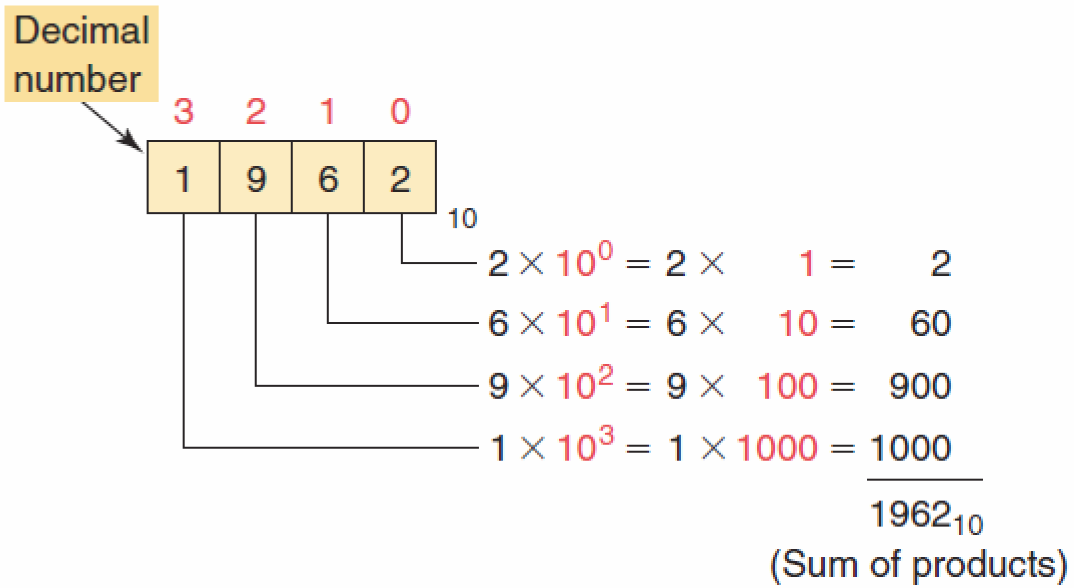
Chapter 3

Number Systems and Codes

Decimal System

The *radix* or *base* of a number system determines the total number of different symbols or digits used by that system.

The decimal system has a base of 10 with the digits 0 through 9 being used.

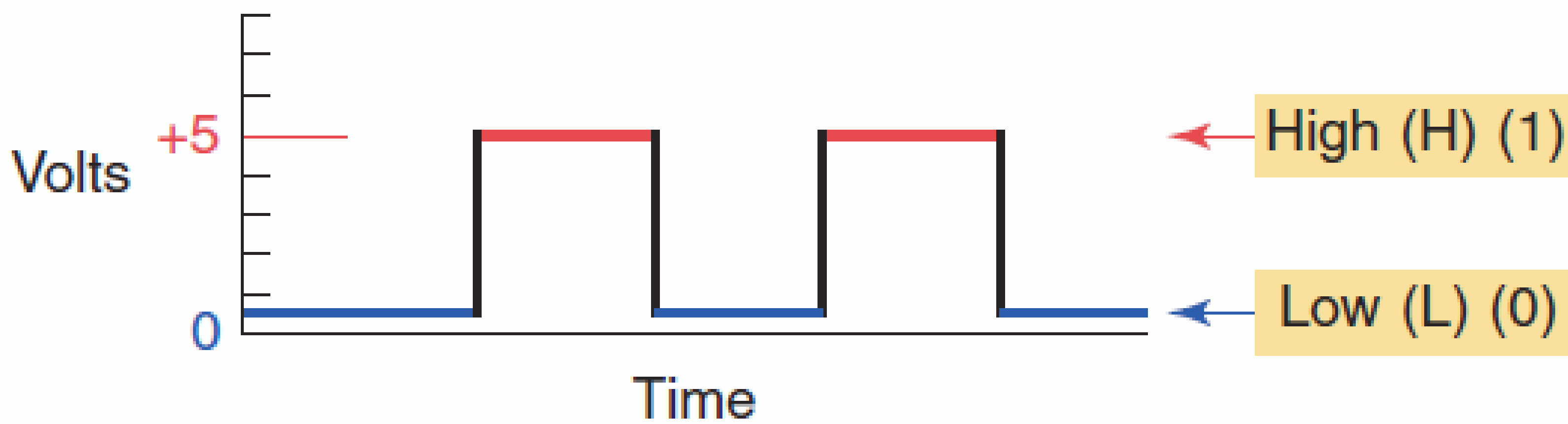


The value of a decimal number depends on the digits that make up the number and the place value of each digit.

3.2

Binary System

The *binary system* has a base of *2* and the only allowable digits are *0* and *1*.



With digital circuits it is easy to distinguish between two voltage levels, which can be related to the binary digits 1 and 0.

Since the binary system uses only two digits, each position of a binary number can go through only *two changes*, and then a 1 is carried to the immediate left position.

Decimal:	0	1	2	3	4	5	6	7	8	9	10
Binary:	0	1	10	11	100	101	110	111	1000	1001	1010

All numbering systems start at *zero*.

