A1

DYNAMICALLY RECONFIGURABLE DATA SPACE

FIELD OF THE INVENTION:

The present invention relates to systems and methods for addressing memory locations and, more particularly, to systems and methods for dynamically accessing a memory as a linear memory space and as a dual memory space.

BACKGROUND OF THE INVENTION:

Digital Signal Processors (DSPs) conventionally are structured to perform mathematical calculations very quickly. The calculations may include repetitive operations such as multiply and accumulate operations that are performed on a series of constant or calculated data stored in a memory. For optimum performance, DSPs must be able to repetitively fetch two operands for immediate processing, calculate a result and write the result back within one processor cycle. In pipelined implementations, result write back should occur at the end of each processor cycle once the pipeline is full.

To allow DSPs to show significant performance improvement over conventional processors for the above types of operations, DSPs typically have the ability to perform dual concurrent operand fetches from a relatively large data memory. Conventional micro-controllers and microprocessors typically allow dual concurrent operand fetches only from a limited number registers, which is overly restrictive for DSPs. This is because processor cycles must be devoted to loading registers with operand data as an intermediate step prior to processing the data, rather than fetching the operand data directly from the memory in the case of a DSP.

Conventional DSPs typically include two data memories, an X memory and a Y memory. They further include X and Y data buses coupled respectively to the X and Y memories. The X and Y memories are separately addressed and allow the DSP to perform dual, concurrent operand fetching and processing directly from the memories - one operand being fetched from the X memory and the other from the Y memory.

Providing separate X and Y memories with separate busing structures allows conventional DSPs to efficiently execute instructions on two operands. However, for instructions that do not require two operands, as is the case for typical MCU or non-DSP instructions, separate X and Y memories may be a disadvantage. This is because operands

may be stored in either the X or the Y memory. Accordingly, programmers must keep track of which memory the operand data is stored in to retrieve the proper data.

Accordingly, there is a need for a more efficient memory organization for digital signal processing which permits good MCU or non-DSP instruction support. There is a further need for a memory organization that provides dual concurrent operand fetching for DSP class instructions but which allows access to all memory locations when single operands are required without requiring software overhead to track whether the operand is in the X or Y memory and without requiring separate read instructions.

SUMMARY OF THE INVENTION:

According to the present invention, a processor is provided that has a data memory that may be addressed as a dual memory space in one mode and as a single linear memory space in another mode. The memory may permit dual concurrent operand fetches from the data memory when DSP instructions are processed. The memory may then dynamically permit the same memory to be accessed as a single linear memory address space for non-DSP instructions.

According to an embodiment of the invention, the method provides linear and dual address spaces within a memory. The method includes providing a digital signal processor (DSP) engine, an arithmetic logic unit (ALU) and a memory. The method further includes providing separate X and Y address generation units for generating addresses to the memory. The X address generation unit may be configured to generate addresses to all of the memory space when processing a non-digital signal processor (DSP) engine instruction and to a first portion of the memory when processing a DSP engine instruction. The Y address generation unit may be configured to generate addresses to a second portion of the memory space when processing a DSP engine instruction and to none of the memory space when processing a non-digital signal processor (DSP) engine instruction. The method may further include providing an instruction decoder that decodes instructions and, in response, activates the X address generation unit for ALU instructions and activates the X and Y address generation units for DSP engine instructions.

The first and second memory portions may be overlapping or non-overlapping. The first and second portion memory portions may further be contiguous or not contiguous with respect to each other. In addition, the first portion may include a portion above the second portion and a portion below the second portion.

According to another embodiment, the method may further include providing separate X and Y data buses. In this configuration, the X data bus may be coupled to the memory permitting reading and writing to and from the entire memory and the Y data bus may be coupled to the second portion of the memory permitting reading and also writing in some embodiments from the second portion. The method may further include reading X and Y operands concurrently from the respective first and second portions of the memory over the X and Y data buses based on the instruction decoder decoding a DSP instruction requiring a dual operand fetch.

According to another embodiment of the invention, a processor addresses a memory as a signal and dual space memory based on an addressing mode. The processor includes a data memory, an instruction decoder and X and Y address generators. The data memory has X and Y memory spaces. The instruction decoder determines whether an instruction requires single or dual memory space operation and selectively activates the X and Y address generators. The instruction decoder activates the X and Y address generators based on the instruction decoder determining dual memory space operation is required. Alternatively, the instruction decoder activates the X address generator only based on the instruction decoder determining single memory space operation is required.

The processor may further include registers coupled to the X and Y address generators for storing pointers to the memory. For a dual memory space operation, a subset of the registers stores pointers to the Y space within the memory and a subset of the registers stores pointers to the X space within the memory. For a single memory space operation, any of the registers stores pointers to the memory.

The processor may further include X and Y data buses coupled to the memory for retrieving operands from the X and Y memory spaces in a dual memory space operation and for retrieving operands from the memory over the X data bus in a single memory space operation. The processor may further include a DSP engine having operand inputs coupled to the X and Y data buses and having a result output coupled only to the X data bus. The processor may still further include an arithmetic logic unit (ALU) coupled to the X data buses for operand and result input and output.

BRIEF DESCRIPTION OF THE FIGURES:

The above described features and advantages of the present invention will be more fully appreciated with reference to the detailed description and appended figures in which:

Fig. 1 depicts a functional block diagram of an embodiment of a processor chip within which embodiments of the present invention may find application.

Fig. 2 depicts a functional block diagram of a data busing scheme for use in a processor, which has a microcontroller and a digital signal processing engine, within which embodiments of the present invention may find application.

Fig. 3A depicts a diagram of the memory space in a linear address mode according to an embodiment of the present invention.

Fig. 3B depicts a diagram of the memory space in a dual address mode according to an embodiment of the present invention.

Fig. 4 depicts a method of dynamically reconfiguring a memory to permit addressing the memory as a dual memory or as a memory with a single linear address space according to an embodiment of the present invention.

Fig. 5 depicts a simplified view of the processor to depict an interaction between instruction decoding logic and the address generators to illustrate single and dual memory space addressing modes according to an embodiment of the present invention.

DETAILED DESCRIPTION:

According to the present invention, a processor is provided that has a data memory that may be addressed as a dual memory space in one mode and as a single linear memory space in another mode. The memory may permit dual concurrent operand fetches from the data memory when DSP instructions are processed. The memory may then dynamically permit the same memory to be accessed as a single linear memory address space for non-DSP instructions.

