高等学校试用

英语理工科教材选

第五分册 工业电子学

汤民钟(主编) 张幼男 薛方 选编
戴鸣钟 李国瑞 谢卓杰 主审

机械工业部部属院校选编

高等学校试用

英语理工科教材选

Book Ⅴ

MICROELECTRONICS

——Digital and Analog Circuits and Systems

汤民钟（主编） 张幼男 薛 方 选编

戴鸣钟 李国瑞 谢卓杰 主审

机械工业部部属院校选编

本分册内容选自 JACOB MILLMAN所著《MICROELECTRONICS: Digital and Analog Circuits and Systems》英文原版书(1979年版本)中的第6、7、11、13四章。

高等学校试用

英语理工科教材选

(第五分册 工业电子学)

汤民钟(主编) 张幼男 薛 方 选编

戴鸣钟 李国瑞 谢卓杰 主审

上海机械学院教务处 出版发行

上海机械学院印刷厂 印刷

成本费 ¥1.12元

编者的话

为了提高机械工业部部属院校学生的外语水平，培养学生阅读英语科技书刊的能力，我们选编了这套“英语理工科教材选”。整套“教材选”共分九个分册，内容包括数学、物理、理论力学、材料力学(与理论力学合为一个分册)、电工学、工业电子学、金属工艺学、机械原理、机械另件(与机械原理合为一个分册)、计算机算法语言、管理工程等十一门业务课程。各业务课都选了三章英语原版教材（个别也有选四章），供机械工业部部属院校试用。

在业务课中使用部分外语原版教材，这是我们的一次尝试，也是业务课教材改革、汲收国外先进科学技术的探索。在选材时，我们考虑了我国现行各课程的体系、内容以及学生的外语程度，尽可能选用适合我国实际的外国材料。

本“教材选”的选编工作，是在机械工业部教育局的直接领导下，由部属院校的有关教研室做了大量调查研究后选定的，并进行注释和词汇整理工作。由马泰来、卢思源、李国瑞、柯秉衡、谢卓杰、戴炜华、戴鸣钟等同志(以姓氏笔划为序)组成的审编小组，对选材的文字、注释、词汇作了审校。戴鸣钟教授担任整套“教材选”的总审。

由于时间仓促，选材、注释和编辑必有不尽完善之处，希广大读者提出宝贵意见，以利改进。

1983年4月

CONTENTS

Unit One

(Chapter 6 *Combinational Digital Systems*)

Unit Two

(Chapter 7 *Sequential Digital Systems*)

Unit Three

(Chapter 11 *Low-Frequency Amplifiers*)

Unit Four

(Chapter 13 *Frequency Response of Amplifiers*)

CHAPTER
SIX

COMBINATIONAL DIGITAL SYSTEMS

A digital system is constructed from very few types of basic network configurations, these elementary types being used over and over again in various topological combinations. As emphasized in Sec. 5-8, it is possible to perform all logic operations with a single type of circuit (for example, a NAND gate). A digital system must store binary numbers in addition to performing logic. To take care of this requirement, a memory cell, called a FLIP-FLOP, is introduced in the next chapter.

Theoretically, any digital system can be constructed entirely from NAND gates and FLIP-FLOPS. Some functions (such as binary addition) are present in many systems, and hence the combination of gates and/or FLIP-FLOPS required to perform this function is available on a single chip. These integrated circuits form the practical (commercially available) basic building blocks for a digital system. The number of such different IC's is not large, and these packages† perform the following functions: binary addition, decoding (demultiplexing), data selection (multiplexing), counting, storage of binary information (memories and registers), digital-to-analog (D/A) and analog-to-digital (A/D) conversion, and a number of other operations. Those building blocks which depend upon combinational logic are described in this chapter.[1]

6-1 STANDARD GATE ASSEMBLIES[2]

Since the fundamental gates are used in large numbers even in a relatively simple digital system, they are not packaged individually; rather, several gates

† The terms *package, chip,* and *IC* are used interchangeably.

are constructed within a single chip. The following list of standard digital IC gates is typical, but far from exhaustive:

Quad two-input NAND	Quad two-input NOR
Triple three-input NAND	Quad two-input AND
Dual four-input NAND	Dual two-wide, two-input AOI
Single eight-input NAND	Two-wide, four-input AOI
Hex inverter buffer	Four-wide, 4-2-3-2-input AOI

These combinations are available in most logic families (TTL, DTL, etc.) listed in Sec. 5-15. The limitation on the number of gates per chip is usually set by the number of pins available. The most common package is the *dual-in-line* (plastic or ceramic) package, which has 14 leads, 7 brought out to each side of the IC (Fig. 6-1*c*). The dimensions of the assembly, which is much larger than the chip size, are approximately 0.8 by 0.3 by 0.2 in. A schematic of the triple three-input NAND is shown in Fig. 6-1*a*. Note that there are $3 \times 3 = 9$ input leads, 3 output leads, a power-supply lead, and a ground lead; a total of 14 leads are used.

In Fig. 6-1*b* is indicated the dual two-wide, two-input AOI (Sec. 5-6). This combination needs 4 input leads and 1 output lead per AOI, or 10 for the dual array. If 1 power-supply lead and 1 ground lead are added, we see that 12 of the 14 available pins are used.

The circuit diagram for this AOI gate is given in Fig. 6-2, implemented in TTL logic. The operation of this network should be clear from the discussion in

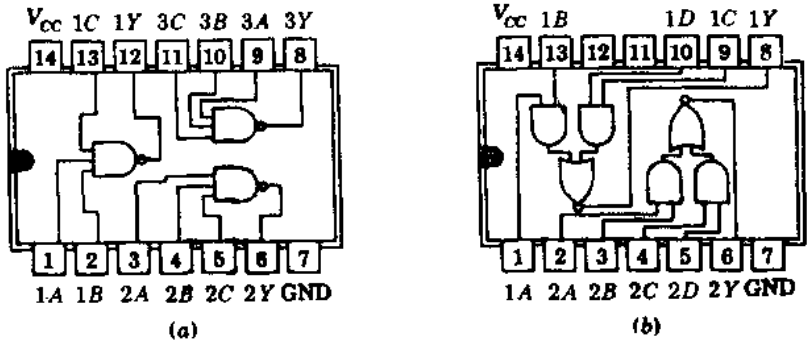

Figure 6-1 The lead connections (top view) of (*a*) the TI-10 triple three-input NAND. (*b*) The TI-51 dual two-wide, two-input AOI gate. (No connections are to be made to pins 11 and 12.) (*c*) A dual-in-line package, DIP.

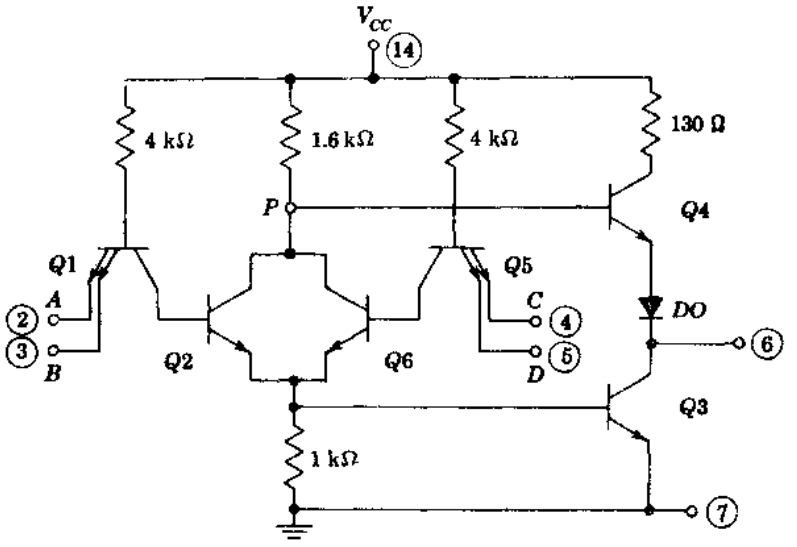

Figure 6-2 The circuit configuration for a TTL AND-OR-INVERT gate.

