Chapter 14: Analog-to-Digital Conversion

Overview

Prerequisites:
- Knowledge of basic circuit analysis
- Knowledge of transistor switches (Chapter 13)

Objectives of Section 14.1:
- Relate hardware meaning and mathematical meaning of digital voltage
- Convert between binary, decimal, and hexadecimal numbers
- Understand parallel and series representation of digital voltage
- Become familiar with clock frequency and timing diagram of digital circuits
- Learn binary representation of ASCII characters
- Obtain initial exposure to tri-state digital voltage

Objectives of Section 14.2:
- Appreciate the necessity of the digital-to-analog converter
- Design simple hardware realization(s) of digital-to-analog converters
- Relate circuit structures to the corresponding mathematical operations with binary numbers
- Become familiar with resolution, accuracy, and voltage range of a DAC
- Learn two basic DAC constructions: binary-weighted input and R/2R ladder

Objectives of Section 14.3:
- Understand the necessity for sampling analog voltages
- Design simple hardware realization(s) of the sample-and-hold circuit
- Understand the value of the Nyquist rate

Objectives of Section 14.4:
- Design simple hardware realization(s) of the analog-to-digital converters: flash ADC and successive-approximation ADC
- Become familiar with key parameters of ADCs: resolution in bits, full-scale voltage range, and voltage resolution
- Obtain initial exposure to ADC speed, throughput rate, and conversion time

S.N. Makarov et al., *Practical Electrical Engineering*, DOI 10.1007/978-3-319-21173-2_14

Keywords:

Analog-to-digital converter (ADC), Digital-to-analog converter (DAC), Analog voltage, Digital voltage, Analog computer, Binary number system, Binary number, Least significant bit (LSB), Most significant bit (MSB), Parallel representation of a binary number, Serial representation of a binary number, Binary line codes, Unipolar NRZ line code, Polar NRZ line code, Unipolar RZ line code, Manchester line code, Bit rate, RS232 interface, Clock frequency, Timing diagram of a digital circuit, Asynchronous transmission, Synchronous transmission, Hexadecimal numbers, ASCII codes, Digital word, Bit, Byte, Nibble, Tri-state buffer, Tri-state digital voltage, Data bus, Output enable, Chip select, Summing amplifier, Binary-weighted-input DAC, R/2R ladder DAC, DAC scaling voltage factor, Scaled digital-to-analog conversion, DAC resolution voltage, Binary counter, DAC reconstruction filter, DAC postfilter, DAC equation, DAC full-scale output voltage range, DAC quantization levels, DAC external voltage reference, DAC resolution, DAC relative accuracy, Sample-and-hold voltage, Sampling interval, Sampling rate, Sampling frequency, Acquisition (sample) time, Sample-and-hold circuit, Track-and-hold circuit, Track/store circuit, Nyquist rate, Nyquist frequency, Digital signal processing, Nyquist-Shannon sampling theorem, Flash ADC, Successive-approximation ADC, Successive-approximation register, ADC full-scale measurement voltage range, ADC encoder block, ADC resolution in bits, ADC voltage resolution, ADC resolution accuracy, ADC equation, Mid-rise coding scheme, Mid-tread coding scheme, ADC quantization error, ADC quantization noise, ADC conversion time, ADC speed

Section 14.1 Digital Voltage and Binary Numbers

14.1.1 Introduction: ADC and DAC Circuits

Consider a typical block diagram of a digital signal processor (DSP)—see Fig. 14.1. The diagram includes an *analog-to-digital converter* (ADC) interfacing with a digital processor. The ADC converts an analog input voltage into data in the form of binary codes which are processed mathematically. The digital processor performs mathematical operations with binary numbers. The output to the processor is converted back to the real-world voltage using the *digital-to-analog converter* (DAC). Every time you use your cell phone, you are using a DSP, and this is only one example of its application. A general-purpose microprocessor possesses a similar structure.

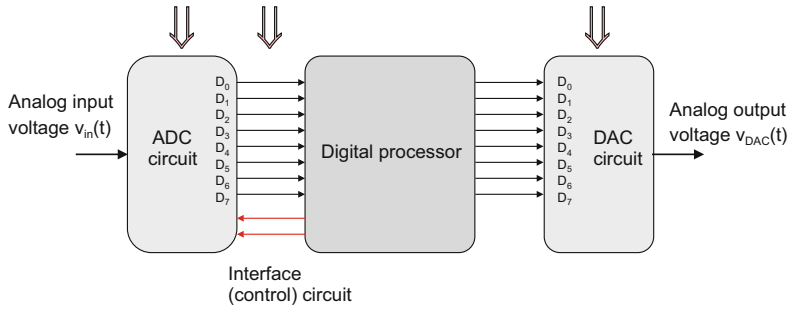


Fig. 14.1. Structure of a DSP including different circuit blocks.

In this chapter, we will study the ADC circuit(s), the DAC circuit(s), and some useful interface/control circuits. Those basic digital circuits utilize and extend the generic amplifier concept studied previously. In this sense, the present chapter offers further solidification of the amplifier theory and practice. Section 14.1 introduces the meaning of a digital voltage and its relation to binary number system. It shows how to read a binary word and how to understand its circuit representation. Along with this, we briefly review hexadecimal numbers and demonstrate how to convert between different number systems using MATLAB. Section 14.2 studies two basic digital-to-analog converter configurations. Resolution voltage, accuracy, and full-scale output voltage range are the most important features of a digital-to-analog converter (DAC) chip. It may be amazing to discover how the corresponding electronic circuits follow the mathematical formulas for number conversion. In this section, we also introduce the useful concept of a binary counter and give a number of examples. Section 14.3 analyzes the first block of the analog-to-digital converter—a sample-and-hold circuit. The meaning of the sampling rate and the fundamental concept of the Nyquist rate are naturally introduced in this context. Section 14.4 is devoted to two basic analog-to-digital converters. Resolution voltage, full-scale input voltage range, quantization error, and a slower speed of the A to D conversion are explained. Indeed, the present chapter does not exactly follow modern

digital hardware, neither does it provide the reader with comprehensive digital fundamentals including the control logic. Only the basic circuit concepts will be illustrated here, based on simpler yet functionally similar circuits.

14.1.2 Analog Voltage Versus Digital Voltage

Analog voltage is any instantaneous (usually continuous) voltage in a circuit. The term *digital voltage* is not quite precise. Strictly speaking, digital voltage is the same as analog voltage, which, however, may have only two states—*high* and *low*—at any time instant and/or at any particular node in the circuit. The digital voltage concept is closely related to the switching concept: the MOSFET switch studied in Chapter 13 may have only two states: *on* or *off*. The output voltage of the switch (the switching voltage) is thus exactly a digital voltage; it may be either high (switch is on) or low (switch is off). Using a number of such switches together allows us to process and store information in the circuit in the form of digital voltages. Thus, the analog circuit becomes a digital machine.

Analog Voltage

Consider the (decimal) number 10. How could we represent it in a computer? One way is shown in Fig. 14.2. We simply form a voltage divider circuit or an amplifier circuit or another analog circuit, with exactly one output. We'd strive to have 10 V at that output, as precisely as possible. This is the simplest and most intuitive way of assigning the voltage to a number.

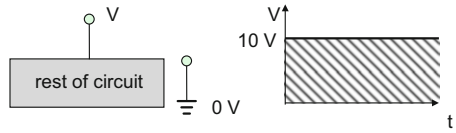


Fig. 14.2. Analog output voltage V of 10 V corresponding to a number 10.

However, the accuracy of this representation heavily depends on resistor tolerance, amplifier gain tolerance, temperature variations, etc. Such an idea basically corresponds to an *analog computer*, which is briefly considered below.

Digital Voltage and Binary Number System

The second less intuitive but much more versatile way to represent a certain number is shown in Fig. 14.3. We still try to represent (decimal) number 10. But instead of *one* output voltage V, we now introduce *four* output voltages denoted by $D_{3,2,1,0}$. However, each output can no longer have arbitrary voltage values. The output voltages may be either low or high, say, 0 V or 5 V, respectively.

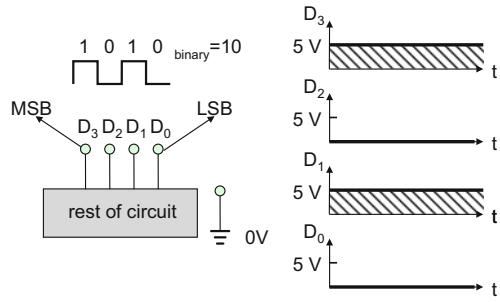


Fig. 14.3. Four parallel digital output voltages $D_{3,2,1,0}$ of either 0 V or 5 V corresponding to a decimal number 10.

In order to label high and low voltages, respectively, it is natural to use *a binary number system*, with only two digits, 0 and 1. We assign binary value 0 to 0 V and binary value 1 to 5 V. In view of this, the resulting *binary number,* which is a combination of all four digital output voltages in Fig. 14.2, is simply 1010_{binary}. Further, we apply the conversion rule

$$1010_{\text{binary}} = \mathbf{1} \times 2^3 + \mathbf{0} \times 2^2 + \mathbf{1} \times 2^1 + \mathbf{0} \times 2^0 = 10_{\text{decimal}} \tag{14.1a}$$

between two number types and obtain (decimal) number 10 exactly. The first (bold) multiplier in every summand in Eq. (14.1a) is a binary digit (or *bit*) in the binary number 1010; we read it from left to right. The rightmost bit (0 in this case) is the *least significant bit or LSB*; the leftmost bit (1 in this case) is the *most significant bit or MSB*. Equation (14.1a) says that it is possible to represent the number 10 with the help of four digital voltages 5 V, 0 V, 5 V, 0 V forming the binary number 1010. The specific value of *high* voltage is not really critical; the voltage combination 3 V, 0 V, 3 V, 0 V will again form the same binary number 1010 if we assign binary value 1 to the voltage of 3 V. Equation (14.1a) may indeed be rewritten in a more general form; for a 4-bit binary number, one has

$$d_3 d_2 d_1 d_{0\text{binary}} = \left(d_3 \times 2^3 + d_2 \times 2^2 + d_1 \times 2^1 + d_0 \times 2^0\right)_{\text{decimal}} \tag{14.1b}$$

where d_3, d_2, d_1, d_0 are the corresponding binary digits (not voltages).

Parallel Versus Serial Representation

Figure 14.3 corresponds to the so-called parallel representation of a binary number. A parallel representation requires each bit in the binary number to have its own electrical connection. In the laboratory you often see (older) ribbon cables. Every such cable has many individual wires; those wires are intended to separately carry digital voltages $D_{3,2,1,0}$. Yet another *serial way of representing digital values* is shown in Fig. 14.4. Serial data transmission is currently the dominant format for digital data transfer. Evidence of this is the overwhelming replacement of the older parallel ribbon cables by USB

(universal serial bus) cables. There is only one output voltage D in Fig. 14.4, but it changes in time as either 0 V or 5 V. Every voltage pulse in Fig. 14.4 is a bit (basic unit of information: high or low); every time interval T in Fig. 14.4 is the *bit width*; every bit in Fig. 14.4 represents a single binary digit: either the binary digit 0 (voltage pulse of 0 V) or the binary digit 1 (voltage pulse of 5 V).

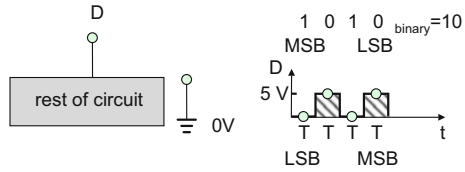


Fig. 14.4. Serial digital output voltage D of either 0 V or 5 V corresponding to a number 10.

If we *sample* (which literally means read) the voltages at the center of bit intervals, then the voltage signal in Fig. 14.4 will give us the binary number 1010 again. Here, the least significant bit is coming at the earliest time moment, and the most significant bit is coming last in time.

Example 14.1: Determine (decimal) numbers represented by digital voltages in Fig. 14.5a (parallel output) and in Fig. 14.5b (serial output), respectively.

Solution: Both Fig. 14.5a and b deal with exactly six bits (six digital voltage values). However, the LSB and MSB positions are the opposite. Therefore, we read in the case of Fig. 14.5a,

$$010101_{\text{binary}} = \mathbf{0} \times 2^5 + \mathbf{1} \times 2^4 + \mathbf{0} \times 2^3 + \mathbf{1} \times 2^2 + \mathbf{0} \times 2^1 + \mathbf{1} \times 2^0 = 21_{\text{decimal}} \quad (14.2)$$

and in the case of Fig. 14.5b,

$$101010_{\text{binary}} = \mathbf{1} \times 2^5 + \mathbf{0} \times 2^4 + \mathbf{1} \times 2^3 + \mathbf{0} \times 2^2 + \mathbf{1} \times 2^1 + \mathbf{0} \times 2^0 = 42_{\text{decimal}} \quad (14.3)$$

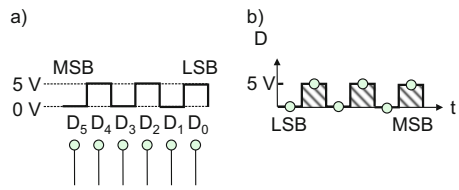


Fig. 14.5. Digital output voltages for parallel (**a**) and serial (**b**) transmission.

14.1.3 Bit Rate, Clock Frequency: Timing Diagram

Bit Rate

While the parallel digital output is only characterized by high and low states at any time instant, the serial bit stream may have a huge variety of forms or *codes* by which the 0 s and 1 s are represented. Some of them (the so-called line codes used for wired or wireless data transmission) are shown in Fig. 14.6. The first one is the unipolar (0–5 V) NRZ (*nonreturn to zero*) code that has also been drawn in Figs. 14.4 and 14.5. The second code is similar in form, but it utilizes both positive (+5 V-high) and negative (−5 V-low) voltages versus ground. The third one has the duty cycle of 50 % (returns to zero in the middle of bit width). The last code represents binary one by a ± 5 V two-phase pulse and binary 0—by a ∓ 5 V reversed two-phase pulse.

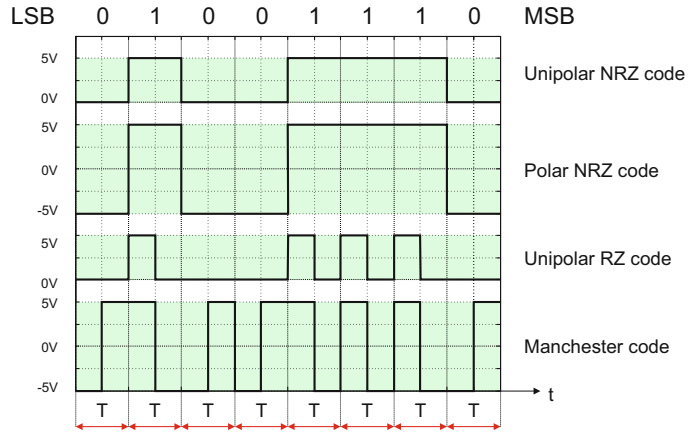


Fig. 14.6. Some serial line codes. All codes have the same bit width, T, and bit rate, f_b.

Despite all those differences, the codes shown in Fig. 14.6 convey the *same* information—binary number 01110010—in the same amount of time and thus have the same *bit width*, T. It is clear that the capacity of a serial digital data stream depends on the speed with which the bits (regardless of format) are transferred through a path. This capacity is determined by a *bit rate*. The bit rate is the number of bits conveyed or processed per second. As the name implies, the bit rate, f_b, for any serial bit stream in Fig. 14.6 is

$$f_b = \frac{1}{T} \tag{14.4}$$

where T is the bit width. Although the bit rate, f_b, in Eq. (14.4) has the units of frequency or hertz, it is rather measured in bits/s or *bps*. The reason for this is in that the serial bit stream conveying nontrivial information is never a periodic waveform—see Fig. 14.6. Therefore, we cannot use the meaning of frequency.

Exercise 14.1: Determine the bit rate of a bit stream in Fig. 14.7 when bit width T is 1 μs.

Answer: Using Eq. (14.4), one can find $f_b = \frac{1}{T} = 1,000,000$ bps or 1 Mbps.

Example 14.2: Figure 14.7 shows the line code for a *RS232* (*Recommended Standard 232*) *interface* (the serial port of a PC). Although outperformed with its much faster successor, the USB, the RS232 interface is still widely used today. In contrast to the USB, the RS232 interface does not require a protocol for transferring the data. For the line code in Fig. 14.7 determine:

A. Code type
B. Bit rate

Solution: The comparison with Fig. 14.6 indicates that the code in Fig. 14.7 is a polar NRZ code. This means that the serial connection requires a common ground wire in order to distinguish between positive and negative voltages. Note that all voltages in Fig. 14.7 are inverted—binary 0 now corresponds to a high voltage and binary 1—to a low voltage. The corresponding bit rate is

$$f_b = \frac{1}{0.1\,\text{ms}} = 10,000 \text{ bps} \quad \text{or} \quad 10 \text{ kbps}.$$

One might wonder, for example, what are bit rates for *RFID* (*radio-frequency identification*) *tags*, in the popular wireless frequency range 860–960 MHz. For passive (without a battery) RFIDs, the bit rates from 26.7 kbps to 128 kbps are common. However, active (battery-powered) RFIDs have higher bit rates. For example, the E-ZPass toll-collection system operates at 500 kbps.

