Chapter 07: Instruction–Level Parallelism— VLIW, Vector, Array and Multithreaded Processors …

Lesson 02

VLIW Processor: Pros and Cons of VLIW

Objective

- To learn VLIW (very long instruction word) processors
- To learn pros and cons of the VLIW processing

VLIW Processors

VLIW Processors

- Very large instruction word means that program recompiled in the instruction to run sequentially without the stall in the pipeline
- Thus require that programs be recompiled for the VLIW architecture
- No need for the hardware to examine the instruction stream to determine which instructions may be executed in parallel

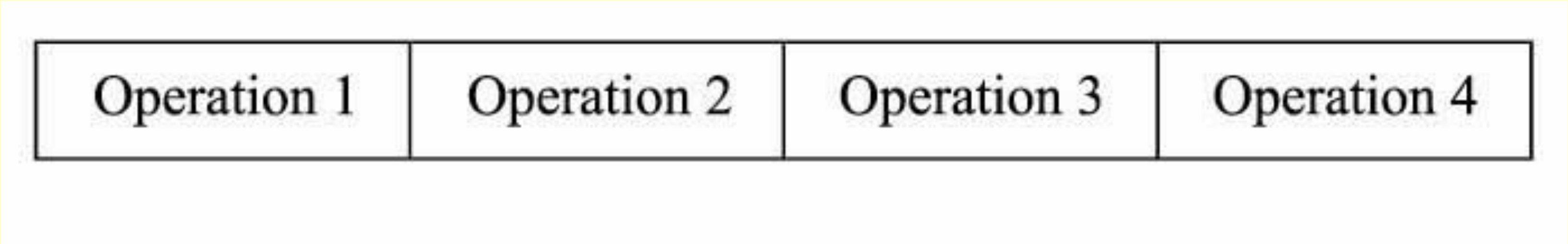
VLIW processors

- Take a different approach to instruction-level parallelism
- Relying on the compiler to determine which instructions may be executed in parallel and providing that information to the hardware
- Each instruction specifies several independent operations (called very long words) that are executed in parallel by the hardware

VLIW processors

- Compiler determining the operations (VLIWs) and which operations can be run in parallel and which are to be at which execution unit

Compiler determined the operations (VLIWs)



Operation 1	Operation 2	Operation 3	Operation 4

VLIW Processors

- Achieve very good performance on programs written in sequential languages such as *C* or FORTRAN when these programs are recompiled for a VLIW processor