Chap. 5. Thus $Q1$ and the input to $Q2$ (corresponding to the similarly numbered transistors of Fig. 5-21) constitute an AND gate. The identical arrangement of $Q5$ and $Q6$ constitutes a second AND gate. Since the collectors of $Q2$ and $Q6$ are tied together at P, the output at this node corresponds to either the inputs 2 AND 3 OR 4 AND 5. Also, because of the inversion through a transistor, the NOT operation is performed at P. The result is AND-OR-INVERT (AOI) logic ($\overline{AB + CD}$). Finally, note that $Q3$, DO, and $Q4$ form the totem-pole output stage of Fig. 5-23.

An alternative way of analyzing the circuit of Fig. 6-2 is to consider $Q1$ and $Q2$ (with the output at P) to constitute a NAND circuit. Similarly, $Q5$ and $Q6$ form a second NAND gate. The outputs of these two NAND configurations are short-circuited together by the lead connecting the collectors of $Q2$ and $Q6$ to form a wired-AND (Sec. 5-11). Hence the output at P is, using De Morgan's law [Eq. (5-26)]

$$\overline{(AB)}\ \overline{(CD)} = \overline{AB + CD}$$

which confirms that AOI logic is performed.

Some of the more complicated functions to be described in this book require in excess of 14 pins, and these IC's are packaged with 16, 20, 24, and up to 64 leads.

The standard combinations considered in this section are examples of *small-scale integration* (SSI). Less than about 12 gates (~ 100 components) on a chip is considered SSI. The FLIP-FLOPS discussed in Sec. 7-3 are also SSI packages. Most other functions (using BJTs) discussed in this chapter are examples of *medium-scale integration* (MSI), defined to have more than 12, but less than 100, gates per chip. The BJT memories of Sec. 6-9 and the MOSFET arrays of Chap. 9 may contain in excess of 100 gates (> 1,000 components) and are defined as *large-scale integration* (LSI).

Design Philosophy

An electronics engineer should design a system so as to use standard IC's for as ⑤ many subsystems as possible. He or she must attempt to minimize the required number of packages (and hence the total cost[3]). A single MSI is used in place of a number of SSI chips which could perform the same function. Similarly, an LSI package is used in the system wherever this IC can replace several MSI chips. In summary, in designing a digital system, it should be defined in terms of standard MSI and LSI packages. Discrete gates (SSI) should be used only for "interfaces" (also called the "glue") which may be required between the subsystem IC's.

A list of manufacturers of IC's is given in Appendix B-1. These companies have available data books, handbooks, and application notes which are invaluable to the system designer since they keep him up to date on new packages and ⑥ applications as they become available. The most important functions performed by MSI chips are given in Chaps. 6 and 7. After introducing the MOSFET in Chap. 8, LSI packages are discussed in Chap. 9. The most versatile LSI system is the microcomputer, a programmable *computer on a chip*. As discussed in Sec. 9-11 the design of a complicated logic system is usually based upon the use of the microcomputer.

6-2 BINARY ADDERS[4]

A digital computer must obviously contain circuits which will perform arithmetic operations, i.e., addition, subtraction, multiplication, and division. The basic operations are addition and subtraction, since multiplication is essentially repeated addition, and division is essentially repeated subtraction.

⑦ Suppose we wish to sum two numbers in decimal arithmetic and obtain, say, the hundreds digit. We must add together not only the hundreds digit of each number but also a carry from the tens digit (if one exists). Similarly, in binary arithmetic we must add not only the digit of like significance of the two numbers to be summed, but also the carry bit (should one be present) of the next lower significant digit. This operation may be carried out in two steps: first, add the two bits corresponding to the 2^n digit, and then add the resultant to the carry from the 2^{n-1} digit. A two-input adder is called a *half adder*, because to complete an addition requires two such half adders.

We show how a *half adder* is constructed from the basic logic gates. A half adder has two inputs—A and B—representing the bits to be added, and two outputs—D (for the digit of the same significance as A and B represent) and C (for the carry bit).

