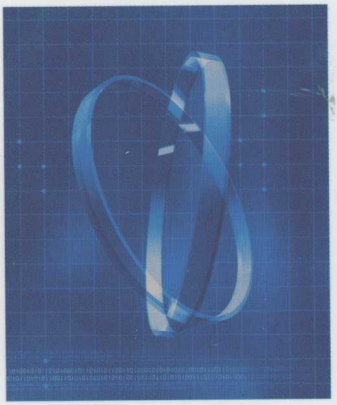

电子信息与通信专业英语

（第2版）

赵淑清　主编

English in Electronic Information and Communication

哈爾濱工業大學出版社

电子信息与通信专业英语

（第 2 版）

主　编　赵淑清

哈爾濱工業大學出版社

内 容 提 要

本书以培养学生专业英语阅读能力为主要目标。内容包括:数字电路及接收机、信号处理和数据处理信息与通信基本理论及应用、信号处理专题以及一些电子仪器、设备及部件的说明书。

本书可作为大学电子信息工程和通信工程专业三、四年级本科生的专业英语教材,也可供广大工程技术人员使用。

图书在版编目(CIP)数据

电子信息与通信专业英语/赵淑清主编.—2版.
—哈尔滨:哈尔滨工业大学出版社,2007.8
ISBN 978-7-5603-1450-1

Ⅰ.电… Ⅱ.赵… Ⅲ.①电子技术-英语-高等学校-教材②通信工程-英语-高等学校-教材 Ⅳ.H31

中国版本图书馆 CIP 数据核字(2007)第 131208 号

出版发行 哈尔滨工业大学出版社
社　　址 哈尔滨市南岗区复华四道街 10 号　邮编 150006
传　　真 0451-86414749
网　　址 http://hitpress.hit.edu.cn
印　　刷 肇东粮食印刷厂
开　　本 880mm×1230mm　1/32　印张 10.5　字数 313 千字
版　　次 2007 年 8 月第 2 版　2007 年 8 月第 7 次印刷
书　　号 ISBN 978-7-5603-1450-1
定　　价 18.00 元

前　言

本书的目的是为本科生打下良好的专业英语基础。本书从基本电子线路到计算机应用，基本覆盖了电子信息工程和通信工程专业技术基础课所学的内容。考虑到学生将来可能从事的科学研究和电子及通信设备的应用、研制和开发，还选择了一些热门研究领域的课题和一些电器设备以及 DSP 芯片或部件的说明书。

本书共有五章，第一章是数字电路和模拟电路；第二章是信号处理和数据处理，包括数字信号、数字滤波器和图像处理的内容，此外还包括数据库及一个实用软件包；第三章是信息、通信的基本理论及应用，包括语音通信、PCM、GPS 及软件无线电的内容；第四章信号处理专题包括了近年来信号处理领域中的一些热门课题；第五章收集了一些电子仪器、设备以及 DSP 芯片或部件的说明书和手册。每节后面列出一些单词，并对一些句子进行了注释。单词和注释以专业词汇和专业性较强的句子为主，主要是使读者能够正确理解书中所叙述的原理和阐述的观点。

本次修订更换了一些内容，增加了三个附录，对较长的篇幅进行了压缩，同时还对发现的错误进行了更正。为了更好帮助选择学习内容，在附录中增加了内容提要，给出了每节的主要内容及应用背景。

本书可作为电子信息工程和通信工程专业大学三、四年级学生的专业英语阅读材料。

参加本书编写的还有王大明、王若楠、张永钋、梁冰霜。在编写的过程中得到了哈尔滨工业大学电子与通信工程系的一些博士、硕士研究生的大力帮助，在此表示诚挚的感谢。由于编者水平有限，书中难免还存在一些缺点和错误，殷切希望广大读者批评指正。

