

an instruction or data. An I/O module transfers data from external devices to processor and memory, and vice versa. It contains internal buffers for temporarily storing data until they can be sent on.

1.2 EVOLUTION OF THE MICROPROCESSOR

The hardware revolution that brought about desktop and handheld computing was the invention of the microprocessor, which contained a processor on a single chip. Though originally much slower than multichip processors, microprocessors have continually evolved to the point that they are now much faster for most computations due to the physics involved in moving information around in sub-nanosecond timeframes.

Not only have microprocessors become the fastest general-purpose processors available, they are now multiprocessors; each chip (called a socket) contains multiple processors (called cores), each with multiple levels of large memory caches, and multiple logical processors sharing the execution units of each core. As of 2010, it is not unusual for even a laptop to have 2 or 4 cores, each with 2 hardware threads, for a total of 4 or 8 logical processors.

Although processors provide very good performance for most forms of computing, there is increasing demand for numerical computation. Graphical Processing Units (GPUs) provide efficient computation on arrays of data using Single-Instruction Multiple Data (SIMD) techniques pioneered in supercomputers. GPUs are no longer used just for rendering advanced graphics, but they are also used for general numerical processing, such as physics simulations for games or computations on large spreadsheets. Simultaneously, the CPUs themselves are gaining the capability of operating on arrays of data—with increasingly powerful vector units integrated into the processor architecture of the x86 and AMD64 families.

Processors and GPUs are not the end of the computational story for the modern PC. Digital Signal Processors (DSPs) are also present for dealing with streaming signals such as audio or video. DSPs used to be embedded in I/O devices, like modems, but they are now becoming first-class computational devices, especially in handhelds. Other specialized computational devices (fixed function units) co-exist with the CPU to support other standard computations, such as encoding/decoding speech and video (codecs), or providing support for encryption and security.

To satisfy the requirements of handheld devices, the classic microprocessor is giving way to the System on a Chip (SoC), where not just the CPUs and caches are on the same chip, but also many of the other components of the system, such as DSPs, GPUs, I/O devices (such as radios and codecs), and main memory.

1.3 INSTRUCTION EXECUTION

A program to be executed by a processor consists of a set of instructions stored in memory. In its simplest form, instruction processing consists of two steps: The processor reads (*fetches*) instructions from memory one at a time and executes each instruction. Program execution consists of repeating the process of instruction fetch

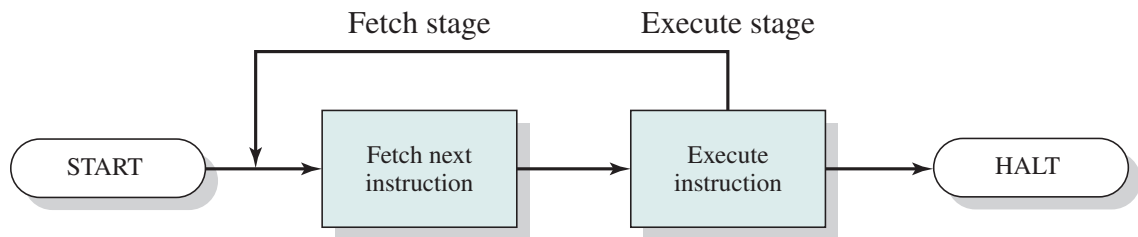


Figure 1.2 Basic Instruction Cycle

and instruction execution. Instruction execution may involve several operations and depends on the nature of the instruction.

The processing required for a single instruction is called an *instruction cycle*. Using a simplified two-step description, the instruction cycle is depicted in Figure 1.2. The two steps are referred to as the *fetch stage* and the *execute stage*. Program execution halts only if the processor is turned off, some sort of unrecoverable error occurs, or a program instruction that halts the processor is encountered.

At the beginning of each instruction cycle, the processor fetches an instruction from memory. Typically, the program counter (PC) holds the address of the next instruction to be fetched. Unless instructed otherwise, the processor always increments the PC after each instruction fetch so it will fetch the next instruction in sequence (i.e., the instruction located at the next higher memory address). For example, consider a simplified computer in which each instruction occupies one 16-bit word of memory. Assume that the program counter is set to location 300. The processor will next fetch the instruction at location 300. On succeeding instruction cycles, it will fetch instructions from locations 301, 302, 303, and so on. This sequence may be altered, as explained subsequently.

The fetched instruction is loaded into the instruction register (IR). The instruction contains bits that specify the action the processor is to take. The processor interprets the instruction and performs the required action. In general, these actions fall into four categories:

- **Processor-memory:** Data may be transferred from processor to memory, or from memory to processor.
- **Processor-I/O:** Data may be transferred to or from a peripheral device by transferring between the processor and an I/O module.
- **Data processing:** The processor may perform some arithmetic or logic operation on data.
- **Control:** An instruction may specify that the sequence of execution be altered. For example, the processor may fetch an instruction from location 149, which specifies that the next instruction be from location 182. The processor sets the program counter to 182. Thus, on the next fetch stage, the instruction will be fetched from location 182 rather than 150.

