MULTIPLIER ARCHITECTURE IN A GENERAL PURPOSE PROCESSOR OPTIMIZED FOR EFFICIENT MULTI-INPUT ADDITION

BACKGROUND

Digital signal processors (DSPs) are often used for a variety of multimedia applications such as digital video, imaging, and audio. DSPs can manipulate the digital signals to create and open such multimedia files.

MPEG-1, MPEG-2, MPEG-4 and H.263 are digital video compression standards and file formats. These standards achieve a high compression rate of the digital video signals by storing mostly changes from one video frame to another, instead of storing each entire frame. The video information can then be further compressed using a number of different techniques.

The DSP is used to perform various operations on the video information during compression. These can include motion search and spatial interpolation algorithms. The primary intention is to measure distortion between blocks within adjacent frames. These operations are computationally intensive and require high data throughput.

The MPEG family of standards is evolving to keep pace with the increasing bandwidth requirements of multimedia

166166

G00002

applications and files. Each new version of the standard presents more sophisticated algorithms that place even greater processing requirements on the DSPs used in MPEG compliant video processing equipment.

Video processing equipment manufacturers often rely on application-specific integrated circuits (ASICs) customized for video encoding under the MPEG and H.263 standards. However, ASICs are complex to design, costly to produce and less flexible in their application than general-purpose DSPs.

SUMMARY

According to one aspect of the invention, there is provided an apparatus comprising: a decoder to decode an instruction specifying an adder-tree operation; a circuit coupled to the decoder and comprising: a multiplier including a partial product generator, and a multiplier array comprising a plurality of arithmetic logic units; and a selection circuit to select first and second operands from a 2n-bit word, wherein said first and second operands have fewer bits than 2n, and said selection circuit is adapted to bypass the partial product generator and add said first and second operands in a selected one of said ALUs in the multiplier array to produce a first sum.

According to a second aspect of the invention, there is provided a video processor comprising: a central processing unit (CPU); a memory to store instructions to perform video encoding operations; a digital signal processor coupled to said CPU and said memory and including an execution unit comprising: a decoder to decode an instruction specifying an adder-tree operation; and a first circuit coupled to the decoder, and comprising a multiplier including a partial product generator, and a multiplier array comprising a plurality of arithmetic logic units (ALUs), said first circuit including an element that, in response to said instructions, selects first and second n-bit operands from a 2n-bit word, and bypassing the partial product generator adds said first and second n-bit operands in a selected one of the ALUs in the multiplier array to produce a first sum.

According to a third aspect of the invention, there is provided a method comprising: selecting a multiplier mode; selecting a first 2n-bit multiplicand from at least

one first 2n-bit word; selecting a second 2n-bit multiplicand from at least one second 2n-bit word; and multiplying said first and second multiplicands in a multiplier comprising a plurality of ALUs, to produce a product, said multiplying comprising: generating a plurality of partial products from the first 2n-bit multiplicand and the second multiplicand in a partial product generator in the multiplier; and adding said plurality of partial products in a multiplier array in the multiplier; selecting an adder-tree mode; selecting first and second operands from a 2n-bit word; bypassing the partial product generator and adding said two first and second operands in a first one of the ALUs in the multiplier array; and outputting a sum from the multiplier array.

According to a fourth aspect of the invention, there is provided a program storage device readable by a machine, comprising instructions that cause the machine to: select a multiplier mode; select a first 2n-bit multiplicand from at least one first 2n-bit word; select a second 2n-bit multiplicand from at least one second 2n-bit word; and multiply said first and second multiplicands in a multiplier comprising a plurality of ALUs, to produce a product, said multiplying comprising: generating a plurality of partial products from the first 2n-bit multiplicand and the second multiplicand in a partial product generator in the multiplier; and adding said plurality of partial products in a multiplier array in the multiplier; select an adder-tree mode; select first and second operands from a 2n-bit word; bypass the partial product generator and add said two first and second operands in a first one of the ALUs in the multiplier array; and output a sum from the multiplier array.

