CHAPTER 3
Basic Principles of Digital Systems

CHAPTER OBJECTIVES
Upon successful completion of this chapter, you will be able to:

1. Describe some differences between analog and digital electronics.
2. Understand the concept of HIGH and LOW logic levels.
3. Explain the basic principles of a positional notation number system.
4. Translate logic HIGHs and LOWs into binary numbers.
5. Distinguish between the most significant bit and least significant bit of a binary number.
6. Count in binary, decimal, or hexadecimal.
7. Convert a number in binary, decimal, or hexadecimal to any of the other number bases.
8. Describe the difference between periodic, aperiodic, and pulse waveforms.
9. Calculate the frequency, period, and duty cycle of a periodic digital waveform.
10. Calculate the pulse width of a digital pulse.
I

f you listen to a radio, you may or may not know that it is typically not a digital device. On the other hand, a computer, a DVD player, a video game unit, and a digital clock are all essentially digital devices, or at least have a lot of digital circuitry inside. A radio receives radio waves through the air, converts the waves to different voltages, and then uses these different voltages with magnets to make a speaker vibrate. Digital devices transform the voltage into a digital code of 1’s and 0’s. For example, your computer stores and processes information in groups of 1’s and 0’s; each small piece of information can be represented by 8 of these 1’s and 0’s, and millions or billions of these groups can be processed each second.

Digital electronics is the branch of electronics based on the combination and switching of logic levels. Physically, these logic levels are represented by voltages. Any quantity in the physical world, such as temperature, pressure, or voltage, can be symbolized in a digital circuit by a group of logic levels that, taken together, form a binary number. Logic levels are usually specified as 0 or 1; at times, it may be more convenient to use LOW/HIGH, FALSE/TRUE, or OFF/ON.

Each logic level corresponds to a digit in the binary (base 2) number system. The binary digits, or bits, 0 and 1, are sufficient to write any number, given enough places. The hexadecimal (base 16) number system is also important in digital systems. Because every combination of four binary digits can be uniquely represented as a hexadecimal digit, this system is often used as a compact way of writing binary information.

Inputs and outputs in digital circuits are not always static. Often they vary with time. Time-varying digital waveforms can have three forms:

1. Periodic waveforms, which repeat a pattern of logic 1’s and 0’s. (A special type of waveform called a clock signal is included in this group.)
2. Aperiodic waveforms, which do not repeat.
3. Pulse waveforms, which produce a momentary variation from a constant logic level.
3.1 DIGITAL VERSUS ANALOG ELECTRONICS

**KEY TERMS**

**Analog** A representation of a physical, continuous quantity. An analog voltage or current can have any value within a defined range.

**Digital** A representation of a physical quantity by a series of binary numbers. A digital representation can have only specific discrete values.

**Continuous** Smoothly connected: an unbroken series of consecutive values with no instantaneous changes.

**Discrete** Separated into distinct segments or pieces. A series of discontinuous values.

Electronic systems and devices are often divided into two areas: analog and digital electronics. As more electronic systems have been designed using digital technology, devices have become smaller and more powerful. For example, consider telephones and cell phones. These devices had been designed originally as analog devices and were heavy, bulky, and less powerful compared to today’s phones. Cell phones that fit into a pocket, can now be used to send text messages, take pictures and videos, hold GPS systems, and even be used as telephones. Music was traditionally available on record albums and cassette tapes, which used analog record players and tape players to play music. The invention of CDs allowed music to be stored and replayed using digital electronics—CDs are smaller, more versatile, and less easily damaged. As digital music players and MP3s came into use, devices as small as a package of gum could now hold hundreds or thousands of songs. Music libraries that used to fill a room and require special care now fit in your pocket! Digital electronics now dominate circuit design, from computers to cars to video-game systems.

The main difference between analog and digital electronics, simply stated, is: analog voltages or currents are continuous between values, while digital voltages or currents are discrete (or allowed only at distinct levels). Typically, digital signals vary between two discrete values.

Some keywords highlight the differences between digital and analog electronics:

<table>
<thead>
<tr>
<th>Analog</th>
<th>Digital</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continuously variable</td>
<td>Discrete steps</td>
</tr>
<tr>
<td>Amplification</td>
<td>Switching</td>
</tr>
<tr>
<td>Voltages</td>
<td>Numbers</td>
</tr>
<tr>
<td>AM (Amplitude modulation)</td>
<td>Microprocessors</td>
</tr>
<tr>
<td>FM (Frequency modulation)</td>
<td>Binary</td>
</tr>
</tbody>
</table>

We can use a common device to better understand the differences between digital and analog signals. A digital signal can be compared to a light switch, whereas an analog signal can be compared to a dimmer switch. The dimmer switch can turn on lights to any level of brightness, and turning the dimmer switch slightly higher gives us slightly brighter light. A typical light switch, however, turns lights on or off. Similar to a digital signal with only two levels, a light controlled by this switch has only two “levels.”

If a light switch can only turn lights on and off, what happens if a light switch is put in the middle position? The answer is: the lights may go on, they may go off, they may flicker, or something else may happen. In other words, we
just don’t know (and we usually don’t care) what happens with
the light switch between on and off: if the lights are off and we
want them on, we push the switch up and they go on. In many
ways, if a digital signal is in one state (say it is off), and we want
it in the other state (on), we can turn it on. What if it is in-
between states? We don’t know—and usually don’t care—what
the value is.

Let’s consider digital vs. analog music once again. Music that
is stored digitally, such as on CDs and MP3 files on a computer,
is very popular because the music is accurate and noise-free when
played. This high quality of sound is possible because the music is
stored as a coded series of numbers that represent the sound waves,
not as a physical copy of the sound waves (as in a vinyl record) or as
a magnetic copy (as in an analog tape). If you listed to music played
on a record player or from a tape deck, there is some noise (hissing)
generated as the needle travels over the surface or the tape travels
over the magnetic head. If you listen to a scratched record, you can
hear the “pop” as the needle goes over the scratch. Most CD players
have built-in circuitry to hide small scratches on CDs, so the music
sounds the same even after the CD is played over and over.

Figure 3.1A shows a sample waveform represented as a volt-
age. This type of waveform could be measured at the output of an
amplified microphone. Figure 3.1B and Figure 3.1C show the volt-
age as an analog copy and a digital copy of the sound waveform.