Instruction Level Parallelism in VLIW

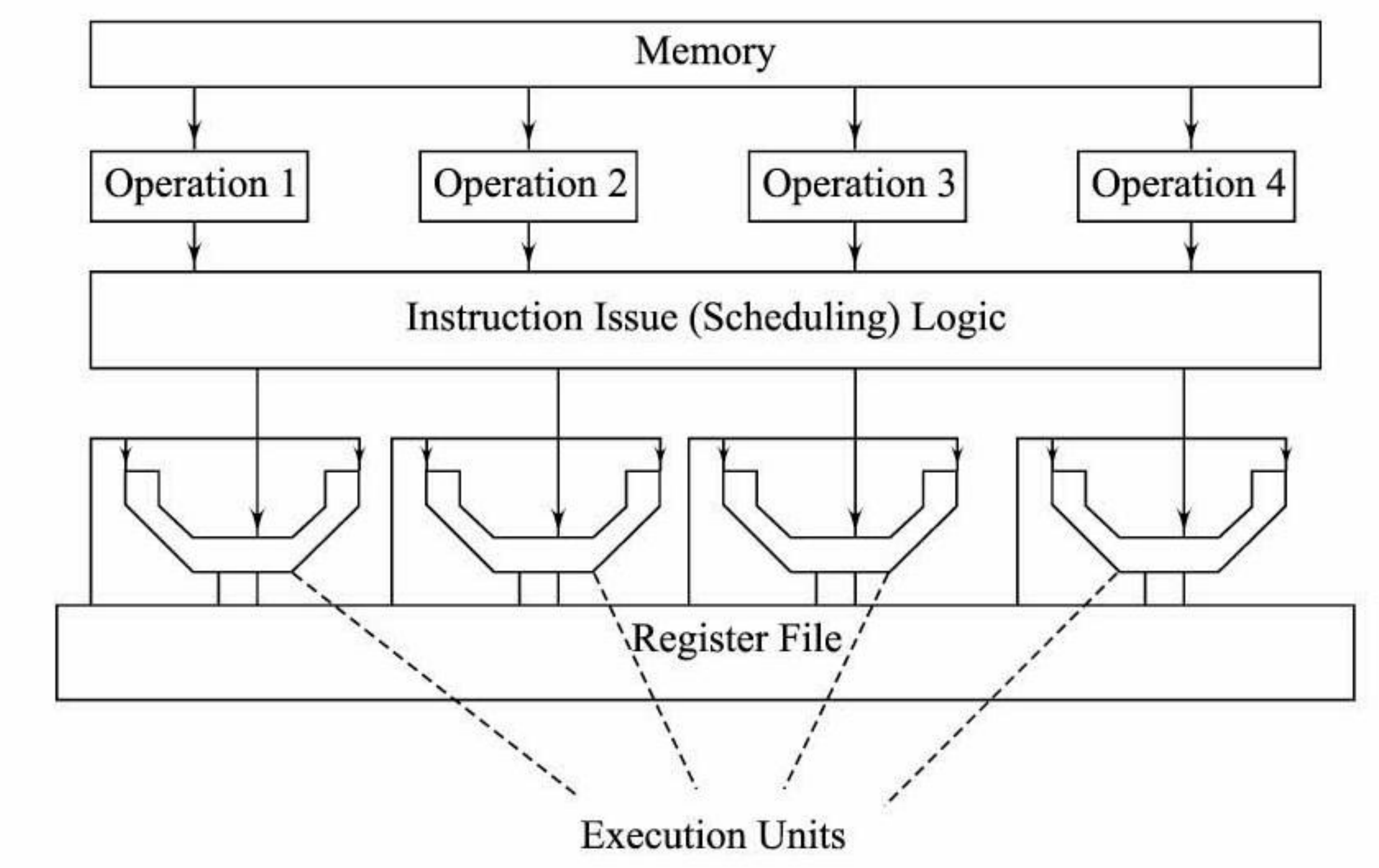
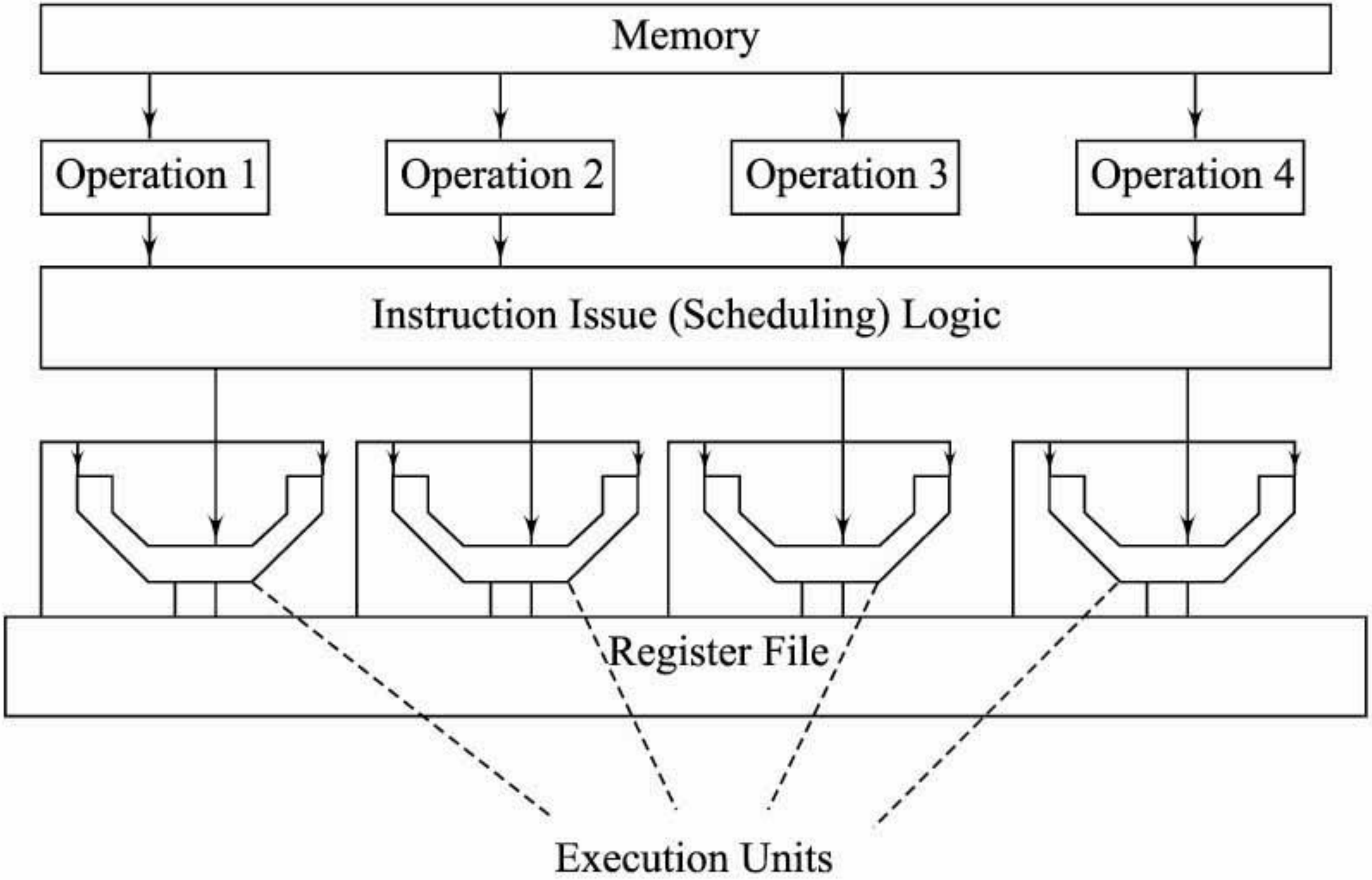


