

Description	Name	Instruction Count $\times 10^9$	CPI	Clock cycle time (seconds $\times 10^{-9}$)	Execution Time (seconds)	Reference Time (seconds)	SPECratio
Interpreted string processing	perl	2,118	0.75	0.4	637	9,770	15.3
Block-sorting compression	bzip2	2,389	0.85	0.4	817	9,650	11.8
GNU C compiler	gcc	1,050	1.72	0.4	724	8,050	11.1
Combinatorial optimization	mcf	336	10.00	0.4	1,345	9,120	6.8
Go game (AI)	go	1,658	1.09	0.4	721	10,490	14.6
Search gene sequence	hmmer	2,783	0.80	0.4	890	9,330	10.5
Chess game (AI)	sjeng	2,176	0.96	0.4	837	12,100	14.5
Quantum computer simulation	libquantum	1,623	1.61	0.4	1,047	20,720	19.8
Video compression	h264avc	3,102	0.80	0.4	993	22,130	22.3
Discrete event simulation library	omnetpp	587	2.94	0.4	690	6,250	9.1
Games/path finding	astar	1,082	1.79	0.4	773	7,020	9.1
XML parsing	xalancbmk	1,058	2.70	0.4	1,143	6,900	6.0
Geometric Mean							11.7

FIGURE 1.20 SPECINTC2006 benchmarks running on AMD Opteron X4 model 2356 (Barcelona). As the equation on page 35 explains, execution time is the product of the three factors in this table: instruction count in billions, clocks per instruction (CPI), and clock cycle time in nanoseconds. SPECratio is simply the reference time, which is supplied by SPEC, divided by the measured execution time. The single number quoted as SPECINTC2006 is the geometric mean of the SPECratios. Figure 5.40 on page 542 shows that mcf, libquantum, omnetpp, and xalancbmk have relatively high CPIs because they have high cache miss rates.

The formula for the geometric mean is

$$n \sqrt[n]{\prod_{i=1}^n \text{Execution time ratio}_i}$$

where Execution time ratio_{*i*} is the execution time, normalized to the reference computer, for the *i*th program of a total of *n* in the workload, and

$$\prod_{i=1}^n a_i \text{ means the product } a_1 \times a_2 \times \dots \times a_n$$

SPEC Power Benchmark

Today, SPEC offers a dozen different benchmark sets designed to test a wide variety of computing environments using real applications and strictly specified execution rules and reporting requirements. The most recent is SPECpower. It reports power consumption of servers at different workload levels, divided into 10% increments, over a period of time. Figure 1.21 shows the results for a server using Barcelona.

SPECpower started with the SPEC benchmark for Java business applications (SPECJBB2005), which exercises the processors, caches, and main memory as well as the Java virtual machine, compiler, garbage collector, and pieces of the operating