PROCESSOR AND METHOD OF GROUPING AND EXECUTING DEPENDENT INSTRUCTIONS IN A PACKET

BACKGROUND

[0001] I. Field of the Invention

[0002] The present invention pertains generally to processors, and more specifically to a processor and method of grouping and executing dependent instructions in a packet.

[0003] II. Background

[0004] Digital signal processors are specialized processors that are capable of executing mathematical operations with speed. Digital signal processors (DSPs) may be used in image processing, audio processing, video processing, and many other applications. Digital signal processors are commonly embedded in a variety of devices such as mobile telephones, personal digital assistants, cameras, video cameras, and portable computing systems. To increase execution speed, some digital signal processors have an interleaved multithreading architecture to support concurrent execution of multiple hardware threads. Instructions from multiple threads are interleaved in the execution pipeline. This architecture enables the use of aggressive clock frequency while maintaining high core and memory utilization.

SUMMARY

[0005] Due to the interleaved multithreading architecture of an embodiment of a digital signal processor, the execution of each thread has sufficient time to permit instructions that are dependent on one another to be grouped in the same instruction packet. Therefore, an instruction that requires another instruction to be executed to generate its result and uses it in its own execution may be packaged into the same instruction packet as the other instruction. It may be seen that by giving program code writers the ability to have inter-dependencies among instructions in an instruction packet, more flexibility to manipulate program logic becomes possible.

[0006] In one aspect of an embodiment of the present disclosure, an interleaved multithreading pipeline operating method comprises reading an instruction packet

containing at least two instructions, steering a first instruction of the first instruction packet to a first execution unit for execution and generating a first result, steering a second instruction of the instruction packet to a second execution unit for execution using the first result and generating a second result, and storing the second result.

[0007] In another aspect of an embodiment of the present disclosure, in an interleaved multithreading digital signal processor executing a plurality of interleaved threads substantially simultaneously, a method comprises reading an instruction packet containing at least two instructions, steering a first instruction of the instruction packet to a first execution unit, reading a first register specified by the first instruction and obtaining an operand, executing the first instruction and generating a first result, storing the first result in a second register, steering a second instruction of the instruction packet to a second execution unit, reading the second register specified by the second instruction, executing the second instruction using the first result read from the second register as an operand and generating a second result, and storing the second result in a third register.

[0008] In yet another aspect of an embodiment of the present disclosure, an interleaved multithreading digital signal processor executing a plurality of interleaved threads substantially simultaneously, the digital signal processor comprises means for reading an instruction packet containing at least two instructions, means for steering a first instruction of the instruction packet to a first execution unit, means for reading a first register specified by the first instruction and obtaining an operand, means for executing the first instruction and generating a first result, means for storing the first result in a second register, means for steering a second instruction of the instruction packet to a second execution unit, means for reading the second register specified by the second instruction, means for executing the second instruction using the first result read from the second register as an operand and generating a second result, and means for storing the second result in a third register.

[0009] In yet another aspect of an embodiment of the present disclosure, a computer-readable medium having encoded thereon a method for execution in an interleaved multithreaded processor, the method comprises reading an instruction packet containing at least two instructions, reading a first register specified by the first instruction and obtaining an operand, executing the first instruction and generating a

first result, storing the first result in a second register, reading the second register specified by the second instruction, and executing the second instruction using the first result read from the second register as an operand and generating a second result.

[0010] In yet another aspect of an embodiment of the present disclosure, an interleaved multithreading digital signal processor executing a plurality of interleaved threads substantially simultaneously, the digital signal processor comprises an instruction cache operable to read an instruction packet containing at least two instructions, steering logic coupled to the instruction cache operable to steer a first instruction of the instruction packet to a first execution unit. The first execution unit is coupled to the steering logic and is operable to read a first register specified by the first instruction and obtaining an operand, execute the first instruction and generating a first result, store the first result in a second register. The steering logic is operable to steer a second instruction of the instruction packet to a second execution unit. The second execution unit is coupled to the steering logic and is operable to read the second register specified by the second instruction, execute the second instruction using the first result read from the second register as an operand and generating a second result, and store the second result in a third register.

BRIEF DESCRIPTION OF THE DRAWINGS

[0011] A more comprehensive understanding of this disclosure may be obtained by referencing the accompanying drawings when considered in conjunction with the subsequent detailed description.

[0012] FIGURE 1 is a simplified logical flow diagram of an embodiment of a processor pipeline;

[0013] FIGURE 2 is a simplified timing diagram of the interleaved multithreading execution of the processor pipeline;

[0014] FIGURE 3 is an exemplary logical flow diagram of multiple dependent instructions from the same instruction packet; and

[0015] FIGURE 4 is a diagram of an exemplary embodiment of an instruction packet format.