DESCRIPTION OF DRAWINGS

FIG. 1 is a schematic diagram of a video processor including a dual mode general-purpose digital signal processor (DSP) according to one embodiment of the invention.

FIG. 2 is a schematic diagram of the execution unit shown in FIG. 1.

FIG. 3 is a flowchart showing the general steps the sequencer in FIG. 1 follows to implement an adder-tree mode in the execution unit.

FIG. 4 is a flowchart showing the general steps the sequencer in FIG. 1 follows to implement a multiplier mode in the execution unit.

FIG. 5 is a schematic diagram of another adder-tree configuration in the execution unit of FIG. 1.

FIG. 6 is a schematic diagram of an adder-tree configuration in an execution unit according to another embodiment of the invention.

Like reference symbols in the various drawings indicate like elements.

DETAILED DESCRIPTION

According to an embodiment, an execution unit 10 for a general-purpose digital signal processor (DSP) 12 may be operable in at least two modes. In a "normal" multiplier mode, the execution unit may operate as a 16 X 16 multiplier. In an adder-tree mode, the execution unit may operate as an 8-bit adder tree for adding multiple operands.

The adder-tree mode is suitable for performing computation intensive operations on relatively lower bit-width data streams, such as 8-bit, that can be used in processing video information under the H.263 and MPEG

family of video compression formats. According to the present embodiment, the general purpose DSP 12 is part of a video processor 5 including a central processing unit (CPU) 7 and a memory device 9 for storing video processing algorithms and video data.

According to the present embodiment, the adder-tree structure of the adder-tree mode may be constructed by re-utilizing pre-existing arithmetic-logic units (ALUs) in the 16X16 multiplier architecture common to conventional general-purpose DSPs. This provides a dual mode of operation in the execution unit with less additional hardware.

FIG. 2 illustrates the dual mode execution unit 10 in greater detail. Execution unit 10 may include two primary input multiplexers (MUXs) 14, two secondary input MUXs 16, a partial product generator 18, a multiplier array 20, two intermediate registers 22, two intermediate MUXs 24, a vector merging ALU 26, an accumulator 28, an accumulator MUX 30, an output register 32, and an output MUX 34 which outputs the sum produced by the execution unit to a result bus 36. However, the scope of the present invention is not limited to embodiments having all the elements shown in FIG. 2.

Interaction between the components of execution unit 10 will be described in connection with operation of the execution unit in the various operating modes.

Logical control of the execution unit 10 may be accomplished by a sequencer 100 (FIG. 1). Sequencer 100 fetches instructions from an instruction cache 102 and issues control signals to enable necessary components for operation in the respective modes. The instructions in the instruction cache 102 may be programmed or hardwired.

In both multiplier and adder-tree modes, execution unit 10 receives two pairs of 16-bit words from operand bus 104 under control of bus controller 106. Each pair of 16-bit words is input to one of primary MUXs 14. Primary MUXs 14 are controlled by sequencer 100 to select and output a single 16-bit word.

FIG. 3 is a flow chart illustrating the general steps sequencer 100 follows to implement the adder-tree mode.

Sequencer 100 controls switches 50 to output the 16-bit words output from primary MUXs 14 to secondary MUXs 16 at block 200. Each secondary MUX is controlled to select, at blocks 202 and 204, two 8-bit words, operands A and B, and C and D, respectively, to be added in multiplier array 20.

Multiplier array 20 includes a number of interconnected ALUs 60 arranged in several levels. In a typical 16 X 16 multiplier, the multiplier array has 7 levels of ALUs.

In binary addition, adding two n-bit numbers result in a maximum (n+1)-bit number, the additional bit representing a new binary place, $2^{(n+1)}$. For example, adding two 8-bit numbers yields a 9-bit sum, as shown in the following equation:

$$\begin{aligned} 11111111_2 &= 255_{10} \\ \underline{+11111111_2} &= 255_{10} \\ 111111110_2 &= 510_{10} \end{aligned}$$

8-bit operands A and B may be input directly to ALU 62 at block 206 and added to produce 9-bit partial sum E at block 208. Likewise, operands C and D may be directly input to ALU 64 at block 210 and added to produce 9-bit partial sum F at block 212.