Fig. 7.3 *VLIW Instruction*



.4 *Instructions-Level Parallelism in VLIW Processors—Compiler Determining*

Each operation in a VLIW instruction

- Equivalent to one instruction in a superscalar or purely sequential processor
- The number of operations in a VLIW instruction = equal to the number of execution units in the processor
- Each operation specifies the instruction that will be executed on the corresponding execution unit in the cycle that the VLIW instruction is issued

VLIW Compiler

VLIW Processor Compiler

- The compiler responsible for ensuring that all of the operations in executing unit operation can be executed simultaneously

VLIW Processor Compiler

- Compiler can predict exactly how many cycles will elapse between the executions of two operations by counting the number of VLIW instructions between them

VLIW Processor Compiler

- Compiler can schedule instructions with a WAR dependency out of order so long as the instruction that reads the register issues before the instruction that writes the register completes, because the old value in the register is not overwritten until the writing instruction completes

Much simpler instruction issue logic

Much simpler instruction issue logic

- VLIW processor uses the same number of execution units with instruction issue logic
- Most VLIW processors do not have scoreboards on their register files
- The compiler responsible for ensuring that an operation not issued before its operands are ready

Effect of simpler instruction issue logic

- In each cycle, the instruction logic fetches a VLIW instruction from the memory and issues it to the execution units for execution

VLIW processor Applications

- VLIW processors are often used in digital signal-processing (DSP) applications, where high performance and low cost are critical

Example

Example

- Assume— VLIW processor with a two-cycle load latency
- Assume— the sequence ADD *r*1, *r*2, *r*3,
- LD *r*2, (*r*4)

Solution

- Schedule so that the ADD operation appeared in the instruction after the load
- Since the load will not overwrite *r*2 until two cycles have elapsed

Advantages of VLIW Processor

VLIW Processor Advantage

- Using compiler a larger-scale view of the program than the instruction logic in a superscalar processor
- Therefore generally better than the issue logic at finding instructions to execute in parallel

VLIW Processor Advantage

- Operations over which the compiler has complete control when operations are executed
- Simpler instruction-issue logic in them
- It results in instructions being implemented with the shorter clock cycles than superscalar processors

VLIW Processor Advantage

- Their simpler instruction issue logic also often allows VLIW processors to fit more execution units onto a given amount of chip space than superscalar processors

Disadvantages of VLIW Processor

VLIW Processor Disadvantage

- VLIW programs only work correctly when executed on a processor with the same number of execution units and the same instruction latencies as the processor they were compiled for, which makes it virtually impossible to maintain compatibility between generations of a processor family

VLIW Processor Disadvantage

- If number of execution units in a processor increases between generations, the new processor will try to combine operations from multiple instructions in each cycle, potentially causing dependent instructions to execute in the same cycle

VLIW Processor Disadvantage

- Changing instruction latencies between generations of a processor family can cause operations to execute before their inputs are ready or after their inputs have been overwritten, resulting in incorrect behavior

VLIW Processor Disadvantage

- If the compiler cannot find enough parallel operations to fill all of the slots in an instruction, it must place explicit NOP (no-operation) operations into the corresponding operation slots
- This causes VLIW programs to take more memory than equivalent programs for superscalar processors

Example of Scheduling by the Compiler

Example

- How these operations would be scheduled by compiler?