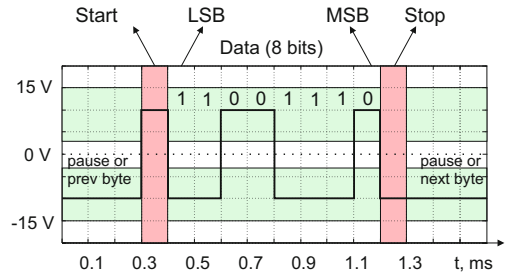


Fig. 14.7. RS232 line code and its format. Acceptable voltage levels for high and low voltages are indicated by green rectangles. Related standards are RS-422, R-423, R-449, and RS-485.

Clock Frequency

In contrast to the bit stream of data, the voltage *clock* (a basic pulse train that is used to synchronize serial bit streams) is always a *periodic waveform*. A typical clock waveform

corresponding to a bit stream in Fig. 14.6 or in Fig. 14.7 is shown in Fig. 14.8. Therefore, one determines *clock frequency*, f, in Fig. 14.8,

$$f = \frac{1}{T} \tag{14.5}$$

in hertz. If the bit width and the clock period coincide, the bit rate is equal to the clock frequency.

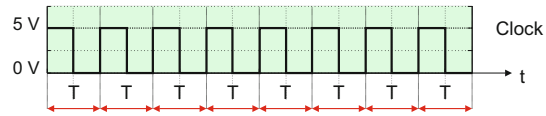


Fig. 14.8. Clock signal for a bit stream in Fig. 14.7. The duty cycle of the periodic waveform is 50 %.

We note that the clock frequency is critical not only for digital IO but also for the functioning of the computer itself. A computer or a microprocessor processes streams of digital data. The logic gates introduced in Chapter 13 make it possible to perform logical and arithmetical operations on binary numbers. As the logic gates are comprised of individual transistors (MOSFETs), their parasitic capacitances delay and distort clock/bit pulses. As a result, there is always a limiting clock frequency that dictates both the maximum serial IO speed and the maximum computer speed. Eventually, those speeds are determined by the response of switching transistors introduced in Chapter 13. This question is of great practical importance; it requires detailed knowledge of MOS transistors introduced in Chapter 18 and the detailed knowledge of transient RC circuits studied in Chapter 7.

Timing Diagram

Figure 14.9 below illustrates a *timing diagram*: the clock signal and the actual synchronized serial bit stream (the unipolar NRZ code). The timing diagram is a necessary attribute of many digital device datasheets. A case in point is an analog-to-digital converter studied in Section 14.3 of this chapter.

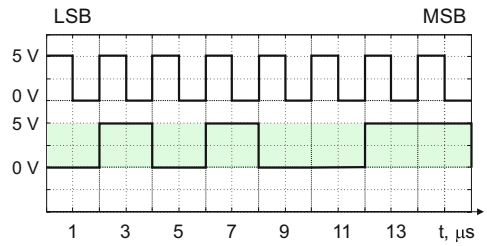


Fig. 14.9. Timing diagram for a binary bit steam.

Does the RS232 code illustrated in Fig. 14.7 need a clock signal to be sent along with the data code? As a matter of fact, the clock signal is not necessarily needed. One can see that in Fig. 14.7 a start signal is sent prior to each byte (eight bits), and a stop signal is sent after each byte. This helps us to decode the data even if no clock is available. Such a transmission is called *asynchronous transmission*. On the other hand, RS232 also makes use of a *synchronous transmission*; it has dedicated lines for a transmitter clock. We also note that modern USB cables (USB 3.0) transfer data at the rate of 4800 Mbps. This is an amazing number given the low cable cost and the use of stranded AWG-28 wires.

Exercise 14.2: For the timing diagram in Fig. 14.9:

A. Determine the clock period.
B. Determine the clock frequency (show units).
C. Determine the bit rate of the bit stream (show units).
D. Decode the corresponding binary number given the LSB/MSB positions (present its decimal equivalent).

Answer:

A. The clock period (distance between two rising edges) is $T = 2$ μs.
B. The clock frequency is $f = 1/T = 500$ kHz.
C. The bit rate of the bit stream is the inverse of the bit width, i.e., 500 kbps.
D. The corresponding binary number is 11001010. Its decimal equivalent is $11001010_{\text{binary}} = 202_{\text{decimal}}$.

14.1.4 Binary Numbers

Defeat of Analog Computers

It can be seen that digital voltage waveforms are always more complicated than analog ones. Either we need more wires or need to transfer a long stream of digital pulses to represent an analog (decimal) number. We may ask ourselves a question: as long the amplifier circuits studied previously can perform multiplication, addition (or subtraction), and integration (or differentiation), why can we not build an *analog computer*, which operates with analog voltages and replaces its digital counterpart at least for simple computational tasks? The answer is indeed: yes, we can. In fact, this was done long before, in the mid-1960s. Figure 14.10 shows one such design. However, the analog computers were quickly outperformed by the digital systems. What is the reason? Surprisingly, it was not speed—amplifier circuits built with single junction transistors can be extremely fast. Plus, analog computers routinely operate with many parallel tasks, whereas the single-processor digital systems tend to operate sequentially.

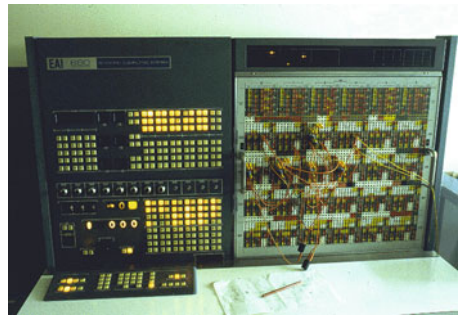


Fig. 14.10. An analog computer from Electronic Associates, Inc. West Long Branch, New Jersey, with 156 individual amplifiers (Analog Computer Museum by Doug Coward).

The culprits turned out noise and even the accuracy of the analog circuit components. Imagine what tolerance the final result could have if you would perform 100 multiplications using 1 %-accurate resistors? The digital approach is principally different: the bit values are defined by logic *threshold* voltage levels, which allow us for a wide variation of circuit voltages within those margins. For example, a bit value 1 for a 0-to-5-V logic may correspond to *any* voltage between 2.6 V and 5.0 V and a bit value zero—to *any* voltage between 0 V and 0.4 V. Another, perhaps even more important point is the existence of fast digital memory (ROM and RAM), which is more advanced than the analog memory carriers. Still, analog computers and especially *hybrid* computers are used in certain military and commercial applications even today.

Binary Numbers

Values that are in the ones and zeros format are said to be *binary. Bi-* means "two," so a binary number can only have one of two values per digit, as opposed to decimal numbers, which can have one of ten values per digit. The conversion of integer or fractional binary numbers to decimal numbers is rather straightforward; a few examples have been given at the beginning of this section. When a binary fraction is present, we perform the conversion following Eq. (14.6), that is,

$$\begin{array}{cccccccccccccc} 2^7 & 2^6 & 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 & & 2^{-1} & 2^{-2} & 2^{-3} & 2^{-4} \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & . & 1 & 0 & 0 & 1 \end{array} = 128 + 16 + 8 + 1 + \frac{1}{2} + \frac{1}{16} = 153.5625_{\text{decimal}} \tag{14.6}$$

On the other hand, the conversion of decimal numbers to binary numbers is a bit more involved. To do so, we repeatedly divide the decimal number by two until the quotient is zero. Equation (14.7) gives an example for decimal number 153. The resulting binary number is given by the remainder column; we read it as shown in Eq. (14.7).

153	Quot.	Remainder	
153/2	76	1	(**LSB**)
76/2	38	0	
38/2	19	0	
19/2	9	1	
9/2	4	1	
4/2	2	0	
2/2	1	0	
1/2	**0**(stop)	1	(**MSB**)

(14.7)

To convert a decimal fraction to a binary fraction, we repeatedly multiply by 2, track if the result exceeds one, and subtract one when this is the case. The template is given by Eq. (14.8) for a decimal fraction 0.5625.

0.5625	Check if > 1.	Remainder	
0.5625×2	1	0.125	(**MSB**)
0.125×2	0	0 .25	
0.25×2	0	0.5	
0.5×2	1	**0**(stop)	(**LSB**)

(14.8)

The remainder zero is not necessarily reached; in that case one has an infinite binary fraction that may be truncated to a given number of binary digits.

Exercise 14.3: (A) Convert binary number 10011001 to decimal number using MATLAB. (B) Convert decimal number 153 to binary number using MATLAB.

Answer:

A. We use MATLAB function `bin2dec('10011001')` and obtain 153. This method cannot be applied to binary fractions.
B. We use MATLAB function `dec2bin(153)` and obtain `10011001`. This method cannot be applied to decimal fractions either.

14.1.5 Hexadecimal Numbers

Binary numbers (base 2) are beneficial for electronic hardware since we only distinguish between two voltage values. For human interpretation and programming, the octal (base 8) and especially *hexadecimal* (base 16) *numbers* are often used. Table 14.1 that follows establishes symbols for single hexadecimal (shortly *hex* or *h*) digits. This table simultaneously lists all binary numbers with four bits—the so-called nibble—total 16 such numbers. The hexadecimal numbers are labeled somewhat differently depending on the application field. Equation (14.9) gives an example for hexadecimal number 378:

$$\underbrace{378_{16}}_{\text{present text}} = \underbrace{378\text{h}}_{\text{a microprocessor textbook}} = \underbrace{0\text{x}378}_{\text{C/C++}} \tag{14.9}$$

Conversion of hexadecimal numbers to decimal and binary numbers is explained in the example that follows. Especially simple and elegant is hex-to-binary conversion.

Table 14.1. Single hexadecimal digits in terms of binary numbers and decimal numbers.

Hexadecimal digit	Binary number	Decimal number
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
A	1010	10
B	1011	11
C	1100	12
D	1101	13
E	1110	14
F	1111	15

Note that binary numbers in the first ten rows of Table 14.1 simultaneously give the BCD (*binary coded decimal*) code for single decimal digits.

Example 14.3: A. Convert hexadecimal number 378_{16} to decimal number.
B. Convert hexadecimal number 378_{16} to binary number.

Solution:

A. We follow the scheme for binary numbers (Eq. (14.6)) but use base 16 instead of base 2:

$$\begin{matrix} 16^2 & 16^1 & 16^0 \\ 3 & 7 & 8 \end{matrix} = \mathbf{3} \times 256 + \mathbf{7} \times 16 + \mathbf{8} \times 1 = 888_{\text{decimal}} \tag{14.10}$$

Conversion of a hex fraction to a decimal fraction is done similarly; we again use the analogy with Eq. (14.6).

Example 14.3 (cont.):

B. Every hexadecimal digit is simply replaced by a binary number from Table 14.1, so that

$$\underset{3}{0011}\ \underset{7}{0111}\ \underset{8}{1000} = 001101111000 = 1101111000 \tag{14.11}$$

At the end of this example, we note that it is much easier to read and talk about 780 A (hex) than 0111100000001010 (binary)!

Exercise 14.4: Convert decimal number 1023_{10} to hexadecimal number using MATLAB.

Solution: We use MATLAB function `dec2hex(1023)` and obtain 3FF. This method cannot be applied to decimal fractions.

14.1.6 ASCII Codes and Binary Words

Once binary numbers have been introduced, every piece of information could be expressed through those numbers, i.e., in terms of ones and zeros. As an example, *ASCII codes* (American Standard Code for Information Interchange) is a set of correspondences between binary numbers (or digital voltages) and characters on the keyboard, including uppercase and lowercase letters, numbers, and special characters. Table 14.2 which follows includes all the capital letters from an ASCII table. Notice that the MSB in the second column is clearly redundant—only seven bits are necessary at most to represent the keyboard ASCII characters. The 8-bit character set is actually the Extended ASCII which came about later.

Table 14.2. ASCII table for English capital letters.

English capital letter	8-bit binary number (only seven bits are meaningful)	Hexadecimal number	Decimal number
A	01000001	41	65
B	01000010	42	66
C	01000011	43	67
D	01000100	44	68
E	01000101	45	69
F	01000110	46	70
G	01000111	47	71
H	01001000	48	72
I	01001001	49	73
J	01001010	4A	74

(continued)

Table 14.2 (continued)

English capital letter	8-bit binary number (only seven bits are meaningful)	Hexadecimal number	Decimal number
K	01001011	4B	75
L	01001100	4C	76
M	01001101	4D	77
N	01001110	4E	78
O	01001111	4 F	79
P	01010000	50	80
Q	01010001	51	81
R	01010010	52	82
S	01010011	53	83
T	01010100	54	84
U	01010101	55	85
V	01010110	56	86
W	01010111	57	87
X	01011000	58	88
Y	01011001	59	89
Z	01011010	5A	90

Example 14.4: Retrieve binary numbers (8-bit long words) for *all* ASCII characters without memorizing Table 14.2 and/or its extension.

Solution: We use a simple MATLAB script that follows (the character is letter A chosen as an example)

```
s = 'A'
d = int8(s)
b = dec2bin(d)
```

and obtain `d =65; b = 1000001` for decimal and binary forms, respectively. Other characters are processed in the same way. Note that an alternative solution that creates the identical result in MATLAB is

```
s = 'A'; d = double(s); b = dec2bin(d)
```

One can see from Table 14.2 that we generally need a multi-bit binary number to represent a letter or another character. Such a binary number is called a *digital word*:

1. A *nibble* is the digital word consisting of four bits.
2. A *byte* is the digital word consisting of eight bits.

All binary numbers in the second column in Table 14.2 above are one-byte digital words. A byte is the smallest amount of information that is typically saved in computer memory. On the other hand, all single hexadecimal digits in Table 14.1 can be described by a 4-bit word.

Historical: When and where the first electronic digital computer was built?

Answer: Interestingly, the first computer ever was built in 1939 in American Midwest, at Iowa State University by Professor John Vincent Atanasoff (1903–1995) and an electrical engineering undergraduate student Clifford Berry (1918–1963)—see Fig. 14.11 that follows. John V. Atanasoff received his BS in EE from the University of Florida (with straight As!). This computer has been called the ABC (Atanasoff-Berry Computer).

The machine was the first to use several innovations that are a part of today's computers:

- A binary system of arithmetic
- Separate memory and computing functions, regenerative memory
- Parallel processing, electronic amplifiers as on-off switches
- Circuits for logical addition and subtraction
- Clocked control of electronic operations
- A modular design

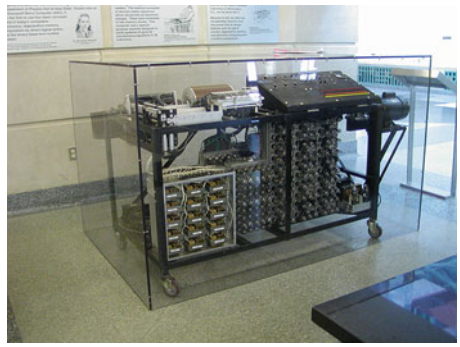


Fig. 14.11. A modern replica of the Atanasoff-Berry computer. Computer History Museum in Mountain View, California, 2010.

14.1.7 Tri-state Digital Voltage

Tri-state Buffer

Along with the high and low voltages considered in this section, a digital voltage is usually required to have one more state: the *High-Z* (or "do not care") state. The High-Z state means that the corresponding terminal is disconnected from the rest of the circuit. Its resistance to ground is therefore infinitely high (the terminal becomes an open circuit). Note that letter *Z* in the term *High-Z* actually stands for impedance *Z*, which is an extension of the resistance concept to both static (resistor) and dynamic (inductor,

capacitor) circuit components as shown in Chapter 8. When a digital device generates high and low voltages only, a *tri-state buffer* is added in order to achieve an extra third state: the High-Z state. This buffer may either be internal or external to a chip. The concept is explained in Fig. 14.12. First, in Fig. 14.12a we consider the standard buffer amplifier that uses a single-polarity power supply. When connected to an output of a digital device, the amplifier simply transfers high and low digital voltages, without adding new states. Therefore, its output *D* still has only two states—high and low. The next step is to add a switching circuitry shown in Fig. 14.12b to the amplifier. When control voltage enable is high, both the NMOS and PMOS switches will be on—see Chapter 15 and note an inverter connected to the PMOS. Nothing really changes compared to the previous case. However, when control voltage enable is low, both switches will be off. This means that the amplifier will be disconnected from the power supply completely. In other words, its output *D* becomes an entirely open circuit because current can flow nowhere (the current cannot flow in/out the amplifier input(s) and into the power rails).