In order to describe embodiments of processing instructions in word and/or byte modes, an overview of pertinent processor elements is first presented with reference to Figs. 1 and 2. The systems and methods for implementing word and/or byte mode processing are then described more particularly with reference to Figs. 3-5.

Overview of Processor Elements

Fig. 1 depicts a functional block diagram of an embodiment of a processor chip within which the present invention may find application. Referring to Fig. 1, a processor 100 is coupled to external devices/systems 140. The processor 100 may be any type of processor including, for example, a digital signal processor (DSP), a microprocessor, a microcontroller or combinations thereof. The external devices 140 may be any type of systems or devices

including input/output devices such as keyboards, displays, speakers, microphones, memory, or other systems which may or may not include processors. Moreover, the processor 100 and the external devices 140 may together comprise a stand alone system.

The processor 100 includes a program memory 105, an instruction fetch/decode unit 110, instruction execution units 115, data memory and registers 120, peripherals 125, data I/O 130, and a program counter and loop control unit 135. The bus 150, which may include one or more common buses, communicates data between the units as shown.

The program memory 105 stores software embodied in program instructions for execution by the processor 100. The program memory 105 may comprise any type of nonvolatile memory such as a read only memory (ROM), a programmable read only memory (PROM), an electrically programmable or an electrically programmable and erasable read only memory (EPROM or EEPROM) or flash memory. In addition, the program memory 105 may be supplemented with external nonvolatile memory 145 as shown to increase the complexity of software available to the processor 100. Alternatively, the program memory may be volatile memory which receives program instructions from, for example, an external non-volatile memory 145. When the program memory 105 is nonvolatile memory, the program memory may be programmed at the time of manufacturing the processor 100 or prior to or during implementation of the processor 100 within a system. In the latter scenario, the processor 100 may be programmed through a process called in-line serial programming.

The instruction fetch/decode unit 110 is coupled to the program memory 105, the instruction execution units 115 and the data memory 120. Coupled to the program memory 105 and the bus 150 is the program counter and loop control unit 135. The instruction fetch/decode unit 110 fetches the instructions from the program memory 105 specified by the address value contained in the program counter 135. The instruction fetch/decode unit 110 then decodes the fetched instructions and sends the decoded instructions to the appropriate execution unit 115. The instruction fetch/decode unit 110 may also send operand information including addresses of data to the data memory 120 and to functional elements that access the registers.

The program counter and loop control unit 135 includes a program counter register (not shown) which stores an address of the next instruction to be fetched. During normal instruction processing, the program counter register may be incremented to cause sequential instructions to be fetched. Alternatively, the program counter value may be altered by loading a new value into it via the bus 150. The new value may be derived based on decoding and executing a flow control instruction such as, for example, a branch instruction.

In addition, the loop control portion of the program counter and loop control unit 135 may be used to provide repeat instruction processing and repeat loop control as further described below.

The instruction execution units 115 receive the decoded instructions from the instruction fetch/decode unit 110 and thereafter execute the decoded instructions. As part of this process, the execution units may retrieve one or two operands via the bus 150 and store the result into a register or memory location within the data memory 120. The execution units may include an arithmetic logic unit (ALU) such as those typically found in a microcontroller. The execution units may also include a digital signal processing engine, a floating point processor, an integer processor or any other convenient execution unit. A preferred embodiment of the execution units and their interaction with the bus 150, which may include one or more buses, is presented in more detail below with reference to Fig. 2.

The data memory and registers 120 are volatile memory and are used to store data used and generated by the execution units. The data memory 120 and program memory 105 are preferably separate memories for storing data and program instructions respectively. This format is a known generally as a Harvard architecture. It is noted, however, that according to the present invention, the architecture may be a Von-Neuman architecture or a modified Harvard architecture which permits the use of some program space for data space. A dotted line is shown, for example, connecting the program memory 105 to the bus 150. This path may include logic for aligning data reads from program space such as, for example, during table reads from program space to data memory 120.

Referring again to Fig. 1, a plurality of peripherals 125 on the processor may be coupled to the bus 125. The peripherals may include, for example, analog to digital converters, timers, bus interfaces and protocols such as, for example, the controller area network (CAN) protocol or the Universal Serial Bus (USB) protocol and other peripherals. The peripherals exchange data over the bus 150 with the other units.

The data I/O unit 130 may include transceivers and other logic for interfacing with the external devices/systems 140. The data I/O unit 130 may further include functionality to permit in circuit serial programming of the Program memory through the data I/O unit 130.

Fig. 2 depicts a functional block diagram of a data busing scheme for use in a processor 100, such as that shown in Fig. 1, which has an integrated microcontroller arithmetic logic unit (ALU) 270 and a digital signal processing (DSP) engine 230. This configuration may be used to integrate DSP functionality to an existing microcontroller core. Referring to Fig. 2, the data memory 120 of Fig. 1 is implemented as two separate memories:

an X-memory 210 and a Y-memory 220, each being respectively addressable by an X address generator 250 and a Y-address generator 260. The X-address generator may also permit addressing the Y-memory space thus making the data space appear like a single contiguous memory space when addressed from the X address generator. The bus 150 may be implemented as two buses, one for each of the X and Y memory, to permit simultaneous fetching of data from the X and Y memories.

The W registers 240 are general purpose address and/or data registers. The DSP engine 230 is coupled to both the X and Y memory buses and to the W registers 240. The DSP engine 230 may simultaneously fetch data from each the X and Y memory, execute instructions which operate on the simultaneously fetched data and write the result to an accumulator (not shown) and write a prior result to X or Y memory or to the W registers 240 within a single processor cycle.

In one embodiment, the ALU 270 may be coupled only to the X memory bus and may only fetch data from the X bus. However, the X and Y memories 210 and 220 may be addressed as a single memory space by the X address generator in order to make the data memory segregation transparent to the ALU 270. The memory locations within the X and Y memories may be addressed by values stored in the W registers 240.

Any processor clocking scheme may be implemented for fetching and executing instructions. A specific example follows, however, to illustrate an embodiment of the present invention. Each instruction cycle is comprised of four Q clock cycles Q1 – Q4. The four phase Q cycles provide timing signals to coordinate the decode, read, process data and write data portions of each instruction cycle.

