教育部高职高专规划教材
高职高专计算机系列规划教材

计算机专业英语

（第2版）

卜艳萍 主编　　周 伟　赵 蕾 副主编　　冯世常 主审

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

教育部高职高专规划教材

高职高专计算机系列规划教材

计算机专业英语

（第2版）

卜艳萍　主编

周　伟　赵　蕾　副主编

冯世常　主审

電子工業出版社

Publishing House of Electronics Industry

北京·BEIJING

内容简介

“计算机专业英语”是高职高专计算机应用类专业学生的基础课，是提高学生计算机知识和英语综合运用能力的课程。本教材采用最新的计算机专业技术资料，涵盖计算机硬件和软件技术、计算机应用技术及计算机网络技术等专业知识。

本书共4个单元，第1单元介绍计算机硬件知识，包括计算机硬件基础、处理器、存储器、输入/输出技术，以及计算机组成部件。第2单元是计算机软件知识，内容有C语言、操作系统、数据结构、编译原理及软件工程。第3单元是计算机应用方面的知识，内容有数据库技术、计算机图形学、多媒体技术、CAD及ERP知识介绍。第4单元是计算机网络相关知识介绍，包括计算机网络、互联网、远程教育、电子商务和网络安全。

本书适合作为计算机应用及相关电类专业高职高专学生使用，也可供计算机专业技术人员学习和参考。

图书在版编目（CIP）数据

计算机专业英语/卜艳萍主编. —2版. —北京：电子工业出版社，2005.11
(高职高专计算机系列规划教材)
ISBN 7-121-01904-3

Ⅰ. 计… Ⅱ. 卜… Ⅲ. 电子计算机—英语—高等学校：技术学校—教材 Ⅳ. H31

中国版本图书馆CIP数据核字（2005）第126325号

责任编辑：吕 迈 lumai@phei.com.cn 特约编辑：王昌铭
印 刷：北京市海淀区四季青印刷厂
出版发行：电子工业出版社
北京市海淀区万寿路173信箱 邮编 100036
经 销：各地新华书店
开 本：787×1092 1/16 印张：14 字数：358
印 次：2005年11月第1次印刷
印 数：3000册 定价：19.00元

凡购买电子工业出版社的图书，如有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系。联系电话：（010）68279077。质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

出 版 说 明

教材建设工作是整个高职高专教育教学工作中的重要组成部分。改革开放以来，在各级教育行政部门、学校和有关出版社的共同努力下，各地已出版了一批高职高专教育教材。但从整体上看，具有高职高专教育特色的教材极其匮乏，不少院校尚在借用本科或中专教材，教材建设仍落后于高职高专教育的发展需要。为此，1999 年教育部组织制定了《高职高专教育基础课程教学基本要求》（以下简称《基本要求》）和《高职高专教育专业人才培养目标及规格》（以下简称《培养规格》），通过推荐、招标及遴选，组织了一批学术水平高、教学经验丰富、实践能力强的教师，成立了“教育部高职高专规划教材”编写队伍，并在有关出版社的积极配合下，推出一批“教育部高职高专规划教材”。

“教育部高职高专规划教材”计划出版 500 种，用 5 年左右时间完成。出版后的教材将覆盖高职高专教育的基础课程和主干专业课程。计划先用 2～3 年的时间，在继承原有高职、高专和成人高等学校教材建设成果的基础上，充分汲取近几年来各类学校在探索培养技术应用性专门人才方面取得的成功经验，解决好新形势下高职高专教育教材的有无问题；然后再用 2～3 年的时间，在《新世纪高职高专教育人才培养模式和教学内容体系改革与建设项目计划》立项研究的基础上，通过研究、改革和建设，推出一大批教育部高职高专教育教材，从而形成优化配套的高职高专教育教材体系。

“教育部高职高专规划教材”是按照《基本要求》和《培养规格》的要求，充分汲取高职、高专和成人高等学校在探索培养技术应用性专门人才方面取得的成功经验和教学成果编写而成的，适用于高等职业学校、高等专科学校、成人高校及本科院校举办的二级职业技术学院和民办高校使用。

教育部高等教育司

2000 年 4 月 3 日

前　言

“计算机专业英语”是面向高职高专院校计算机应用及相关专业学生的。本书力图采用最新的计算机专业技术资料，涵盖计算机专业软、硬件基础知识、计算机应用、多媒体技术、网络技术等专业知识。通过本课程的学习，使学生掌握较多的专业词汇和基本概念，为阅读计算机专业文献和书籍打下较好的基础。