9-bit partial sums E and F may be output to ALU 66 at blocks 214 and 216 and added to produce 10-bit sum G at block 218. Sum G is output from the multiplier array 20 at block 220.

According to one embodiment, sequencer 100 controls execution unit 10 to bypass multiplier architecture downstream of the multiplier array 20 (e.g., ALU 26 and accumulator 28), and output sum G directly to output MUX 34. This embodiment may be beneficial for video encoding operations such as spatial interpolation algorithms which utilize simple sums.

According to an alternate embodiment, sum G is fed through intermediate register 22, intermediate MUX 24, and vector merging ALU 26 to accumulator 28 where it is stored. Sum G is then fed back to accumulator MUX 30 and input to vector merging ALU 26 with 10-bit sum G' from a subsequent adding operation. The sum of G and G' may be output and stored in accumulator 28. Subsequent 10-bit sums are added to the value in the accumulator and this cycle may be repeated numerous times. This embodiment may be beneficial for video encoding operations such as motion search algorithms which utilize accumulation operations.

FIG. 4 is a flowchart illustrating the general instructions that the sequencer 100 follows to implement the multiplier mode. Sequencer 100 controls switches 50 to bypass secondary MUXs and output the 16-bit words to partial product generator as multiplicand I and J at block

300. The partial product generator 18 may comprise a matrix of AND logic gates, each operating on one bit of 16-bit multiplicand I and one bit of 16-bit multiplicand J. The partial products, sixteen for 16-bit multiplicands, may be fed to multiplier array 20 at block 302. They are reduced at block 304 to form two 32-bit words, sum X and carry Y, the sum of which equal the sum of the partial products. X and Y may be fed to intermediate registers 22, through intermediate MUXs 24 to be added in vector merging ALU 26 to produce product Z. Z may be output to result bus 36 through output register 32 and output MUX 34, or subject to an accumulation operation through accumulator 28 as described above in relation to the adder-tree mode. Preferably, vector merging ALU 26 is a 40-bit ALU to accommodate for overflow during multiplication and accumulation operations.

Other adder-tree structures can be implemented using ALUs available in the multiplier array. According to another embodiment, an execution unit having the same construction as that shown in FIG. 5 may be implemented as dual 2-operand adder-trees.

According to this particular embodiment, execution unit 10 is controlled to bypass ALU 66. Each of the 9-bit

sums E and F are output directly from multiplier array to the output MUX 34 or to intermediate registers 22 for accumulation operations as described above.

The sequencer 100 follows the same general steps shown in FIG. 3, except outputs sums S and F after steps 208 and 212.

FIG. 6 illustrates yet another embodiment in which a more complex adder-tree structure is implemented in the multiplier array. A secondary MUX 16 is provided for each 16-bit input.

Left adder-tree structure 1 in FIG. 6 is described. It is understood that right adder-tree structure 2 has the same construction and operation. The sequencer 100 follows the same general steps shown in FIG. 3 for each adder-tree structure.

Each secondary MUX 16 selects two 8-bit operands from 16-bit word input from operand bus 102 (FIG. 1) to produce operands A_1, B_1, C_1, and D_1. A_1 and B_1 are added in ALU to produce 9-bit partial sum E_1. Likewise, C_1 and D_1 are added in ALU to produce 9-bit partial sum F_1. E_1 and F_1 are added in ALU to produce 10-bit sum G_1.

Alternately, sum G_1 and G_2 (from right adder-tree structure) may be fed directly to output MUX 34 for output

to result bus 36, or alternatively, to vector merging ALU 26 for another addition operation and/or accumulator operation.