According to one embodiment of the processor 100, the processor 100 concurrently performs two operations – it fetches the next instruction and executes the present instruction. Accordingly, the two processes occur simultaneously. The following sequence of events may comprise, for example, the fetch instruction cycle:

Q1: Fetch Instruction

Q2: Fetch Instruction

Q3: Fetch Instruction

Q4: Latch Instruction into prefetch register, Increment PC

The following sequence of events may comprise, for example, the execute instruction cycle for a single operand instruction:

Q1: latch instruction into IR, decode and determine addresses of operand data

Q2: fetch operand

Q3: execute function specified by instruction and calculate destination address for data

Q4: write result to destination

The following sequence of events may comprise, for example, the execute instruction cycle for a dual operand instruction using a data pre-fetch mechanism. These instructions pre-fetch the dual operands simultaneously from the X and Y data memories and store them into registers specified in the instruction. They simultaneously allow instruction execution on the operands fetched during the previous cycle.

Q1: latch instruction into IR, decode and determine addresses of operand data

Q2: pre-fetch operands into specified registers, execute operation in instruction

Q3: execute operation in instruction, calculate destination address for data

Q4: complete execution, write result to destination

Dynamically Reconfigurable Memory

Figs. 3A and 3B depict an organization of the data memory 120 according to an embodiment of the present invention. Referring to Fig. 3A, the data memory 120 may be organized as a contiguous memory that includes an X space portion and a Y space portion. When the processor is processing a non-DSP engine instruction, only the X address generator 250 is active. In this mode, the entire memory may be accessed using the X address generator. Accordingly, data may be read from the entire memory and written into the entire memory 120, including the Y space portion thereof, using the X address generator. This permits the data memory 120 to be accessed as a linear address space during non-DSP engine instruction processing.

During DSP engine processing, the data memory 120 is defined to have separate X and Y spaces. The X and Y space are separately addressable by the X address generator and the Y address generator. The X address generator generates addresses for the X portion of the data memory and the Y address generator only generates addresses for the Y portion of the data memory. In this mode, the X and Y spaces may each be less than the entire memory and in a preferred embodiment the X space includes portions above and below the Y space portion.

The instruction decoder generates control signals, based on the decoded instruction which determine the addressing mode, for activating the X address generator 250 and the Y address generator 260. When the instruction decoder decodes a non-DSP engine instruction, the instruction decoder generates control signals which disable the y-address generator 260 and the y-data bus. The instruction decoder also generates control signals that allow any of the registers 240 to include a pointer value to any location within the data memory 120. Accordingly, instruction processing may entail addressing the data memory as a linear address space via the X address generator to a) fetch operands from the memory 120, b) execute the instruction on the operands and c) store back the result into the data memory over the x-data bus into a memory location specified to the X address generator 250.

When the instruction decoder decodes a DSP engine instruction, the instruction decoder generates control signals which enable the y-address generator 260 and the y-data bus. In addition, the instruction decoder may configure the x-address generator 250 and the y-address generator 260 to only accept indirect addresses from certain registers within the W array 240. For example, in a preferred embodiment, the x-address generator and the y-address generator are each configured to only process indirect addresses from a respective four of the W-registers 240 with two being different for X address generation and 2 different ones being for Y address generation. However, any number of registers may be allocated for these purposes.

During execution of a DSP instruction, the instruction decoder decodes the DSP engine instruction and causes the DSP engine to fetch and process two operands simultaneously from the data memory 120. One of the operands is fetched from the y-data space over the y-data bus based on a pointer address specified in one of two of the designated W registers 240 and decoded by the Y address generator. The other operand is fetched from the x-data space over the x-data bus based on a pointer address specified in one of the designated W registers 240 and decoded by the X address generator. The X and Y spaces may be non-overlapping and contiguous. Alternatively, the X and Y spaces may be overlapping.

Fig. 4 depicts a method of dynamically reconfiguring a memory to permit addressing the memory as a dual memory or as a memory with a single linear address space according to an embodiment of the present invention. Referring to Fig. 4, in step 400, a DSP engine an arithmetic logic unit (ALU) and a memory are provided on a processor. This configuration may permit the processor to process DSP instructions in addition to microcontroller or

microprocessor instructions and may further permit the melding of two instructions sets or an expanded instruction set within the same processor.

In step 405, separate X and Y space address generators are provided for addressing portions of the memory as two separate X and Y memory spaces in one mode and for addressing the entire memory as a single linear address space using the X address generator in another mode.

The X and Y address generators may be configured at the time of manufacturing the processor to set the portions addressable by the X and Y address generators in the dual memory space mode. Alternatively, the X and Y address generators may each permit access to all of the memory space with the users enforcing discipline to maintain the memory spaces as non-overlapping or otherwise to include mechanisms to prevent address conflicts between the X and Y memory spaces.

In step 410, the processor decodes an incoming instruction. Then in step 415 an instruction decoder determines whether the instruction is a DSP instruction or a non-DSP instruction in order to determine whether to use the dual memory space mode or the linear memory space mode. The decision may be made based on the opcode of the instruction being present in a look up table, being within a predetermined range or otherwise meeting a predetermined criteria for dual memory space mode.

If in step 415 the instruction meets the predetermined criteria for dual memory space mode, then step 435 begins. In step 435, the processor addresses the data memory as a dual memory to concurrently fetch two operands. Then in step 440, the processor executes the instruction. As an example, the instruction may be a multiply and accumulate instruction which multiplies the two operands that were concurrently fetched and produces a result. Then in step 445, the processor writes the result back to the data memory over the X bus.

If in step 415 the instruction does not meet the predetermined criteria for dual memory space mode, then step 420 begins. In step 420, the processor addresses the memory as a single linear address space during operand fetch and fetches at most a single operand. In step 425, the processor executes the instruction on the operand fetched from memory or the registers or on the operands fetched from the registers. Then in step 430, the processor writes the result of the instruction over the X bus to the data memory using the X address generator to address the memory as a linear address space.