Half Adder

The symbol for a half adder is given in Fig. 6-3*a*, and the truth table in Fig. 6-3*b*. Note that the D column gives the sum of A and B as long as the sum can be represented by a single digit. When, however, the sum is larger than can be

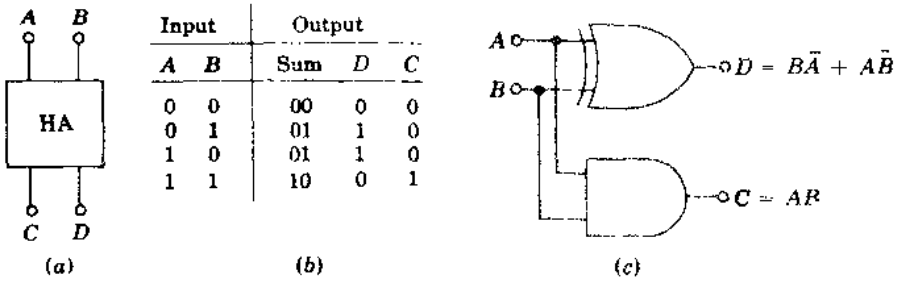

Figure 6-3 (*a*) The symbol for a half adder. (*b*) The truth table for the digit D and the carry C. (*c*) The implementation for D with an EXCLUSIVE OR gate and for C with an AND gate.

represented by a single digit, then D gives the digit in the result which is of the same significance as the individual digits being added. Thus, in the first three rows of the truth table, D gives the sum of A and B directly. Since the decimal equation "1 plus 1 equals 2" is written in binary form as "01 plus 01 equals 10," then in the last row $D = 0$. Because a 1 must now be carried to the place of next higher significance, $C = 1$.

From Fig. 6-3*b* we see that D obeys the EXCLUSIVE OR function and C follows the logic of an AND gate. These functions are indicated in Fig. 6-3*c*, and may be implemented in many different ways with the circuitry discussed in Chap. 5. For example, the EXCLUSIVE OR gate can be constructed with any of the four topologies of Sec. 5-6 and in any of the logic families in Table 5-3. The configuration in Fig. 5-10*b* ($Y = A\bar{B} + B\bar{A}$) is implemented in TTL logic with the AOI circuit of Fig. 6-2. The inverter for B (or A) is a single-input NAND gate. Since Y has an AND-OR (rather than an AND-OR-INVERT) topology, a transistor inverter is placed between node P and the base of $Q4$ of Fig. 6-2.

Parallel Operation

Two multidigit numbers may be added serially (one column at a time) or in parallel (all columns simultaneously). Consider parallel operation first. For an N-digit binary number there are (in addition to a common ground) N signal leads in the computer for each number. The nth line for number A (or B) is excited by A_n (or B_n), the bit for the 2^n digit ($n = 0, 1, \ldots, N - 1$).

Full Adder

In integrated circuit implementation, addition is performed using a complete adder, which (for reasons of economy of components) is not constructed from two half adders. The symbol for the nth full adder (FA) is indicated in Fig. 6-4*a*. The circuit has three inputs: the addend A_n, the augend B_n, and the input carry C_{n-1} (from the next lower bit). The outputs are the sum S_n (sometimes designated Σ_n) and the output carry C_n. A parallel 4-bit adder is indicated in Fig. 6-4*b*. Since FA0 represents the least significant bit (LSB), it has no input carry; hence $C_{-1} = 0$.

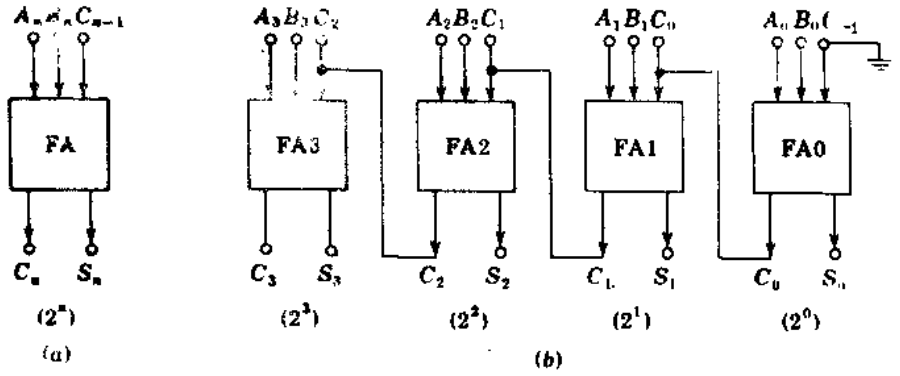

Figure 6-4 (*a*) The symbol for a full adder. (*b*) A 4-bit parallel binary adder constructed from cascaded full adders.