编　者

2007 年 9 月于哈工大

CONTENTS

1 Digital Circuits and Analog Circuits

1.1 DIGITAL CIRCUITS ····· 1

1.2 RECEIVER ····· 19

2 Signal Processing and Data Processing

2.1 DIGITAL SIGNAL AND DISCRETE-TIME SYSTEMS ····· 35

2.2 DIGITAL FILTER ····· 52

2.3 IMAGE ENHANCEMENT ····· 67

2.4 DATA BASE ····· 84

2.5 SKYvec SOFTWARE TOOLKIT ····· 100

3 Communications

3.1 SPEECH COMMUNICATIONS ····· 118

3.2 PCM AND DIGITAL TRANSMISSION ····· 138

3.3 GLOBAL POSITIONING SYSTEM ····· 159

3.4 SOFTWARE RADIO ····· 170

4 Special Topics in Signal Processing

4.1 THE FAST ALGORITHMS ····· 182

4.2 THE LEAST MEAN SQUARES(LMS) ALGORITHM ····· 199

4.3 PATTERN CLASSIFICATION BY DISTANCE FUNCTIONS ····· 217

4.4 LEARNING ALGORITHM BY BACK-PROPAGATION ····· 228

5 Instruction Manual

5.1 MOBILE PHONE ····· 242

5.2 SPECIFICATIONS FOR SOME ELECTRICAL APPLIANCE ····· 258

5.3 SKYsystem FAMILY ····· 268
5.4 ADSP-2106x ····· 277
5.5 TMS320C5x ····· 295
Appendix A 内容提要 ····· 314
Appendix B 数学公式与运算 ····· 326
Appendix C 希腊字母表 ····· 330

1

Digital Circuits and Analog Circuits

1.1 DIGITAL CIRCUITS

Digital design can be divided into two general areas. The first is the creation and production of direct hardware from available building blocks. The second is the computer software or programming aspects, which may or may not involve the design of hardware items.

The second area employs techniques that are quite different from those of conventional hardware design and also require a substantial investment in special test equipment for efficient development and debugging. The quantity of information needed is worthy of a separate book and is not covered here.

Many aspects of computer technology, however, are very important in conventional hardware design. Among these are the programmable read-only memories referred to as PROMs and the read-write memories called RAMs (random-access memories). The word "firmware" is commonly used for these applications.

The design of digital circuits differs greatly from the design of analog circuits, being more like a systems design on a small scale. For the large

part it consists of connecting standard building blocks without the use of modifying passive components. Many circuits do, of course, contain both analog and digital parts, and the construction of these systems is a mixture of the two techniques.

1.1.1 Basic Circuits

Digital design is based on the simple concept of yes or no, true or false, high or low, and so on. Electronically the basic circuit is the NAND gate.[1] A simple version is shown in Fig.1.1.

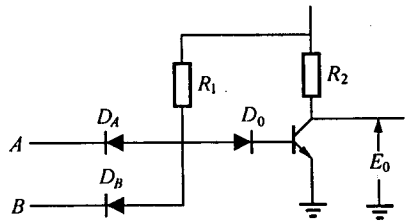

Fig.1.1 Simple NAND gate

If both A and B are open or tied to a (+) voltage, a current flows from R_1 through the base-emitter junction of the transistor, which is turned "hard-on" or put into the so-called saturation mode.[2] E_0 is then near ground potential, and its output is called a "0" or "low." If either A or B or both are grounded, the current through R_1 is diverted to that ground. This provides a voltage drop of one diode at the anode of D_0. However, a voltage equal to two diode drops, that of D_0 and the transistor, is required to turn the transistor on. With the transistor off, E_0 approaches the $V+$ level and is called a "1" or "high."

It will be noticed that an inversion is involved in this circuit. That is, if the inputs are low, the output is high, and vice versa. To make the circuit noninverting would require an extra stage. This is why digital circuitry is based on inverting logic. The word NAND is a contraction of the phrase "INVERTING AND."

With each logic element there is a "truth table" that explains how the unit works. These tables are generally in positive logic. This means that the function is described for input signals that are 1's. Negative logic is when the function is described in terms of input zeros. The use of negative logic is often confusing and is not used in this text. The truth table for the two-input NAND gate is shown in Fig.1.2.