The analog copy, shown in Figure 3.1B, may show some distor-
tion with respect to the original. The distortion is usually intro-
duced by both the analog storage and playback processes.

A digital audio system doesn’t make a copy of the waveform,
but rather stores a code (a series of amplitude numbers) that tells
the CD player how to recreate the original sound every time a disc
is played. During the recording process, the sound waveform is
“sampled” at precise intervals. That is, the voltage of the wave-
form is measured at certain intervals and each measurement is
converted to a representative binary number. A typical encoding
scheme might assign the voltage to a value between 0 and 65,535.
Such a large number of possible values means the voltage differ-
ence between any two consecutive digital numbers is very small.
The numbers can thus correspond extremely closely to the actual
amplitude of the sound waveform. The signal on a CD is sampled
44,100 times each second; with this very fast sampling and very
fine divisions between voltages, the original wave can be repro-
duced very accurately.

Digital representations of physical quantities are also superior
to analog in that they can easily be stored, transferred, and copied
without the distortion that accompanies analog processes. Digital
values can be stored in a variety of media, such as optical (CD or
DVD), magnetic (hard drive on a PC), or semiconductor (flash
memory). They can be transmitted over electronic communica-
tions systems such as fiber optic, radio, or telephone. As long as
the integrity of the digital data is maintained, any copy of the data
is as good as any other. Copies can be made from other copies
without deterioration between copy generations.
3.2 DIGITAL LOGIC LEVELS

**KEY TERMS**

- **Logic level** A voltage level that represents a defined digital state in an electronic circuit.
- **Logic LOW** (or **logic 0**) The lower of two voltages in a digital system with two logic levels.
- **Logic HIGH** (or **logic 1**) The higher of two voltages in a digital system with two logic levels.
- **Positive logic** A system in which logic LOW represents binary digit 0 and logic HIGH represents binary digit 1.
- **Negative logic** A system in which logic LOW represents binary digit 1 and logic HIGH represents binary digit 0.

We have described digital signals as similar to light switches: they have only two possible values. Like light switches, we sometimes refer to digital signals as ON or OFF. We can also refer to the two logic levels of a digital signal as HIGH (or logic HIGH) and LOW (or logic LOW), or using the numbers 1 and 0. The binary number system also uses only the two numbers 1 and 0, and we will see shortly how any number can be represented in binary—using only the numbers 1 and 0. Because we are describing a digital quantity electronically, we need to have a system that uses voltages (or currents) to symbolize binary numbers.

For binary systems, digits are used to represent different voltage or logic levels: the lower voltage (usually 0 volts) is called a **logic LOW** or **logic 0** and represents the digit 0. The higher voltage (traditionally 5 V, but in many current systems a different value such as 1.8 V, 2.5 V, or 3.3 V) is called a **logic HIGH** or **logic 1**, which represents the digit 1. In reality, there is a range of acceptable values for HIGH and LOW: logic HIGH voltages are at or near the higher voltage, and logic LOW voltages are at or near the lower voltage. Voltages between these values are not defined. Figure 3.2 shows that voltages between 2 V and 5 V are considered HIGH, and voltages between 0 V and 0.8 V are considered LOW. Voltages between these levels are considered undefined and should not occur in a digital system, much like a light switch should not be between ON and OFF ... and if it is, the results are undefined.

Assigning the digit 1 to a logic HIGH and digit 0 to logic LOW as described is called **positive logic**; this is by far the more common system. Throughout the remainder of this text, logic levels will be referred to as HIGH/LOW or 1/0.
interchangeably. A complementary system called negative logic also exists. If we’re using negative logic, we simply change the names of the voltages, where +5 V is named “0” and 0 V is named “1.” We’ll see some examples much later.

### 3.3 THE BINARY NUMBER SYSTEM

**Key Terms**

- **Decimal number system** Base-10 number system; the most commonly used number system.
- **Positional notation** A system of writing numbers where the value of a digit depends not only on the digit, but also on its placement within a number.
- **Binary number system** A number system used extensively in digital systems, based on the number 2. It uses two digits, 0 and 1, to write any number.
- **Bit** Binary digit. A 0 or a 1.

**Positional Notation**

The decimal number system, or the base-10 number system, is the number system with the numbers you use every day to count, to do math, and so on. We can write any number using only 10 digits, 0 through 9 in the proper columns. For example, when we count, we use the numbers 7...8...9. If we need to count to a number larger than 9, we add a new column to the left—the 10’s column. Numbers in the decimal system have a value based on the digits (0 through 9) and their position (1’s, 10’s, 100’s, 1,000’s, etc.).

In the decimal number 845, the digit 4 really means 40 (845 = 800 + 40 + 5), whereas in the number 9,426, the digit 4 really means 400 (9,426 = 9,000 + 400 + 20 + 6), as shown in Figure 3.3. The value of the digit is determined by what the digit is as well as where it is.

The decimal system is a positional notation system, where the value of a digit depends on its placement within a number. In the decimal system, a digit in the position immediately to the left of the decimal point is multiplied by 1 (10^0, or the 1’s column). A digit two positions to the left of the decimal point is multiplied by 10 (10^1, or the 10’s column). A digit in the next position left is multiplied by 100 (10^2, or the 100’s column), and so on. The positional multipliers, as you move left from the decimal point, are ascending powers of 10.