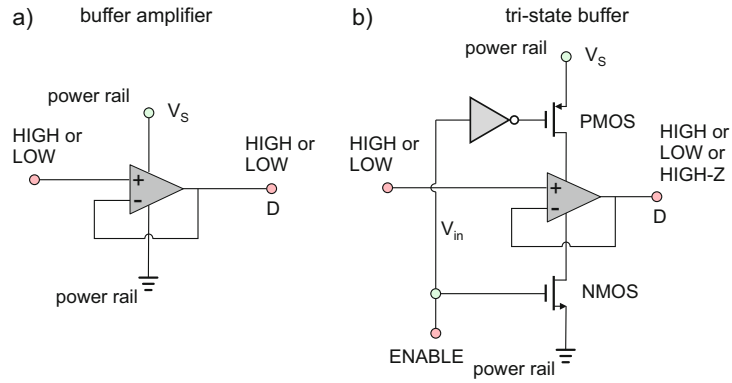


Fig. 14.12. Concept of a tri-state buffer.

Thus, the output to the amplifier, *D*, in Fig. 14.12b achieves the High-Z state. The corresponding truth table is Table 14.3.

Table 14.3. Truth table for the tri-state buffer in Fig. 14.12b.

Input		
Input to the amplifier	Enable	Output D
0	0	High-Z
1	0	High-Z
0	1	0
1	1	1

Why Is the Tri-state Voltage Important?

An example is given in Fig. 14.13a. Here, two digital devices A and B are connected to a *bus*, which is a set of interconnections that interface two or more digital devices. The bus is supposed to run to another device C, not shown in Fig. 14.13a.

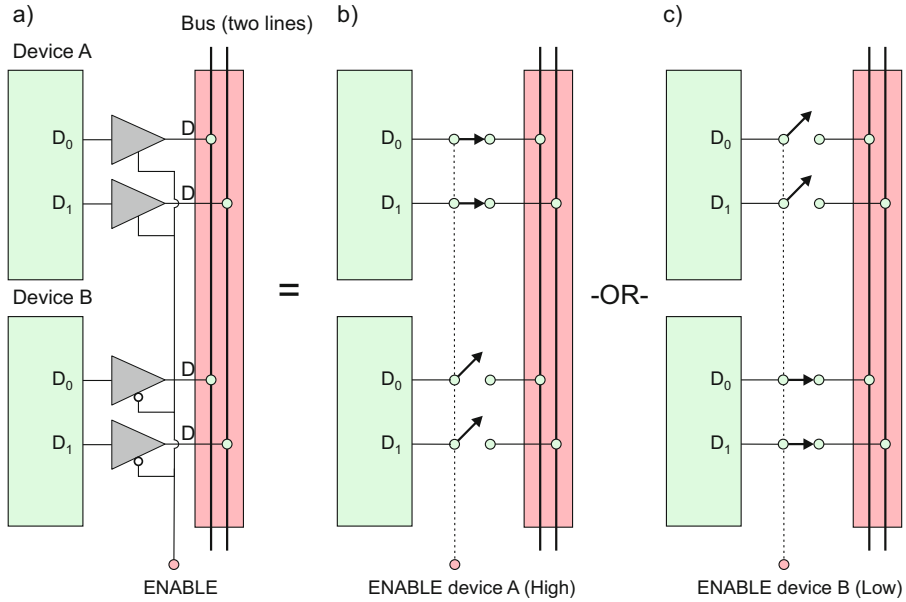


Fig. 14.13. Two digital devices connected to the same bus.

At any given time instant, only one device can have access to the bus (to the device C). Let us say it is device A. Device B should then be electrically disconnected from the circuit so that it cannot send competing voltage signals and thus cause *bus contention*. Thus, we should add tri-state buffers to both devices A and B as illustrated in Fig. 14.13a. The symbol for the tri-state buffer is a triangle with an extra terminal. The circuit then operates as shown in Fig. 14.13b and c, respectively. The corresponding bus then becomes the *tri-state bus*. How do we enable/disable outputs from different digital devices in practice? For many chips, it is done with *output enable* (OE) and *chip select* (CS) *pins*. In particular, the output drivers are disabled by deasserting o*utput enable*.

Section 14.2 Digital-to-Analog Converter

14.2.1 Digital-to-Analog Converter

Conversion between analog and digital voltage signals is done by means of a *digital-to-analog converter* (DAC) and an *analog-to-digital converter* (ADC). Both DAC and ADC are electric circuits that perform the corresponding operations. In this subsection we consider the idea of the DAC circuit. Its goal is to convert a sequence of binary numbers (digital voltages) generated by a processor or by a computer into a real-world voltage signal, which could be used as an input to a control system, to an audio amplifier, etc. The place of the DAC in the generic block diagram of a DSP is shown in Fig. 14.14.

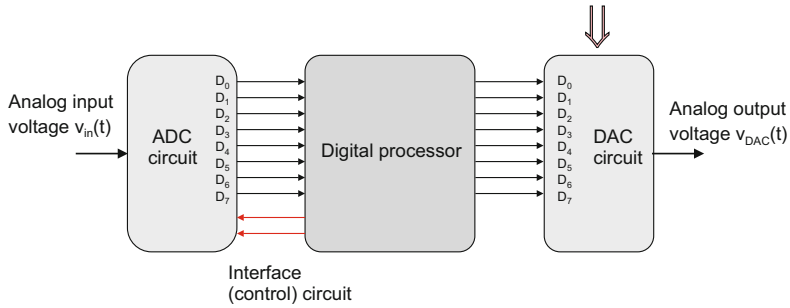


Fig. 14.14. Place of the DAC in the DSP block diagram.

14.2.2 Circuit (A Binary-Weighted-Input DAC)

Digital-to-analog conversion is conceptually simple. A 4-bit binary-weighted-input DAC at the base of an amplifier is shown in Fig. 14.15. The circuit implies four digital input voltages (*data lines*) D_3, D_2, D_1, D_0 and one analog output voltage υ_{DAC}. The parallel path of four binary signals in Fig. 14.15 is called a *data bus*, D_0 always represents the least significant bit (LSB). The converter in Fig. 14.15 has the form of a *summing amplifier*. The summing amplifier is further connected to an (optional) inverting amplifier stage having the gain of minus one. The summing amplifier is also an inverting amplifier with the negative feedback but with multiple inputs. Its operation may be explained as follows. According to the KCL and to the first summing-point constraint (no current into the amplifier), one has for the feedback current with reference to Fig. 14.15

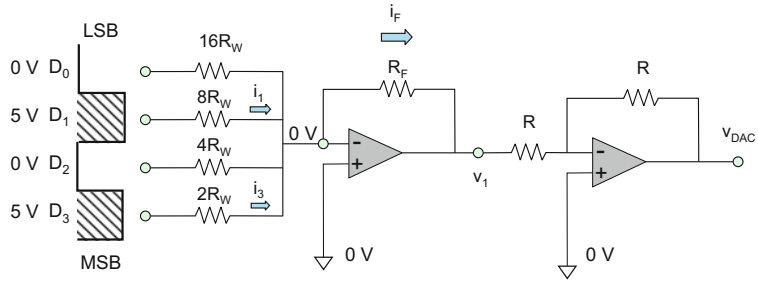


Fig. 14.15. Binary-weighted-input digital-to-analog converter. Index w indicates weighted resistances.

$$i_F = i_3 + i_2 + i_1 + i_0 \tag{14.12a}$$

On the other hand, the second summing-point constraint (the differential voltage to the amplifier is zero and the inverting input is the virtual ground) yields

$$i_3 = \frac{D_3}{2R_W}, \quad i_2 = \frac{D_2}{4R_W}, \quad i_1 = \frac{D_1}{8R_W}, \quad i_0 = \frac{D_0}{16R_W} \tag{14.12b}$$

in terms of input voltages D_3, D_2, D_1, D_0. Therefore, voltage v_1 in Fig. 14.15 is found from Eq. (14.12a) that is now written in the form:

$$\begin{aligned} i_F = \frac{0 - v_1}{R_F} &= i_3 + i_2 + i_1 + i_0 = \left(\frac{D_3}{2R_W} + \frac{D_2}{4R_W} + \frac{D_1}{8R_W} + \frac{D_0}{16R_W}\right) \\ &= \frac{1}{16R_W}\left(D_3 \times 2^3 + D_2 \times 2^2 + D_1 \times 2^1 + D_0 \times 2^0\right) \end{aligned} \tag{14.12c}$$

Consequently, the output voltage to the entire converter becomes

$$v_{DAC} = -v_1 = \frac{R_F}{16R_W}\left(D_3 \times 2^3 + D_2 \times 2^2 + D_1 \times 2^1 + D_0 \times 2^0\right), \tag{14.12d}$$

which is the final result for the *binary-weighted-input DAC shown* in Fig. 14.15.

14.2.3 Underlying Math and Resolution Voltage

It should be emphasized that Eq. (14.12d), which describes the circuit operation, is *precisely* the hardware realization of the mathematical formula for binary-to-decimal conversion given, for example, by Eq. (14.1) at the beginning of the previous section. In order to prove this fact, Eq. (14.12d) may be rewritten in the form

$$v_{DAC} = Q\left(d_3 \times 2^3 + d_2 \times 2^2 + d_1 \times 2^1 + d_0 \times 2^0\right) \tag{14.13a}$$

where the *scaling voltage factor* Q is given by

$$Q = \frac{R_F D}{2^N R_W}, \quad N = 4 \tag{14.13b}$$

for the present circuit. Here, N is the number of bits (data lines), and $d_3, \ldots, d_0$ are input binary digits (0 and 1) of a 4-bit word, so that

$$D_3 = Dd_3, \ \ldots, D_0 = Dd_0 \tag{14.13c}$$

where D is the "high" digital voltage value. The conversion term in parentheses in Eq. (14.13a) exactly coincides with Eq. (14.12d) for binary-to-decimal conversion given at the beginning of the previous section. Equation (14.13) holds for any number of bits (inputs), N, of the DAC. An extension of the circuit in Fig. 14.15 is straightforward.

Resolution Voltage

The circuit in Fig. 14.15 performs the *scaled digital-to-analog conversion*. The scaling voltage factor Q with the units of volts in Eq. (14.13) is critical. It is called the *resolution voltage* of the DAC. Resolution voltage Q is simply the analog voltage increment per one single binary count. The resolution voltage is often called the *LSB voltage* since this is exactly the voltage represented by a change in the least significant bit in Eq. (14.13a). The resolution voltage may be changed if necessary, by varying the resistor values in Fig. 14.15. The result will then follow Eq. (14.13b).

Example 14.5: An input to a four-bit DAC is a timing sequence shown in Fig. 14.16a. Such a timing sequence is known as a *binary counter*; it represents all four-bit binary numbers in an ascending order, with the time interval of 1 μs. In other words, we *count* all binary numbers with four bits in Fig. 14.16a, with the bit width of 1 μs and with the bit rate of 1 Mbps. It takes exactly 16 μs to count them all. Design a binary-weighted-input DAC circuit which, at the given digital inputs, attempts to output the analog voltage in the form of a linear time dependence, $v_{out} = 2.5 \times 10^5 t$ (V), over time interval from 0 to 16 μs:

A. Present the corresponding circuit diagram.
B. Specify required resolution voltage and necessary resistor values.
C. Plot the output voltage to the DAC to scale versus time.

Solution: In order to model the required time dependence, the increment of the analog voltage per one bit (per one LSB) should be equal to

$$2.5 \times 10^5 \frac{V}{s} \times 1\,\mu s = 0.25\,V \tag{14.14a}$$

The resolution voltage given by Eq. (14.13b) must have exactly the same value,

$$Q = \frac{R_F D}{16 R_W} = 0.25\,V \tag{14.14b}$$

Example 14.5 (cont.): Since $D = 5$ V in Fig. 14.16a, we may choose $R_F = 2$ kΩ, $R_W = 2.5$ kΩ to satisfy Eq. (14.14b). Furthermore, the next obvious choice is $R = 1$ kΩ. With this in mind, all resistor values have been specified and the circuit is completed. The output of the DAC is plotted with the help of Eq. (14.13a) where $Q = 0.25$ V. The corresponding result is shown in Fig. 14.16b. Figure 14.16b clearly demonstrates that the four-bit DAC from Fig. 14.15 provides a staircase approximation of the required linear voltage dependence. Therefore, a *filter* (called *DAC reconstruction filter* or a *postfilter*) may follow the DAC in order to *smooth* the voltage output. Such analog filters are considered in Chapters 9 and 10 of the text.

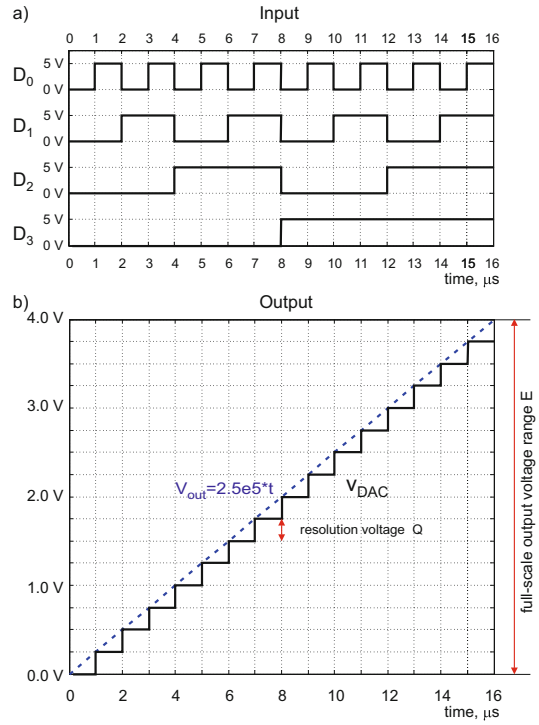


Fig. 14.16. Input digital voltages (a *4-bit binary counter*) and output analog voltage to the 4-bit digital-to-analog converter.

The binary-weighted DAC is one of the *fastest* conversion methods. However, its accuracy is poor, in particular, due to difficulty in manufacturing precise resistors with different binary-weighted values. Realize that the amplifiers and the resistors in Fig. 14.15 are implemented in practice as *integrated transistor circuits*. Also note that the realization of the binary counter with light-emitting diodes in the laboratory is an impressive and useful exercise.

14.2.4 DAC Full-Scale Output Voltage Range, Resolution, and Accuracy

Using the (mostly illustrative) binary-weighted-input DAC circuit as a starting point, we now intend to obtain general facts about DAC resolution and accuracy, which are applicable to any DAC chip, operating over an arbitrary voltage range.

DAC Equation

When all binary numbers are counted in ascending order starting with 0000 toward the maximum 4-bit number of 1111, the circuit in Fig. 14.15 produces an analog output voltage increasing in steps—see Fig. 14.16b above. In order to obtain the range of variation of the output voltage in a general case, we may want to rewrite Eq. (14.13) as

$$v_{\text{DAC}} = E\left(\frac{d_3}{2} + \frac{d_2}{4} + \frac{d_1}{8} + \frac{d_0}{16}\right) \tag{14.15a}$$

which is the commonly used *DAC equation*. Here, the constant E with the units of volts,

$$E = 2^N Q, \quad N = 4 \tag{14.15b}$$

is the *full-scale output voltage range* of the DAC, and Q is its resolution voltage. For a 4-bit DAC, the maximum binary number is 1111, and Eq. (14.15a) yields

$$v_{\text{DAC}}|_{\text{max}} = \frac{2^N - 1}{2^N} E = \frac{15}{16} E \approx E \tag{14.15c}$$

Clearly, the maximum output voltage approaches E more and more precisely as the number of bits, N, increases. Equation (14.15) no longer relies upon the specific DAC circuitry. They are valid for any DAC chip. Equation (14.15b) indicates that the resolution voltage and the full-scale output voltage range are simply related to each other by a factor of 2^N. This factor is exactly the number of distinct binary words or *quantization levels* for a DAC with N inputs (input bits). Note that Eq. (14.15a) is perhaps the most useful DAC formula, often present in the datasheets.

Setting Output Voltage Range

In a realistic DAC chip, the combinations of voltage sources and resistors shown in Fig. 14.15 (resistor "current sources") are replaced with the transistor-based "current sources." Therefore, it is not necessary to change resistor values in order to obtain different voltage ranges and resolution voltages, as might appear at first sight from Eqs. (14.13b) and (14.15b). Instead, the output voltage range, E, and simultaneously the resolution voltage Q are simply controlled by setting an *external voltage reference*, which precisely coincides with the desired value of E. A case in point is a DAC0808 chip (an 8-bit DAC) from National Semiconductor Corp. (Texas Instruments).

DAC Resolution

As long as the full-scale output voltage range of the DAC, E, is given, its resolution voltage Q (the LSB voltage) per one digital step is solely determined by the number of bits N. According to Eq. (14.15b),

$$Q = \frac{E}{2^N} \tag{14.16a}$$

The larger the number of bits is, the smaller resolution voltage can be obtained, and the "smoother" analog voltage is produced. DACs with 8, 10, 12, and 16 bits are common. The DAC *resolution* is simply defined as the number of bits, N, or more often as the total number of discrete values it can produce over the full-scale output voltage range:

$$\text{Resolution} = 2^N = \frac{E}{Q} \tag{14.16b}$$

It has been pointed out already that the full-scale output voltage range of a DAC chip may be controlled by specifying an external voltage reference for a DAC chip, V_{REF}. Therefore, voltage E, may be designated in practice as reference voltage or V_{REF}.