A	*B*	Out
0	0	1
0	1	1
1	0	1
1	1	0

Fig.1.2 Truth table for a NAND gate

This table states that if both *A* and *B* are 1 (High or Hi), the output is a 0 (Low or Lo). It is sometimes easier for the beginner to think in terms of an AND gate followed by an inverter. The logic of an AND gate states that if both *A* and *B* are Hi, the output is Hi. An AND gate is really a NAND gate followed by an inverter.

If we look again at the truth table, it also says if *A* or *B* is a 0 the output is a 1. In other words, the NAND circuit does a NAND function with respect to 1's at the input and a NOR function with respect to 0's at the input. If the two inputs are tied together, the NAND circuit becomes an inverter. Fig.1.3 shows three common symbols. The small circle at the output means inverting, so that if the circles are removed the three symbols become AND, OR, and EXCLUSIVE OR respectively.

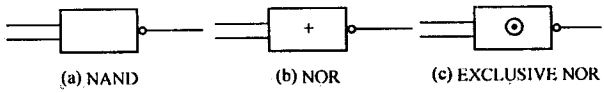

Fig.1.3 Some standard gates

The truth tables for the NOR and EXCLUSIVE NOR are shown in Fig.1.4. The NOR truth tables states that if *A* or *B* are a 1, the output

is a 0. The EXCLUSIVE NOR is the same thing except the condition that both *A* and *B* are 1's.

A	*B*	Out
0	0	1
0	1	0
1	0	0
1	1	0

(a) NOR

A	*B*	Out
0	0	1
0	1	0
1	0	0
1	1	1

(b) EX-NOR

Fig.1.4 Truth tables

From the truth table of the NAND gate we know that if one or more of the inputs to the NAND gate is a 0, the output is a 1. Inverting the 1 produces a 0, and we have a NOR gate. This is a very poor way to make a NOR gate, and not the manner in which they are made. The purpose of the example was to demonstrate a simple use of the NAND gate and the process by which more complex building blocks can be evolved.

An important difference should be noted at this time between digital and analog components. There is very little need for the circuit designer to know how a digital function is accomplished. If the job is well done, the important properties such as propagation delay, power consumption, number of leads, and the need for supporting modules will be favorable. If the performance specifications of the device are adequate, attempting to study the technique is generally a waste of time. The reason for this is that a digital device is exact. It is yes or no. This does not mean that it necessarily produces a correct result, but it does produce a defined one. The analog world of full of relative numbers and approximations, all of which depend heavily on the basic semiconductor properties.[3] These properties are both variant and different from unit to unit. The designer of the digital blocks faces the same problems, but once the digital unit has been properly designed and built, the circuit designer is largely relieved of these considerations.

When digital circuits are operated near their maximum speeds they

approach a failure mode that is largely analog in nature and all the troubles and uncertainties of the analog circuit are back.[4] High-frequency performance is specified in different ways. One common expression is "maximum toggle frequency." This means that the output is going between the logic Hi and Lo states at its fastest possible rate but without the duty cycle or rise and fall times defined. It does not mean that the device can operate properly at that speed. Just how fast the device can operate depends on an analog type of analysis of the system.

The speed limitations of a digital circuit show up in four different forms: propagation delay, setup time, rise time, and fall time.[5] Propagation delay is basically the time between a signal edge's entering a device and leaving the device. When a number of digital devices are connected in series, their propagation delays add up. When a similar set of digital devices are operating in parallel, their propagation delays, because of the tolerances, is not necessarily the same.[6] This problem is sometimes referred to as "skew." It is, of course, essential in digital circuits that signal edges occur in a known order. It is a further absolute requirement that this order preserve a minimum time between signal edges of concern. This is called setup time. But simply, a signal must remain at an input for a certain minimum amount of time or it will not be recognized.

The rise and fall times limit the response by not reaching the next logic level in time to be recognized. The rise and fall times can be somewhat controlled through good layout to reduce capacitance and inductance, by limiting the number of stages that are driven, and by the occasional use of a pull-up resistor in the output circuits.[7]

To summarize, the problems with digital circuits increase rapidly as the toggle frequencies are approached. Much difficulty is avoided if the operating frequencies are limited to one half of the minimum value and if the setup times are increased by a factor of 2 or 3 over the manufacturer's

stated minimum values.