The binary number system is based on the number 2. This means that we can write any number using only two binary digits (or bits), 0 and 1. The binary system is also a positional notation system; the value of a digit (0 or 1) depends on its placement within a number, like the decimal system. The difference is that the positional multipliers in the binary system are powers of 2 (2^1 = 1, 2^2 = 2, 2^3 = 4, 2^4 = 8, 2^5 = 16, 2^6 = 32, ... or 1’s column, 2’s column, 4’s column, 8’s column, etc.). For example, the binary number 11010 (see Figure 3.4) has the decimal equivalent:

\[
(1 \times 2^4) + (1 \times 2^3) + (1 \times 2^2) + (1 \times 2^1) + (0 \times 2^0) \\
= (1 \times 16) + (1 \times 8) + (0 \times 4) + (1 \times 2) + (0 \times 1) \\
= 16 + 8 + 0 + 2 + 0 \\
= 26
\]

**FIGURE 3.3 The Number 9,426**

<table>
<thead>
<tr>
<th></th>
<th>9</th>
<th>4</th>
<th>2</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>10^3</td>
<td>1,000s</td>
<td>100s</td>
<td>10s</td>
<td>1s</td>
</tr>
<tr>
<td>10^2</td>
<td>10^2</td>
<td>10^1</td>
<td>10^0</td>
<td></td>
</tr>
</tbody>
</table>

\[
9 \times 1,000 = 9,000 \\
4 \times 100 = 400 \\
2 \times 10 = 20 \\
6 \times 1 = 6 \\
9,426
\]

**FIGURE 3.4 The Binary Number 11010**

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>1</th>
<th>0</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>2^4</td>
<td>16s</td>
<td>8s</td>
<td>4s</td>
<td>2s</td>
<td>1s</td>
</tr>
<tr>
<td>2^3</td>
<td></td>
<td>2^3</td>
<td>2^2</td>
<td>2^1</td>
<td>2^0</td>
</tr>
</tbody>
</table>

\[
1 \times 16 = 16 \\
1 \times 8 = 8 \\
0 \times 4 = 0 \\
1 \times 2 = 2 \\
0 \times 1 = 0 \\
26
\]
Example 3.1

Calculate the decimal equivalents of the binary numbers 1010, 111, and 10010.

Solutions

1010 = (1 \times 2^3) + (0 \times 2^2) + (1 \times 2^1) + (0 \times 2^0)
   = (1 \times 8) + (0 \times 4) + (1 \times 2) + (0 \times 1)
   = 8 + 2 = 10

111 = (1 \times 2^2) + (1 \times 2^1) + (1 \times 2^0)
   = (1 \times 4) + (1 \times 2) + (1 \times 1)
   = 4 + 2 + 1 = 7

10010 = (1 \times 2^4) + (0 \times 2^3) + (0 \times 2^2) + (1 \times 2^1) + (0 \times 2^0)
   = (1 \times 16) + (0 \times 8) + (0 \times 4) + (1 \times 2) + (0 \times 1)
   = 16 + 2 = 18

Binary Inputs

**KEY TERMS**

- **Truth table** A list of output levels of a circuit corresponding to all different input combinations.
- **Most significant bit (MSB)** The leftmost bit in a binary number. This bit has the number’s largest positional multiplier.
- **Least significant bit (LSB)** The rightmost bit of a binary number. This bit has the number’s smallest positional multiplier.

A major class of digital circuits, called combinational logic, operates by accepting logic levels at one or more input terminals and producing a logic level at an output. Most of these systems have more than one input, and the signal at each input can be HIGH or LOW. We can use a binary number to represent each input: 1 if the input value is HIGH, 0 if it is LOW. In the analysis and design of these circuits, we will usually find the output logic level of a circuit for all possible combinations of input logic levels.

The digital circuit in the black box in Figure 3.5 has three inputs. Each input can have two possible states, LOW or HIGH, which can be represented by positive logic as 0 or 1. Table 3.1 shows a list of all combinations of the input variables both as logic levels and binary numbers, and their decimal equivalents.

The number of possible input combinations is $2^3 = 8$. In general, a circuit with $n$ inputs has $2^n$ input combinations, ranging from 0 to $2^n - 1$.

<table>
<thead>
<tr>
<th>Logic Level</th>
<th>Binary Value</th>
<th>Decimal Equivalent</th>
</tr>
</thead>
<tbody>
<tr>
<td>A B C</td>
<td>A B C</td>
<td></td>
</tr>
<tr>
<td>L L L</td>
<td>0 0 0</td>
<td>0</td>
</tr>
<tr>
<td>L L H</td>
<td>0 0 1</td>
<td>1</td>
</tr>
<tr>
<td>L H L</td>
<td>0 1 0</td>
<td>2</td>
</tr>
<tr>
<td>L H H</td>
<td>0 1 1</td>
<td>3</td>
</tr>
<tr>
<td>H L L</td>
<td>1 0 0</td>
<td>4</td>
</tr>
<tr>
<td>H L H</td>
<td>1 0 1</td>
<td>5</td>
</tr>
<tr>
<td>H H L</td>
<td>1 1 0</td>
<td>6</td>
</tr>
<tr>
<td>H H H</td>
<td>1 1 1</td>
<td>7</td>
</tr>
</tbody>
</table>
A list of output logic levels corresponding to all possible input combinations, applied in ascending binary order, is called a truth table. This is a standard form for showing the function of a digital circuit.

If the input bits on each line of Table 3.1 are read (from left to right) as a series of 3-bit binary numbers, we find that they count from 0 to 7. Because this circuit has 3 inputs, it has $2^3 = 8$ input combinations, ranging from 000₂ to 111₂ (0 to 7).

Bit A is called the most significant bit (MSB), or leftmost bit, and bit C is called the least significant bit (LSB) or rightmost bit. As these terms imply, a change in bit A is more significant, because it has the greatest effect on the number of which it is part.

Table 3.2 shows the effect of changing each of these bits in a 3-bit binary number and compares the changed number to the original by showing the difference in magnitude. A change in the MSB of any 3-bit number results in a difference of 4. A change in the LSB of any binary number results in a difference of 1. (Try it with a few different numbers.)

**TABLE 3.2** Effect of Changing the LSB and MSB of a Binary Number

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>Decimal</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Change MSB</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>Change LSB</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

**Example 3.2**

Figure 3.6 shows a 4-input digital circuit. List all the possible binary input combinations to this circuit and their decimal equivalents. What is the value of the MSB?

**Solution**

Because there are 4 inputs, there will be $2^4 = 16$ possible input combinations, ranging from 0000 to 1111 (0 to 15 in decimal). Table 3.3 shows the list of all possible input combinations.

**TABLE 3.3** Possible Input Combinations for a 4-Input Digital Circuit

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>Decimal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>11</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>13</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>14</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>15</td>
</tr>
</tbody>
</table>

The MSB has a value of 8 (decimal).
Knowing how to construct a binary sequence is a very important skill when working with digital logic systems. Two ways to do this are:

1. **Learn to count in binary.** You should know all the binary numbers from 0000 to 1111 and their decimal equivalents (0 to 15). *Make this your first goal in learning the basics of digital systems.*

Each binary number is a unique representation of its decimal equivalent. You can work out the decimal value of a binary number by adding the weighted values of all the bits.