NUMBER SYSTEMS

Decimal (base 10)

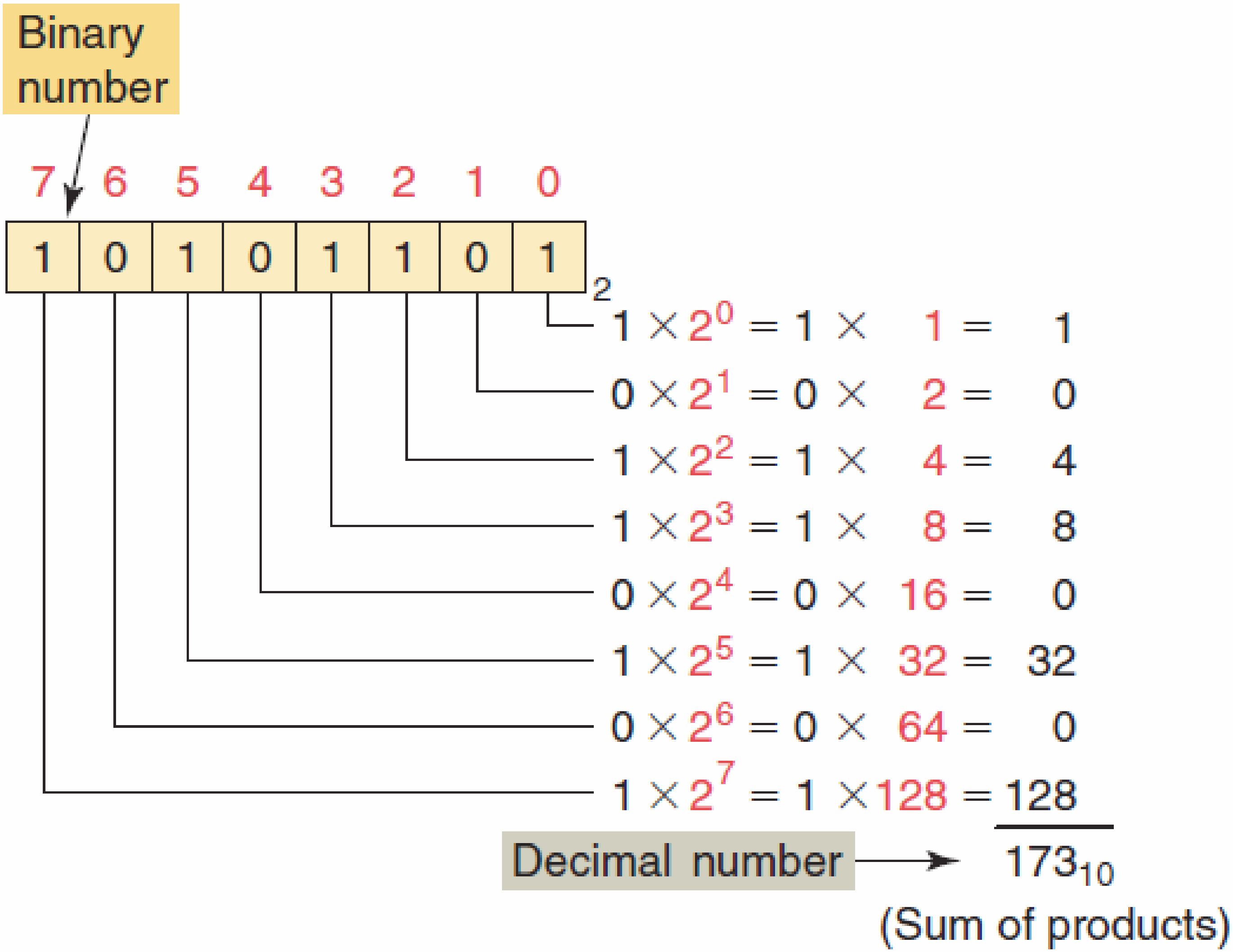
Binary (base 2)

Octal (base 8)

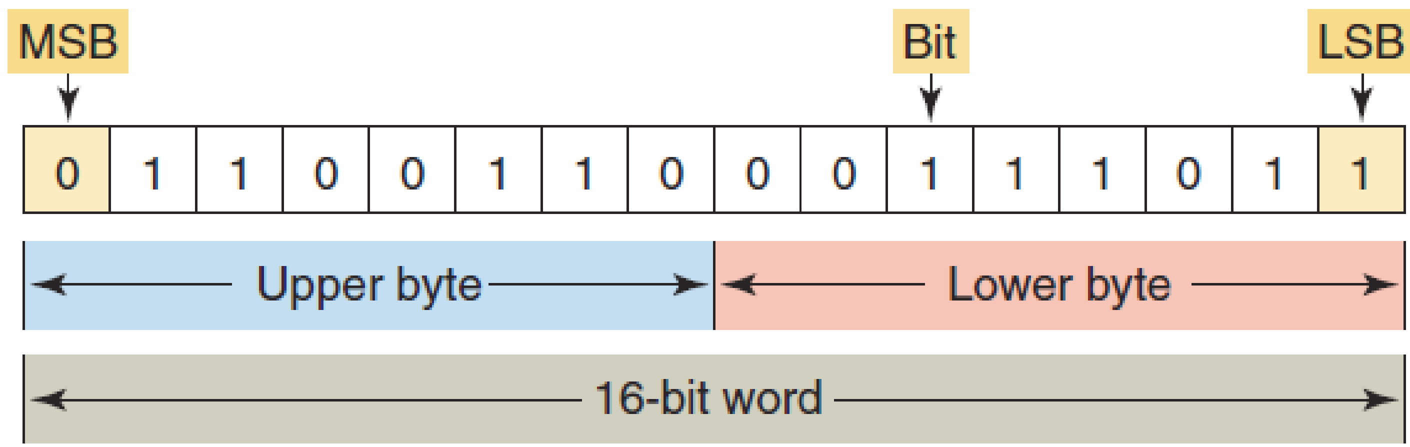
Hexadecimal (base 16)

Decimal	Octal	Hexadecimal	Binary
0	0	0	0
1	1	1	1
2	2	2	10
3	3	3	11
4	4	4	100
5	5	5	101
6	6	6	110
7	7	7	111
8	10	8	1000
9	11	9	1001
10	12	A	1010
11	13	B	1011
12	14	C	1100
13	15	D	1101
14	16	E	1110
15	17	F	1111
16	20	10	10000
17	21	11	10001
18	22	12	10010
19	23	13	10011
20	24	14	10100

Converting a *binary* number to a *decimal* number.