DAC Relative Accuracy

The *DAC relative accuracy* is not exactly the deviation of the staircase approximation in Fig. 14.16b from the straight line as might appear at first sight. It is rather accuracy in reproducing an exact analog value, from the given exact binary data. The corresponding error is further divided by the full-scale output voltage range. Such an error is typically in the range of ± ½ LSB (± ½Q). Therefore,

$$\text{Relative Accuracy Percentage} = \frac{1/2Q}{E} \times 100\,\% = \frac{1}{2^{N+1}} \times 100\,\% \tag{14.16c}$$

For example, according to its datasheet, an 8-bit DAC0808 chip from National Semiconductor Corp. (Texas Instruments) has the relative accuracy percentage

$$\frac{1}{2^{8+1}} \times 100\,\% = 0.19\,\% \tag{14.16d}$$

Example 14.6: A 4-bit DAC, a 6-bit DAC, and an 8-bit DAC use a 4-, 6-, or 8-bit binary counter input sequences in order to produce the analog voltage in the form of a linear time dependence, $v_{\text{out}}(t) = 2.5 \times 10^5 t$ (V), over the same time interval from 0 to 16 μs. Determine:

1. DAC resolution in bits (quantization levels)
2. Full-scale output voltage range, E

Example 14.6 (cont.):

3. DAC voltage resolution, Q
4. Necessary bit rate, f_b

and plot output analog voltages to scale versus time.

Solution: The first parameter is only DAC specific, and the second parameter is only problem specific, that is, $E = v_{out}(16\,\mu s) = 4$ V. The rest of the parameters are both DAC and problem specific. One thus has

4-bit DAC

- DAC *resolution in bits* is 4 bits or $2^4 = 16$ quantization levels (distinct analog outputs)
- Full-scale output voltage range $E = 4$V (from 0 V to 4 V)
- DAC voltage resolution, $Q = E/16 = 0.25\,\text{V}$
- Necessary bit rate, $f_b = 1/T_b = 1\,\text{Mbps}$

6-bit DAC

- DAC *resolution in bits* is 6 bits or $2^6 = 64$ quantization levels (distinct analog outputs)
- Full-scale output voltage range $E = 4\,\text{V}$ (from 0 V to 4 V)
- DAC voltage resolution, $Q = E/64 = 62.5\,\text{mV}$
- Necessary bit rate, $f_b = 1/T_b = 4\,\text{Mbps}$

8-bit DAC

- DAC *resolution in bits* is 8 bits or $2^8 = 256$ quantization levels (distinct analog outputs)
- Full-scale output voltage range $E = 4\,\text{V}$ (from 0 V to 4 V)
- DAC voltage resolution, $Q = E/256 = 15.6\,\text{mV}$
- Necessary bit rate, $f_b = 1/T_b = 16\,\text{Mbps}$

The voltage resolution indeed improves with increasing the number of bits. The corresponding plots are shown in Fig. 14.17.

Exercise 14.5: Generate Fig. 14.17 using MATLAB.

Answer:

```
N       = 6;        %   DAC resolution (bits)
T       = 16;       %   Time interval in microseconds
Q       = 4/2^N;    %   Voltage resolution of the DAC
q       = T/2^N;    %   Time resolution (bit width)
t       = [0:q:T];  %   Time array
vDAC    = [0:Q:4];  %   Analog output array
stairs(t, vDAC);    %   Stairstep graph - plot of discrete data
axis([0 16 0 4]); grid on;
```

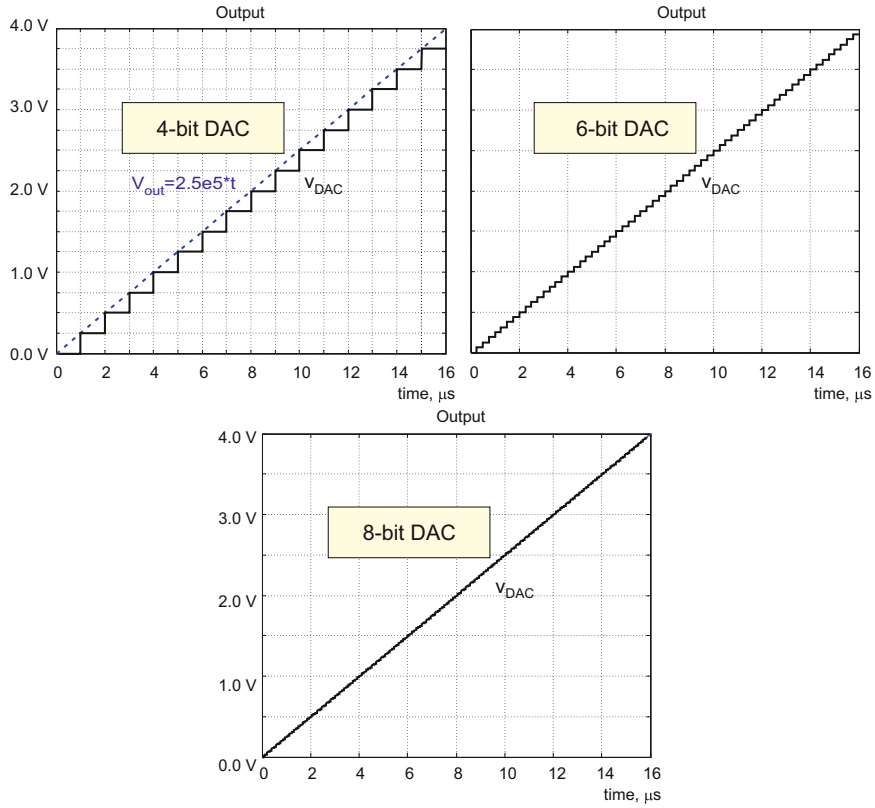


Fig. 14.17. Comparative resolution of the 4-bit DAC, the 6-bit DAC, and the 8-bit DAC. In all three cases, the full-scale output voltage range (4 V) is the same. The linear time dependence, $v_{\text{out}}(t) = 2.5 \times 10^5 t$ (V), over the time interval from 0 to 16 μs is approximated.

14.2.5 Other DAC Circuits

R/2R Ladder DAC

An alternative to the circuit in Fig. 14.15 is an R/2R *ladder* DAC, which is shown (without the inverter) in a 4-bit configuration in Fig. 14.18. The *resistive ladder* is a circuit, which has a similar performance when more sections are added. This circuit requires a more careful analysis based on Thévenin equivalents, but the final result exactly coincides with Eq. (14.15a), that is (after adding the inverter),

$$v_{\text{DAC}} = E\left(\frac{d_3}{2} + \frac{d_2}{4} + \frac{d_1}{8} + \frac{d_0}{16}\right) \tag{14.17a}$$

where the full-scale voltage range is given by

$$E = 2^N Q = \frac{R_F D}{R} \tag{14.17b}$$

The binary-weighted-input DAC and the R/2R ladder DAC become identical in operation when $R_W = R$.

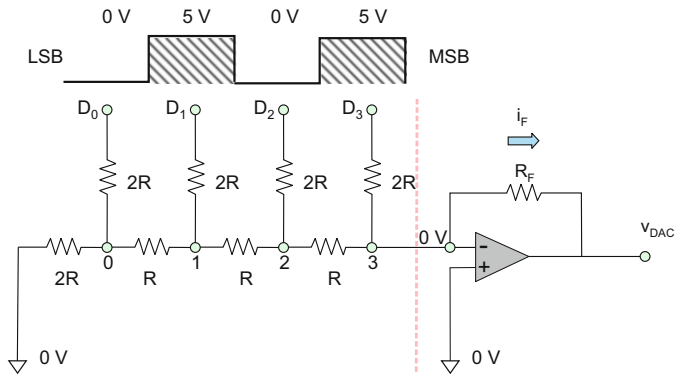


Fig. 14.18. The R/2R ladder DAC.

Example 14.7: For the circuit in Fig. 14.18, $D_3 = 5\,\text{V}$, $D_2 = 0\ \text{V}$, $D_1 = 0\,\text{V}$, $D_0 = 0\ \text{V}$. Determine the output voltage, v_{DAC}, when $R_F = R$.

Solution: The voltages at nodes 0,1,2,3 in Fig. 14.18 are all equal to zero (the corresponding resistors are shorted out). The current $i_3 = \frac{5\ \text{V}}{2R}$ from the input D_3 flows entirely into the feedback resistor, which yields

$$v_{\text{DAC}} = -\frac{D_3 R_F}{2R} = -\frac{5\ \text{V} \times R_F}{2R} = -2.5\,\text{V} \tag{14.18}$$

This (after adding the inverter) is exactly the first term on the right-hand side of Eq. (14.17a). The analysis of other individual input voltage combinations is suggested as homework problems.

The use of the ladder network improves the *precision* of the binary-weighted-input DAC since it is easier to produce equal resistors. Therefore, a typical DAC chip rather uses the R/2R ladder network in the form of an integrated circuit, where the combinations of voltage sources and resistors shown in Fig. 14.18 (resistor "current sources") have been replaced with the transistor-based "current sources." A case in point is again a DAC0808 chip (an 8-bit DAC) from National Semiconductor Corp. (Texas Instruments).

The "Power" of Thévenin Equivalent

An attempt to solve the ladder circuit in Fig. 14.18 for combinations other than the simple input combination from Example 14.7 meets considerable difficulties. We outline below a general Thévenin-equivalent-based solution that is shown in Fig. 14.19. First, the circuit

of Fig. 14.18 is redrawn in the form of the equivalent voltage sources D_3, D_2, D_1, D_0 referenced to ground in Fig. 14.19, top.

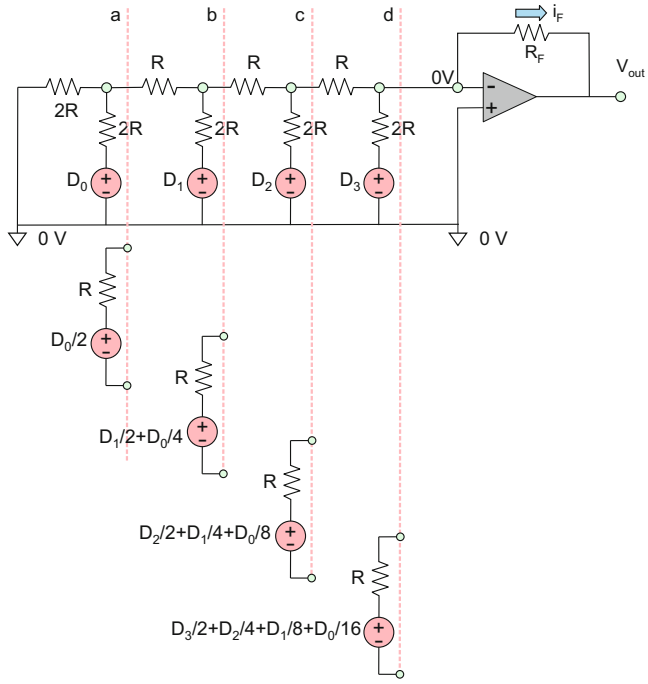


Fig. 14.19. Solving R/2R ladder DAC using the method of Thévenin equivalent while adding one ladder section at a time.

The resulting circuit block on the left of the amplifier (the ladder) contains only voltage sources and resistors. The method of Thévenin equivalent is applied in order to convert it to a form of a single voltage source V_T in series with a resistance R_T. The brute-force calculation of Thévenin resistance for the entire ladder block is rather simple. However, a calculation of Thévenin voltage V_T is not. The resistance calculation (with shorted out voltage sources) gives $R_T = R$. But what about V_T? The idea (which is also applicable to other ladder networks) is to apply Thévenin equivalent in steps, adding one single section a,b,c,d of the ladder block at a time, starting with the leftmost section a in Fig. 14.19. Every such step is analyzed straightforwardly; it gives Thévenin voltages and resistances shown in Fig. 14.19. The last step in Fig. 14.19 followed by solving the inverting amplifier circuit leads us exactly to Eq. (14.17) if an extra inverter is added.

PWM DAC

One should mention a digital PWM (pulse-width modulation) code, which is digitally stored and then converted to an analog signal by means of an analog RC filter studied in Chapter 9. Such a technique is becoming increasingly popular today.

Section 14.3 Sample-and-Hold Circuit: Nyquist Rate

14.3.1 Analog-to-Digital Converter

The analog-to-digital converter (ADC) studied in this and next sections should translate a continuously varying analog voltage (voice, electromagnetic signal, readout of a sensor) into a continuous stream of binary numbers (and equivalent digital voltages) passed to a processor. The digital-to-analog converter (DAC) studied in the previous section performs an inverse operation. The place of the ADC in the generic block diagram of a DSP (digital signal processor) is shown in Fig. 14.20. The ADC circuit analysis is in general much more involved than the DAC circuit analysis. ADC design is an exciting and growing area of the electrical engineering with many hundreds of engineers employed.

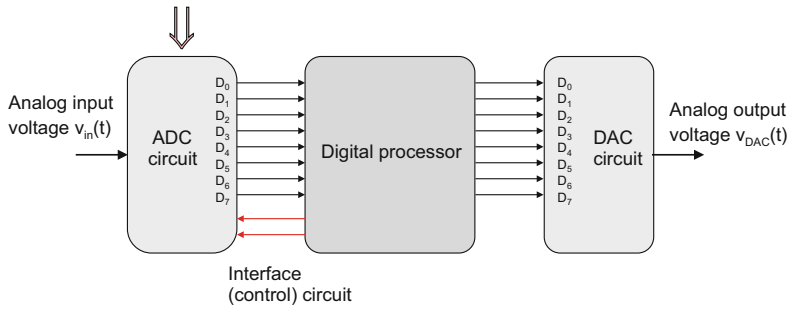


Fig. 14.20. Place of the ADC in the DSP block diagram.

14.3.2 A Quick Look at an Analog Sinusoidal Voltage

First, an input analog voltage to the circuit in the figure above should be analyzed. The simplest and simultaneously the most important case of the analog voltage is a pure sine or cosine function shown in Fig. 14.21 and also called the *harmonic*. The sinusoidal voltages are critical in AC circuit analysis studied previously. Why are we interested in a sinusoidal voltage input also in this chapter? The reason is that, according to the method of Fourier analysis studied in Chapter 9, all existing continuous voltages $v_{in}(t)$, including voltage signals corresponding to the human voice, may be expanded into a sum of such multiple sinusoidal functions. Every sinusoidal function will have its own frequency, phase, and amplitude. The sinusoidal voltage can be written in the form

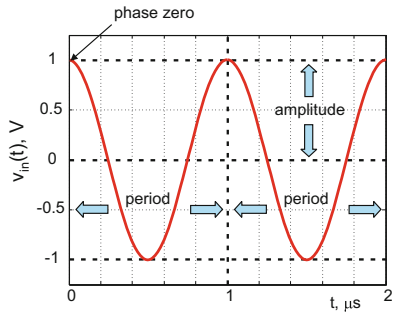


Fig. 14.21. Analog continuous voltage in the form of a cosine function.

$$v_{\text{in}}(t) = V_{\text{m}} \cos(\omega t + \varphi) \tag{14.19a}$$

where:

V_{m} is the voltage *amplitude* (maximum *absolute* voltage), with the unit of volts.
ω is the *angular frequency*, with the unit of radians/sec.
φ is the *phase*, with the unit of radians.

The angular frequency relates to the voltage's frequency, f, and period, T, by

$$\omega = 2\pi f, \quad T = \frac{1}{f} \tag{14.19b}$$

where f is measured in hertz or Hz (1 Hz = 1 s^{-1}) and the period is measured in seconds. For example, the frequency of the voltage signal in Fig. 14.21 is

$$f = \frac{1}{T} = \frac{1}{1\ \mu\text{s}} = 1\ \text{MHz} \tag{14.19c}$$

The angular frequency ω is essentially a replica of the frequency f; its use is primarily a matter of convenience. The phase in Eq. (14.19a) ranges from -π to +π radians; this corresponds in degrees to $-180°$ to $+180°$. Positive phases correspond to a shift of the entire sinusoidal signal in Fig. 14.21 to the left and negative phases to the right. For example, the phase $\varphi = +\pi/2$ implies shifting of the sinusoidal signal in Fig. 14.21 to the left by a quarter period. The phase in degrees should be divided by 180 and multiplied by π to obtain the phase in radians. The phase is a *relative* measure; it is given with reference to a base signal (usually a plain cosine function $\cos \omega t$).

Exercise 14.6: Determine frequency in Hz and angular frequency in rad/sec, phase, and amplitude of a harmonic voltage signal shown in Fig. 14.21.

Answer: $f = 1/T = 1$ MHz, $V_{\text{m}} = 1$ V, the phase is exactly zero.

14.3.3 Sample-and-Hold Voltage

As compared to D to A conversion, the A to D conversion is generally (much) slower, due to the following reasons:

- First, we need to *sample* the analog signal (acquire its voltage value) as shown in Fig. 14.22.
- Second, we need to *hold* the sampled voltage value as long as it is necessary for the conversion of this value to a binary number. Then, the next value of the analog signal is acquired and held, so that the process repeats itself periodically.