Fig. 5 depicts a simplified view of the processor to depict an interaction between instruction decoding logic and the address generators to illustrate single and dual memory space addressing modes according to an embodiment of the present invention. Referring to

Fig. 5, a data memory array 500 is coupled to a x, Y data multiplexer 505 which affords read and write access to memory locations within the memory array 500. The memory array may include dual bit lines for permitting simultaneous reads from two memory locations in the same active word column over the X and Y buses 510 and 515, respectively. Alternatively, the memory array and X and Y addressing schemes may be organized to provide simultaneous reading from any memory location within the memory array via the X and Y data buses. The X and Y data multiplexer may also be configured to allow only reading from the Y data bus and reading and writing from the X data bus.

The X and Y data buses are coupled to the DSP engine 550 to permit concurrent fetching of operands from the X and Y memory spaces. The DSP engine may be configured to provide result output only over the X bus for writing back to the memory array 500. The ALU 555 may be coupled to the X data bus 510 to permit reading and writing only over the X data bus.

The instruction decoder 540 decodes instructions retrieved from program memory (not shown). The instruction decoder 540 may include determining logic for determining whether the operation code (op code) of the instruction meets a predetermined criteria for treating the memory array 500 as a dual space memory. The logic may include logic for comparing all or portions of the op code (or other portions of the instruction) with values in a look up table or a predetermined range of values. Alternatively, the logic may check for the presence or absence of certain values within bit fields of the instruction. Based on the determination made by the determining logic, the instruction decoder 540 sends control signals over lines 560 and 565 respectively to the X and Y address generators.

The X address generator is coupled to the control signal line 560 from the instruction decoder and to the registers 525. When the instruction is determined to not be an instruction that requires dual operand fetch (or to be an instruction that requires a single memory space for operand fetch), a toggle signal and an address of one of the registers 525 may be sent over the line 560 to the X address generator 520. The toggle signal sets the X address generator 520 to address the memory array 500 as a single linear address space. The register address sent over the control lines 560 is used to select one of the registers 525 to supply a pointer value for determining a memory location within the memory array 500.

When the instruction is determined to be an instruction that requires dual operand fetch from dual address spaces, the toggle signal sent over the control lines 560 sets the X address generator 520 to address only the portions of the memory array 500 that are assigned to X memory space. Moreover, logic within the X address generator 520 may restrict the

number of registers within the registers 520 from which the X address generator may pull values for addressing the memory array 500. In the example shown, the X address generator is limited to pulling values from registers W4 and W5 for X address generation during dual memory space mode. The control signals 560 further include values identifying which registers W4 or W5 to use for pointing to the required memory location within the X memory space of the memory array 500.

The Y address generator is coupled to the control signal lines 565 from the instruction decoder and may be coupled to all or only a portion of the registers 525. Fig. 5 shows an illustrative example in which the Y address generator 530 is coupled to only a portion of the registers (W6 and W7) which are labeled 535. When the instruction is determined to not be an instruction that requires dual operand fetch (or to be an instruction that requires a single memory space for operand fetch), a toggle signal may be sent over the line 560 to the Y address generator 530. The toggle signal sets the Y address generator 530 to disable memory addressing.

When the instruction is determined to be an instruction that requires dual operand fetch from dual address spaces, the toggle signal sent over the control lines 565 sets the Y address generator 530 to address only the portions of the memory array 500 that are assigned to Y memory space. Moreover, logic within the Y address generator 530 may restrict the number of registers within the registers 525 to the registers 535 from which the Y address generator may pull values for addressing the memory array 500. The control signals 565 further include values identifying which registers W6 or W7 to use for pointing to the required memory location within the Y memory space of the memory array 500.

While particular embodiments of the present invention have been illustrated and described, it will be understood by those having ordinary skill in the art that changes may be made to those embodiments without departing from the spirit and scope of the invention. For example, it will be understood that operands may be written concurrently over both the X and Y buses back to the X and Y portions of the data memory. It will also be understood that while DSP instructions have been illustrated as being capable of dual operand memory operation and non-DSP instruction have been illustrated as not being capable of dual operand memory operation, this limitation is for illustration purposes only. Other instruction classes, including non-DSP instruction classes, and individual instructions regardless of their "class" may also be capable of dual operand processing from the memory.

CLAIMS:

What is claimed is:

1. A method of providing linear and dual address spaces within a memory, comprising:

providing a digital signal processor (DSP) engine, an arithmetic logic unit (ALU) and a memory;

providing separate X and Y address generation units for generating addresses to the memory,

the X address generation unit being configured to generate addresses to all of the memory space when processing a non-digital signal processor (DSP) engine instruction and to a first portion of the memory when processing a DSP engine instruction;

the Y address generation unit being configured to generate addresses to a second portion of the memory space when processing a DSP engine instruction and to none of the memory space when processing a non-digital signal processor (DSP) engine instruction; and

providing an instruction decoder for decoding instructions, the instruction decoder activating the X address generation unit for ALU instructions and the instruction decoder activating the X and Y address generation units for DSP engine instructions.

2. The method according to claim 1, wherein the first and second portion are non-overlapping.

3. The method according to claim 2, wherein the first and second portion are contiguous.

4. The method according to claim 1, wherein the first portion includes a portion above the second portion and a portion below the second portion.

5. The method according to claim 1, further comprising:

providing separate X and Y data buses,

the X data bus being coupled to the memory permitting reading and writing to and from the entire memory, and

the Y data bus being coupled to the second portion of the memory permitting only reading from the second portion.

6. The method according to claim 5, further comprising:

reading X and Y operands concurrently from the respective first and second portions of the memory over the X and Y data buses based on the instruction decoder decoding a DSP instruction requiring a dual operand fetch.

7. The method according to claim 6, further comprising:

executing the DSP instruction to calculate a result based on the X and Y operands; and

writing the result back to the first portion of the memory over the X bus.

8. A processor for addressing a memory as a signal and dual space memory based on an addressing mode, comprising:

a data memory having X and Y memory spaces;

an instruction decoder for determining whether an instruction requires single or dual memory space operation; and

X and Y address generators coupled to the instruction decoder, the data memory activating the X and Y address generators based on the instruction decoder determining dual memory space operation is required and activating the X address generator only based on the instruction decoder determining single memory space operation is required.

9. The processor according to claim 8, further comprising:

registers coupled to the X and Y address generators for storing pointers to the memory.

10. The processor according to claim 9, wherein for a dual memory space operation, a subset of the registers stores pointers to the Y space within the memory.