Each digit of a binary number is known as a *bit*.



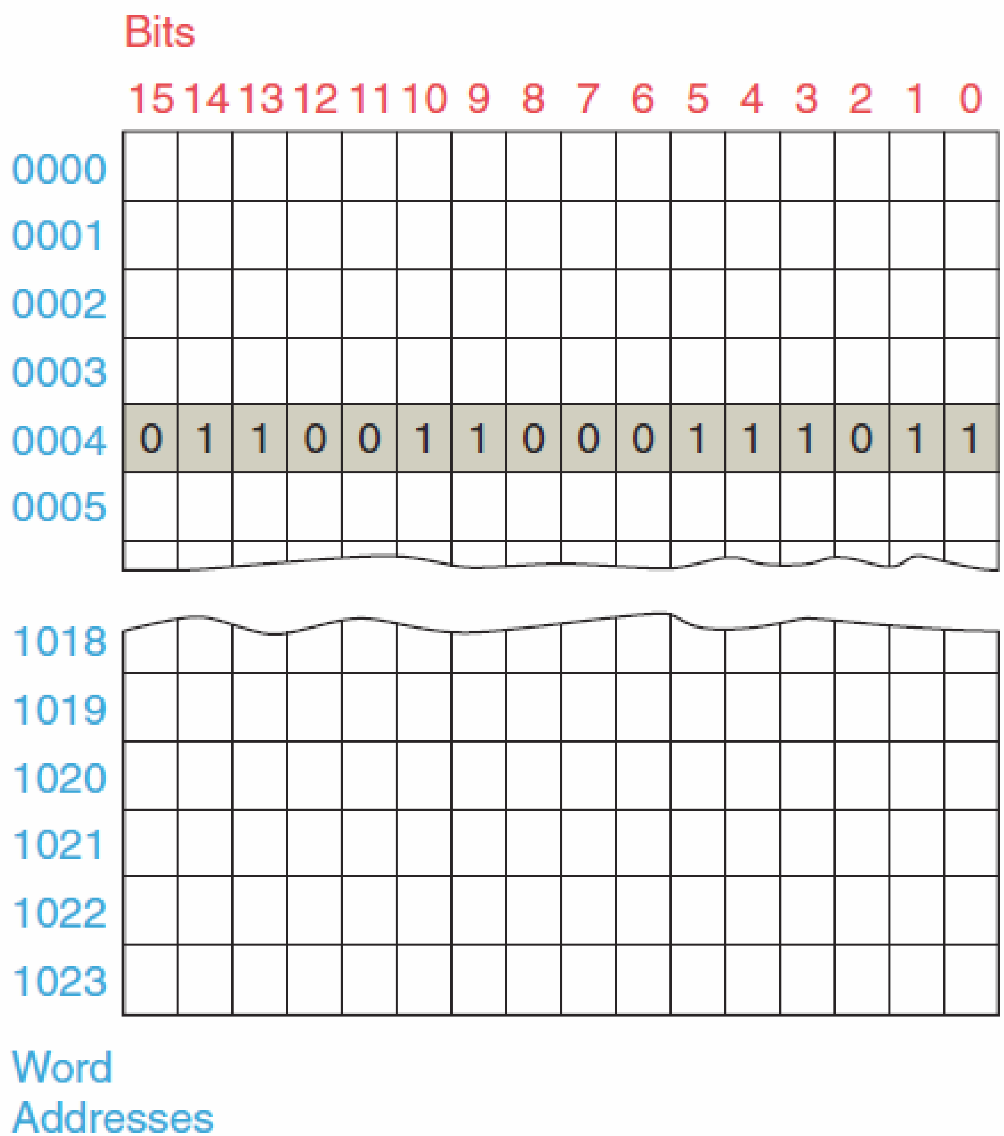
Each word is capable of storing data in the form of bits.

A group of 8 bits is a byte.