A general purpose DSP incorporating an execution unit according to the various embodiments has advantages. Such a DSP may be capable of providing performance comparable to that of an ASIC intended to support motion search algorithms with the added benefit of the flexibility of a general-purpose DSP. This provides increased flexibility to the customer for implementing a wide variety of motion search and 8-bit image processing algorithms with a general purpose 16-bit machine.

Such a general-purpose DSP may be implemented in such a way that the adder-tree mode can be easily disabled for products which are not intended for video processing.

Multiplication is used in digital signal processing for several applications, but may be the slowest operation performed by the DSP. Motion search and spatial filtering algorithms performed in digital video processing under the MPEG and H.263 standards are computation intensive, but involve relatively simple arithmetic on low bit-width numbers. Embodiments of the dual mode execution unit of the present invention advantageously accommodate both

operations using the same component, the multiplier array. The system may use lower complexity and less silicon surface area as compared with an ASIC solution.

The embodiments described in relation to FIGs. 5 and 6 including parallel adder-tree structures are particularly well suited to operations requiring bilinear interpolations of two-dimensional image based data structures. Such bilinear operations are used in, for example, spatial low-pass filtering operations, color interpolation, and fractional pixel motion searches.

The adder tree according to the invention may also be useful for low power applications such as sum-of-absolute difference (SAD) calculations. SAD is a distortion measure used to test between two adjacent frames in a video sequence to determine a best fit. SAD calculations are commonly used in motion search operations.

As described above, a dual mode DSP including an execution unit according to the various embodiments are particularly well suited for use in video processing imaging equipment which utilize MPEG-1/MPEG-2/MPEG-4/H.263 and currently unknown but contemplated future standards for video compression.

Such a general-purpose DSP is contemplated for use in video camcorders, teleconferencing, PC video cards, and HDTV. In addition, the general-purpose DSP is also contemplated for use in connection with other technologies utilizing digital signal processing such as voice processing used in mobile telephony, speech recognition, and other applications.

A number of embodiments of the invention have been described. Nevertheless, it will be understood that various modifications may be made without departing from the spirit and scope of the invention. Accordingly, other embodiments are within the scope of the following claims.

For example, while this application describes dividing in half, it should be understood that any kind of division by "x", where x could be 2 or any other number, is complicated. Also, the use of 16 and 8 bits was only illustrated. The scope of the present invention is not limited to the number of bits in any operand.

WHAT IS CLAIMED IS:

1. An apparatus comprising:

a decoder to decode an instruction specifying an adder-tree operation;

a circuit coupled to the decoder and comprising:

a multiplier including a partial product generator, and

a multiplier array comprising a plurality of arithmetic logic units; and

a selection circuit to select first and second operands from a 2n-bit word, wherein said first and second operands have fewer bits than 2n, and

said selection circuit is adapted to bypass the partial product generator and add said first and second operands in a selected one of said ALUs in the multiplier array to produce a first sum.

2. The apparatus according to claim 1 wherein said first and second operands are n-bit words.

3. The apparatus according to claim 1 wherein n is 8.

4. The apparatus according to claim 1 wherein said selection circuit is adapted to operate in response to said instruction to,

select a third operand and a fourth operand from a second 2n-bit word, wherein said first and second operands have fewer bits than 2n, and

bypass the partial product generator and add said third and fourth n-bit operands in a second selected one of said ALUs to produce a second sum.

5. The apparatus of claim 4 wherein said third and fourth operands are n-bit words.

6. The apparatus of claim 4 wherein said selection circuit is adapted to operate in response to said instruction to add said first sum

and said second sum in a third selected one of said ALUs to produce a third sum.

7. The apparatus of claim 1 wherein said selection circuit is adapted to operate in response to an instruction specifying a multiplier mode to

select first and second 2n-bit multiplicands, and

multiply said first and second 2n-bit multiplicands to produce a 4n-bit product.