11. The processor according to claim 10, wherein for a dual memory space operation, a subset of the registers stores pointers to the X space within the memory.

12. The processor according to claim 11, wherein for a single memory space operation, any of the registers stores pointers to the memory.

13. The processor according to claim 9, further comprising:

X and Y data buses coupled to the memory for retrieving operands from the X and Y memory spaces in a dual memory space operation and for retrieving operands from the memory over the X data bus in a single memory space operation.

14. The processor according to claim 10, further comprising:

a DSP engine having operand inputs coupled to the X and Y data buses and having a result output coupled only to the X data bus.

15. The processor according to claim 9, further comprising:

an arithmetic unit coupled to the X data buses for operand and result input and output.

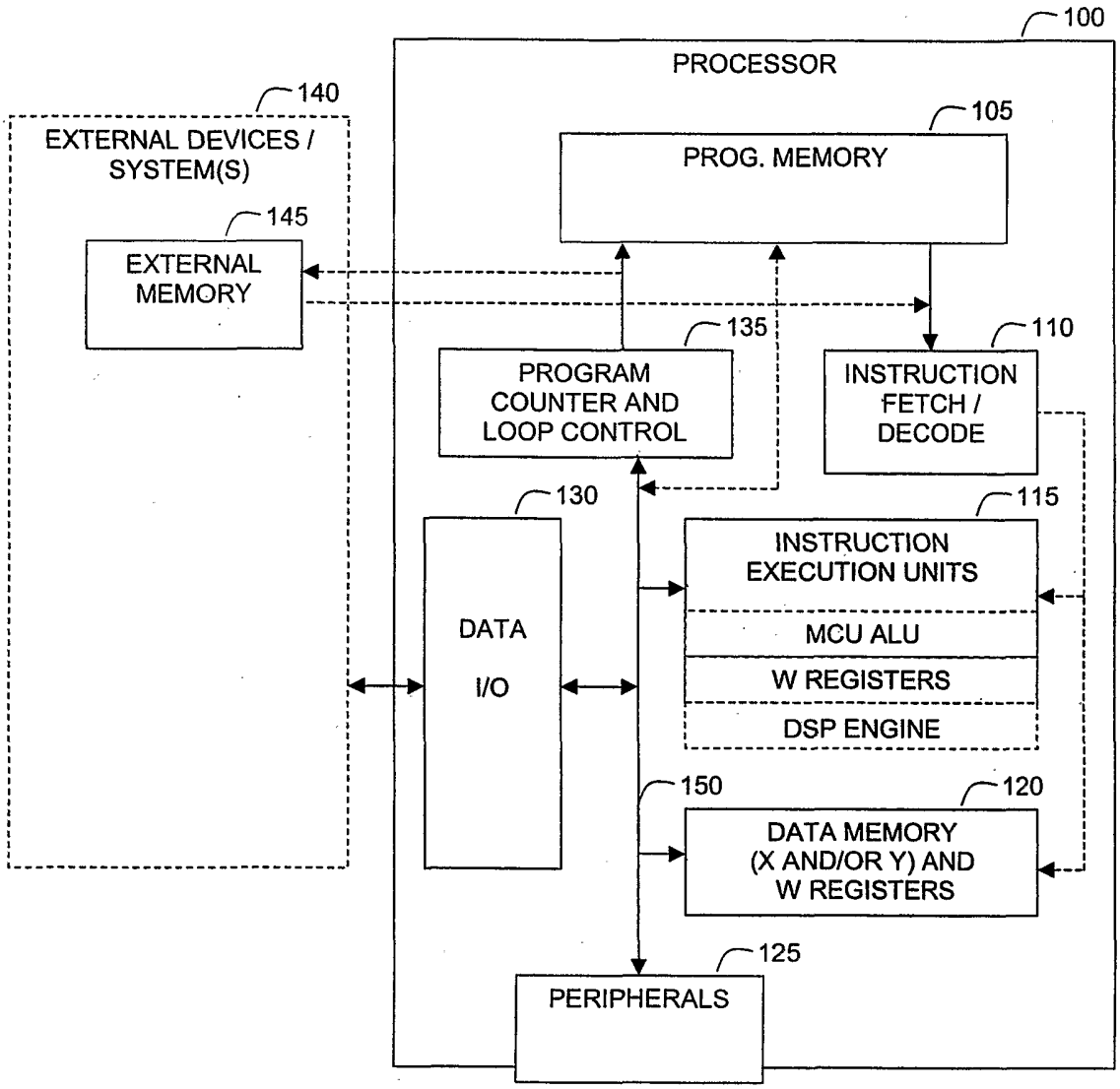

100
PROCESSOR
140
EXTERNAL DEVICES / SYSTEM(S)
105
PROG. MEMORY
145
EXTERNAL MEMORY
135
PROGRAM COUNTER AND LOOP CONTROL
110
INSTRUCTION FETCH / DECODE
130
DATA I/O
115
INSTRUCTION EXECUTION UNITS
MCU ALU
W REGISTERS
DSP ENGINE
150
120
DATA MEMORY (X AND/OR Y) AND W REGISTERS
125
PERIPHERALS