1.1.2 Family Groups

A number of different classes of digital circuits are in current use. The most popular are TTL, MOS, and ECL.[8]

1. *TTL*

TTL, which means transistor transistor logic, is most commonly used for small and medium-scale integration (MSI). There are two basic forms of TTL. The first and original group has a low-power/low-speed version and the standard line. The later designs are Schottky clamped, which come in quite a few versions and are still growing. The Schottky diodes are used to prevent the transistors from going into saturation, with a resulting increase in speed for a given power dissipation.

The older families, including the 54H/74H are obsolete and used only for replacement. The low-power Schottky, 54LS/74LS, and the standard Schottky, 54S/74S, have been around for a long time. The newer versions include the family 54F/74F called FAST and two Texas Instruments versions, 54AS/74AS and 54ALS/74ALS. There are designated Advanced Schottky and Advanced Low Power Schottky, respectively. All of the TTL families represent different tradeoffs between power consumption and speed.

If speed is not required, the Low Power class is a good choice, not only because of low power consumption, but because its low speed makes it insensitive to many high-frequency spikes and glitches.[9]

Although open inputs to TTL logic act as a high, it is not a good practice to leave them disconnected in final circuitry. This is because the maximum speed is lowered and the noise susceptibility is increased. It is also not a safe practice to tie these inputs to the 5 V line because the breakdown voltage of the input lines are only 5.5 V compared to 7 V for the supply pin. Unused open inputs can be connected to the output of a

spare gate that is held at a high or a 1 kΩ resistor can be inserted between the gate lead and the + 5 V supply. One resistor can be used for up to 25 gates in some logic families. A better way, however, is to use two resistors and a couple of microfarads of capacitance to form a stable Hi of about 3.5 V.

One exception to this is the LS series where most of the devices have diode inputs, which can be directly connected to the + 5 V supply. Some of them have emitter inputs, however, which must be connected through the 1 kΩ resistor.

The families are almost completely pin compatible, but for some strange reason there are a few exceptions. The power pin for dual-in-line sockets is usually # 16 (or # 14) and the ground pin # 8 (or # 7) depending on whether it is a 16-pin or 14-pin package.[10] But, again, not always.

2. *MOS*

Mos logic comes in two forms. The first is called CMOS, where C stands for complementary, which means that both *N* channel and *P* channel transistors are used in a complementary fashion. These are mediumspeed devices that can be operated at 3 to 15 V. For example, a 54C73 flip-flop has a typical toggle frequency of 4 MHz at 5 V and 11 MHz at 10 V. If the speed limitations are acceptable, these devices are, on an all-around basis, clearly superior to the other families of logic. They have the lowest power consumption, high noise immunity (at 10 V), symmetrical drive capability at a good current level, and a very high input impedance. They are competitive with TTL in cost and are approaching TTL in the availability of logic functions, which are increasing rapidly. As contrasted to the other forms of logic, all inputs must be tied to a high or a low because they are open-gate leads of the MOSFETs and the devices simply do not know where they are if the gate lead is left open. One good feature is that the inputs can be safety connected to *B* + for a

high. Particular attention should be given to power consumption, which depends strongly on the operating frequency and load.

The second form of MOS logic is not really a family of logic elements but a collection of MSI and LSI (large-scale integration) that takes advantage of the small size and the low-power consumption of the MOS transistor to fabricate very large arrays. Typical of these devices are random-access memories (RAMs), read-only memories (ROMs), and microprocessors. These devices can be made by either *N* channel or *P* channel processes. They are generally constructed so that their drive levels are TTL compatible and can be mixed with TTL circuits. Many MSI and LSI devices are available in both MOS and TTL. In these cases the TTL is used only if its superior speed is required.