8. The apparatus according to claim 1 wherein the operand selector includes a multiplexer.

9. A video processor comprising:

a central processing unit (CPU);

a memory to store instructions to perform video encoding operations;

a digital signal processor coupled to said CPU and said memory and including an execution unit comprising:

a decoder to decode an instruction specifying an adder-tree operation; and

a first circuit coupled to the decoder, and comprising

a multiplier including a partial product generator, and

a multiplier array comprising a plurality of arithmetic logic units (ALUs),

said first circuit including an element that, in response to said instructions, selects first and second n-bit operands from a 2n-bit word, and bypassing the partial product generator adds said first and second n-bit operands in a selected one of the ALUs in the multiplier array to produce a first sum.

10. The video processor of claim 9 wherein n equals 8.

11. The video processor of claim 9 wherein said first circuit operates in response to an instruction specifying a multiplier mode to

select first and second 2n-bit multiplicands, and

multiply said two 2n-bit multiplicands to produce a product.

12. The video processor of claim 9 further comprising:

a first multiplicand selector to select said first multiplicand from at least one first 2n-bit word;

a second multiplicand selector to select said second multiplicand from at least one second 2n-bit word;

a partial product generator to produce a plurality of partial sums from said first and second 2n-bit multiplicands; and

an adder to add two 4n-bit words produced in the multiplier array by reducing said partial products.

13. The video processor of claim 9 wherein the instructions to perform video encoding operations conform to MPEG standards.

14. The video processor of claim 9 wherein the instructions to perform video encoding operations comprise instructions to perform motion search algorithms.

15. The video processor of claim 9 wherein the instructions to perform video encoding operations comprise instructions to perform spatial interpolation algorithms.

16. A method comprising:

selecting a multiplier mode;

selecting a first 2n-bit multiplicand from at least one first 2n-bit word;

selecting a second 2n-bit multiplicand from at least one second 2n-bit word; and

multiplying said first and second multiplicands in a multiplier comprising a plurality of ALUs, to produce a product, said multiplying comprising:

generating a plurality of partial products from the first 2n-bit multiplicand and the second multiplicand in a partial product generator in the multiplier; and

adding said plurality of partial products in a multiplier array in the multiplier;

selecting an adder-tree mode;

selecting first and second operands from a 2n-bit word;

bypassing the partial product generator and adding said two first and second operands in a first one of the ALUs in the multiplier array; and

outputting a sum from the multiplier array.

17. The method of claim 16 wherein said first and second operands have less than 2n bits.

18. The method of claim 16 further comprising:

selecting third and fourth operands from a second 2n-bit word, said third and fourth operands having less than 2n bits;

adding said third and fourth words in a second one of the ALUs in the multiplier array;

adding a first sum from said first ALU and a second sum from said second ALU in a third ALU in the multiplier array.

19. The method of claim 18 wherein said third and fourth operands are n-bit words.

20. The method of claim 16 wherein n is 8.

21. The method of claim 16 further comprising:

performing a motion search algorithm in the adder-tree mode.

22. The method of claim 16 further comprising:

performing a spatial interpolation algorithm in the adder-tree mode.

23. The method of claim 16 further comprising:

performing a spatial low-pass filtering algorithm in the adder-tree mode.

24. A program storage device readable by a machine, comprising instructions that cause the machine to:

select a multiplier mode;

select a first 2n-bit multiplicand from at least one first 2n-bit word;

select a second 2n-bit multiplicand from at least one second 2n-bit word; and

multiply said first and second multiplicands in a multiplier comprising a plurality of ALUs, to produce a product, said multiplying comprising:

generating a plurality of partial products from the first 2n-bit multiplicand and the second multiplicand in a partial product generator in the multiplier; and

adding said plurality of partial products in a multiplier array in the multiplier;

select an adder-tree mode;

select first and second operands from a 2n-bit word;

bypass the partial product generator and add said two first and second operands in a first one of the ALUs in the multiplier array; and

output a sum from the multiplier array.

25. The program storage device of claim 25 wherein said first and second operands are n-bit words.