The circuitry within the block FA may be determined from Fig. 6-5, which is the truth table for adding 3 binary bits. From this table we can verify that the Boolean expressions for S_n and C_n are given by

$$S_n = \bar{A}_n \bar{B}_n C_{n-1} + \bar{A}_n B_n \bar{C}_{n-1} + A_n \bar{B}_n \bar{C}_{n-1} + A_n B_n C_{n-1} \qquad (6\text{-}1)$$

$$C_n = \bar{A}_n B_n C_{n-1} + A_n \bar{B}_n C_{n-1} + A_n B_n \bar{C}_{n-1} + A_n B_n C_{n-1} \qquad (6\text{-}2)$$

Note that the first term of S_n corresponds to line 1 of the table, the second term to line 2, the third term to line 4, and the last term to line 7. (These are the only rows where $S_n = 1$.) Similarly, the first term of C_n corresponds to the line 3 (where $C_n = 1$), the second term to the line 5, etc.

The AND operation ABC is sometimes called the *product* of A and B and C. (9) Also, the OR operation + is referred to as *summation.* Hence expressions such as (10) those in Eqs. (6-1) and (6-2) represent a *Boolean sum of products*. Such an equation is said to be in a *standard,* or *canonical, form,* and each term in the equation is called a *minterm*. A minterm contains the product of all Boolean variables, or their complements.

The expression for C_n can be simplified considerably as follows: Since $Y + Y + Y = Y$, then Eq. (6-2), with $Y = A_n B_n C_{n-1}$, becomes

	Inputs			Outputs	
Line	A_n	B_n	C_{n-1}	S_n	C_n
0	0	0	0	0	0
1	0	0	1	1	0
2	0	1	0	1	0
3	0	1	1	0	1
4	1	0	0	1	0
5	1	0	1	0	1
6	1	1	0	0	1
7	1	1	1	1	1

Figure 6-5 Truth table for a three-input adder. The lines are numbered decimally as if $A_n B_n C_{n-1}$ represents a 3-bit binary number with C_{n-1} equal to the LSB and A_n equal to the MSB (most significant bit).

$$C_n = \left(\bar{A}_n B_n C_{n-1} + A_n B_n C_{n-1}\right) + \left(A_n \bar{B}_n C_{n-1} + A_n B_n C_{n-1}\right) + \left(A_n B_n \bar{C}_{n-1} + A_n B_n C_{n-1}\right) \quad (6\text{-}3)$$

Since $\bar{X} + X = 1$ where $X = A_n$ for the first parentheses, $X = B_n$ for the second parentheses, and $X = C_{n-1}$ for the third parentheses, then Eq. (6-3) reduces to ⑪

$$C_n = B_n C_{n-1} + C_{n-1} A_n + A_n B_n \quad (6\text{-}4)$$

This expression could have written down directly from the truth table of Fig. 6-5 by noting that $C_n = 1$ if and only if at least two out of the three inputs is 1.

It is interesting to note that if all 1s are changed to 0s and all 0s to 1s, then lines 0 and 7 are interchanged, as are 1 and 6, 2 and 5, and also 3 and 4. Because this switching of 1s and 0s leaves the truth table unchanged, whatever logic is represented by Fig. 6-5 is equally valid if all inputs and outputs are complemented. Therefore Eq. (6-3) is true if all variables are negated, or ⑫

$$\bar{C}_n = \bar{B}_n \bar{C}_{n-1} + \bar{C}_{n-1} \bar{A}_n + \bar{A}_n \bar{B}_n \quad (6\text{-}5)$$

This same result is obtained (Prob. 6-3) by Boolean manipulation of Eq. (6-4).