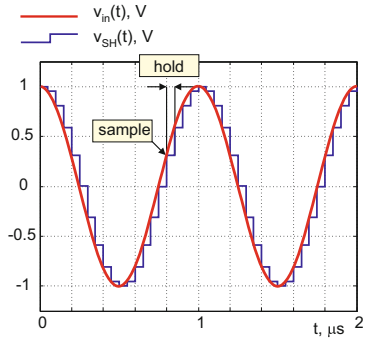


Fig. 14.22. Analog continuous voltage versus sample-and-hold voltage.

The sample-and-hold concept is illustrated in Fig. 14.22. The input (analog) voltage $v_{\text{in}}(t)$ is to be converted into a staircase voltage $v_{\text{SH}}(t)$—the *sample-and-hold* approximation of the input signal. Realize that $v_{\text{SH}}(t)$ is not yet the digital voltage corresponding to A to D conversion but rather the first step in doing so. Every interval between two consecutive samples in Fig. 14.22 is called the *sampling interval*, T_{S}. The *sampling rate* (*sampling frequency*), which is the inverse of the sampling interval, defines the number of samples taken per second, that is,

$$f_{\text{S}} = \frac{1}{T_{\text{S}}} \tag{14.20}$$

The unit for sampling rate is hertz. The sampling interval is in fact the sum of the *hold time* and a (typically much shorter) *acquisition* (or *sample*) *time*, which is necessary, for example, to charge the capacitor in Fig. 14.23 that follows. The acquisition time is not seen in Fig. 14.22 due to insufficient resolution.

Exercise 14.7: Determine sampling interval and sampling rate for the sample-and-hold voltage in Fig. 14.22.

Answer: $T_{\text{S}} = 0.2/4 = 0.05$ µs, the sampling rate is $f_{\text{S}} = 20$ MHz. The sampling frequency is *much higher* than the signal frequency of 1 MHz.

Example 14.8: Generate the plot for the sample-and-hold voltage shown in Fig. 14.22 using MATLAB.

Solution: The idea is to plot the "analog" voltage (still the discrete MATLAB array, but with the sufficiently fine time resolution) versus the sample-and-hold voltage. In the last case, we use the MATLAB function `stairs`, which allows us to plot the discrete data.

```
%   Analog voltage signal
T       = 2e-6;             %   Time interval, sec
t       = [0:T/1e3:T];      %   Time array ("analog" time)
f       = 1e6;              %   Frequency of the analog signal, Hz
vin     = cos(2*pi*f*t);    %   Analog input voltage, V
%   Sample-and-hold voltage signal
dt      = 0.05e-6;          %   Sampling time, sec
ts      = [0:dt:T];         %   Time array
vSH     = cos(2*pi*f*ts);   %   Sample-and-hold voltage, V

plot  (t, vin, 'r');                    %   Analog input voltage
hold on;
stairs(ts, vSH,  'b', 'LineWidth', 2);    %   Sample-and-hold voltage
grid on; axis([min(t) max(t), -1.25 1.25])
```

14.3.4 Sample-and-Hold Circuit (SH Circuit)

Which circuit could convert the analog voltage $v_{in}(t)$ into the sample-and-hold voltage $v_{SH}(t)$ in Fig. 14.22? One such generic design is shown in Fig. 14.23a:

1. First, we need to isolate the acquired analog voltage from the rest of the circuit—introduce the input non-inverting buffer amplifier. Once the "sample" switch is on, the buffer amplifier acquires the voltage sample and charges the capacitor C_{hold} to exactly the same value.
2. This capacitor has no discharge path; it therefore keeps the sampled voltage as long as required. The output buffer further conveys the sampled voltage to the rest of the circuit without changing it (without discharging the capacitor).
3. However, when the "reset" switch closes, the voltage-holding capacitor voltage quickly discharges, i.e., turns its voltage to zero. In other words, the past voltage value is "erased." Then, the reset switch opens again, which makes the circuit ready for the next acquisition cycle. Both switches could be implemented with n-channel MOSFET transistors as shown in Fig. 14.23b.

The basic process of charging/discharging a capacitor was studied in Chapter 7. Indeed, it requires some time, which is very short when the capacitor value is small enough. Such a transition time was ignored in Fig. 14.22.

Sample-and-Hold Circuits in Practice

The sample-and-hold circuits exist as single chips (LF198/LF298/LF398 chips from National Semiconductor Corp. or Texas Instruments). The typical capacitance used

there is $C_{\text{hold}} = 0.02$ μF. Those circuits may be used to hold a given voltage value from any particular sensor in a robot or elsewhere.

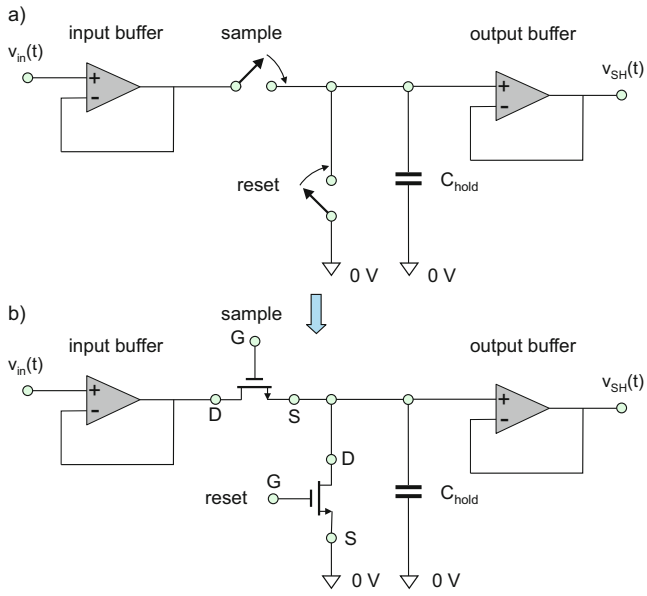


Fig. 14.23. Sample-and-hold circuit. (**a**) Circuit with "sample" and "reset" switches. (**b**) Switches replaced by MOSFET transistors.

However, usually, the sample-and-hold circuit is an internal part of the ADC chip as explained further in the next section. Another popular modification is the *track-and-hold circuit* suggested as one of the homework problems. Could an ADC function without the sample-and-hold circuit? Yes, it could. This is exactly the way how first ADCs have been made. However, it means that the voltage value to be converted will not be held; it will change during the conversion process, which may lead to wrong bits.

14.3.5 Nyquist Rate

How fast should we sample? The answer seems to be trivial: as fast as possible in order to acquire the most precise replica of the input signal. However, a very fast sampling rate may be either impossible in practice for ultrafast signals, or it may lead to huge and unrealistic memory consumptions. On the other hand, reducing the sampling rate, while still keeping the major information about the analog voltage behavior, allows us to proceed with a realistic circuit design and realistic memory requirements. Therefore, a *minimum acceptable sampling rate*, $f_{\text{S min}}$, for a given analog voltage signal is of great practical importance. In order to establish this minimum value, we will again consider the sinusoidal voltage of 1 MHz shown in Fig. 14.24. We introduce the *Nyquist rate*, f_{N}, of this voltage signal, which is two times its frequency, i.e.,

$$f_N = 2f \tag{14.21}$$

We further analyze three distinct cases in Fig. 14.24:

1. When the sampling frequency is higher than the Nyquist rate—see Fig. 14.24a—then the sample-and hold voltage generally follows the signal shape. After proper filtering (smoothing the stairs in Fig. 14.24a), it will very well replicate the original analog sinusoid.
2. When the sampling frequency is exactly equal to the Nyquist rate—see Fig. 14.24b—then the sample-and hold voltage becomes exactly the pulse train. And yet, after proper filtering, this pulse train may be converted to the sinusoidal function of the same frequency—the reconstruction may be successful.
3. However, when the sampling frequency is less than the Nyquist rate—see Fig. 14.24c—then the sample-and-hold voltage may become just a straight line. No matter what do we do further, the signal information is entirely lost!

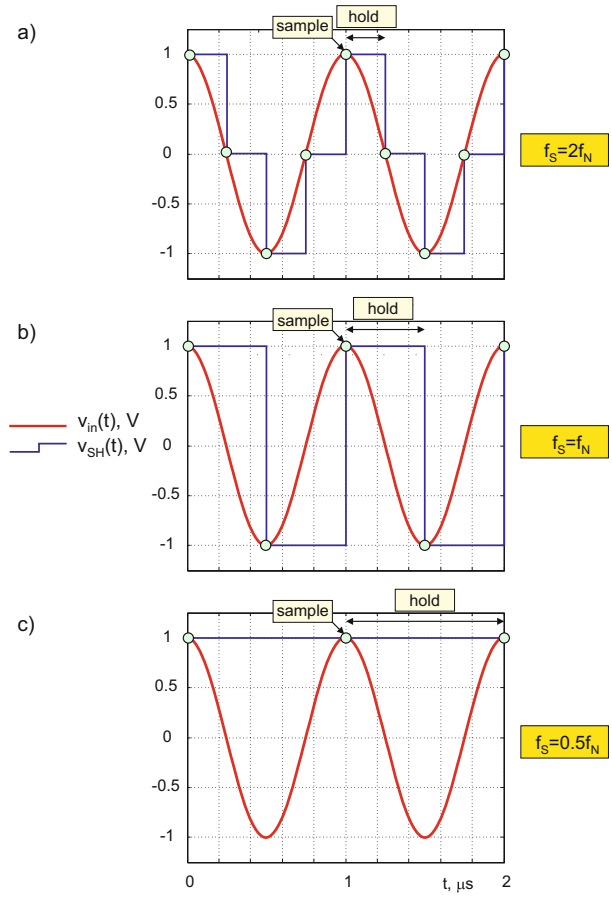


Fig. 14.24. Effect of sampling rate on the sample-and hold voltage.

Based on this reasoning, we arrive at the conclusion that the minimum acceptable sampling rate must exceed the Nyquist rate, that is,

$$f_{\text{Smin}} > f_{\text{N}} \tag{14.22}$$

Equation (14.22) is the first (and critical) step of *digital signal processing*. It constitutes a significant part of *Nyquist-Shannon sampling theorem* studied in ECE communication classes. For example, the sinusoidal voltage in Fig. 14.24 may only be reconstructed properly when the sampling frequency is higher than 2 MHz. What if an analog voltage is not a pure sinusoidal function but is a (weighted) sum of many sinusoids with different frequencies, e.g., a voice signal? In this case, Eq. (14.22) must be valid for the sinusoid having the *highest frequency* among the others. All other sinusoidal functions will then satisfy this condition automatically.

Historical: The Nyquist rate was named after famous Swedish-American engineer Harry Nyquist (1889–1976). Harry Nyquist received his BS and MS in electrical engineering from the University of North Dakota and PhD from Yale University. Interestingly, the term "Nyquist rate" itself was first introduced by Harold S. Black (remember negative feedback?), in his book *Modulation Theory* (1953).

Oversampling and Undersampling

Oversampling is the process of sampling a signal with sampling frequency significantly higher than the Nyquist rate:

$$f_{\text{S}} >> f_{\text{N}} \tag{14.23}$$

Clearly, once possible in practice, oversampling has many potential advantages. Undersampling is the opposite of oversampling. Although generally undesired, undersampling finds numerous (and really smart!) applications for modulated signals in wireless communications.

Example 14.9: An analog voltage signal is a combination of three sinusoidal voltages (*harmonics*) with frequencies 5 kHz, 10 kHz, and 15 kHz. The voltage amplitudes of the individual sinusoids are 5 V, 10 V, and 5 V. What is the limit on the minimum acceptable sampling rate of the sample-and-hold circuit?

Solution: We apply Eq. (14.22) to the *highest-frequency* sinusoid present in the signal and obtain

$$f_{\text{Smin}} > f_{\text{N}} = 2 \times 15 \text{ kHz} = 30 \text{ kHz} \tag{14.24}$$

The amplitudes (and phases) of individual harmonics play no role for this estimate, as long as their amplitudes are not zero. However, if the amplitude of a certain harmonic is zero (or very small), this harmonic may be eliminated from the analysis entirely.

Section 14.4 Analog-to-Digital Converter

The sample-and-hold (SH) circuit studied previously is only the "front-end" of the analog-to-digital converter. Even if this circuit is available, the A to D converter itself still needs to be designed. Therefore, the ADC block in Fig. 14.25 still needs to be studied. This section completes the analog-to-digital converter model. We will consider only two basic ADC circuit concepts (flash and successive approximation) leaving many others.

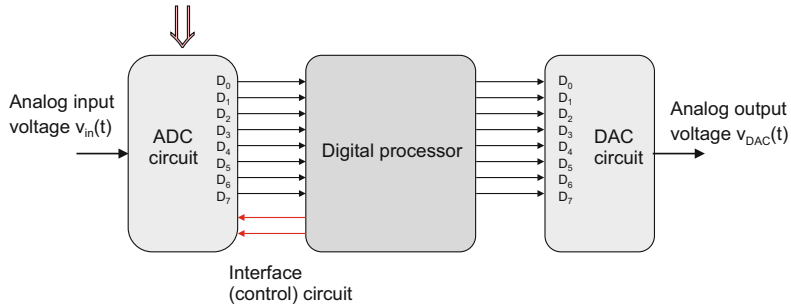


Fig. 14.25. DSP block diagram. The ADC converter still needs to be completed.

14.4.1 Flash ADC

Circuit

The next step in ADC design is to convert the sample-and-hold voltage $v_{SH}(t)$ to the digital voltage itself. Figure 14.26 shows the concept of a *flash* A to D converter. This is the fastest A to D conversion method. All data are essentially processed in parallel. The sample-and-hold circuit is not shown in Fig. 14.26, but it is implied. The circuit in Fig. 14.26 is intended to encode the sample-and-hold voltage into 3-bit binary numbers or 3-bit digital voltages D_2, D_1, D_0. It includes (from left to right):

1. A voltage divider with eight equal resistors, R. The voltage divider subdivides the reference voltage, E, into eight levels corresponding to eight possible 3-bit words: 000, 001, 010, 011, 100, 101, 110, and 111. The reference voltage E to the ADC chip simultaneously determines the *full-scale measurement voltage range* of the ADC studied below.
2. Seven open-loop comparator amplifiers (the comparator for a 000 condition is not needed) compare the input voltage with the seven nontrivial voltage levels of the voltage divider. The comparators generate high voltage when $v_{SH}(t)$ exceed the corresponding voltage divider level and low voltage otherwise.
3. The output of the comparator block—the second column of Table 14.4.
4. Finally, an ADC *encoder block*. This circuit converts binary words from the comparator block to the binary numbers. It may be constructed with logic gates.

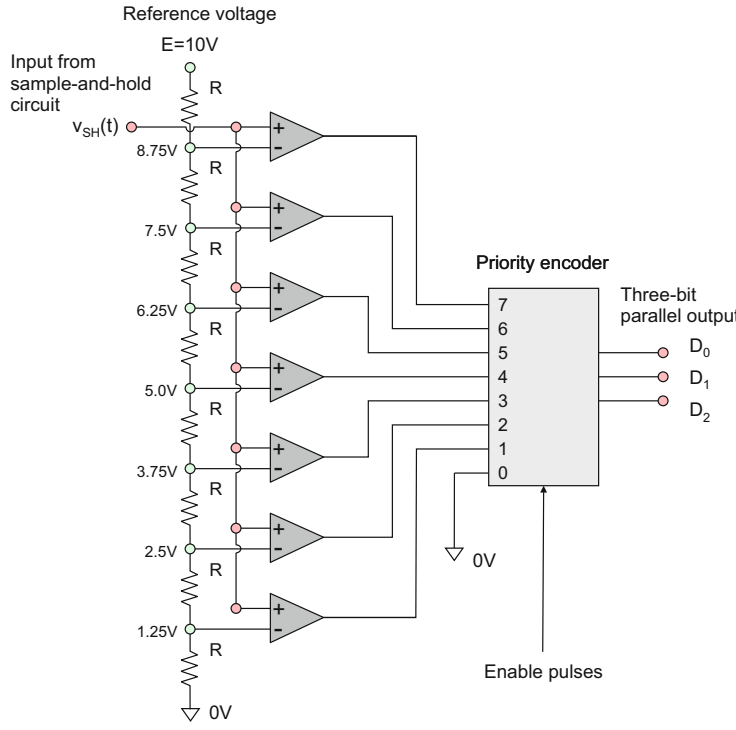


Fig. 14.26. A 3-bit flash ADC.

Operation

Table 14.4 lists circuit parameters for the voltage divider analysis in Fig. 14.26.

Table 14.4. Output of comparators (second column) and output of the entire ADC chip (third column).