DESCRIPTION OF SPECIFIC EMBODIMENTS

[0016] FIGURE 1 is a simplified logical flow diagram of an embodiment of a processor pipeline 10. The processor pipeline 10 may be implemented in a digital signal processor specialized in mathematical operations or in a general purpose processor. In one embodiment, the processor pipeline 10 has an interleaved multi-threading architecture that may execute six threads substantially simultaneously. The processor pipeline 10 comprises an instruction memory unit 12 comprising six threads. Program instructions are fetched from an instruction cache 14. An instruction steering and predecode logic 16 is then operable to resolve data dependencies and resource conflicts encoded in certain predetermined bits in the instructions and direct the instructions to specific execution units. The program instructions are then read into the instruction queue 18 pending execution. In an embodiment of the digital signal processor, each program instruction is an instruction packet that may contain from one to four separate instructions. Each instruction may comprise a source field that contains an index into a register file to read a value to be used by the execution of the instruction. A register file may be thought of as an array of registers accessible by an index into the register file. In an embodiment of the processor pipeline 10, the register file may have thirty-two registers, each register being thirty-two bits long, for example. A register file read operation 20 is then made to read the operands to execute the instruction for a thread. A plurality of execution units 22, such as four in this embodiment (execution units 22a-22d) are operable to carry out the execution of the program instructions in the instruction packet. An execution unit is capable of executing one instruction. A register file write operation 24 is then performed to save the execution results.

[0017] FIGURE 2 is a simplified timing diagram of the interleaved multithreading execution. The interleaved multithreading architecture of the processor pipeline 10 enables six instructions from six threads 31-36 (T1 to T6) to be substantially simultaneously executed. In the embodiment of the digital signal processor as shown, each program instruction is an instruction packet containing four separate instructions. Each instruction packet may be executed in seven clock cycles, for example, for decoding (clock cycle C1), reading the operands from the register file (clock cycle C2), executing the four instructions (clock cycles C3-C6), and finally writing the results to the register file (clock cycle C7). While an instruction packet from the first thread is

being decoded and then the register file read for operands, the second instruction packet is being decoded. Operating in this manner, the start of the execution of each successive instruction packet lags behind the prior instruction packet by one clock cycle. Taking advantage of this execution characteristic of the processor pipeline, a previously unattainable operation is possible. Previously, it was not possible to group instructions that are dependent on one another's results in the same instruction packet, but because each thread is executed in seven clock cycles, with each instruction executing in successive clock cycles, it is possible for an instruction in a packet to use the results of another instruction in the same packet.

[0018] FIGURE 3 is an exemplary logical flow diagram of multiple dependent instructions from the same instruction packet. This simple example is used solely to illustrate the inter-dependencies that may exist between instructions in the same instruction packet. An instruction 40 (I1) in the instruction packet 31 (FIGURE 2) contains the mathematical operation R[2] = R[1] + R[3], for example, to add the contents of registers 1 to the contents of register 3 and to write the result to register 2. The contents of register 2 is then used as an operand in instruction 42 (I2) to perform the operation, R[4] = R[2] + R[6]. The contents of register 4 is then used as an operand in instruction 44 (I3) to perform R[1] = R[4] + R[3]. Thereafter, the contents of register 2 and register 1 are both used as operands in instruction 46 (I4) to perform R[5] = R[1] + R[2].

[0019] It should be noted that arithmetic instructions, scalar multiplication instructions, vector multiplication instructions, bit manipulation instructions, logical instructions, load instructions, store instructions, jump instructions, and other instructions may be grouped together in each instruction packet. The types of dependent instructions that can be grouped together into one instruction packet may be constrained by a number of factors such as resource availability. For example, if only execution units 22c and 22d are operable to perform scalar multiplication and vector multiplication instructions, then an instruction packet should only contain two such instructions. Further, such instructions should be ordered as I3 and I4 of the instruction packet and be steered to those execution units 22c and 22d.

[0020] FIGURE 4 is a diagram of an exemplary embodiment of an instruction packet format 50. Instruction packet 50 includes a packet header 51 followed by four

instructions 52-55 grouped together in the packet. The packet header 51 may include one or more control fields used to indicate whether any of the group instructions is dependent on another instruction in the same instruction packet. For example, a particular bit may be used in the packet header 51 to indicate whether instruction #0 is dependent on, i.e. gets its source data from, instruction #1 or the register file. Other manner of implementation may be used to indicate inter-dependencies of instructions in the instruction packet.

[0021] It may be seen from the foregoing that by giving program code writers the ability to have inter-dependencies among instructions in an instruction packet, more flexibility to manipulate program logic becomes possible. The embodiments described herein may be implemented in hardware, software or a combination of hardware and software. The number of instructions that may be grouped into the same instruction packet is dependent on many design and implementation considerations, and the implementation of four grouped instructions and other details are provided herein only as an example and should not be construed to limit the scope of the claims.

[0022] Although embodiments of the present disclosure have been described in detail, those skilled in the art should understand that they may make various changes, substitutions and alterations herein without departing from the spirit and scope of the present disclosure. Accordingly, all such changes, substitutions and alterations are intended to be included within the scope of the present disclosure as defined in the following claims. In the claims, means-plus-function clauses are intended to cover the structures described herein as performing the recited function and not only structural equivalents, but also equivalent structures.