By evaluating $D_n \equiv (A_n + B_n + C_{n-1})\bar{C}_n$ and comparing the result with Eq. (6-1), we find that $S_n \equiv D_n + A_n B_n C_{n-1}$, or

$$S_n = A_n \bar{C}_n + B_n \bar{C}_n + C_{n-1} \bar{C}_n + A_n B_n C_{n-1} \quad (6\text{-}6)$$

Equations (6-4) and (6-6) are implemented in Fig. 6-6 using AOI gates of the type shown in Fig. 6-2.

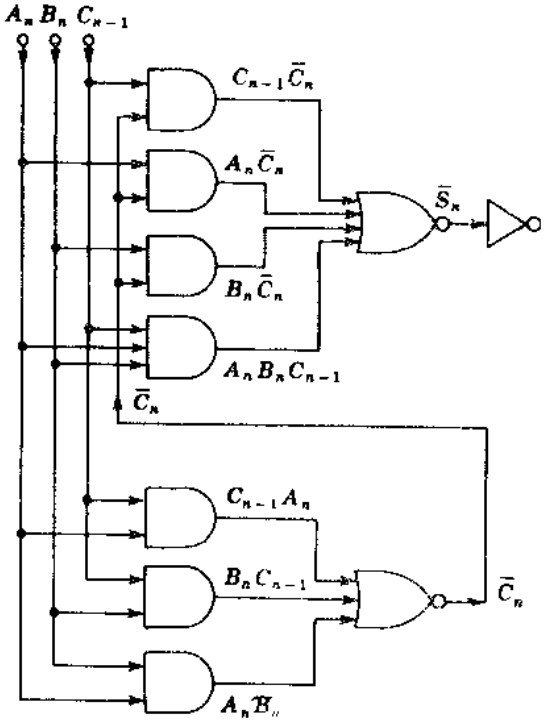

Figure 6-6 Block-diagram implementation of the nth stage of a full adder.

MSI Adders

There are commercially available 1-bit, 2-bit, and 4-bit full adders, each in one package. In Fig. 6-7 is indicated the logic topology for 2-bit addition. The inputs to the first stage are A_0 and B_0; the input marked C_{-1} is grounded. The output is the sum S_0. The carry C_0 is connected internally and is not brought to an output pin. This 2^0 stage (LSB) is identical with that in Fig. 6-6 with $n = 0$.

Since the carry from the first stage is C_0, it should be negated before it is fed to the 2^1 stage. However, the delay introduced by this inversion is undesirable, because the limitation upon the maximum speed of operation is the propagation delay (Sec. 5-15) of the carry through all the bits in the adder. The NOT-gate delay is eliminated completely in the carry by connecting $\overline{C}_0$ directly to the