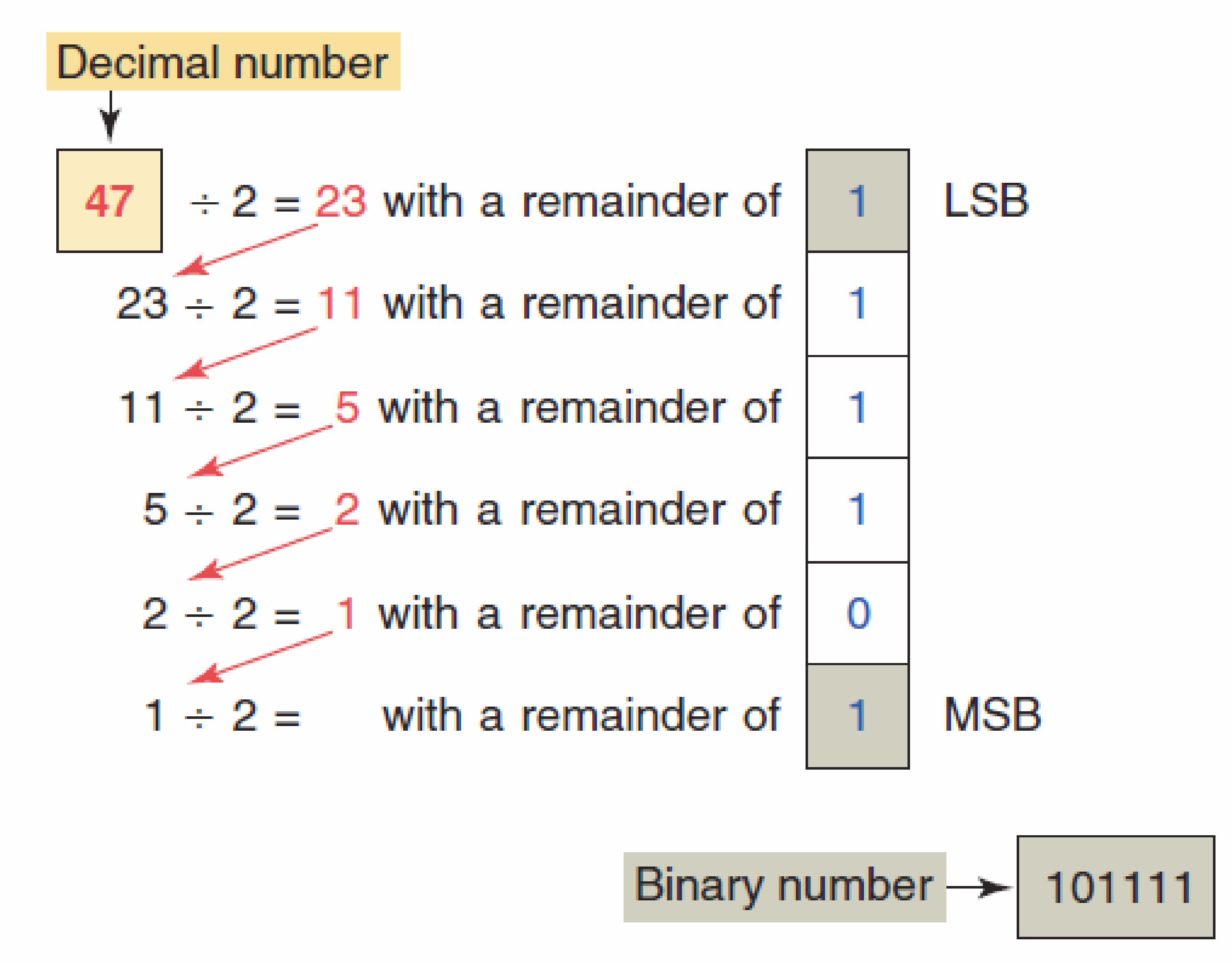
The least significant bit (LSB) represents the smallest value and the most significant bit (MSB) represents the largest value.

PLC memory is organized using *bytes*, *single words*, or *double words*.

If the memory size is 1 K word, it can store 1024 words or 16,384 (1024 x 16) bits of information using 16-bit words,



Converting a *decimal* number to a *binary* number.



Computer memory is a series of binary *1s* and *0s*.

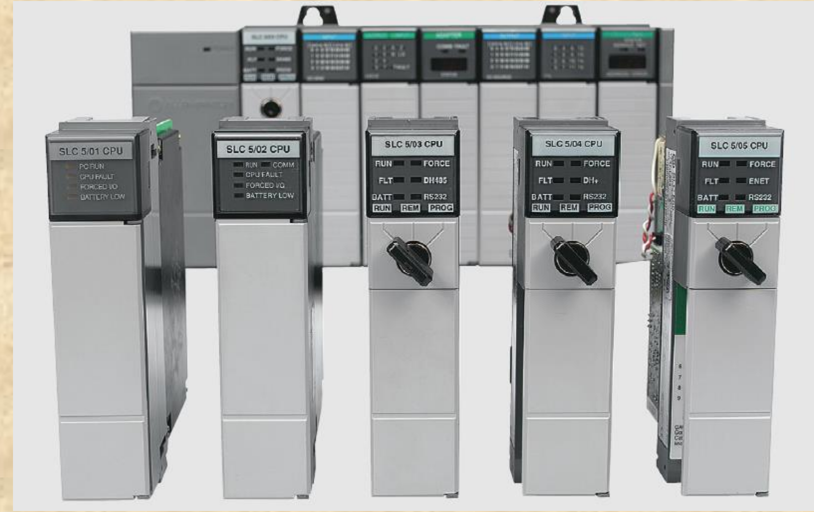
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address
1	1	0	0	0	0	1	0	1	1	1	1	0	0	0	1	O:1
0	0	1	1	0	1	0	0	0	0	0	0	1	1	1	1	O:2
1	0	1	0	1	1	0	0	1	1	1	0	0	0	0	1	O:3
0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	O:4
1	1	1	0	1	0	0	1	1	1	0	0	1	1	0	1	O:5

Output status file made up of single bits grouped into 16-bit words.

One 16-bit output file word is reserved for each slot in the chassis.

Each bit represents the ON or OFF state of one output point.

3.3



Negative Numbers

Binary systems can not use *positive* and *negative* symbols to represent the polarity of a number.

In signed binary numbers, the left-most bit is the equivalent of a +/- sign. "0" indicates that the number is positive, "1" indicates negative.

Signed Binary Numbers

Magnitude
Sign

Same as binary numbers

	Decimal Value
0111	+7
0110	+6
0101	+5
0100	+4
0011	+3
0010	+2
0001	+1
0000	0
1001	−1
1010	−2
1011	−3
1100	−4
1101	−5
1110	−6
1111	−7

Another method of expressing a negative number is by using the *complement* of a binary number.

To complement a binary number, change all the 1s to 0s and all the 0s to 1s. This is known as the 1's complement form of a binary number.