本教材有以下几个方面的特点：(1)计算机专业知识丰富；(2)注重实践性和实用性；(3) 阅读材料难度适中，强调理解及分析；(4) 每章配有关键词、注释及大量习题；(5) 注意与其他计算机技术专业课程内容的衔接与知识补充；(6)书后附有所有专业文章的参考译文和习题答案；(7) 附录列出了常用的计算机专业词汇和缩写词表。

本书由上海交通大学技术学院卜艳萍主编，华东理工大学周伟博士，山西财经大学经济技术学院赵蕾副主编，冯世常主审。卜艳萍编写 Unit 1 及对全书统稿、校对；周伟编写 Unit 3 中的 3.1、3.2、3.3 节及 Unit 4；赵蕾编写 Unit 2、Unit 3 中的 3.4、3.5 节及附录 A、B。何飞、燕慧、周风波、刘雅琴等参加了资料的整理工作，周烨晴、邱尧同学帮助录入了部分书稿，在此一并表示感谢。

由于编者水平有限，不当之处敬请同行批评指正。主编 E-mail：yanpingbu@sina.com。

编　者

2005 年 6 月于上海

目　　录

Unit 1　Hardware Knowledge

1.1　Hardware Basics

1.1.1　Text

Logic Circuits

The design of digital computers is based on a logical methodology called Boolean Algebra which uses three basic operations: logical addition, called the OR function; logical multiplication, called the AND function; and logical complementation, called the NOT function.[1] The variables in Boolean Algebra are binary, namely, the resulting variable of an operation or a set of operations can have only one of the two values: One or Zero. These two values may also be interpreted as being True or False, Yes or No, and Positive or Negative. A switch is ideally suited to represent the value of any two-state variable because it can only be "off " or "on".

The OR circuit has two or more inputs and a single output. The inputs and the output can each be at one of two states, 0 or 1. The OR circuit is arranged so that the output is in state 1 when any one of the inputs is in state 1.[2] We can list the various combinations of switch states and the resulting output states. This list is known as a truth table, from which it can be seen that all switches must be open (0 state) for the output 0 state. This type of circuits is called an OR gate. It is used to make the logic decision on whether or not at least one of several inputs is in the 1 state.

The AND circuits also has several inputs and only one output. However, the circuit output is at a logical 1 state only if all inputs are in the logical 1 state simultaneously. The AND gate makes the logic decision on whether or not several inputs are all in the 1 state at the same time.

The NOT circuit has a single input and a single output. It is arranged so that the output state is always opposite to the input state. The operation of making the output state opposite to that of the input is called inversion, and a circuit designed to do this is known as an inverter.

Combinational Circuit

A combinational circuit is a connected arrangement of logic gates with a set of inputs and outputs. At any given time, the binary values of the outputs are a function of the binary combination of the inputs. A block diagram of a combinational circuit is shown in Fig. 1-1. The n binary input variables come from an external source, the m binary output variables go to an external destination, and in between there is an interconnection of logic gates. A combinational circuit transforms binary information from the given input data to the required output data.

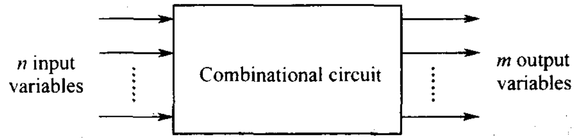

Fig. 1-1 Block diagram of a combinational circuit

A combinational circuit can be described by a truth table showing the binary relationship between the n input variables and the m output variables. The truth table lists the corresponding output binary values for each of 2^n input combinations. A combinational circuit can also be specified with m Boolean functions, one for each output variable. Each output function is expressed in terms of the n input variables.

Flip-Flop and Clock

Microprocessors employ both latches and flip-flops. The basic RS latch, or the basic D latch, is not a flip-flop because it is an asynchronous device (it is un-clocked). That is, a latch functions at arbitrary times, whenever data pulses may be inputted. On the other hand, we will see that a flip-flop is a synchronous device; it is clocked, and it can change state only on arrival of a clock pulse. Clock pulses are basically square waves; they may have a very low repetition, or they may have a very high repetition rate.