For instance, the binary equivalent of the decimal sequence 0, 1, 2, 3 can be written using two bits: the 1’s bit and the 2’s bit. The binary count sequence is:

<table>
<thead>
<tr>
<th>Binary</th>
<th>Decimal</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>0</td>
</tr>
<tr>
<td>01</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
</tr>
<tr>
<td>11</td>
<td>3</td>
</tr>
</tbody>
</table>

To count beyond this, you need another bit: the 4’s bit. The decimal sequence 4, 5, 6, 7 has the binary equivalents:

<table>
<thead>
<tr>
<th>Binary</th>
<th>Decimal</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>4</td>
</tr>
<tr>
<td>101</td>
<td>5</td>
</tr>
<tr>
<td>110</td>
<td>6</td>
</tr>
<tr>
<td>111</td>
<td>7</td>
</tr>
</tbody>
</table>

The two least significant bits of this sequence are the same as the bits in the 0 to 3 sequence; a repeating pattern has been generated.

The sequence from 8 to 15 requires yet another bit: the 8’s bit. The three LSBs of this sequence repeat the 0 to 7 sequence. The binary equivalents of 8 to 15 are:

<table>
<thead>
<tr>
<th>Binary</th>
<th>Decimal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>8</td>
</tr>
<tr>
<td>1001</td>
<td>9</td>
</tr>
<tr>
<td>1010</td>
<td>10</td>
</tr>
<tr>
<td>1011</td>
<td>11</td>
</tr>
<tr>
<td>1100</td>
<td>12</td>
</tr>
<tr>
<td>1101</td>
<td>13</td>
</tr>
<tr>
<td>1110</td>
<td>14</td>
</tr>
<tr>
<td>1111</td>
<td>15</td>
</tr>
</tbody>
</table>

Practice writing out the binary sequence, as listed in Table 3.3, until it becomes familiar. In the 0 to 15 sequence, it is standard practice to write each number as a 4-bit value, as in Example 3.2, so that all numbers have the same number of bits. Numbers up to 7 have leading zeros to pad them out to 4 bits.

When we need to write a binary number using a specified number of bits, we pad the left side with zeros (think of an odometer in a car—a car with 2,309 miles may show 002309).

While you are still learning to count in binary, you can use a second method.

2. **Follow a simple repetitive pattern.** Look at Table 3.1 and Table 3.3 again. Notice that the least significant bit follows a pattern. The bits alternate with every line, producing the pattern 0, 1, 0, 1 … The 2’s bit alternates every two lines: 0, 0, 1, 1, 0, 0, 1, 1 … The 4’s bit alternates every four lines: 0, 0, 0, 0, 1, 1, 1, 1 … This pattern can be expanded to cover any number of bits, with the number of lines between alternations doubling with each bit to the left.

---

*Note…*

Practice writing out the binary sequence, as listed in Table 3.3, until it becomes familiar.
**Decimal-to-Binary Conversion**

Two methods are commonly used to convert decimal numbers to binary: sum of powers of 2 and repeated division by 2.

**Sum of Powers of 2**

You can convert a decimal number to binary by adding up powers of 2 by inspection, adding bits as you need them to fill up the total value of the number (refer to Figure 3.7). For example, convert $57_{10}$ to binary.

$64_{10} > 57_{10} > 32_{10}$

- We see that $32$ ($= 2^5$) is the largest power of 2 that is smaller than 57. Set the 32’s bit to 1 and subtract 32 from the original number, as shown.

  $57 - 32 = 25$

- The largest power of 2 that is less than 25 is 16. Set the 16’s bit to 1 and subtract 16 from the accumulated total.

  $25 - 16 = 9$

- 8 is the largest power of 2 that is less than 9. Set the 8’s bit to 1 and subtract 8 from the total.

  $9 - 8 = 1$

- 4 is greater than the remaining total. Set the 4’s bit to 0.

- 2 is greater than the remaining total. Set the 2’s bit to 0.

- 1 is left over. Set the 1’s bit to 1 and subtract 1.

  $1 - 1 = 0$

- Conversion is complete when there is nothing left to subtract. Any remaining bits should be set to 0.

**Repeated Division by 2**

Any decimal number divided by 2 will leave a remainder of 0 or 1. Repeated division by 2 will leave a string of 0 and 1 remainders that become the binary equivalent of the decimal number. Let us use this method to convert $46_{10}$ to binary.

1. Divide the decimal number by 2 and note the remainder.

   $46/2 = 23 + \text{remainder 0 (LSB)}$

   The remainder is the least significant bit of the binary equivalent of 46.

2. Divide the quotient from the previous division and note the remainder. The remainder is the second LSB.

   $23/2 = 11 + \text{remainder 1}$

3. Continue this process until the quotient is 0. The last remainder is the most significant bit of the binary number.

   $11/2 = 5 + \text{remainder 1}$
   $5/2 = 2 + \text{remainder 1}$
   $2/2 = 1 + \text{remainder 0}$
   $1/2 = 0 + \text{remainder 1 (MSB)}$

**Example 3.3**

Convert $92_{10}$ to binary using the sum-of-powers-of-2 method.

**Solution**

$128 > 92 > 64$

$64 - 64 = 0$

$92 - 64 = 28$

$64 - 64 = 0$

$28 - 16 = 12$

$64 - 64 = 0$

$12 - 8 = 4$

$64 - 64 = 0$

$4 - 4 = 0$

$92_{10} = 1011100_2$
To write the binary equivalent of the decimal number, read the remainders from the bottom up.

\[ 46_{10} = 101110_{2} \]

### Example 3.4

Use repeated division by 2 to convert \( 115_{10} \) to a binary number.