	Bit Pattern	Decimal Value
Binary Number →	0000 0000	0
1's Complement →	1111 1111	-0
	0000 0001	1
	1111 1110	-1
	0000 0011	3
	1111 1100	-3
	0001 1111	31
	1110 0000	-31

2's complement is the binary number that results when 1 is added to the 1's complement.

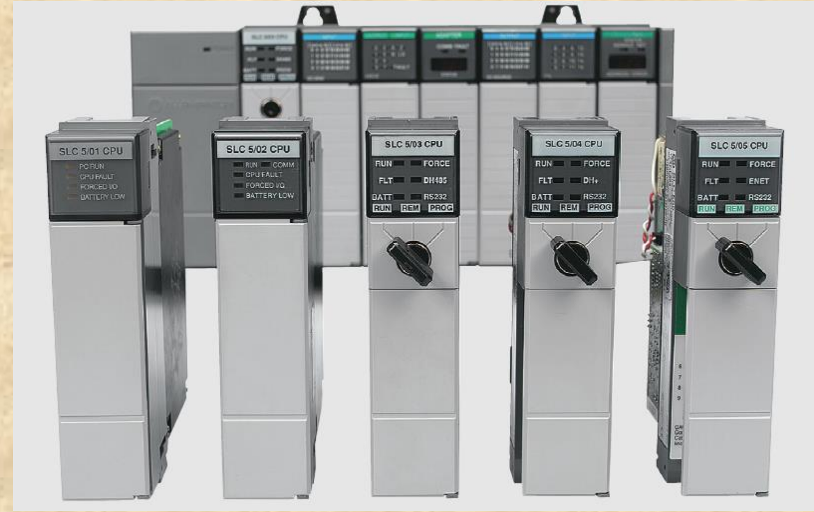
0 sign bit indicates a positive number.

1 sign bit indicates a negative number.

Signed Decimal	1's Complement	2's Complement
+7	0111	0111
+6	0110	0110
+5	0101	0101
+4	0100	0100
+3	0011	0011
+2	0010	0010
+1	0001	0001
0	0000	0000
−1	1110	1111
−2	1101	1110
−3	1100	1101
−4	1011	1100
−5	1010	1011
−6	1001	1010
−7	1000	1001

Same as binary numbers

3.4



Octal System

The *octal numbering system* is a base 8 system

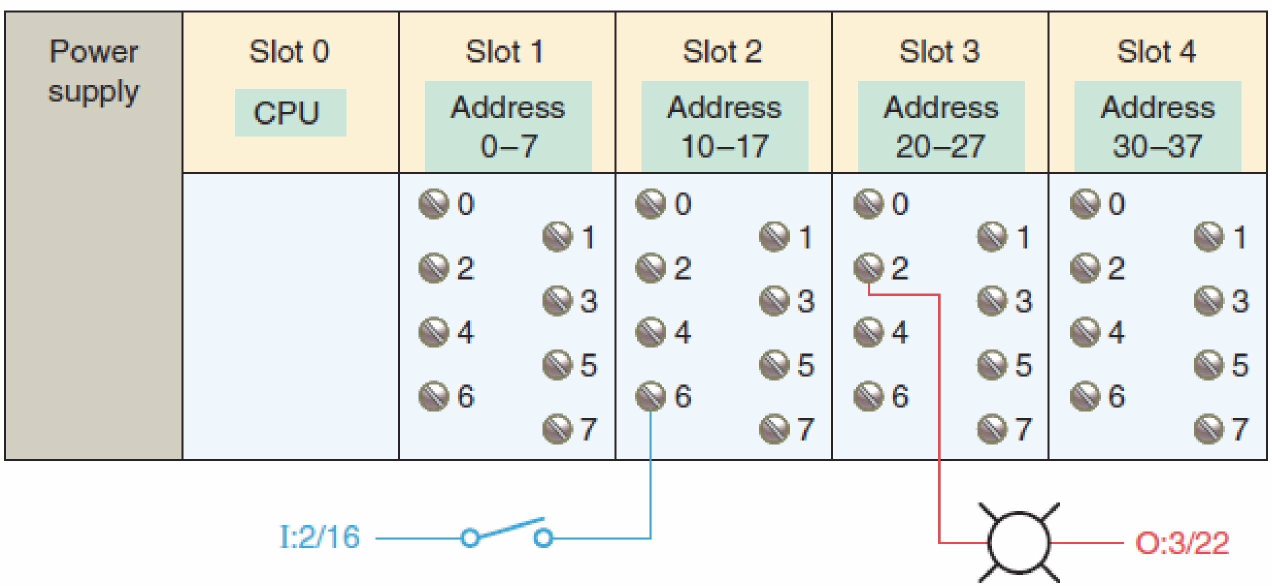
Decimal	Binary	Octal
0	0	0
1	1	1
2	10	2
3	11	3
4	100	4
5	101	5
6	110	6
7	111	7
8	1000	10
9	1001	11
10	1010	12
11	1011	13
12	1100	14
13	1101	15
14	1110	16
15	1111	17
16	10000	20
17	10001	21
18	10010	22
19	10011	23
20	10100	24

Numbers 8 and 9 not allowed

The octal numbering system is used because *8 data bits* make up a *byte* of information that can be addressed.