Note that the simple arrangement depicted in Fig. 1-2 operates as a flip-flop, in as much as the RS latch function is locked in step with the clock input. This is active-low configuration; the R and S outputs can be complemented only while the clock is logic-low. A clock input is a steady square-wave signal.

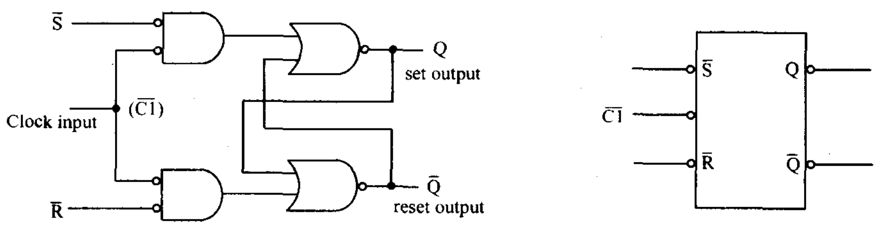

Fig. 1-2 RS flip-flop

Shift registers

A register capable of shifting its binary information in one or both directions is called a shift register. The logical configuration of a shift register consists of a chain of flip-flops in cascade, with the output of one flip-flop connected to the input of the next flip-flop. [3] All flip-flops receive common clock pulses that initiate the shift from one stage to the next.

The simplest possible shift register is one that uses only flip-flops, as shown in Fig. 1-3. The output of a given flip-flop is connected to the D input of the flip-flop at its right. The clock is common to all flip-flops. The serial input determines what goes into the leftmost position during the shift. The serial output is taken from the output of the rightmost flip-flop.

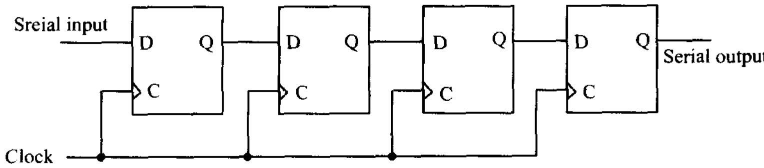

Fig. 1-3 4-bit shift register

Sometimes it is necessary to control the shift so that it occurs with certain clock pulses but not with others. [4] This can be done by inhibiting the clock from the input of the registers if we do not want it to shift. When the shift register of Fig. 1-3 is used, the shift can be controlled by connecting the clock to the input of an AND gate, and a second input of the AND gate can then control the shift by inhibiting the clock.

A register capable of shifting in one direction only is called a unidirectional shift register. A register that can shift in both directions is called a bi-directional shift register. Some shift registers provide the necessary input terminals for parallel transfer. The most general shift register has all the capabilities listed below. Others may have some of these capabilities, with at least one shift operation.

- An input for clock pulses to synchronize all operations.
- A shift-right operation and a serial input line associated with the shift-right.
- A shift-left operation and a serial input line associated with the shift-left.
- A parallel load operation and *n* input lines associated with the parallel transfer.
- *N* parallel output lines.
- A control state that leaves the information in the register unchanged even though clock pulses are applied continuously.

Key Words

arbitrary	任意的
asynchronous	异步的
bi-directional	双向的
Boolean algebra	布尔代数
Boolean function	布尔函数
cascade	级连的，串级
clock pulse	时钟脉冲
combinational circuit	组合电路
data pulse	数据脉冲
inhibit	禁止，抑制
initiate	开始，激发
interconnection	互联
inverter	反向器，非门
logic gate	逻辑门
methodology	方法学，方法论

negative 负的，否定的
opposite 相对的，相反的
parallel transfer 并行传送
positive 正的，肯定的
serial input 顺序输入，串行输入
simultaneously 同时地
unidirectional 单向的
variable 变量，可变的

Notes

[1] The design of digital computers is based on a logical methodology called Boolean Algebra which uses three basic operations: logical addition, called the OR function; logical multiplication, called the AND function; and logical complementation, called the NOT function.

本句中的“digital computers”作定语，修饰“design”，而“which uses three basic...”是定语从句，修饰“Boolean Algebra”。

译文：数字计算机的设计基于称为布尔代数的逻辑方法学，它采用三种基本运算：逻辑加，称为“或”功能；逻辑乘，称为“与”功能，和逻辑求补，称为“非”功能。

[2] The OR circuit is arranged so that the output is in state 1 when any one of the inputs is in state 1.