Voltage range of sample-and-hold voltage v_{SH}, V	Output of the comparator block (7–0)	Output of the priority encoder (binary number $d_2d_1d_0$)	Voltage decoded back from $d_2d_1d_0$ and resolution Q, $Q(4d_2 + 2d_1 + 1d_0)$ V	Quantization error, max
8.75–10	11111110	111	8.75	1.25 V or Q
7.5–8.75	01111110	110	7.5	1.25 V or Q
6.25–7.5	00111110	101	6.25	1.25 V or Q
5–6.25	00011110	100	5	1.25 V or Q
3.75–5	00001110	011	3.75	1.25 V or Q
2.5–3.75	00000110	010	2.5	1.25 V or Q
1.25–2.5	00000010	001	1.25	1.25 V or Q
0–1.25	00000000	000	0	1.25 V or Q

14.4.2 ADC Resolution in Bits, Full-Scale Input Voltage Range, and Voltage Resolution

Very similar to the digital-to-analog converter considered in Section 14.2, any ADC (not only the flash ADC) is characterized by:

- *Resolution in bits*, N.
- *Full-scale measurement (or input) voltage range*, E.
- *Voltage resolution*, $Q = \frac{E}{2^N}$.
- *Relative accuracy* (typically 1 LSB), which gives the accuracy magnitude percentage of $\frac{1}{2^N} \times 100\ \%$. The relative accuracy will be explained next.

We emphasize again that the reference voltage E to the ADC chip in Fig. 14.26 is exactly its full-scale measurement voltage range. The reference voltage is specified in the circuit depending on the problem under study.

Exercise 14.8: For the 3-bit ADC in Fig. 14.26, determine its resolution, full-scale measurement voltage range, and voltage resolution.

Answer

- ADC resolution in bits is 3 bits or $2^3 = 8$ quantization levels (distinct digital outputs).
- Full-scale measurement voltage range $E = 10$ V (from 0 V to 10 V).
- ADC voltage resolution, $Q = E/8 = 1.25$ V.

14.4.3 ADC Equation and Quantization Error

ADC Equation

It is seen from Table 14.4 that the 3-bit ADC in Fig. 14.26 follows an *ADC equation* in the form

$$\text{ADC}_{\text{code}} = \text{floor}\left(\frac{v_{\text{SH}}}{Q}\right) \tag{14.25a}$$

where ADC_{code} is a *binary* number corresponding to an integer on the right-hand side of Eq. (14.25a). A function `floor(x)` rounds its argument x to the nearest integers less than or equal to x. This function is used in MATLAB and in other software packages. Equation (14.25a) corresponds to a *mid-rise coding scheme*. Its error—the difference between the original signal v_{SH} and the digitized and restored back signal—is clearly Q or one LSB. However, a different coding scheme, the *mid-tread coding scheme* may reduce the error magnitude to ½ Q or ½ LSB. It follows an ADC equation in the form

$$\text{ADC}_{\text{code}} = \text{round}\left(\frac{v_{\text{SH}}}{Q}\right) \tag{14.25b}$$

A function `round(x)` rounds its argument x to the nearest integer. This function is also used in MATLAB and in other software packages. Equation (14.25b) corresponds to a *mid-tread coding scheme*. Equations (14.25a) and (14.25b) are valid for any ADC type, not only the flash ADC. Note that the flash-ADC circuit in Fig. 14.26 may be modified to follow Eq. (14.25b). Only a slight modification of the circuit is needed! The corresponding homework problem is suggested at the end of this chapter.

Quantization Error

The *quantization error* or *quantization noise* is the difference between the original signal v_{SH} and the digitized signal restored back – compare the first and last columns in Table 14.4 above. It is *exactly* the error of Eq. (14.25a) or Eq. (14.25b), nothing else. One has,

- For the coding scheme following Eq. (14.25a), the quantization error is $1Q$ or 1 LSB.
- For the coding scheme following Eq. (14.25b), the quantization error is $\pm½\ Q$ or ±½ LSB (error magnitude is ½ Q or ½ LSB).

Quantization error is due to the finite resolution of the digital number; it is an intrinsic imperfection of any ADC and cannot be avoided. The ADC relative accuracy considered in the previous subsection includes the quantization error and other sources of error.

Example 14.10: A 4-bit flash ADC follows a mid-rise coding scheme. The reference voltage is 10 V:

1. Present the ADC equation.
2. Determine ADC quantization error.
3. Find ADC output code when the sample-and-hold voltage, v_{SH}, is 3.01 V.

Solution: We use Eq. (14.25a)

$$\text{ADC}_{\text{code}} = \text{floor}\left(\frac{v_{\text{SH}}}{Q}\right) \quad \text{where} \quad Q = E/2^N = 0.625\ \text{V} \tag{14.25c}$$

The quantization error is 1LSB or 0.625 V. The ADC code is 0100, that is,

$$\text{floor}(3.01/0.625) = 4 = 0100_{\text{binazry}} \tag{14.25d}$$

ADC Speed, Throughput Rate, and Conversion Time

An important parameter of the ADC is its *conversion time*. Conversion time is the time required for a complete measurement by an analog-to-digital converter. Since the conversion time does not include acquisition time of the sample-and-hold circuit, the

conversion time may be less than the ADC throughput time. A case in point is an 8-bit ADC0820 from National Semiconductor Corp. (Texas Instruments), which internally uses two 4-bit flash A to D converters with the conversion time of 1.5–2.5 μs. The ADC *speed* (or *throughput rate*) is the maximum rate at which the A–D converter can output data values. The speed in measured in *ksps* (kilo samples per second) or *Msps* (mega samples per second). For ADCs with the sample-and-hold circuits, the speed is the inverse of the conversion time plus the acquisition time. For ADCs without the sample-and-hold circuit (early versions of the ADCs), the speed is the inverse of the conversion time only. A modern trend is to increase the throughput rate by using a *pipelined A to D converter*, so a second conversion can start while the first is still in progress.

14.4.4 Successive-Approximation ADC

Concept

The flash ADC shown in Fig. 14.26 suffers from a huge number of comparators to be used when a fine resolution is necessary. Another clever idea is to use the D to A converter considered previously. Once driven by a binary counter, this converter produces various analog outputs corresponding to different binary numbers. Then, the circuit compares such outputs with the actual sample-and-hold voltage using *only one* comparator amplifier in Fig. 14.27. The closest result is the desired binary number.

Circuit

The corresponding circuit diagram (for a 4-bit ADC) is shown in Fig. 14.27. The A to D converter so constructed is known as a *successive-approximation ADC* for a reason that will be explained below. It is slower than the flash ADC but is still faster than many other ADC techniques. This is perhaps the most popular ADC design today.

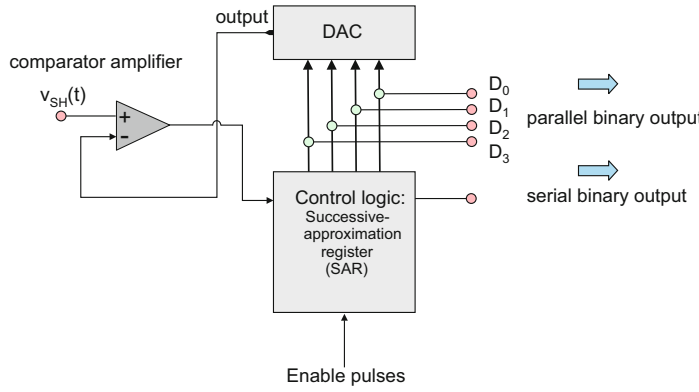


Fig. 14.27. A 4-bit successive-approximation ADC. Note that both parallel and serial outputs are allowed.

How Does It Work?

The brute-force comparison, e.g., counting all binary numbers starting with zero and waiting until the corresponding voltage difference changes its sign, is not very appealing. The logic control block in Fig. 14.27 uses a smarter, the *successive-approximation* comparison algorithm, which actually dates back to the sixteenth century. The corresponding logic circuit is known as *successive-approximation register* (SAR).

Historical: Long ago, an Italian mathematician Niccolò Fontana Tartaglia (1500–1557) was posed with a problem of determination of an unknown weight by a minimal sequence of weighing operations. His suggestion was to use a binary series of weights, e.g., 8 lb, 4 lb, 2 lb, 1 lb, etc. (1000, 0100, 0010, 0001 in terms of binaries). The proposed weighing algorithm found its application in modern successive-approximation ADCs.

For simplicity, we assume that the 4-bit DAC in Fig. 14.27 has the output of 4 V for the MSB (D_3 high), of 2 V for D_2 high, of 1 V for D_1 high, and of 0.5 V for the LSB (D_0 high). The sample-and-hold voltage is $v_{SH} = 2.6$ V. The DAC resolution voltage, Q, is then 0.5 V—exactly the LSB voltage. The DAC equation (Eq. (14.15a) of Section 14.2) is written in terms of Q in the form

$$v_{DAC} = Q(8d_3 + 4d_2 + 2d_1 + d_0) \tag{14.26}$$

The corresponding comparison sequence follows:

- First, the MSB only (binary word 1000) is compared. The result (4 V) is greater than v_{SH}; therefore, this bit is reset to zero (output of the comparator is low).
- Second, the D_2 (binary word 0100) is compared. The result (2 V) is less than v_{SH}; therefore, this bit is kept (output of the comparator is high).
- Third, the D_1 (binary word 0110 with D_2 high) is compared. The result (3 V) is greater than v_{SH}; therefore, this bit is reset to zero (output of the comparator is low).
- Finally, the LSB (binary word 0101 with D_2 high) is compared. The result (2.5 V) is less than 2.6 V; therefore, it is kept (output of the comparator high). The obtained binary number is 0101.

Figure 14.28 shows the corresponding comparison sequence, which is done in four steps, instead of 15 steps of the worst-case scenario with the binary counter.

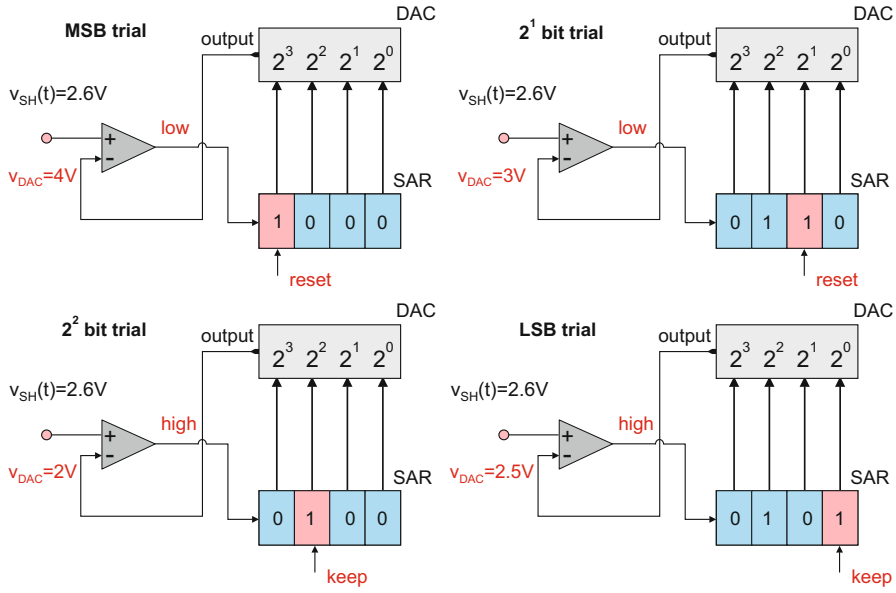


Fig. 14.28. Operation of a 4-bit successive-approximation ADC.

Example 14.11: A 5-bit successive-approximation ADC has the input voltage of $v_{SH} = 2.05$ V; the DAC resolution voltage is 0.1 V. Determine the sequence of binary states and the final ADC output.

Solution: The solution uses the DAC formula for a 5-bit DAC

$$v_{DAC} = Q(16d_4 + 8d_3 + 4d_2 + 2d_1 + d_0) \tag{14.27}$$

and is done in five steps:

- 10000 is compared. The DAC output is $Q \times 16 = 1.6$ V. The comparator output is high; the bit is kept.
- 11000 is compared. The DAC output is $Q \times 24 = 2.4$ V. The comparator output is low; the bit is reset.
- 10100 is compared. The DAC output is $Q \times 20 = 2.0$ V. The comparator output is high; the bit is kept.
- 10110 is compared. The DAC output is $Q \times 22 = 2.2$ V. The comparator output is low; the bit is reset.
- 10101 is compared. The DAC output is $Q \times 21 = 2.1$ V. The comparator output is low; the bit is reset.

The ADC output is 10100 (or 2.0 V when converted back to analog).

Summary

Conversion of binary and hexadecimal numbers

Binary to decimal conversion:	$d_3d_2d_1d_{0\text{binary}} = (8d_3 + 4d_2 + 2d_1 + 1d_0)_{\text{decimal}}$
Hex to decimal conversion:	$378_{\text{hex}} = \mathbf{3} \times 256 + \mathbf{7} \times 16 + \mathbf{8} \times 1 = 888_{\text{decimal}}$
Binary to hex conversion:	$\underbrace{0011}\ \underbrace{0111}\ \underbrace{1000}_{\text{binary}} = 378_{\text{hex}}$
Decimal to binary conversion:	$156_{\text{decimal}} =$ 256 128 64 32 16 8 4 2 1 / 1 0 0 1 1 1 0 0 $= 10011100_{\text{binary}}$

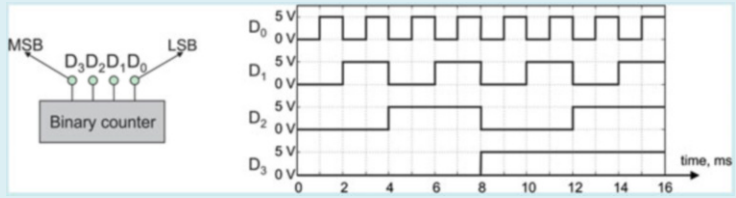
Four-bit binary numbers

Hexadecimal digit	Binary number	Decimal number
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
A	1010	10
B	1011	11
C	1100	12
D	1101	13
E	1110	14
F	1111	15

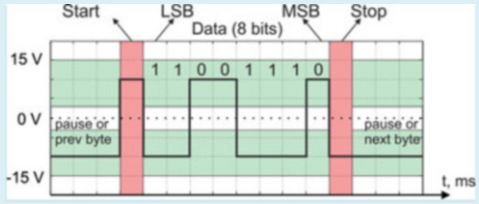
MATLAB conversion

```
bin2dec('10011001')= 153; dec2bin(153) = 10011001; dec2hex(1023)= 3FF
```

4 bit—binary counter

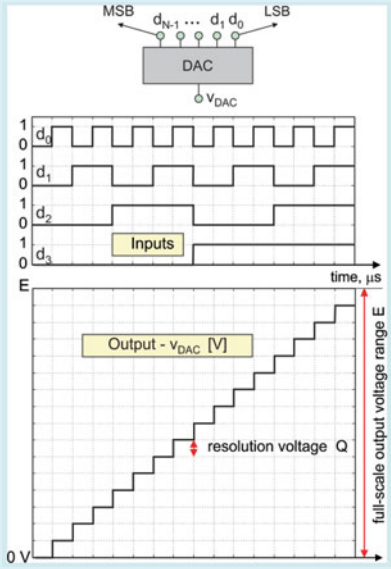


RS232 interface



(continued)

Digital to analog converter (DAC) with N bits



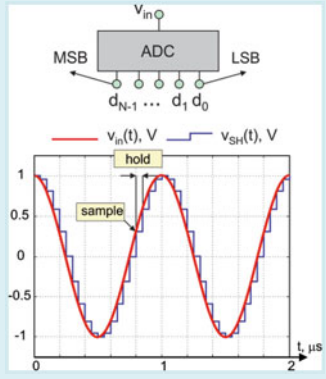
– DAC resolution in bits is N bits or 2^N quantization levels (distinct analog output values);
– Full-scale output voltage range E is equal to DAC reference voltage V_{REF};
– DAC voltage resolution Q (or LSB voltage) is given by $Q = E/2^N$;
– DAC relative accuracy is ½ LSB voltage or 1LSB
– *DAC equation* (a four-bit DAC):

$$v_{DAC} = Q(8d_3 + 4d_2 + 2d_1 + 1d_0) \text{ or}$$

$$v_{DAC} = E\left(\frac{d_3}{2} + \frac{d_2}{4} + \frac{d_1}{8} + \frac{d_0}{16}\right), \quad d_i = 0 \text{ or } 1$$

– $Q = \dfrac{R_F D}{2^N R_W}$ Binary-weighted-inp. DAC
– $Q = \dfrac{R_F D}{2^N R}$ R/2R ladder DAC

Analog to digital converter (ADC) with N bits



– ADC resolution in bits is N bits or 2^N quantization levels (distinct analog outputs);
– Full-scale output voltage range E is equal to ADC reference voltage V_{REF};
– ADC voltage resolution Q (or LSB voltage) is given by $Q = E/2^N$;
– ADC quantization error is ½ LSB or 1LSB
– *ADC equation*:

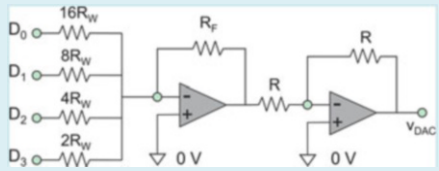
Mid-rise (w offset) coding: $\text{ADC}_{\text{code}} = \text{floor}\left(\dfrac{v_{SH}}{Q}\right)$