FIG. 1

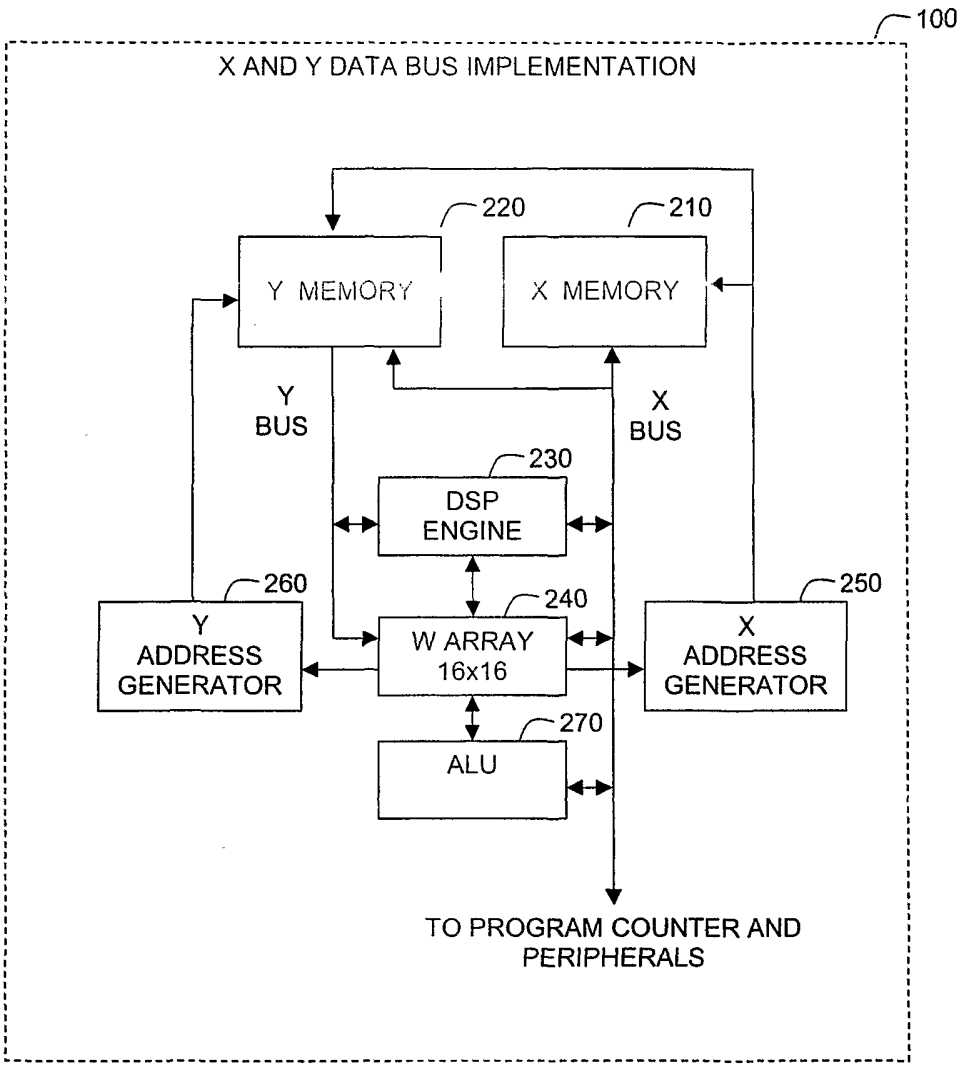

100
X AND Y DATA BUS IMPLEMENTATION
220
210
Y MEMORY
X MEMORY
Y
BUS
X
BUS
230
DSP
ENGINE
260
240
250
Y
ADDRESS
GENERATOR
W ARRAY
16x16
X
ADDRESS
GENERATOR
270
ALU
TO PROGRAM COUNTER AND
PERIPHERALS

