for the FPU, and four words for the I-cache and D-cache, respectively. These wide buses provide the high instruction and data bandwidths required for superscalar implementation.

The RS/6000 design is optimized to perform well in numerically intensive scientific and engineering applications, as well as in multiuser commercial environments. A number of RS/6000-based workstations and servers are produced by IBM. For example, the POWERstation 530 has a clock rate of 25 MHz with performance benchmarks reported as 34.5 MIPS and 10.9 Mflops.

■

4.2.2 The VLIW Architecture

The VLIW architecture is generalized from two well-established concepts: horizontal microcoding and superscalar processing. A typical VLIW (*very long instruction word*) machine has instruction words hundreds of bits in length. As illustrated in Fig. 4.14a, multiple functional units are used concurrently in a VLIW processor. All functional units share the use of a common large register file. The operations to be simultaneously executed by the functional units are synchronized in a VLIW instruction, say, 256 or 1024 bits per instruction word, as implemented in the Multiflow computer models.

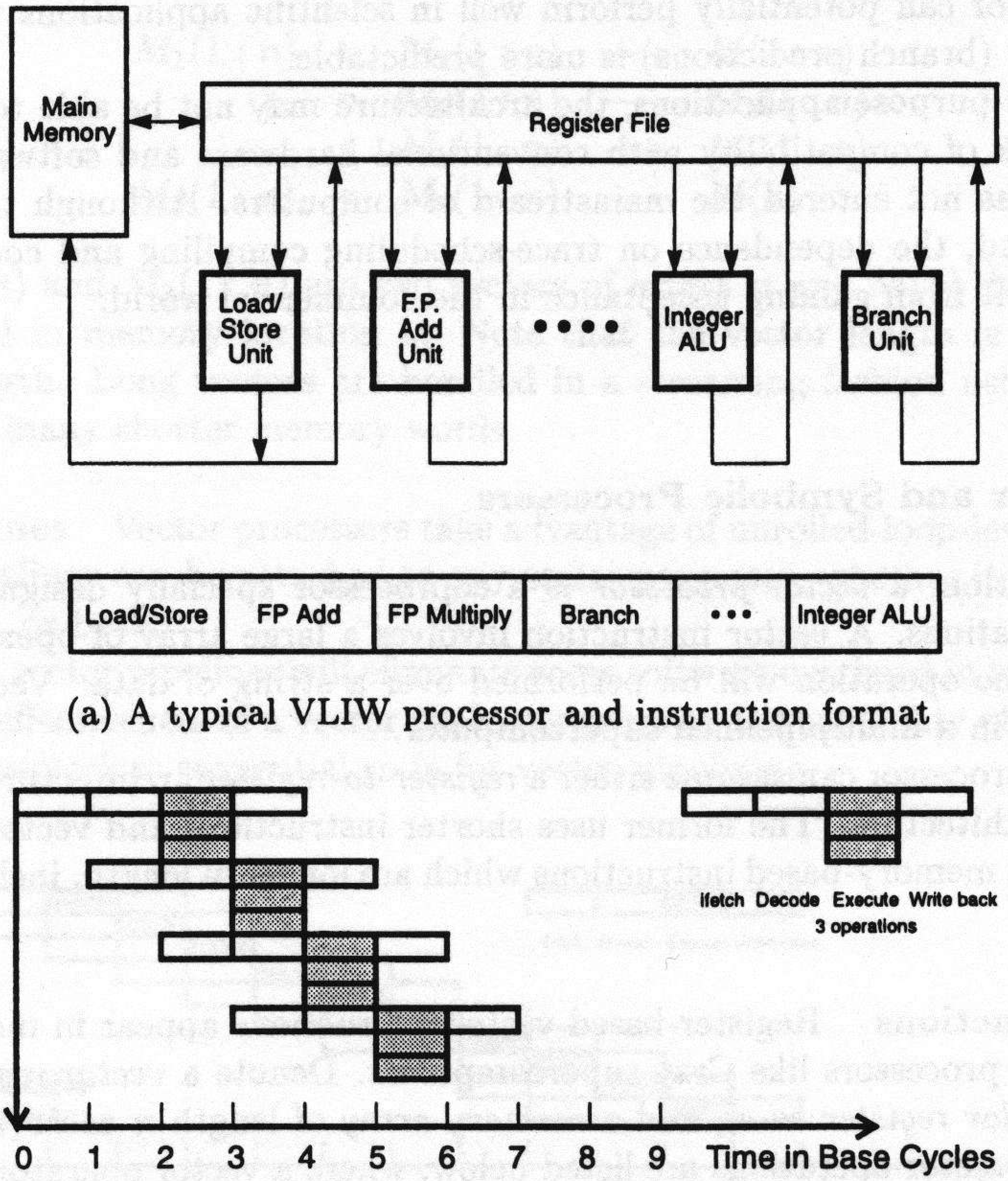
The VLIW concept is borrowed from horizontal microcoding. Different fields of the long instruction word carry the opcodes to be dispatched to different functional units. Programs written in conventional short instruction words (say 32 bits) must be compacted together to form the VLIW instructions. This code compaction must be done by a compiler which can predict branch outcomes using elaborate heuristics or run-time statistics.