I/O module screw terminal number	Slot number and address			
	1	2	3	4
0	0	10	20	30
1	1	11	21	31
2	2	12	22	32
3	3	13	23	33
4	4	14	24	34
5	5	15	25	35
6	6	16	26	36
7	7	17	27	37

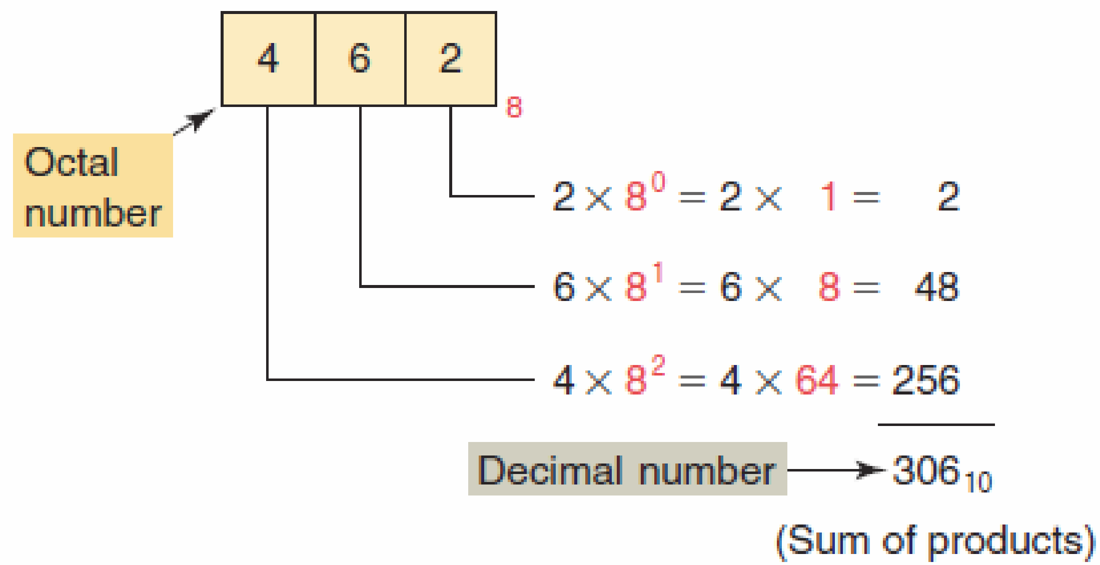
PLC-5 controllers use octal-based I/O addressing.



Octal is a convenient means of handling *large binary numbers*.

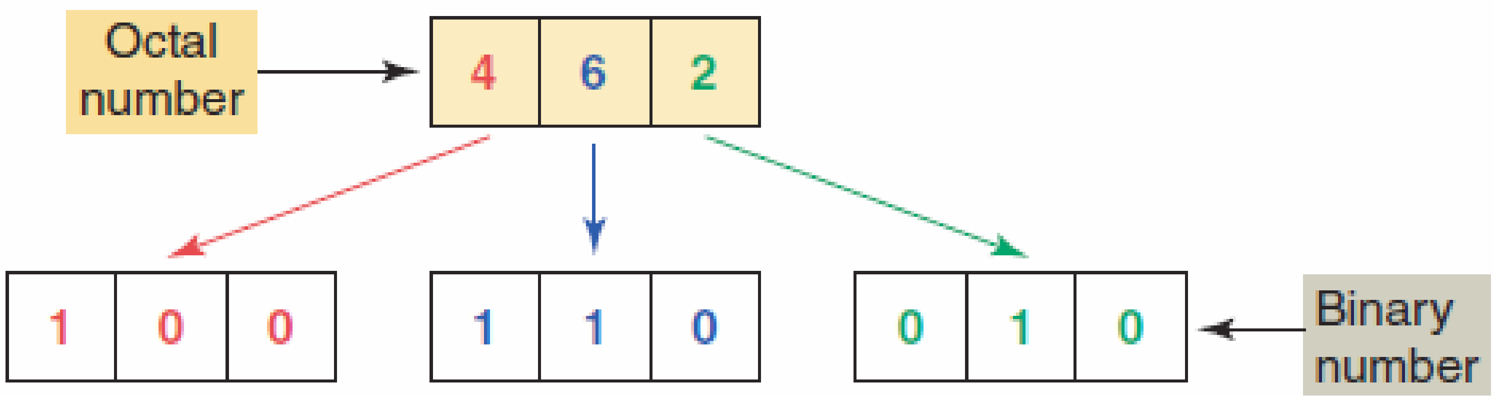
One octal digit can be used to express three binary digits.

Binary	Octal
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7



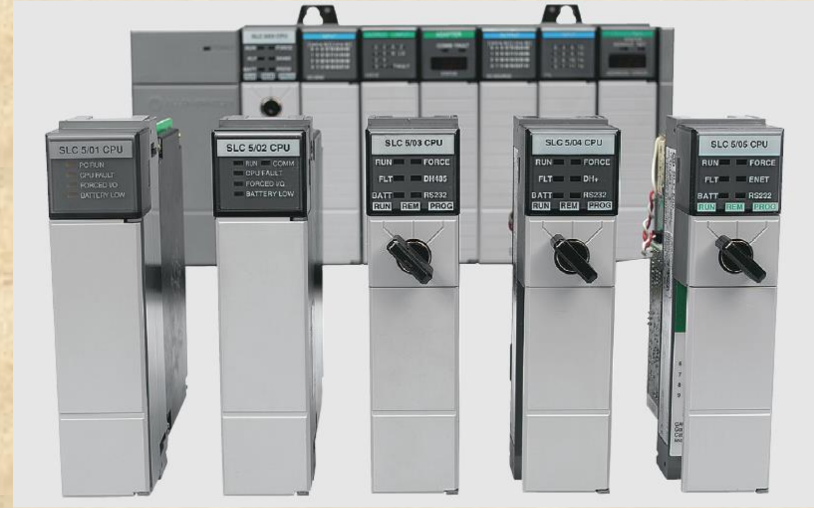
Converting an octal number to a decimal number.