FIG. 2

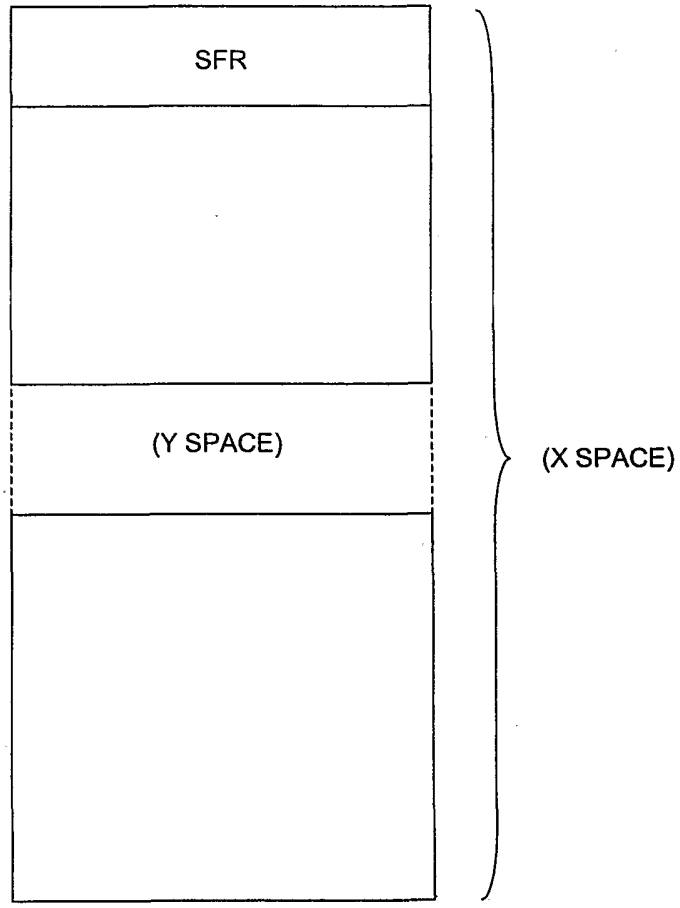

SFR
(Y SPACE)
(X SPACE)
INDIRECT EA
FROM ANY
ADDRESS
NON DSP ENGINE INSTRUCTIONS

FIG. 3A

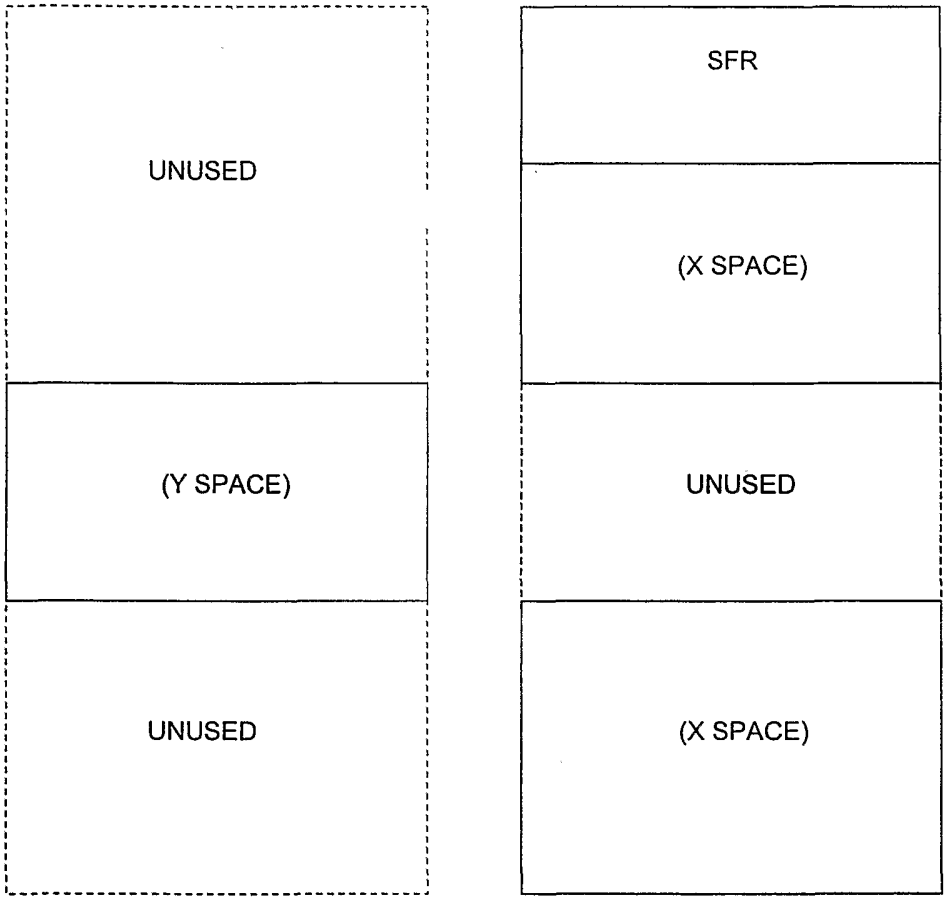

UNUSED
(Y SPACE)
UNUSED
SFR
(X SPACE)
UNUSED
(X SPACE)
INDIRECT EA
FROM W6, W7
INDIRECT EA
FROM W4, W5
DSP ENGINE OPERATIONS

FIG. 3B

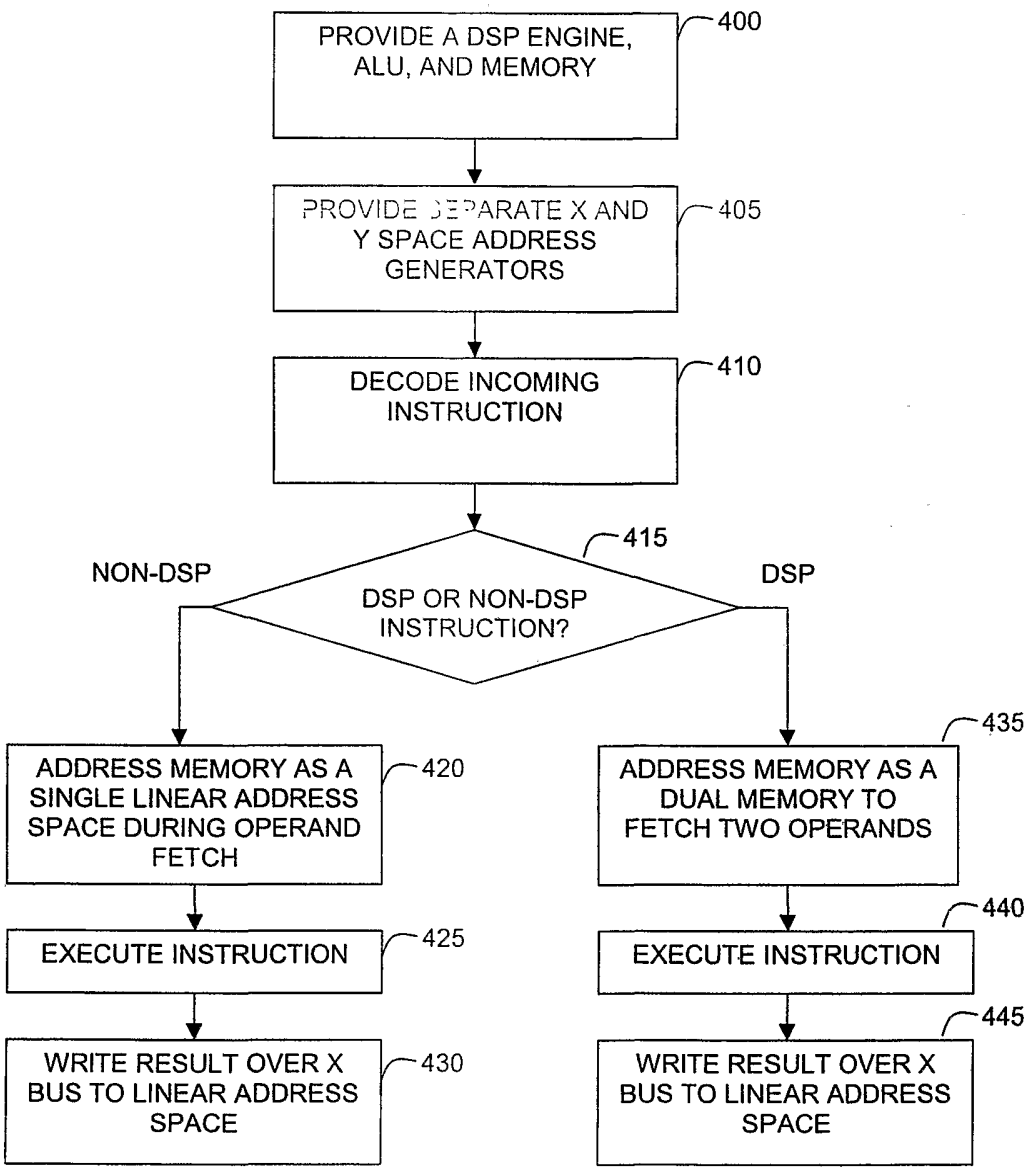

PROVIDE A DSP ENGINE, ALU, AND MEMORY
400
PROVIDE SEPARATE X AND Y SPACE ADDRESS GENERATORS
405
DECODE INCOMING INSTRUCTION
410
DSP OR NON-DSP INSTRUCTION?
415
NON-DSP
DSP
ADDRESS MEMORY AS A SINGLE LINEAR ADDRESS SPACE DURING OPERAND FETCH
420
EXECUTE INSTRUCTION
425
WRITE RESULT OVER X BUS TO LINEAR ADDRESS SPACE
430
ADDRESS MEMORY AS A DUAL MEMORY TO FETCH TWO OPERANDS
435
EXECUTE INSTRUCTION
440
WRITE RESULT OVER X BUS TO LINEAR ADDRESS SPACE
445