句中“so that”引导目的状语，“when”引导的是时间状语从句。

译文：“或”电路被设计成当任一输入为 1 态时输出为 1 态。

[3] The logical configuration of a shift register consists of a chain of flip-flops in cascade, with the output of one flip-flop connected to the input of the next flip-flop.

此句中的“with the output of one flip-flop connected to the input of the next flip-flop”是独立结构作状语，说明触发器连接的方式。

译文：移位寄存器的逻辑结构包含一个级连的触发器链，一个触发器的输出端被连接到下一个触发器的输入端。

[4] Sometimes it is necessary to control the shift so that it occurs with certain clock pulses but not with others.

此句中的“it”是形式主语，“to control the shift”是真正主语，so that 引导状语从句。

译文：有时有必要控制移位以使它对一特定的时钟脉冲而不对其他脉冲发生。

1.1.2 Exercises

1. Translate the following phrases into English

（1）R-S 触发器

（2）向左移位操作

（3）输出变量

（4）8 位移位寄存器

（5）二进制信息

（6）反向器，非门

(7) 布尔代数

(8) 时钟脉冲

2. Translate the following phrases into Chinese

(1) digital computer

(2) truth table

(3) logic decision

(4) synchronous device

(5) square wave

(6) parallel output

(7) flip-flop

(8) gated latch

3. Identify the following to be True or False according to the text

(1) A shift-left operation and a serial input line associated with the shift-right.

(2) Clock pulses are basically square waves.

(3) A register that can shift in both directions is called a unidirectional shift register.

(4) The R and S outputs can be complemented only while the clock is logic-high.

(5) The output values are a function of the combination of the inputs in a combination circuit.

(6) The OR circuit has two or more inputs and a single output.

(7) An OR gate is used to make the logic decision on whether or not at least one of several inputs is in the 1 state.

4. Reading Comprehension

(1) ________________ may have a very low repetition, or they may have a very high repetition rate.

a. Clock pulses

b. Data

c. Input signals

d. Output signals

(2) The ________________ has a single input and a single output.

a. AND circuit

b. OR circuit

c. NOT circuit

d. Logic circuit

(3) A combinational circuit transforms ________________information from the given input data to the required output data.

a. hexadecimal

b. binary

c. octal

d. decimal

(4) The output of ____________________is at a logical 1 state only if all inputs are in the logical 1 state simultaneously.

a. an OR circuit
b. a NOT circuit
c. an AND circuit
d. Logic gates

1.1.3 Reading Material

RISC Technology

The RISC methodology is based on a top-down approach guided by a set of architectural design criteria. The major goal is to reduce the semantic gap between the hardware machine and the application level, while maximizing the execution speed. Therefore, the first step in the RISC design methodology is to make a detailed and careful analysis of application codes in order to discover the most frequent operations. This analysis must be done on a large amount of program codes related to the selected application field. Examples of the latter were mainly in the past: C programming, symbolic processing and signal processing. Then, the study of the highly frequent operations enables the designer to determine the instruction set architecture. The next step is then to verify that each instruction can fit in the basic RISC architecture model without increasing the cycle time. The real keys to enhanced performance are single-cycle execution and keeping the cycle-time as short as possible. Remember that choice of a given implementation technology and related software tools also significantly influence the design.

The following nine points are the RISC design criteria.

1. Relatively few instructions and single-cycle operations for every instruction.
2. Few addressing modes and a fixed instruction format.
3. Hardwired control without micro-code.
4. Migration of complex functions to software.
5. Only Load and Store instructions should reference external memory.
6. Each instruction must fit a pipelined model that allows parts of several operations to be processed at the same time.
7. Delayed-branches enable conditional and unconditional jumps to execute without flushing the pipeline.
8. The architecture must have at least 32 general-purpose registers and large cache memories.
9. The architecture must be tailored for a dedicated application field and must support high-level languages.

The major tenet of RISC states that much of the static runtime complexity can and should be handled prior to runtime by an optimizing compiler. Then, the resulting hardware architecture must be quite simple to ensure single-cycle execution for every instruction and small clock cycle. Note that all these features, particularly pipelining and high-performance memories, have been used in supercomputer designs for many years. Remember that the main advance of RISC was to integrate all these ideas into a single architecture model, coupled with a coherent top-down methodology.