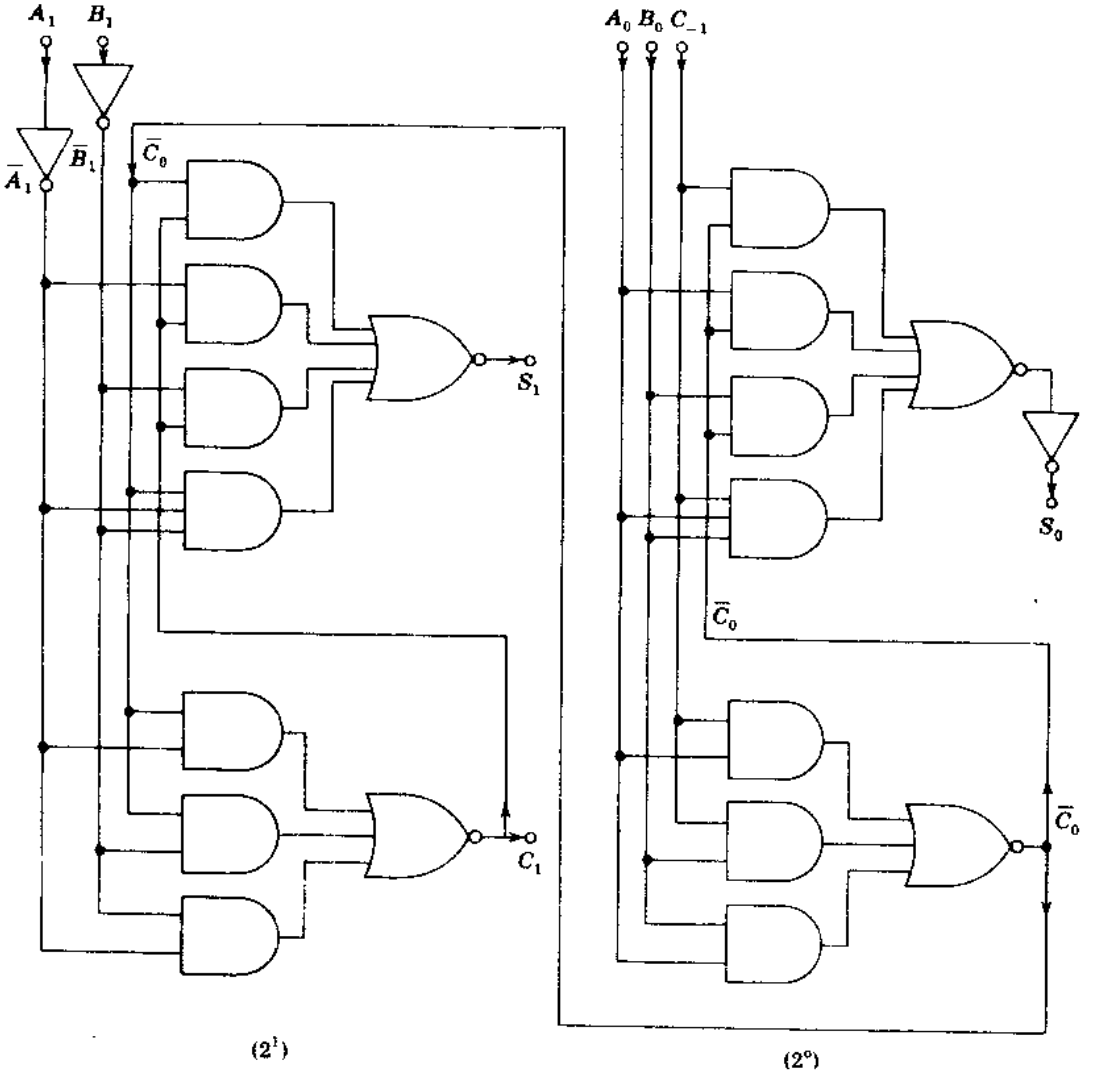

Figure 6-7 Logic diagram of an integrated 2-bit full adder (TI-82).

following stage and by complementing the inputs A_1 and B_1 before feeding these to this stage. This latter method is used in Fig. 6-7. Note that now the outputs S_1 and C_1 are obtained directly without requiring inverters. The logic followed by this second stage for the carry is given by Eq. (6-5), and for the sum by the modified form of Eq. (6-6), where each symbol is replaced by its complement.

In a 4-bit adder C_1 is not brought out but is internally connected to the third stage, which is identical with the first stage. Similarly, the fourth and second stages have identical logic topologies. A 4-bit adder requires a 16-pin package: 8 inputs, 4 sum outputs, a carry output, a carry input, the power-supply input, and ground. The carry input is needed only if two arithmetic units are cascaded; for example, cascading a 2-bit with a 4-bit adder gives the sum of two 6-bit numbers. If the 2-bit unit is used for the 2^4 and 2^5 digits, then 4 must be added to all the subscripts in Fig. 6-7. For example, C_{-1} is now called C_3 and is obtained from the output carry of the 4-bit adder.

The MSI chip (TI-283†) for a 4-bit binary full adder contains over 200 components (resistors, diodes, or transistors). The propagation delay time from data-in to data-out is typically 16 ns, and the power dissipation is 310 mW.