#### Solution

\[ \begin{align*}
115/2 & = 57 + \text{remainder 1 (LSB)} \\
57/2 & = 28 + \text{remainder 1} \\
28/2 & = 14 + \text{remainder 0} \\
14/2 & = 7 + \text{remainder 0} \\
7/2 & = 3 + \text{remainder 1} \\
3/2 & = 1 + \text{remainder 1} \\
1/2 & = 0 + \text{remainder 1 (MSB)}
\end{align*} \]

Read the remainders from bottom to top: 1110011.

\[ 115_{10} = 1110011_{2} \]

In any decimal-to-binary conversion, the number of bits in the binary number is the exponent of the smallest power of 2 that is larger than the decimal number. For example, for the number \( 92_{10} \):

\[ 2^7 = 128 > 92 \quad \text{7 bits: 1011100} \]

and \( 46_{10} \):

\[ 2^6 = 64 > 46 \quad \text{6 bits: 101110} \]

### Your Turn

3.3 How many different binary numbers can be written with 6 bits?

3.4 How many can be written with 7 bits?

3.5 Write the sequence of 7-bit numbers from 1010000 to 1010111.

3.6 Write the decimal equivalents of the numbers written for the previous problem.

### 3.4 Hexadecimal Numbers

**Key Term**

Hexadecimal number system (Hex) Base-16 number system. Hexadecimal numbers are written with 16 digits, 0–9 and A–F, with power-of-16 positional multipliers.
You may have noticed that binary numbers tend to be much longer than decimal numbers: for example, the decimal number 233 is 11101001 binary. The **hexadecimal number system** allows us to work in the binary world (which we need to do in digital electronics), but work with numbers that are shorter and may be easier to work with in many cases. Hex numbers can pack more digital information into fewer digits. The hexadecimal number system is based on powers of 16. After binary numbers, hexadecimal numbers are the most often used in digital applications. Hexadecimal, or **hex**, numbers are primarily used as a shorthand form of binary notation. Because 16 is a power of 2 \(2^4 = 16\), each hexadecimal digit can be easily converted to four binary digits.

Hex numbers have become particularly popular with the increasing use of microprocessors and other computers, which use binary data having 8, 16, 32, or 64 bits. Such data can be represented by 2, 4, 8, or 16 hexadecimal digits, respectively.

### Counting in Hexadecimal

The positional multipliers in the hex system are powers of 16: \(16^0 = 1\), \(16^1 = 16\), \(16^2 = 256\), \(16^3 = 4096\), and so on.

We need 16 digits to write hex numbers; the decimal digits 0 through 9 are not sufficient. Because we don’t have enough digits, we need other symbols to represent \(10_{10}\) through \(15_{10}\). We will use capital letters A through F to represent \(10\) through \(15\). Table 3.4 shows how hexadecimal digits relate to their decimal and binary equivalents.

<table>
<thead>
<tr>
<th>Hex</th>
<th>Decimal</th>
<th>Binary</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0000</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0001</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>0010</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>0011</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>0100</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>0101</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>0110</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>0111</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>1000</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>1001</td>
</tr>
<tr>
<td>A</td>
<td>10</td>
<td>1010</td>
</tr>
<tr>
<td>B</td>
<td>11</td>
<td>1011</td>
</tr>
<tr>
<td>C</td>
<td>12</td>
<td>1100</td>
</tr>
<tr>
<td>D</td>
<td>13</td>
<td>1101</td>
</tr>
<tr>
<td>E</td>
<td>14</td>
<td>1110</td>
</tr>
<tr>
<td>F</td>
<td>15</td>
<td>1111</td>
</tr>
</tbody>
</table>

For instance, the hex numbers between 19 and 22 are 19, 1A, 1B, 1C, 1D, 1E, 1F, 20, 21, 22. (The decimal equivalents of these numbers are \(25_{10}\) through \(34_{10}\).)
Example 3.5
What is the next hexadecimal number after 999? After 99F? After 9FF? After FFF?

Solution
The hexadecimal number after 999 is 99A. The number after 99F is 9A0. The number after 9FF is A00. The number after FFF is 1000.

Example 3.6
List the hexadecimal digits from \(190_{16}\) to \(200_{16}\), inclusive.

Solution
The numbers follow the counting rules: Use all the digits in one position, add 1 to the digit one position left, and start over. For brevity, we will list only a few of the numbers in the sequence:

\[
\begin{align*}
190, & 191, 192, \ldots, 199, 19A, 19B, 19C, 19D, 19E, 19F, \\
1A0, & 1A1, 1A2, \ldots, 1A9, 1AA, 1AB, 1AC, 1AD, 1AE, 1AF, \\
1B0, & 1B1, 1B2, \ldots, 1B9, 1BA, 1BB, 1BC, 1BD, 1BE, 1BF, \\
1C0, & \ldots, 1CF, 1D0, \ldots, 1DF, 1E0, \ldots, 1EF, 1F0, \ldots, 1FF, 200
\end{align*}
\]

Example 3.7
Convert \(7C6_{16}\) to decimal.

Solution
\[
\begin{align*}
7 \times 16^2 & = 7 \times 256_{10} = 1792_{10}, \\
C \times 16^1 & = 12_{10} \times 16_{10} = 192_{10}, \\
6 \times 16^0 & = 6_{10} \times 1_{10} = 6_{10}, \\
\hline
& 1990_{10}
\end{align*}
\]

Example 3.8
Convert \(1FD5_{16}\) to decimal.

Solution
\[
\begin{align*}
1 \times 16^3 & = 1_{10} \times 4096_{10} = 4096_{10}, \\
F \times 16^2 & = 15_{10} \times 256_{10} = 3840_{10}, \\
D \times 16^1 & = 13_{10} \times 16_{10} = 208_{10}, \\
5 \times 16^0 & = 5_{10} \times 1_{10} = 5_{10}, \\
\hline
& 8149_{10}
\end{align*}
\]

Your Turn

3.7 List the hexadecimal numbers from \(FA9_{16}\) to \(FB0_{16}\), inclusive.

3.8 List the hexadecimal numbers from \(1F9_{16}\) to \(200_{16}\), inclusive.

Hexadecimal-to-Decimal Conversion
To convert a number from hex to decimal, multiply each digit by its power-of-16 positional multiplier and add the products. In the examples on the left, hexadecimal numbers are indicated by a final “H” (e.g., \(1F7_{16}\)), rather than a “16” subscript.

Your Turn

3.9 Convert the hexadecimal number \(A30_{16}\) to its decimal equivalent.
Chapter 3: Basic Principles of Digital Systems

Decimal-to-Hexadecimal Conversion

Converting decimal numbers to hex numbers is similar to converting decimal numbers to binary numbers. In fact, the conversion from binary to hex is so straightforward, it is often easier to convert from decimal to binary, than from binary to hex (or to use a calculator to do the conversion).