An instruction's execution may involve a combination of these actions.

Consider a simple example using a hypothetical processor that includes the characteristics listed in Figure 1.3. The processor contains a single data register, called



(a) Instruction format



(b) Integer format

Program counter (PC) = Address of instruction

Instruction register (IR) = Instruction being executed

Accumulator (AC) = Temporary storage

(c) Internal CPU registers

0001 = Load AC from memory

0010 = Store AC to memory

0101 = Add to AC from memory

(d) Partial list of opcodes

Figure 1.3 Characteristics of a Hypothetical Machine

the accumulator (AC). Both instructions and data are 16 bits long, and memory is organized as a sequence of 16-bit words. The instruction format provides 4 bits for the opcode, allowing as many as $2^4 = 16$ different opcodes (represented by a single hexadecimal¹ digit). The opcode defines the operation the processor is to perform. With the remaining 12 bits of the instruction format, up to $2^{12} = 4,096$ (4K) words of memory (denoted by three hexadecimal digits) can be directly addressed.

Figure 1.4 illustrates a partial program execution, showing the relevant portions of memory and processor registers. The program fragment shown adds the contents of the memory word at address 940 to the contents of the memory word at address 941 and stores the result in the latter location. Three instructions, which can be described as three fetch and three execute stages, are required:

1. The PC contains 300, the address of the first instruction. This instruction (the value 1940 in hexadecimal) is loaded into the IR and the PC is incremented. Note that this process involves the use of a memory address register (MAR) and a memory buffer register (MBR). For simplicity, these intermediate registers are not shown.
2. The first 4 bits (first hexadecimal digit) in the IR indicate that the AC is to be loaded from memory. The remaining 12 bits (three hexadecimal digits) specify the address, which is 940.
3. The next instruction (5941) is fetched from location 301 and the PC is incremented.

¹A basic refresher on number systems (decimal, binary, hexadecimal) can be found at the Computer Science Student Resource Site at ComputerScienceStudent.com.

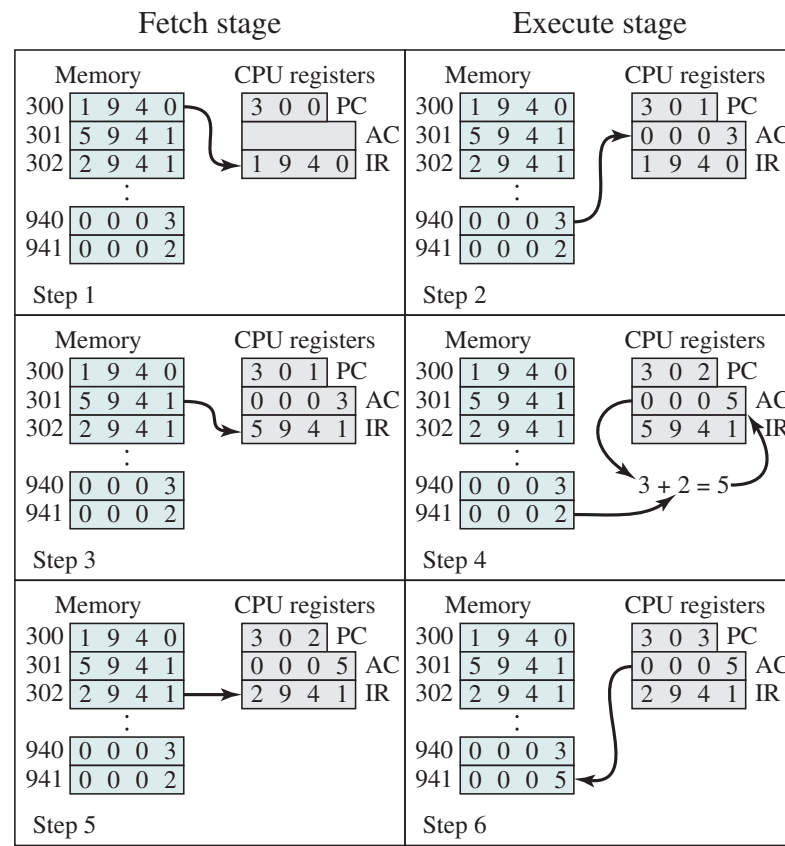


Figure 1.4 Example of Program Execution (contents of memory and registers in hexadecimal)

4. The old contents of the AC and the contents of location 941 are added, and the result is stored in the AC.
5. The next instruction (2941) is fetched from location 302, and the PC is incremented.
6. The contents of the AC are stored in location 941.

In this example, three instruction cycles, each consisting of a fetch stage and an execute stage, are needed to add the contents of location 940 to the contents of 941. With a more complex set of instructions, fewer instruction cycles would be needed. Most modern processors include instructions that contain more than one address. Thus, the execution stage for a particular instruction may involve more than one reference to memory. Also, instead of memory references, an instruction may specify an I/O operation.

1.4 INTERRUPTS

Virtually all computers provide a mechanism by which other modules (I/O, memory) may interrupt the normal sequencing of the processor. Table 1.1 lists the most common classes of interrupts.