Serial Operation

In a serial adder the inputs A and B are synchronous pulse trains on two lines in the computer. Figure 6-8*a* and *b* shows typical pulse trains representing, respectively, the decimal numbers 13 and 11. Pulse trains representing the sum (24) and difference (2) are shown in Fig. 6-8*c* and *d*, respectively. A serial *adder* is a device which will take as inputs the two waveforms of Fig. 6-8*a* and *b* and deliver the output waveform in Fig. 6-8*c*. Similarly, a *subtractor* (Sec. 6-3) will yield the output shown in Fig. 6-8*d*. ⑬

We have already emphasized that the sum of two multidigit numbers may be formed by adding to the sum of the digits of like significance the carry (if any) which may have resulted from the next lower place. With respect to the pulse trains of Fig. 6-8, the above statement is equivalent to saying that, at any instant of time, we must add (in binary form) to the pulses A and B the carry pulse (if any) which comes from the resultant formed one period T earlier. The logic outlined above is performed by the full-adder circuit of Fig. 6-9. This circuit differs from the configuration in the parallel adder of Fig. 6-4 by the inclusion of a time delay TD which is equal to the time T between pulses. Hence the carry pulse is delayed a time T and added to the digit pulses in A and B, exactly as it should be.

A comparison of Figs. 6-4 and 6-9 indicates that parallel addition is faster than serial because all digits are added simultaneously in the former, but in sequence in the latter. However, whereas only one full adder is needed for serial ⑭

† The specific designations (Sec. 5-15) given in this chapter refer to Texas Instrument units.[2] However, equivalent units are available from other vendors (Appendix B-1). ⑮

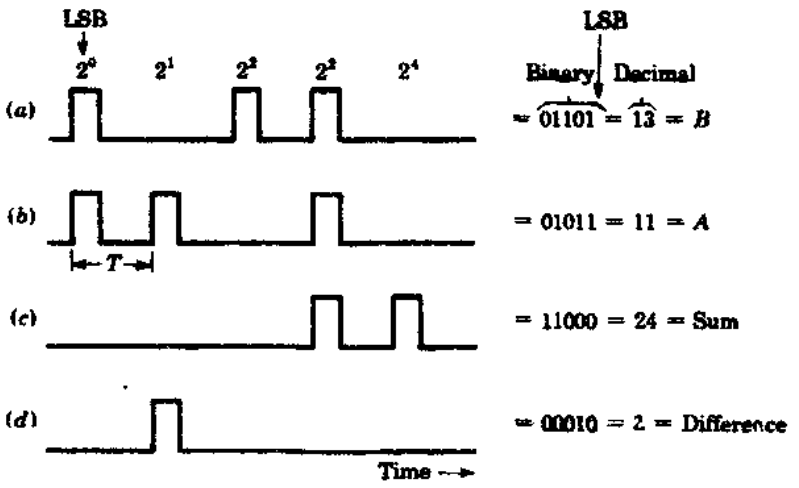

Figure 6-8 (*a*, *b*) Pulse waveforms representing numbers *B* and *A*. (*c*, *d*) Waveforms representing sum and difference. (LSB = least significant bit.)

arithmetic, we must use a full adder for each bit in parallel addition. Hence parallel addition is much more expensive than serial operation.

The time delay unit TD is a type D FLIP-FLOP, and the serial numbers A_n, B_n, and S_n are stored in *shift registers* (Secs. 7-3 and 7-4).

6-3 ARITHMETIC FUNCTIONS

In this and the next two sections other arithmetic units besides the adder are discussed. These include the *subtractor*, the *ALU*, the *multiplier*, the digital comparator, and the parity checker.

Binary Subtraction[4]

The process of subtraction (*B* minus *A*) is equivalent to addition if the complement *A* of the subtrahend is used. To justify this statement consider the following argument (applied specifically to a 4-bit number). The NOT function

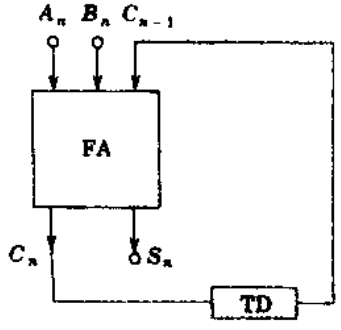

Figure 6-9 A serial binary full adder.