Sum of Weighted Hexadecimal Digits

One method of converting decimal numbers to hex numbers that can be useful for simple conversions (about three digits) is the sum-of-weighted-hex-digits method. The main difficulty we encounter is remembering to convert decimal numbers 10 through 15 into the equivalent hex digits, A through F.

For example, the decimal number 35 is easily converted to the hex value 23.

\[ 35_{10} = 32_{10} + 3_{10} = (2 \times 16) + (3 \times 1) = 23_{16} \]

Example 3.9

Convert 175\textsubscript{10} to hexadecimal.

Solution

\[ 256_{10} > 175_{10} > 16_{10} \]

Because 256 = 16\textsuperscript{2}, the hexadecimal number will have two digits.

\[
\begin{align*}
16’s \quad 1’s \\
A \quad 175 - (A \times 16) = 175 - 160 = 15 \\
16’s \quad 1’s \\
A \quad F \quad 175 - ((A \times 16) + (F \times 1)) \\
&= 175 - (160 + 15) = 0
\end{align*}
\]

Your Turn

3.10 Convert the decimal number 137 to its hexadecimal equivalent.

Conversions between Hexadecimal and Binary

Hexadecimal numbers are used very often because of the ease in converting between binary and hex. In fact, you may find it easier to convert decimal to binary to hex than to convert decimal to hex directly. Table 3.4 shows all 16 hexadecimal digits and their decimal and binary equivalents. Note that for every possible 4-bit binary number, there is a hexadecimal equivalent.

Binary-to-hex and hex-to-binary conversions simply consist of making a conversion between each hex digit and its binary equivalent.
Example 3.10

Convert 7EF8H to its binary equivalent.

Solution

Convert each digit individually to its equivalent value:

- 
  \[ 7H = 0111_2 \]
  \[ \text{EH} = 1110_2 \]
  \[ \text{FH} = 1111_2 \]
  \[ 8H = 1000_2 \]

The binary number is all of these binary numbers in sequence:

\[ 7EF8H = 111111011111000_2 \]

The leading zero (the MSB of 0111) has been left out.

Sometimes, we might insert spaces every four bits for readability. In this case:

\[ 7EF8H = 111 1110 1111 1000_2 \]

Your Turn

3.11 Convert the hexadecimal number 934BH to binary.

3.12 Convert the binary number 11001000001101001001 to hexadecimal.

3.5 DIGITAL WAVEFORMS

KEY TERMS

Digital waveform A series of logic 1’s and 0’s plotted as a function of time.
Timing diagram A digital waveform, typically with multiple signals on one plot.

Like a light switch, a digital signal may be ON, may be OFF, or may change every so often. In cases like playing digital music, the digital signals may change quickly—thousands or millions of times per second. The input and output logic levels vary between 0 and 1 with time, creating digital waveforms, or plots of the signal vs. time. Digital waveforms of inputs and outputs of a system are an excellent way to demonstrate how a system works; these waveform plots are called timing diagrams. There are two types of digital waveforms. Periodic waveforms repeat the same pattern of logic levels over a specified period of time. Aperiodic waveforms do not repeat. Pulse waveforms are a special case where a signal may follow a HIGH-LOW-HIGH or LOW-HIGH-LOW pattern and may be periodic or aperiodic. This is similar to walking into a dark room, turning the light ON for a short time, then turning it back OFF.
Periodic Waveforms

**KEY TERMS**

Periodic waveform A time-varying sequence of logic HIGHs and LOWs that repeats over a specified period of time.

Time LOW \( (t_l) \) Time during one period that a waveform is in the LOW state. Unit: seconds \( (s) \).

Time HIGH \( (t_h) \) Time during one period that a waveform is in the HIGH state. Unit: seconds \( (s) \).

Period \( (T) \) Time required for a periodic waveform to repeat. Unit: seconds \( (s) \).

Frequency \( (f) \) Number of times per second that a periodic waveform repeats. \( f = 1/T \) Unit: hertz \( (Hz) \).

Duty cycle \( (DC) \) Fraction of the total period that a digital waveform is in the HIGH state. \( DC = \frac{t_h}{T} \) (often expressed as a percentage: \( %DC = \frac{t_h}{T} \times 100\% \)).

Clock A special case of a symmetrical, periodic waveform with a specified frequency.

Periodic waveforms repeat the same pattern of HIGHs and LOWs over a specified period of time. The waveform may or may not be symmetrical; that is, it may or may not be HIGH and LOW for equal amounts of time.

**Example 3.11**

Calculate the time LOW, time HIGH, period, frequency, and percent duty cycle for each of the periodic waveforms in Figure 3.8. How are the waveforms similar? How do they differ?

**Solution**

a. Time LOW: \( t_l = 3 \) ms

Time HIGH: \( t_h = 1 \) ms

Period: \( T = t_l + t_h = 3 \) ms + 1 ms = 4 ms

Frequency: \( f = 1/T = 1/(4 \) ms) = 0.25 kHz = 250 Hz

Duty cycle: \( %DC = \frac{t_h}{T} \times 100\% = (1\) ms/4 ms) \times 100\% = 25\%

(1 ms = 1/1,000 second; 1 kHz = 1,000 Hz)

b. Time LOW: \( t_l = 2 \) ms

Time HIGH: \( t_h = 2 \) ms

Period: \( T = t_l + t_h = 2 \) ms + 2 ms = 4 ms

Frequency: \( f = 1/T = 1/(4 \) ms) = 0.25 kHz = 250 Hz

Duty cycle: \( %DC = \frac{t_h}{T} \times 100\% = (2\) ms/4 ms) \times 100\% = 50\%

c. Time LOW: \( t_l = 1 \) ms

Time HIGH: \( t_h = 3 \) ms

Period: \( T = t_l + t_h = 1 \) ms + 3 ms = 4 ms

Frequency: \( f = 1/T = 1/(4 \) ms) = 0.25 kHz = 250 Hz

Duty cycle: \( %DC = \frac{t_h}{T} \times 100\% = (3\) ms/4 ms) \times 100\% = 75\%

The waveforms all have the same period but different duty cycles. A square waveform, shown in Figure 3.8b, has a duty cycle of 50%.