3. *ECL*

The last of the logic families to be discussed is called emitter-coupled logic, or ECL for short. This differs from the other forms of logic in that the transistors are operated in a linear mode and not allowed to go into saturation. ECL is therefore the fastest of all logic forms. It also consumes the most power. ECL devices operate from a -5.2 V power supply, a logic low is -1.8 V, and a logic high is -0.9 V. The output stages are open-ended emitter followers, which are externally terminated with a 51 resistor to a -2 V supply. This clutters up the circuit a bit, and the -2 V supply because of its low voltage must have a low efficiency. The basic ECL gate is the OR/NOR as contrasted to the NAND gate for TTL.

1.1.3 Building Blocks

There is really not much difference between "basic circuits" and "building blocks," except that the term "building blocks" implies that the function is contained in one package and is generally more complex. The tabulations that follow represent a very coarse selection of some of the

more common classifications. A suitable knowledge of the available products must come from the manufacturer's data sheets and not from a textbook.

1. *Function Generators and Clock Drivers*

"Function generator" is a broad term that includes clock generators. Clock drivers are included in this section because often the clocks drive many devices and the drive requirements can be quite severe.

The SN74LS124 is a useful dual voltage-controlled square-wave generator. It has a range of 0.12 Hz to 50 MHz, tunable by a voltage and selectable capacitor. A crystal can also be used in place of the capacitor if desired.

The Signetics 555 is very popular basic timer that makes a good square-wave generator up to about 100 kHz. It is low in cost, has excellent stability, and can use 10 MΩ resistors to reduce the size of the required capacitor.

2. *Flip-Flops*

In the early days of digital electronics flip-flops were the basic building block. They were used in great quantities to make counters, memories, and much miscellaneous logic. The design of synchronous counters from flip-flops can be tricky, but now counters of all kinds are available and the average circuit designer seldom needs to design them. The two kinds of flip-flops used the most are the *J-K* and the *D*.

The *J-K* flip-flop has two control lines, *J* and *K*, which allow four logical operations. The block diagram and the truth table are shown in Fig.1.5 *J-K* flip-flops are actuated by a clock edge to do the function defined by the *J* and *K* inputs. The figure shows a negative edge transition such as that used on the SN7473. Others, for example, the SN74109, operate on the positive edge. This device has two principal applications. The first is as part of some overall logic operation. The results of this logic operation are placed on the *J* and *K* lines, and the

clock then makes Q a high or a low in accordance with the truth tables. This mode of operation is used widely in the construction of synchronous counters and various types of memories. The second and more common application is as a divide-by-2 counter. The f-f is often used following counters to produce a square wave. For example, if a counter is used to divide a given frequency by 100, the output has a duty cycle of 1 to 99 (or sometimes less). Aside from being hard to see on an oscilloscope, this may not be a satisfactory waveform. The usual way is to program the counter to divide by 50 and then use the f-f to divide by 2, which produces an exactly symmetrical wave. Most oscillators do not produce a symmetrical waveform. Again, this problem is easily solved by designing the oscillator for twice the frequency and following it with an f-f.

Clock	J	K	Q
×	L	L	Q
↓	H	L	H
↓	L	H	L
↓	H	H	Q

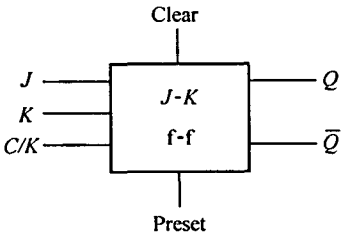

Fig.1.5 Truth table of a *J-K* f-f

Two *J-K* f-f's can be connected as shown in Fig. 1.6 to get a synchronous divider by 4, and since f-f's often come two to a package, this is commonly done. To divide by more than 4 requires additional logic and a counter is generally more suitable. The *J-K* f-f is available with many modifications for logical flexibility. There are asynchronous clear and preset inputs, which set Q to a low or high. These inputs override the clock. Then the J and K

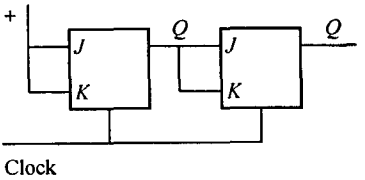

Fig.1.6 *J-K* divided by 4 counter