ADD *r*1, *r*2, *r*3

SUB *r*16, *r*14, *r*7

LD *r*2, (*r*4)

LD *r*14, (*r*15)

MUL *r*5, *r*1, *r*9

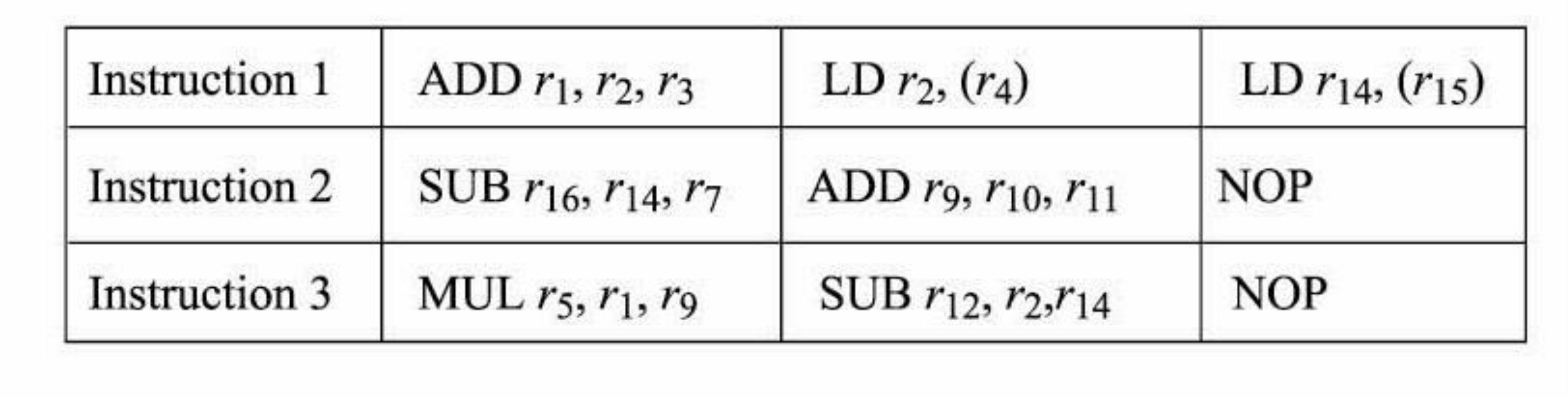
ADD *r*9, *r*10, *r*11

SUB *r*12, *r*2, *r*14

Assume

- Number of execution units = 3
- All operations has latency = 2 cycles
- Any execution unit can execute any operation

Solution



Instruction 1	ADD r_1, r_2, r_3	LD r_2, (r_4)	LD r_{14}, (r_{15})
Instruction 2	SUB r_{16}, r_{14}, r_7	ADD r_9, r_{10}, r_{11}	NOP
Instruction 3	MUL r_5, r_1, r_9	SUB r_{12}, r_2,r_{14}	NOP

… Solution

- The LD *r*14, (*r*15) scheduled in the instruction before the SUB *r*16, *r*14, *r*7 operation despite the fact that the SUB instruction appears earlier in the original program and reads the destination register of the LD

… Solution

- Because VLIW operations do not overwrite their register values until they complete, the previous value of r14 remains available until 2 cycles after the instruction containing the LD issues, allowing the SUB to see the old value of *r*14 and generate the correct result

… Solution

- Scheduling these operations out-of-order in this way allows the program to be scheduled into fewer instructions than would be possible otherwise

… Solution

- Similarly the ADD *r*9, *r*10, *r*11 operation is scheduled ahead of the MUL *r*5, *r*1, *r*9 operation, although these operations could have been placed in the same instruction without increasing the number of instructions required by the program

Summary

We learnt

- VLIW processor recompile the program into the operations
- Run the operations in parallel on different execution units
- Advantage of simpler instruction issue logic
- Faster clock cycle or more execution units

We learnt

- Advantage of complete control of operations and scheduling of instructions
- VLIW processor disadvantage difficulty of increasing more execution units

End of Lesson 03 on

VLIW Processor: Pros and Cons of VLIW