Mid-tread coding $\text{ADC}_{\text{code}} = \text{round}\left(\dfrac{v_{SH}}{Q}\right)$

– Minimum acceptable sampling rate must exceed the Nyquist rate (sampling theorem):
– $f_{S\min} > f_N, f_N = 2f$—Nyquist frequency
– f—highest signal frequency

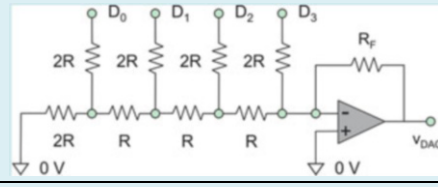
Schematic DAC and ADC circuits

Binary-weighted-input DAC

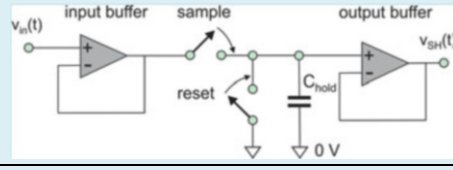


(continued)

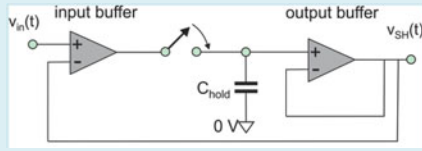
R/2R ladder DAC



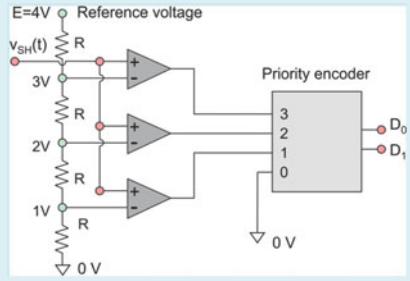
Sample-and-hold circuit



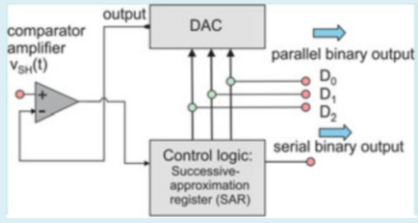
Track-and-hold circuit



Flash ADC



Successive-approximation ADC



Problems

14.1 Digital Voltage and Binary Numbers

14.1.2 Analog Voltage Versus Digital Voltage

14.1.3 Bit Rate. Clock Frequency. Timing Diagram

Problem 14.1. Describe in your own words the meaning of:

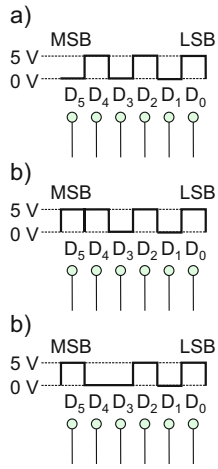
1. Analog voltage
2. Digital voltage
3. A bit
4. A binary number
5. Most significant bit (MSB)
6. Least significant bit (LSB)

Problem 14.2. Convert the following binary numbers:

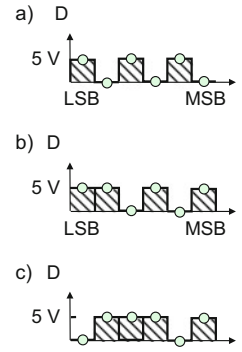
A. 1000001
B. 11111
C. 111111
D. 00001
E. 11110000

to decimal numbers.

Problem 14.3. Determine integer (decimal) numbers represented by parallel digital output voltages shown in the following figure.

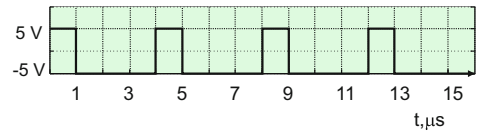


Problem 14.4. Determine integer (decimal) numbers represented by serial digital output voltages shown in the figure. Sampling is made at the center of each bit interval. In part (c), the least significant bit arrives first at the earliest time moment.



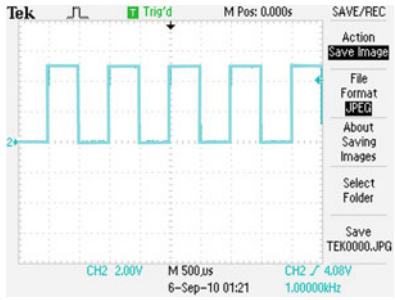
Problem 14.5. A generic oscilloscope is measuring a periodic voltage clock signal. The oscilloscope window is shown in the following figure.

A. Determine the clock period.
B. Determine the clock frequency (show units).
C. Determine the duty cycle of the clock waveform.
D. Determine Pk-Pk (peak-to-peak) voltage of the clock signal.



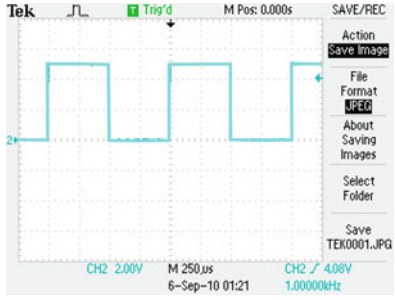
Problem 14.6. A Tektronix oscilloscope is measuring a voltage clock signal. The oscilloscope window is shown in the following figure.

A. Determine the clock period.
B. Determine the clock frequency (show units).
C. Determine the duty cycle of the clock waveform.
D. Determine Pk-Pk (peak-to-peak) voltage of the clock signal.



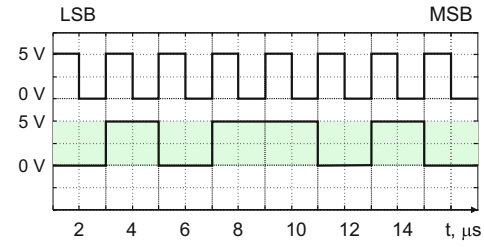
Problem 14.7. A Tektronix oscilloscope is measuring a voltage clock signal. The oscilloscope window is shown in the following figure.

A. Determine the clock period.
B. Determine the clock frequency (show units).
C. Determine the duty cycle of the clock waveform.
D. Determine Pk-Pk (peak-to-peak) voltage of the clock signal.



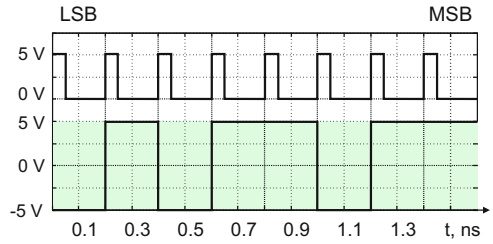
Problem 14.8. The following figure shows a timing diagram: the clock signal and the actual synchronized serial bit stream (the unipolar NRZ code).

A. Determine the clock period.
B. Determine the clock frequency (show units).
C. Determine the bit rate of the bit stream (show units).
D. Decode the corresponding binary number given the LSB/MSB positions (present its decimal equivalent).



Problem 14.9. The following figure shows a timing diagram: the clock signal and the actual synchronized serial bit stream (the polar NRZ code).

A. Determine the clock period.
B. Determine the clock frequency (show units).
C. Determine the bit rate of the bit stream (show units).
D. Decode the corresponding binary number given the LSB/MSB positions (present its decimal equivalent).



14.1.4 Binary Numbers

14.1.5 Hexadecimal Numbers

14.1.6 ASCII Codes and Binary Words

14.1.7 Tri-state Digital Voltage

Problem 14.10. Name two major reasons why the analog computer was surpassed by the digital computer.

Problem 14.11. Describe in your own words the meaning of:

1. A digital word
2. A nibble
3. A byte

Problem 14.12. Without a calculator or MATLAB, convert the following binary numbers to decimal numbers:

A. 1010
B. 101010
C. 11.1
D. 10.001

Problem 14.13. Using either a calculator or MATLAB, convert the following binary numbers to decimal numbers:

A. 1000001.111111
B. 0001111.000010

Problem 14.14. Without a calculator or MATLAB, convert the following decimal numbers to binary numbers:

A. 19
B. 10
C. 1960
D. 14.25

Problem 14.15. Using either a calculator or MATLAB, convert decimal numbers that follow to binary numbers. The desired degree of precision is six bits after the binary point:

A. 133.33
B. 999.125
C. 256.256

Problem 14.16

A. How many bits are necessary to represent decimal number 300,000,000 in binary form?
B. How many bytes are necessary?

Problem 14.17. Write down the year of your birth. Without a calculator or MATLAB, convert this decimal number to:

A. Binary number
B. Hexadecimal number

Problem 14.18. Using the ASCII conversion table, write the string *USA* in terms of:

A. Three decimal numbers
B. Three hexadecimal numbers
C. Three binary numbers

Problem 14.19. Without a calculator or MATLAB, convert hexadecimal numbers that follow to decimal numbers and binary numbers, respectively:

1. 1
2. 12
3. 1A
4. AAA
5. ECE
6. BE
7. CE

Problem 14.20. Using either a calculator or MATLAB, convert hexadecimal numbers that follow to decimal numbers:

1. 3F7
2. EE.25
3. 555h
4. 0x555

Problem 14.21. Convert binary numbers that follow to hexadecimal numbers:

1. 01
2. 11110101
3. 1000000110000000
4. 1111111110000001
5. 1111111111111111.1111

14.2 Digital to Analog Converter

14.2.1 Digital to Analog Converter (DAC)

14.2.2 Circuit (A Binary-Weighted-Input DAC)

14.2.3 Underlying Math and Resolution Voltage

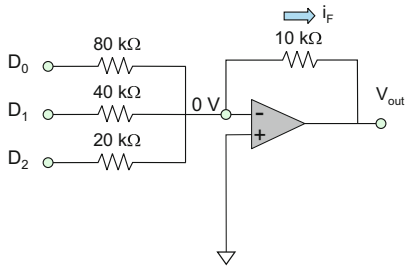
Problem 14.22. The DAC circuit from Fig. 14.15 is operating using the virtual-ground condition of the inverting amplifier in order to add up weighted-input currents. With this in mind, a beginning ECE student removes the amplifier from the circuit and puts a common ground reference at the summing node instead. Will the circuit work?

Problem 14.23

A. For an 8-bit binary-weighted-input DAC, express its output voltage through the resolution voltage (LSB voltage), Q, and binary digits $d_7, d_6, \ldots, d_0$, which corresponds to the input voltages.

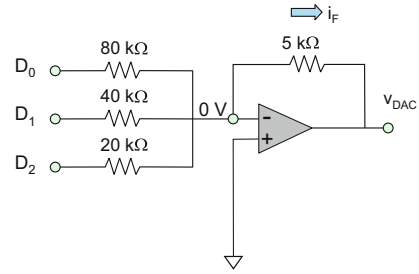
B. Find the output voltage when the LSB voltage is 20 mV and $d_7 = d_6 \ldots = d_2 = 1, d_1 = 0, d_0 = 1$.

Problem 14.24. A 3-bit binary-weighted digital-to-analog converter (DAC) is shown in the figure. The circuit does not include the inverter. Fill out the table that follows:



D2, V	D1, V	D0, V	v_{DAC}, V
0	0	0	
0	0	5	
0	5	0	
0	5	5	
5	0	0	
5	0	5	
5	5	0	
5	5	5	

Problem 14.25. Repeat the previous problem for the DAC shown in the following figure.



Problem 14.26. For a 4-bit binary-weighted-input DAC, the input voltages D_3, D_2, D_1, D_0 are either 0 V or 5 V. The resolution voltage Q of 10 mV is required:

A. Present the circuit diagram of a DAC; label the input voltages.

B. Specify one set of possible resistor values.

Problem 14.27. For a 5-bit binary-weighted-input DAC, the input voltages D_4, D_3, D_2, D_1, D_0 are either 0 V or 2.5 V. The resolution voltage Q of 1 mV is required:

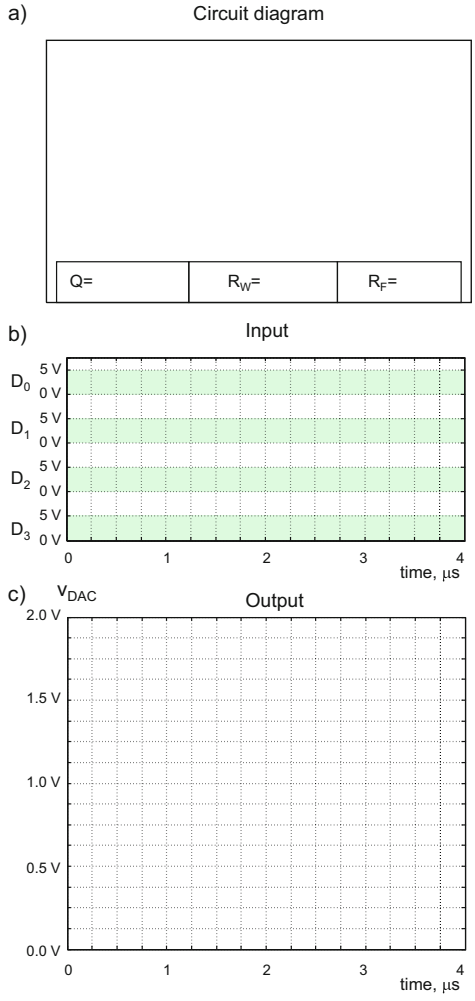
A. Present the circuit diagram of a DAC; label the input voltages.

B. Specify one set of possible resistor values.

Problem 14.28. Design a 4-bit binary-weighted-input DAC circuit which attempts to output the analog voltage in the form of a linear time dependence, $v_{out}(t) = 5 \times 10^5 t$ (V) over time interval from 0 to 4 μs. The input to the DAC is a binary-counter sequence of all 4-bit

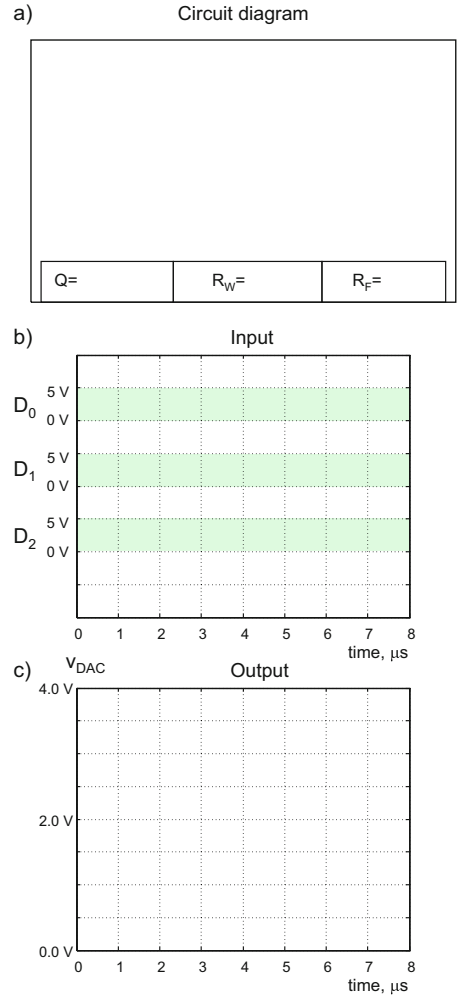
binary numbers. The bit width is 0.25 μs; high and low voltages are 5 V and 0 V, respectively:

A. Present the corresponding circuit diagram; specify required DAC resolution voltage (LSB voltage) and necessary resistor values.
B. Plot the input digital voltages to scale versus time.
C. Plot the output voltage to the DAC to scale versus time.

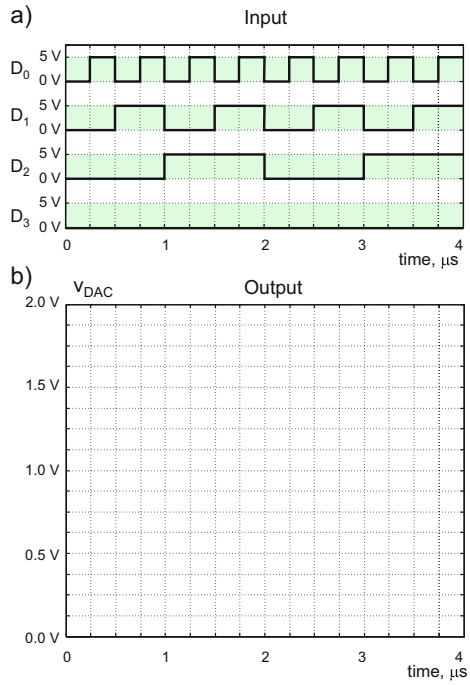


Problem 14.29. Design a 3-bit binary-weighted-input DAC circuit which attempts to output the analog voltage in the form of a linear time dependence, $v_{out}(t) = 5 \times 10^5 t$ (V) over time interval from 0 to 8 μs. The input to the DAC is a binary-counter sequence of all 3-bit binary numbers. The bit width is 1 μs; high and low voltages are 5 V and 0 V, respectively:

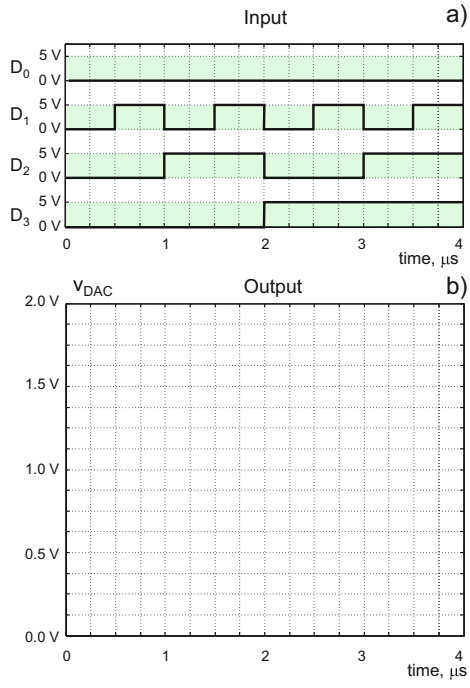
A. Present the corresponding circuit diagram; specify the required DAC resolution voltage (LSB voltage) and necessary resistor values.
B. Plot the input digital voltages to scale versus time.
C. Plot the output voltage to the DAC to scale versus time.