FIG. 4

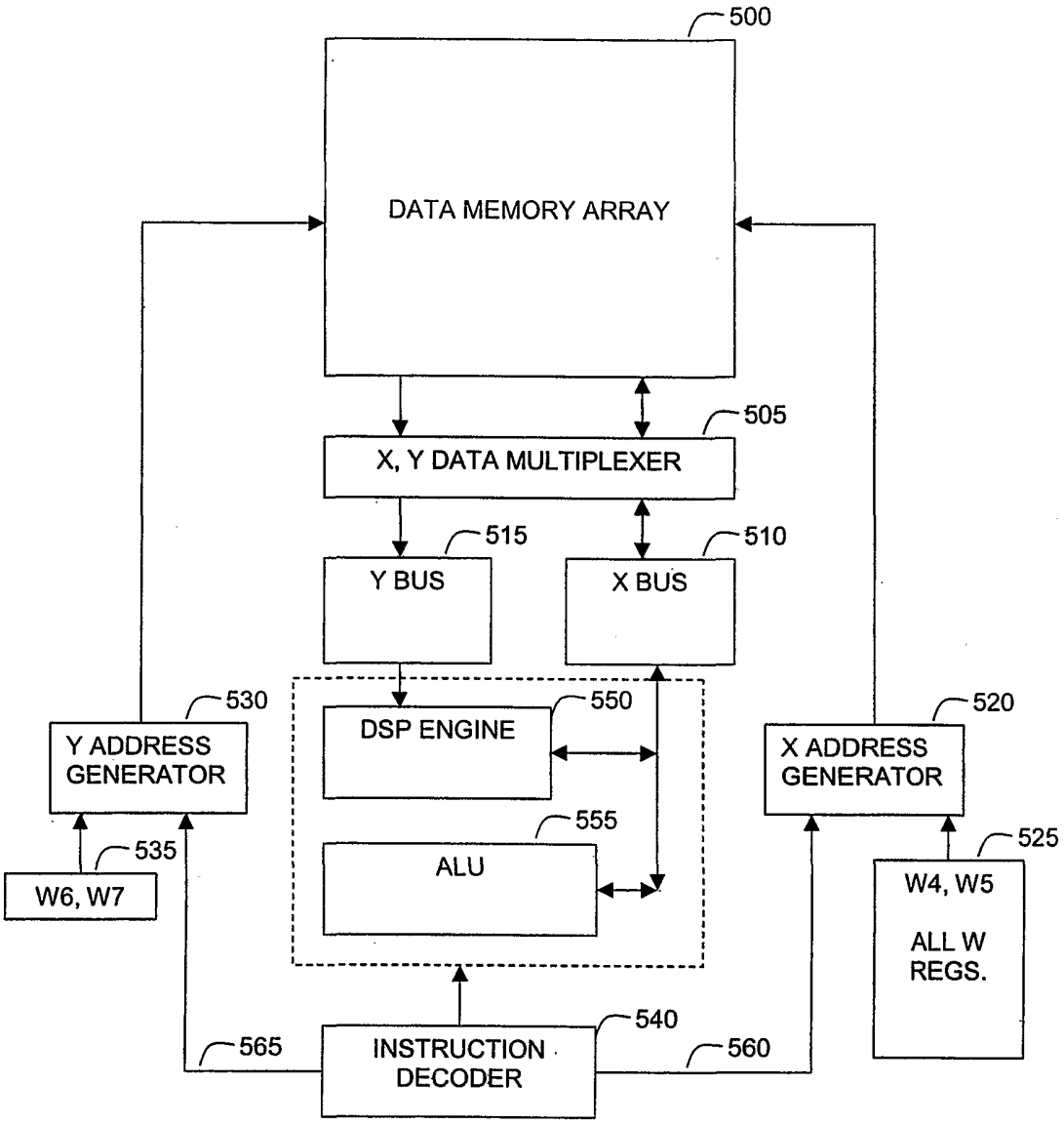

500
DATA MEMORY ARRAY
505
X, Y DATA MULTIPLEXER
515
Y BUS
510
X BUS
530
Y ADDRESS GENERATOR
550
DSP ENGINE
520
X ADDRESS GENERATOR
555
ALU
535
W6, W7
525
W4, W5
ALL W REGS.
540
565
INSTRUCTION DECODER
560

FIG. 5

INTERNATIONAL SEARCH REPORT

International Application No
PCT/US 02/16970

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 G06F9/38 G06F9/34 G06F9/345 G06F9/30 G06F12/10

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 7 G06F

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

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
INSPEC, EPO-Internal, IBM-TDB

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	MARTIN D ET AL: "A RISC architecture with uncompromised digital signal processing and microcontroller operation" ACOUSTICS, SPEECH AND SIGNAL PROCESSING, 1998. PROCEEDINGS OF THE 1998 IEEE INTERNATIONAL CONFERENCE ON SEATTLE, WA, USA 12-15 MAY 1998, NEW YORK, NY, USA,IEEE, US, 12 May 1998 (1998-05-12), pages 3097-3100, XP010279502 ISBN: 0-7803-4428-6 the whole document --- -/--	1-15

[X] Further documents are listed in the continuation of box C.

[X] Patent family members are listed in 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 document 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	Date of mailing of the international search report
17 September 2002	26/09/2002

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Thibaudeau, J

Form PCT/ISA/210 (second sheet) (July 1992)

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication,where appropriate, of the relevant passages	Relevant to claim No.
A	GEORGE A D: "DSP96002 vs. i860: Digital Signal Processing Applications" IEEE, August 1992 (1992-08), pages 110-114, XP010285255 the whole document ---	1-15
A	KLOKER K L ET AL: "The Motorola DSP 96002 IEEE floating-point digital signal processor" IEEE, 1989, pages 2480-2483, XP010082807 the whole document ---	1-15
A	US 5 832 257 A (FETTWEIS GERHARD ET AL) 3 November 1998 (1998-11-03) column 1, line 19 - line 57 column 2, line 66 -column 4, line 27 -----	1-15

INTERNATIONAL SEARCH REPORT

Information on patent family members

International Application No
PCT/US 02/16970

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 5832257 A	03-11-1998	NONE	