WHAT IS CLAIMED IS:

1. An interleaved multithreading pipeline operating method comprising:

reading an instruction packet containing at least two instructions;

steering a first instruction of the instruction packet to a first execution unit for execution and generating a first result;

steering a second instruction of the instruction packet to a second execution unit for execution using the first result and generating a second result; and

storing the second result.

2. The method of claim 1, further comprising:

steering a third instruction of the instruction packet to a third execution unit for execution using at least one of the first and second results and generating a third result; and

storing the third result.

3. The method of claim 2, further comprising:

steering a fourth instruction of the instruction packet to a fourth execution unit for execution using at least one of the first, second and third results and generating a fourth result; and

storing the fourth result.

4. The method of claim 1, further comprising:

reading a first register specified by the first instruction to obtain a first operand;

executing the first instruction on the first operand and generating the first result;

saving the first result in a second register specified by the first instruction;

reading the second register specified by the second instruction; and

executing the second instruction using the first result read from the second register and generating the second result.

5. The method of claim 4, further comprising:

storing the second result in a third register specified by a third instruction of the instruction packet; and

executing the third instruction using the second result read from the third register and generating a third result.

6. The method of claim 5, further comprising:

storing the third result in a fourth register specified by a fourth instruction of the instruction packet; and

executing the fourth instruction using the third result read from the fourth register and generating a fourth result.

7. The method of claim 6, wherein executing the first, second, third and fourth instructions includes executing instructions selected from the group consisting of arithmetic instructions, scalar multiplication instructions, vector multiplication instructions, bit manipulation instructions, logical instructions, load instructions, store instructions, and jump instructions.

8. The method of claim 1, further comprising reading an instruction packet header of the instruction packet and determining inter-dependencies between instructions in the instruction packet.

9. In an interleaved multithreading digital signal processor executing a plurality of interleaved threads substantially simultaneously, a method comprising:

reading an instruction packet containing at least two instructions;

steering a first instruction of the instruction packet to a first execution unit;

reading a first register specified by the first instruction and obtaining an operand;

executing the first instruction and generating a first result;

storing the first result in a second register;

steering a second instruction of the instruction packet to a second execution unit;

reading the second register specified by the second instruction;

executing the second instruction using the first result read from the second register as an operand and generating a second result; and

storing the second result in a third register.

10. The method of claim 9, wherein reading a first register specified by the first instruction comprises accessing a source field of the first instruction and reading the first register indexed by a value in the source field.

11. The method of claim 9, wherein storing the first result in a second register comprises accessing a destination field of the first instruction and storing the first result in the second register indexed by a value in the destination field.

12. The method of claim 9, wherein reading the second register specified by the second instruction comprises accessing a source field of the second instruction and reading the second register indexed by a value in the source field.

13. The method of claim 9, wherein storing the second result in a third register comprises accessing a destination field of the second instruction and storing the second result in the third register indexed by a value in the destination field.

14. The method of claim 9, further comprising reading an instruction packet header of the instruction packet and determining inter-dependencies between instructions in the instruction packet.

15. An interleaved multithreading digital signal processor executing a plurality of interleaved threads substantially simultaneously, the digital signal processor comprising:

means for reading an instruction packet containing at least two instructions;

means for steering a first instruction of the instruction packet to a first execution unit;

means for reading a first register specified by the first instruction and obtaining an operand;

means for executing the first instruction and generating a first result;

means for storing the first result in a second register;

means for steering a second instruction of the instruction packet to a second execution unit;

means for reading the second register specified by the second instruction;

means for executing the second instruction using the first result read from the second register as an operand and generating a second result; and

means for storing the second result in a third register.

16. A computer-readable medium having encoded thereon a method for execution in an interleaved multithreaded processor, the method comprising:

reading an instruction packet containing at least two instructions;

reading a first register specified by the first instruction and obtaining an operand;

executing the first instruction and generating a first result;

storing the first result in a second register;

reading the second register specified by the second instruction; and

executing the second instruction using the first result read from the second register as an operand and generating a second result.

17. The method of claim 16, further comprising reading an instruction packet header of the instruction packet and determining inter-dependencies between instructions in the instruction packet.

18. An interleaved multithreading digital signal processor executing a plurality of interleaved threads substantially simultaneously, the digital signal processor comprising:

an instruction cache operable to read an instruction packet containing at least two instructions;

steering logic coupled to the instruction cache operable to steer a first instruction of the instruction packet to a first execution unit;

the first execution unit coupled to the steering logic operable to read a first register specified by the first instruction and obtaining an operand, execute the first instruction and generating a first result, store the first result in a second register;

the steering logic operable to steer a second instruction of the instruction packet to a second execution unit;

the second execution unit coupled to the steering logic operable to read the second register specified by the second instruction, execute the second instruction using the first result read from the second register as an operand and generating a second result, and store the second result in a third register.