Problem 14.30. A 4-bit binary-weighted-input DAC has the resolution voltage, Q, of 0.125 V and the input voltages shown in the following figure. Pin D_3 is accidentally connected to ground. Plot the output voltage of a DAC to scale versus time.



Problem 14.31. Repeat the previous problem for the input voltages shown in the following figure.



14.2.4 DAC Full-scale Output Voltage Range, Resolution, and Accuracy

Problem 14.32

A. For an 8-bit DAC chip, express its output voltage through the full-scale output voltage range, E, and binary digits d_7, $d_6, \ldots, d_0$, which corresponds to the input voltages.

B. Find the output voltage when the full-scale output voltage range, E, is 6 V and $d_7 = d_6 \ldots = d_2 = 1, d_1 = 0, d_0 = 1$.

Problem 14.33

A. For a 10-bit DAC chip, express its output voltage through the full-scale output voltage range, E, and binary digits d_9, $d_8, \ldots, d_0$, which corresponds to the input voltages.

B. Find the output voltage when the full-scale output voltage range, E, is 10 V and $d_9 = d_8 \ldots = d_3 = 1, d_2 = d_1 = 0$, $d_0 = 1$

C. Find the resolution voltage of the DAC, Q.

Problem 14.34. A 6-bit DAC and a 8-bit DAC use a 6- and 8-bit binary-counter input sequences in order to produce the analog voltage in the form of a linear time dependence, $v_{\text{out}}(t) = 5 \times 10^5 t$ (V), over the same time interval from 0 to 10 μs. Determine:

1. DAC resolution in bits (quantization levels)
2. Full-scale output voltage range, E
3. DAC voltage resolution, Q
4. Necessary bit rate, f_{b}

Problem 14.35. An 8-bit DAC and a 10-bit DAC use an 8- and 10-bit binary-counter input sequences in order to produce the analog voltage in the form of a linear time dependence, $v_{\text{out}}(t) = 5 \times 10^6 t$ (V), over the same time interval from 0 to 1 μs. Determine:

1. DAC resolution in bits (quantization levels)
2. Full-scale output voltage range, E
3. DAC voltage resolution, Q
4. Necessary bit rate, f_{b}

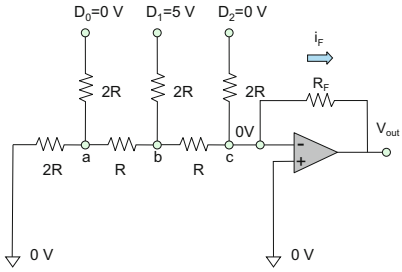
Problem 14.36. For a 3-bit R/2R ladder DAC, the input voltages D_2, D_1, D_0 are either 0 V or 5 V. The resolution voltage Q of 100 mV is required:

A. Present the circuit diagram of a DAC; label the input voltages.

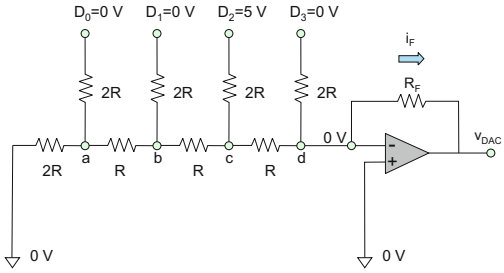
B. Specify one set of possible resistor values.

14.2.5 Other DAC Circuits

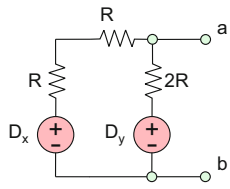
Problem 14.37. By solving the amplifier circuit, determine the output voltage of the 3-bit R/2R ladder DAC shown in the following figure given that $D_2 = 0\ \text{V}, D_1 = 5\ \text{V}, D_0 = 0\,\text{V}$.



Problem 14.38. By solving the amplifier circuit, determine the output voltage of the 4-bit R/2R ladder DAC in Fig. 14.18 given that $D_3 = 0\ \text{V}, D_2 = 5\ \text{V}, D_1 = 0\ \text{V}, D_0 = 0\ \text{V}$.



Problem 14.39. An important step in the analysis of the R/2R ladder network is finding Thévenin equivalent for the circuit shown in the figure. Find the Thévenin equivalent and draw the corresponding circuit.



14.3 Sample-and-Hold Circuit. Nyquist Rate

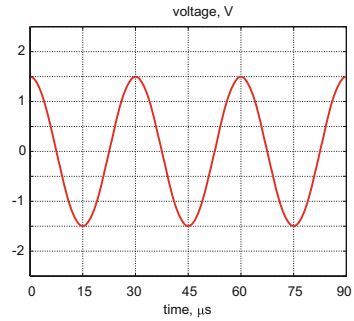
14.3.1 Analog to Digital Converter (ADC)

14.3.2 A Quick Look at an Analog Sinusoidal Voltage

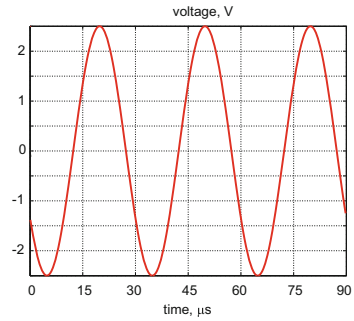
Problem 14.40

A. Determine frequency in Hz, angular frequency in rad/sec, phase, and amplitude of the harmonic voltage signal shown in the following figure.

B. Write the voltage in the form of a cosine function with the corresponding amplitude, frequency, and phase.



Problem 14.41. Determine frequency in Hz, angular frequency in rad/s, and amplitude of the harmonic voltage signal shown in the following figure.

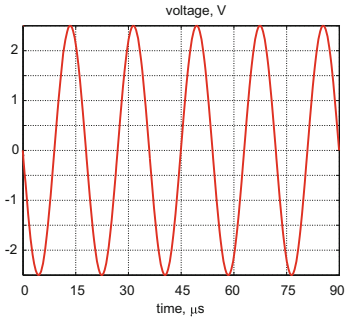


Problem 14.42

A. Determine frequency in Hz, angular frequency in rad/s, phase, and amplitude of

the harmonic voltage signal shown in the figure.

B. Write the AC voltage in the form of a cosine function, with the corresponding amplitude, frequency, and phase.

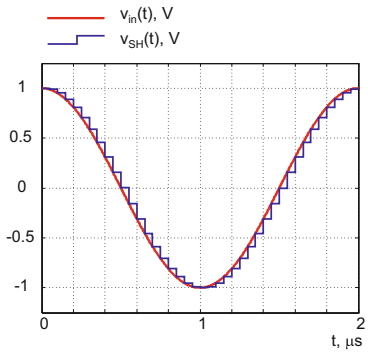


14.3.3 Sample-and-Hold Voltage

Problem 14.43. For the voltage signal shown in the following figure, determine:

A. Frequency, f, and amplitude, V_m, of the analog voltage

B. Sampling interval, T_S, and sampling rate, f_S, for the sample-and-hold voltage

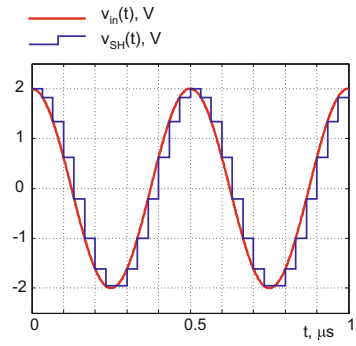


Problem 14.44*. Plot the figure to the previous problem using MATLAB, introduce a title, and label the axes. Present the text of the corresponding MATLAB script.

Problem 14.45. For the voltage signal shown in the following figure, determine:

A. Frequency, f, and amplitude, V_m, of the analog voltage

B. Sampling interval, T_S, and sampling rate, f_S, for the sample-and-hold voltage



Problem 14.46*. Plot the figure to the previous problem using MATLAB, introduce a title, and label the axes. Present the text of the corresponding MATLAB script.

14.3.4 Sample-and-Hold Circuit (SH Circuit)

Problem 14.47. Draw the schematic of the sample-and-hold circuit. Explain its operation in steps.

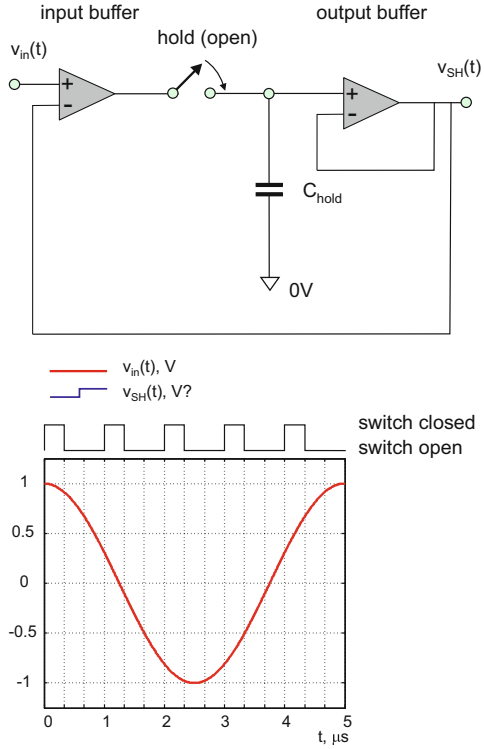
Problem 14.48. The circuit shown in the figure is another modification of the sample-and-hold circuit.

Assuming that the capacitor responds instantaneously:

A. Explain the circuit operation.

B. Sketch its output voltage to scale versus time in the figure the follows.

The switching control voltage is shown on the top of the figure. The switch is closed at high control voltage and is open at low control voltage.

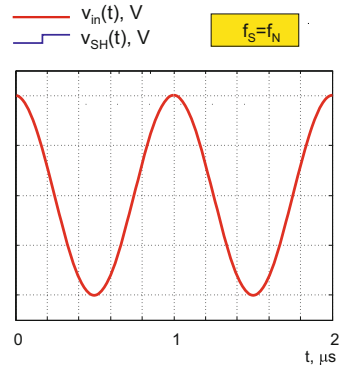


14.3.5 Nyquist Rate

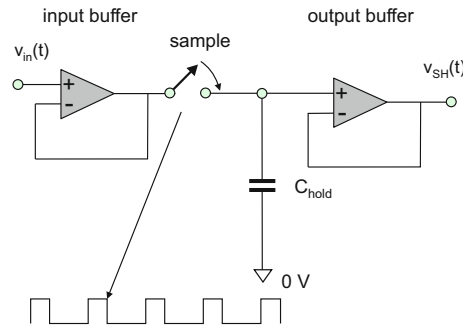
Problem 14.49. An analog voltage is a combination of three sinusoidal harmonics with frequencies 1 MHz, 0.5 MHz, and 0.2 MHz. The voltage amplitudes of the individual sinusoids are 1 V, 1 V, and 5 V. What is the limit on minimum acceptable sampling rate of the sample-and-hold circuit?

Problem 14.50. An analog voltage is a combination of four sinusoidal harmonics with frequencies 0.5 MHz, 0.2 MHz, 1 MHz, and 1.2 MHz. The voltage amplitudes of the individual sinusoids are 5 V, 1 V, 1 V, and 0 V. What is the limit on minimum acceptable sampling rate of the sample-and-hold circuit?

Problem 14.51. Using the figure below, could you demonstrate when the sampling at exactly the Nyquist rate may not be successful?



Problem 14.52. An audio signal (containing all analog sinusoids with frequencies between 20 Hz and 20 kHz) is recorded using a simplified sample-and-hold circuit (the reset switch is omitted) shown in the following figure.



The sample switch closes every:

A. 227 µs
B. 22.7 µs
C. 2.27 µs

then opens momentarily. Which case should be preferred for the minimum memory requirement (for an audio CD)?

14.4 Analog to Digital Converter

14.4.1 Flash ADC

Problem 14.53. How many comparators would we need for an 8-bit flash ADC? For a 12-bit flash ADC?

Problem 14.54

A. Draw a complete circuit diagram of a 2-bit flash ADC with the full-scale measurement voltage range of 4 V.
B. Fill out the following table:

Range of sample-and-hold voltage v_{SH}, V	Output of comparator block (3–0)	Output of the priority encoder (binary number d_1d_0)
3–4		
2–3		
1–2		
0–1		

Problem 14.55

A. Draw a complete circuit diagram of a 3-bit flash ADC with the full-scale measurement voltage range of 8 V.
B. Fill out the following table:

Range of sample-and-hold voltage v_{SH}, V	Output of comparator block (7–0)	Output of the priority encoder (binary number $d_3d_2d_1d_0$)
7–8		
6–7		
5–6		
4–5		
3–4		
2–3		
1–2		
0–1		

14.4.2 ADC Resolution in Bits, Full-scale Input Voltage Range, and Voltage Resolution

Problem 14.56. For a 6-bit ADC determine:

A. Resolution in bits (quantization levels)
B. Voltage resolution, Q
C. Relative accuracy percentage assuming a 1 LSB error

when the full-scale measurement voltage range is 8 V.

Problem 14.57. For an 8-bit ADC determine:

A. Resolution in bits (quantization levels)
B. Voltage resolution, Q
C. Relative accuracy percentage assuming a 1 LSB error

when the full-scale measurement voltage range is 10 V.

14.4.3 ADC Equation and Quantization Error

Problem 14.58. A 5-bit flash ADC follows a mid-rise coding scheme. The reference voltage is 12 V:

1. Present the ADC equation.
2. Determine ADC quantization error.
3. Find ADC output code when the sample-and-hold voltage, v_{SH}, is 3.1 V.

Problem 14.59. An 8-bit flash ADC follows a mid-tread coding scheme. The reference voltage is 5 V:

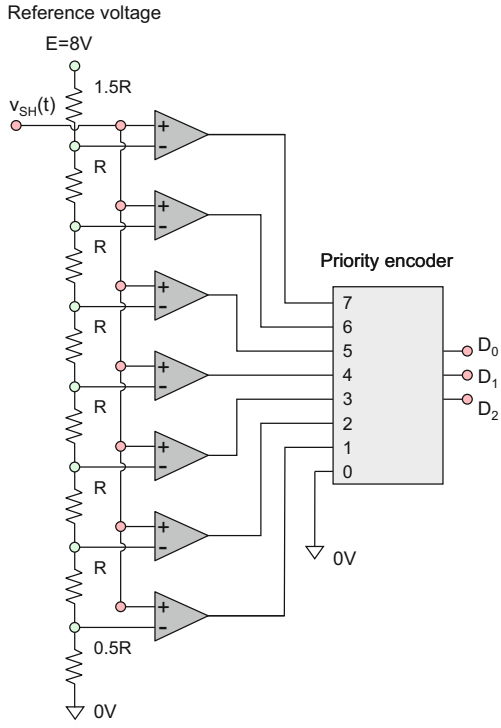
1. Present the ADC equation.
2. Determine ADC quantization error.
3. Find ADC output code when the sample-and-hold voltage, v_{SH}, is 0.2 V.

Problem 14.60. A 3-bit flash ADC is constructed as shown in the following figure:

A. Fill out the table below, which describes the ADC operation. Express all absolute voltage values in terms of the resolution voltage, Q.

B. Based on this table, estimate the quantization error of the ADC in terms of Q.

C. Is this ADC design is better than the *mid-rise coding scheme*?



Voltage range of v_{SH}, *in terms of Q*	Output of comp. block (7–0)	Output of priority encoder (binary number $d_2d_1d_0$)	Voltage decoded back, *in terms of Q*
6.5Q–8Q			
5.5Q–6.5Q			
4.5Q–5.5Q			
3.5Q–4.5Q			
2.5Q–3.5Q			
1.5Q–2.5Q			
0.5Q–1.5Q			
0–0.5Q			

14.4.4 Successive-Approximation ADC

Problem 14.61. Draw the circuit diagram for a 3-bit successive-approximation ADC.

Problem 14.62. A 4-bit successive-approximation ADC has the input voltage of $v_{SH} = 1.05$ V; the DAC resolution voltage is 0.1 V. Determine the sequence of binary states and the final ADC output.

Problem 14.63. A 5-bit successive-approximation ADC has the input voltage of $v_{SH} = 3.1$ V; the DAC resolution voltage is 0.2 V. Determine the sequence of binary states and the final ADC output.