Octal converts easily to *binary* equivalents.



The octal number 462 is converted to its binary equivalent by assembling the 3-bit groups.

3.5



Hexadecimal System

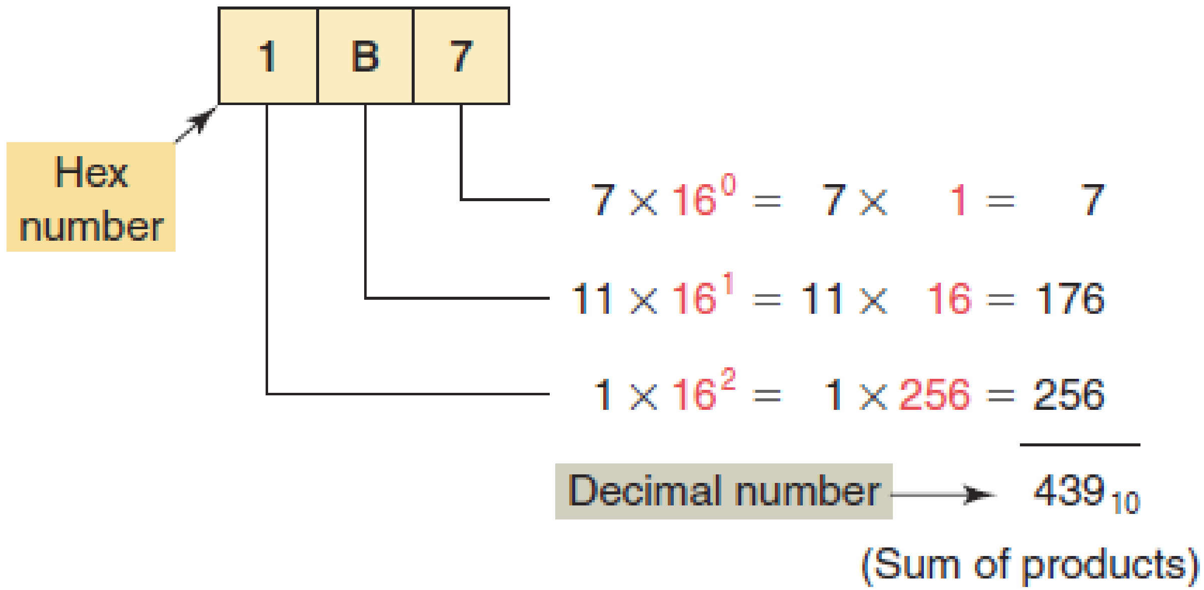
The *hexadecimal (hex)* system is a *base 16* system.

Hexadecimal	Binary	Decimal
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

Letters *A to F* are used to represent decimal numbers 10 to 15.

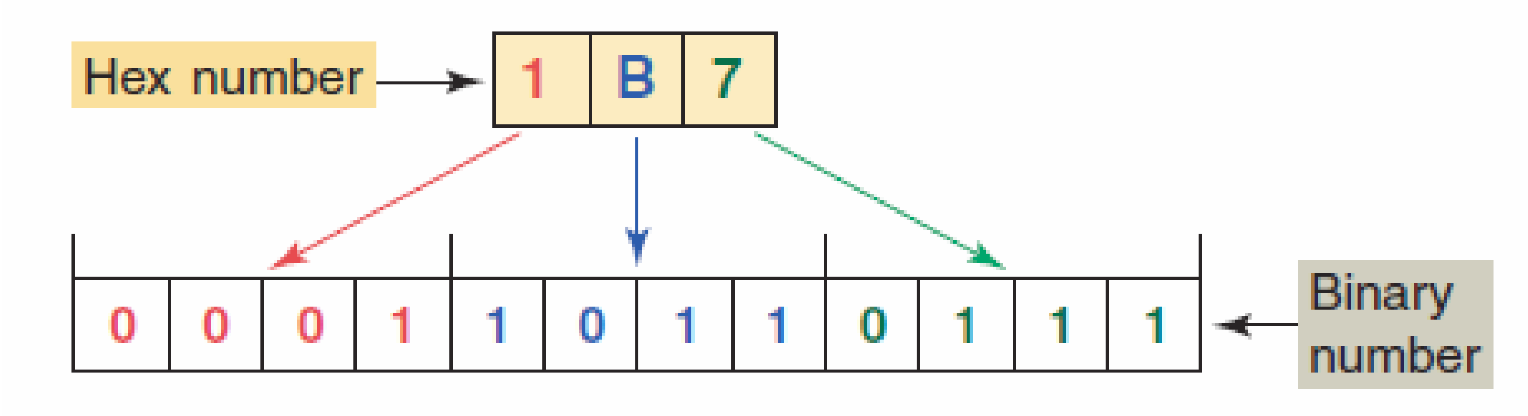
The hex numbering system is used in PLCs because a *word* of data consists of *16 data bits*.

Converting hexadecimal to decimal.



Hexadecimal digits in the columns are multiplied by the base 16 weight, depending on digit significance.

Converting decimal to hexadecimal.



Conversion is accomplished by writing the 4-bit binary equivalent of the hex digit for each position.