A clock signal, or more simply, a clock, is a special case of a symmetrical periodic waveform as shown in Figure 3.8b. Although the duty cycle of a clock
Digital Electronics

Note... A clock signal, or more simply, a clock, is a special case of a symmetrical periodic waveform.

doesn’t have to be 50%, it typically is close to 50%. Clock signals are useful because we know the frequency, or how often the signal will change. For example, a 1-GHz computer will have an internal clock signal that will change from LOW to HIGH one billion times per second.

Aperiodic Waveforms

**KEY TERM**

Aperiodic waveform A time-varying sequence of logic HIGHs and LOWs that does not repeat.

An aperiodic waveform does not repeat a pattern of 0’s and 1’s. Thus, the parameters of time HIGH, time LOW, frequency, period, and duty cycle have no meaning for an aperiodic waveform. Most waveforms of this type are one-of-a-kind specimens.

Figure 3.9 shows some examples of aperiodic waveforms.

Example 3.12

A digital circuit generates the following strings of 0’s and 1’s:

- a. 001111101101101110011000
- b. 001100110011001100110011
- c. 000000011111101000000111
- d. 101110111011111011101110

The time between two bits is always the same. Sketch the resulting digital waveform for each string of bits. Which waveforms are periodic and which are aperiodic?

**Solution**

Figure 3.10 shows the waveforms corresponding to the strings of bits just mentioned. The waveforms are easier to draw if you break up the bit strings into smaller groups of, say, 4 bits each. For instance:

- a. 0011 1111 0110 1011 0100 0011 0000

All of the waveforms except waveform (a) are periodic.

Pulse Waveforms

**KEY TERMS**

Pulse A momentary variation of voltage from one logic level to the opposite level and back again.

Rising edge The part of a signal where the logic level is in transition from a LOW to a HIGH. In an ideal pulse, this is instantaneous.

Falling edge (or trailing edge) The part of a signal where the logic level is in transition from a HIGH to a LOW.

Amplitude The instantaneous voltage of a waveform. Often used to mean maximum amplitude, or peak voltage, of a pulse.
Pulse width ($t_w$) (of an ideal pulse) The time from the rising to falling edge of a positive-going pulse, or from the falling to rising edge of a negative-going pulse.

Edge The part of the pulse that represents the transition from one logic level to the other.

In an ideal pulse, the rising edge and the falling edge are vertical. That is, the transitions between logic HIGH and LOW levels are instantaneous. Although there is no such thing as an ideal pulse (i.e., a pulse with absolutely vertical edges) in a real digital circuit, we can usually consider pulses and waveforms to be ideal and not consider the amount of time the transition really takes. Again, like turning on a light, this is the same as saying that a light turns on instantly when the switch is turned to ON, without considering how long the light took to go from OFF to ON. The transition is so fast that we can assume the wave changes instantaneously.

Figure 3.11 shows an ideal pulse. The rising edge and falling edge of an ideal pulse are vertical. That is, the transitions, or edges, between logic HIGH and LOW levels are instantaneous.

Pulses can be either positive-going or negative-going, as shown in Figure 3.12. In a positive-going pulse, the measured logic level is normally LOW, goes HIGH for the duration of the pulse, and returns to the LOW state. A negative-going pulse acts in the opposite direction.

Figure 3.12 Pulse Edges

The amplitude of the pulse is the voltage value of its maximum height; in a digital circuit, this is often 5 volts. The pulse width, as shown in Figure 3.13, is the time from the rising edge to the falling edge of a positive-going pulse, or the falling to rising edge of a negative-going pulse.

Figure 3.13 Pulse Width

Your Turn

3.13 A digital circuit produces a waveform that can be described by the following periodic bit pattern: 0011001100110011.
   a. What is the duty cycle of the waveform?
   b. Write the bit pattern of a waveform with the same duty cycle and twice the frequency of the original.
   c. Write the bit pattern of a waveform having the same frequency as the original and a duty cycle of 75%.
1. The two basic areas of electronics are analog and digital electronics. Analog electronics deals with continuously variable quantities; digital electronics represents the world in discrete steps.
2. Digital logic uses defined voltage levels, called logic levels, to represent binary numbers within an electronic system.
3. The higher voltage in a digital system represents the binary digit 1 and is called a logic HIGH or logic 1. The lower voltage in a system represents the binary digit 0 and is called a logic LOW or logic 0.
4. The logic levels of multiple locations in a digital circuit can be combined to represent a multibit binary number.
5. Binary is a positional number system (base 2) with two digits, 0 and 1, and positional multipliers that are powers of 2.
6. The bit with the largest positional weight in a binary number is called the most significant bit (MSB); the bit with the smallest positional weight is called the least significant bit (LSB). The MSB is also the leftmost bit in the number; the LSB is the rightmost bit.
7. A decimal number can be converted to binary by sum of powers of 2 (add place values to get a total) or repeated division by 2 (divide by 2 until the quotient is 0; remainders are the binary value).
8. The positional multipliers in a fractional binary number are negative powers of 2.
9. The hexadecimal number system is based on 16. It uses 16 digits, from 0–9 and A–F, with power-of-16 multipliers.
10. Each hexadecimal digit uniquely corresponds to a 4-bit binary value. Hex digits can thus be used as shorthand for binary.
11. A digital waveform is a sequence of bits over time. A waveform can be periodic (repetitive), aperiodic (nonrepetitive), or pulsed (a single variation and return between logic levels).
12. Periodic waveforms are measured by period (T: time for one cycle), time HIGH (th), time LOW (tl), frequency (f: number of cycles per second), and duty cycle (DC or %DC: fraction of cycle in HIGH state).
13. Pulse waveforms are measured by pulse width (tw) and amplitude.

**BRING IT HOME**

3.1 Digital versus Analog Electronics

3.1 Which of the following quantities is analog in nature and which digital? Explain your answers.
   a. Water temperature at the beach
   b. Weight of a bucket of sand
   c. Grains of sand in a bucket
   d. Waves hitting the beach in 1 hour
   e. Height of a wave
   f. People in a square mile

3.2 Which of the following quantities is analog in nature and which digital? Explain your answers.
   a. Number of students in a classroom
   b. Winning score of a basketball game
   c. Height of the tallest player on a team
   d. Speed of a roller coaster
   e. Roller coaster riders per hour

58 Digital Electronics
3.2 Digital Logic Levels

3.3 A digital logic system is defined by the voltages 3.3 volts and 0 volts. For a positive logic system, state which voltage corresponds to a logic 0 and which to a logic 1.