Pipelining in VLIW Processors The execution of instructions by an ideal VLIW processor was shown in Fig. 4.14b. Each instruction specifies multiple operations. The effective CPI becomes 0.33 in this particular example. VLIW machines behave much like superscalar machines with three differences: First, the decoding of VLIW instructions is easier than that of superscalar instructions.

Second, the code density of the superscalar machine is better when the available instruction-level parallelism is less than that exploitable by the VLIW machine. This is because the fixed VLIW format includes bits for nonexecutable operations, while the superscalar processor issues only executable instructions.

Third, a superscalar machine can be object-code-compatible with a large family of nonparallel machines. On the contrary, a VLIW machine exploiting different amounts of parallelism would require different instruction sets.

Instruction parallelism and data movement in a VLIW architecture are completely specified at compile time. Run-time resource scheduling and synchronization are thus completely eliminated. One can view a VLIW processor as an extreme of a superscalar processor in which all independent or unrelated operations are already synchronously compacted together in advance. The CPI of a VLIW processor can be even lower than that of a superscalar processor. For example, the Multiflow trace computer allows up



(a) A typical VLIW processor and instruction format

(b) VLIW execution with degree $m = 3$

Figure 4.14 The architecture of a very long instruction word (VLIW) processor and its pipeline operations. (Courtesy of Multiflow Computer, Inc., 1987)

to seven operations to be executed concurrently with 256 bits per VLIW instruction.

VLIW Opportunities In a VLIW architecture, random parallelism among scalar operations is exploited instead of regular or synchronous parallelism as in a vectorized supercomputer or in an SIMD computer. The success of a VLIW processor depends heavily on the efficiency in code compaction. The architecture is totally incompatible with that of any conventional general-purpose processor.

Furthermore, the instruction parallelism embedded in the compacted code may require a different latency to be executed by different functional units even though the instructions are issued at the same time. Therefore, different implementations of the same VLIW architecture may not be binary-compatible with each other.

By explicitly encoding parallelism in the long instruction, a VLIW processor can eliminate the hardware or software needed to detect parallelism. The main advantage of VLIW architecture is its simplicity in hardware structure and instruction set. The VLIW processor can potentially perform well in scientific applications where the program behavior (branch predictions) is more predictable.

In general-purpose applications, the architecture may not be able to perform well. Due to its lack of compatibility with conventional hardware and software, the VLIW architecture has not entered the mainstream of computers. Although the idea is academically sound, the dependence on trace-scheduling compiling and code compaction has prevented it from gaining acceptance in the commercial world.