Finally, what is the major difference between a RISC microprocessor and a CISC one? RISC refers to a new design methodology that indicates a new relationship between hardware and software, rather than a set of technical features. In fact, the effect of the RISC approach on computer design is far more than just a collection of methods to run the CPU faster. In this sense, the term "Reduced Instruction Set Computer" is somewhat misleading. The real goal of RISC design definitely is not to arbitrarily reduce the number of instructions. Rather, the main objective is to create a high-speed computer by making the software system to match the hardware. The main goal of CISC designs was to reduce the number of instructions for executing programs. Rather, RISC tries mainly to reduce the average number of clock cycles per instruction (CPI). Both architectures want also to increase the clock-rate by using high-speed technologies, however RISC provides a greater performance improvement since its reduced complexity enables it to be fabricated earlier in such technologies.

Other microprocessor architects join the RISC faithful or not, they have been influenced by the RISC methodology: as an example, the Motorola's 68030 has already acquired some of its features. The force of RISC is that it is based on a true design methodology pushed by advances in compiler technology. Within RISC, teams of hardware engineers and compiler architects work together to forge a unified design: one that will minimize the hardware complexity by integrating in hardware only such instructions that are justified by their occurrence in program trace analysis.

1.2 Central Processing Unit

1.2.1 Text

A computer can solve a series of problems and make hundreds, even thousands, of logical decisions without becoming tired or bored. It can find the solution to a problem in a fraction of the time it takes a human being to do the job. A computer can replace people in dull, routine tasks, but it has no originality; it works according to the instructions given to it and cannot exercise any value judgements. But a computer can carry out vast numbers of arithmetic/logical operations almost instantaneously.

The CPU means the Central Processing Unit. It is the heart of a computer system. The CPU in a microcomputer is actually one relatively small integrated circuit or chip. Although most CPU chips are smaller than a lens of a pair of glasses, the electronic components they contain would have filled a room a few decades ago.[1] Using advanced microelectronic techniques, manufacturers can cram tens of thousands of circuits into tiny layered silicon chips that work dependably and use less power. The CPU coordinates all the activities of the various components of the computer. It determines which operations should be carried out and in what order. The CPU can also retrieve information from memory and can store the results of manipulations back into the memory unit for later reference.

The basic job of computers is the processing of information. For this reason, computers can defined as devices which accept information in the form of instructions called a program and characters called data, perform mathematical and/or logical operations on the information, and then supply results of these operations. The program, which tells the computers what to do and the data, which provide the information needed to solve the problem, are kept inside the computer in a place called memory. [2]

Computers are thought to have many remarkable powers. However, most computers, whether large or small, have three basic capabilities.

First, computers have circuits for performing arithmetic operations, such as: addition, subtraction, multiplication, division and exponentiation.

Second, computers have a means of communicating with the user. After all, if we couldn't feed information in and get results back, these machines would not be of much use.

Third, computers have circuits which can make decisions. The kinds of decisions which computer circuits can make are of the type: Is one number less than another? Are two numbers equal? And, is one number greater than another?

A CPU can be a single microprocessor chip, a set of chips, or a box of boards of transistors, chips, wires, and connectors. Differences in CPUs distinguish mainframes, mini- and microcomputers. A processor is composed of two functional units: a control unit and an arithmetic / logic unit, and a set of special workspaces called registers.

The control unit

The control unit is the functional unit that is responsible for supervising the operation of the entire computer system. In some ways, it is analogous to a telephone switchboard with intelligence because it makes the connections between various functional units of the computer system and calls into operation each unit that is required by the program currently in operation. The control unit fetches instructions from memory and determines their type or decodes them. It then breaks each instruction into a series of simple small steps or actions. By doing this, it controls the step-by-step operation of the entire computer system.

The Arithmetic / Logic Unit

The Arithmetic / Logic Unit (ALU) is the functional unit that provides the computer with logical and computational capabilities.[3] Data are brought into the ALU by the control unit, and the ALU performs whatever arithmetic or logic operations are required to help carry out the instructions. Arithmetic operations include adding, subtracting, multiplying, and dividing. Logic operations make a comparison and take action based on the results. For example, two numbers might be compared to determine if they are not equal. If they are equal, processing will continue; if they are not equal, processing will stop.

Registers

A register is a storage location inside the processor. Registers in the control unit are used to keep track of the overall status of the program that is running. Control unit registers store information such as the current instruction, the location of the next instruction to be executed, and the operands of the instruction. In the ALU, registers store data items that are added, subtracted, multiplied, divided, and compared. Other registers store the results of arithmetic and logic