3.4 A digital logic system is defined by the voltages 1.8 volts and 0 volts. For a positive logic system, state which voltage corresponds to a logic 0 and which to a logic 1.

3.3 The Binary Number System

3.5 Calculate the decimal values of each of the following binary numbers:
   a. 100
   b. 1000
   c. 11001
   d. 110
   e. 10101
   f. 11101
   g. 100001
   h. 10111001

3.6 Calculate the decimal values of each of the following binary numbers:
   a. 101
   b. 1001
   c. 10110
   d. 111
   e. 11101
   f. 111011
   g. 1010101
   h. 100001

3.7 Translate each of the following combinations of HIGH (H) and LOW (L) logic levels to binary numbers using positive logic:
   a. H H L H
   b. L H L H
   c. H L H L
   d. L L L H

3.8 Translate each of the following combinations of HIGH (H) and LOW (L) logic levels to binary numbers using positive logic:
   a. H L L L
   b. L L L L
   c. H H H L L
   d. H L L H H L

3.9 List the sequence of binary numbers from 101 to 1000.

3.10 List the sequence of binary numbers from 10000 to 11111.

3.11 What is the decimal value of the most significant bit for the numbers in Problem 3.10?

3.12 Convert the following decimal numbers to binary. Use the sum-of-powers-of-2 method for parts a, c, e, and g. Use the repeated-division-by-2 method for parts b, d, f, and h.
   a. 75
   b. 237
   c. 198
   d. 63
   e. 83
   f. 64
   g. 4087
   h. 8193

3.13 Convert the following decimal numbers to binary. Use the sum-of-powers-of-2 method.
   a. 65
   b. 249
   c. 189
   d. 98
   e. 32
   f. 2177

3.14 Convert the following decimal numbers to binary. Use the repeated-division-by-2 method.
   a. 35
   b. 194
   c. 311
   d. 89
   e. 128
   f. 3247

3.4 Hexadecimal Numbers

3.15 Write all the hexadecimal numbers in sequence from 308H to 321H inclusive.

3.16 Write all the hexadecimal numbers in sequence from 9F7H to A03H inclusive.

3.17 Convert the following hexadecimal numbers to their decimal equivalents:
   a. 1A0H
   b. 10AH
   c. FFFFH
   d. F3C8H
   e. D3B4H
   f. C000H

3.18 Convert the following hexadecimal numbers to their decimal equivalents:
   a. 2BCH
   b. 10FH
   c. 1000H
   d. A38DH
   e. A222H
   f. 30BAFH

continue...
3.19 Convert the following decimal numbers to their hexadecimal equivalents:
   a. \(709_{10}\)
   b. \(1889_{10}\)
   c. \(4225_{10}\)
   d. \(10128_{10}\)
   e. \(32000_{10}\)
   f. \(32768_{10}\)

3.20 Convert the following decimal numbers to their hexadecimal equivalents:
   a. \(907_{10}\)
   b. \(1789_{10}\)
   c. \(4095_{10}\)
   d. \(4096_{10}\)
   e. \(31999_{10}\)
   f. \(33000_{10}\)

3.21 Convert the following hexadecimal numbers to their binary equivalents:
   a. \(F3C8H\)
   b. \(D3B4H\)
   c. \(8037H\)
   d. \(FABDH\)
   e. \(30ACH\)
   f. \(3E7B6H\)

3.22 Convert the following hexadecimal numbers to their binary equivalents:
   a. \(3FFFH\)
   b. \(FACEH\)
   c. \(A123H\)
   d. \(3214H\)
   e. \(3F36BH\)
   f. \(4952FEH\)

3.23 Convert the following binary numbers to their hexadecimal equivalents:
   a. \(101111010000110_{2}\)
   b. \(101101101010_{2}\)
   c. \(110001011110000100_{2}\)
   d. \(10101011110001011_{2}\)
   e. \(11001100010110111_{2}\)

3.24 Convert the following binary numbers to their hexadecimal equivalents:
   a. \(110110101101_{2}\)
   b. \(100101010101_{2}\)
   c. \(111110111101_{2}\)
   d. \(1011001100110101_{2}\)
   e. \(110000001000011011_{2}\)
   f. \(101000000000000000_{2}\)

3.25 Calculate the time LOW, time HIGH, period, frequency, and percent duty cycle for the waveforms shown in Figure 3.14. How are the waveforms similar? How do they differ?

FIGURE 3.14: Problem 3.25: Periodic Waveforms

3.26 Calculate the time LOW, time HIGH, period, frequency, and percent duty cycle for the waveform shown in Figure 3.15c.

FIGURE 3.15: Problems 3.26 and 3.27: Aperiodic and Periodic Waveforms

3.27 Which of the waveforms in Figure 3.15 are periodic and which are aperiodic? Explain your answers.

3.28 Sketch the pulse waveforms represented by the following strings of 0’s and 1’s. State which waveforms are periodic and which are aperiodic.
   a. \(110011110001100000001110110101_{2}\)
   b. \(1111001100110001110001110000111_{2}\)
   c. \(1111111000000011111111111111_{2}\)

3.29 Draw a timing diagram for the signals represented by the following strings of 0’s and 1’s. State which waveforms are periodic and which are aperiodic.
   a. \(011001100110011001100110011010001_{2}\)
   b. \(0110110110011010100110110101_{2}\)
   c. \(11111111000000011111111111111_{2}\)

3.30 Classify each of the waveforms in Figure 3.16 as aperiodic or periodic. For the periodic waveforms, calculate time HIGH, time LOW, period, frequency, and duty cycle.
3.31 For each of the periodic waveforms shown in Figure 3.17, calculate the period, frequency, time HIGH, time LOW, and percent duty cycle. (The time scale is shown in nanoseconds; 1 ns = $10^{-9}$ seconds.)

3.32 Describe each of the periodic waveforms shown in Figure 3.17 as a clock signal, specifying its speed.

3.33 Calculate the pulse width of the pulse shown in Figure 3.18.


FIGURE 3.18: Problem 3.33: Pulse

% of full amplitude