3.6

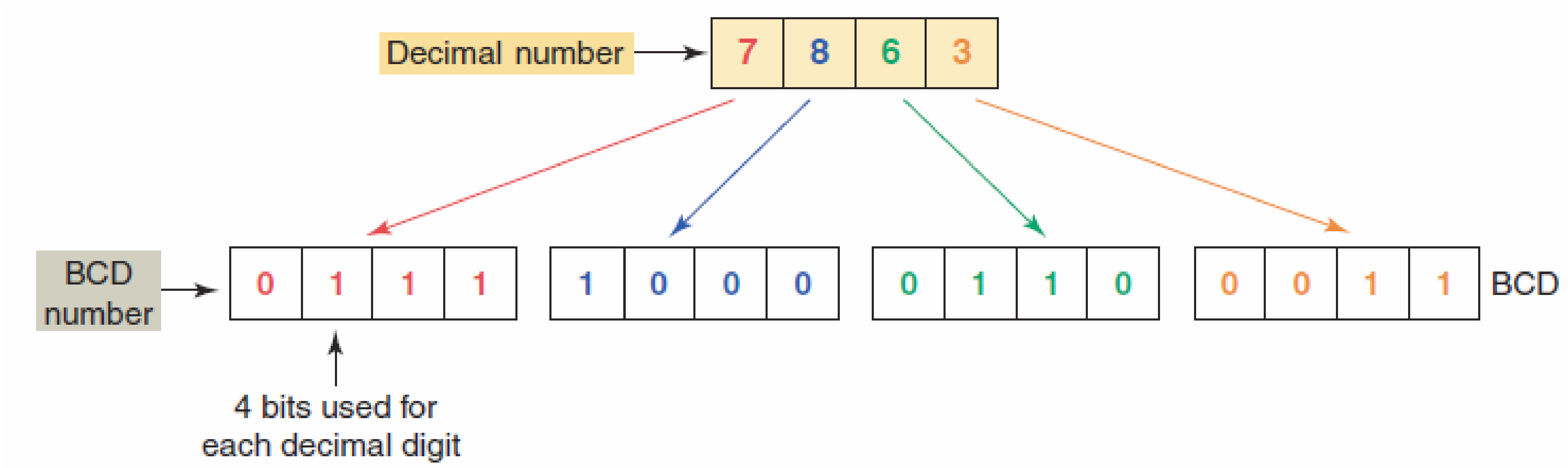
Binary Coded Decimal System

The *binary coded decimal (BCD)* system uses *4 bits* to represent each *decimal digit.*

The 4 bits used are the binary equivalents of the numbers from 0 to 9.

Decimal	Binary	BCD	Hexadecimal
0	0	0000	0
1	1	0001	1
2	10	0010	2
3	11	0011	3
4	100	0100	4
5	101	0101	5
6	110	0110	6
7	111	0111	7
8	1000	1000	8
9	1001	1001	9
10	1010	0001 0000	A
11	1011	0001 0001	B
12	1100	0001 0010	C
13	1101	0001 0011	D
14	1110	0001 0100	E
15	1111	0001 0101	F
16	1 0000	0001 0110	10
17	1 0001	0001 0111	11
18	1 0010	0001 1000	12
19	1 0011	0001 1001	13
20	1 0100	0010 0000	14

The BCD representation of a decimal number is obtained by replacing each decimal digit by its BCD equivalent.



To distinguish the BCD numbering system from a binary system, a *BCD designation* is placed to the right of the units digit.

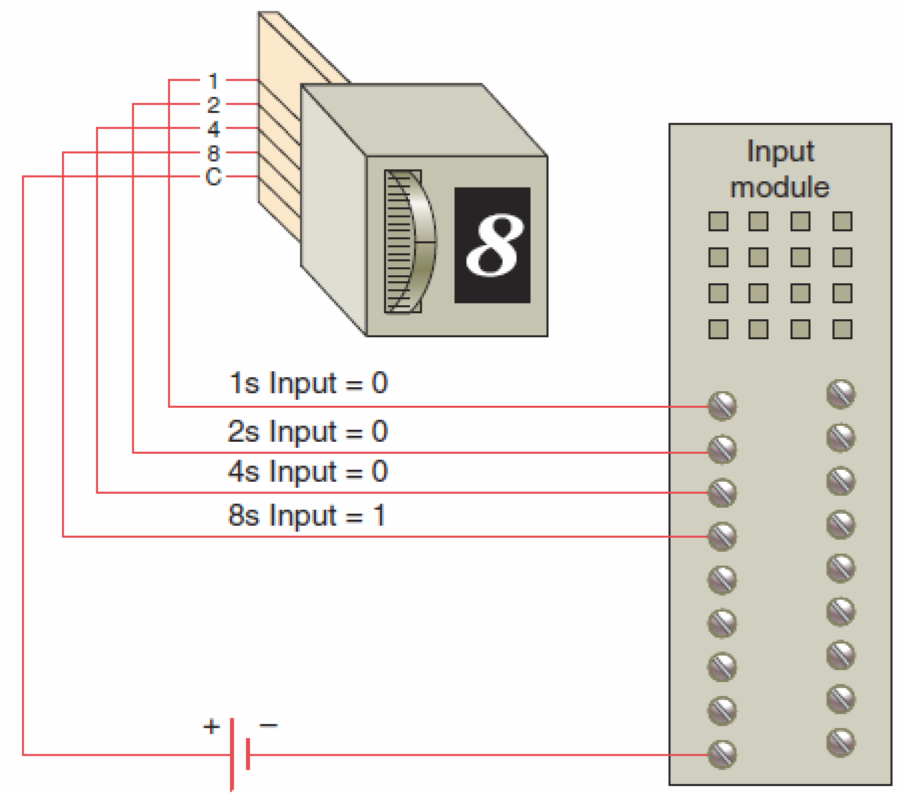
Decimal 49_{10} expressed is binary is 110001_2

Decimal 49_{10} expressed in BCD is 01001001_{BCD}

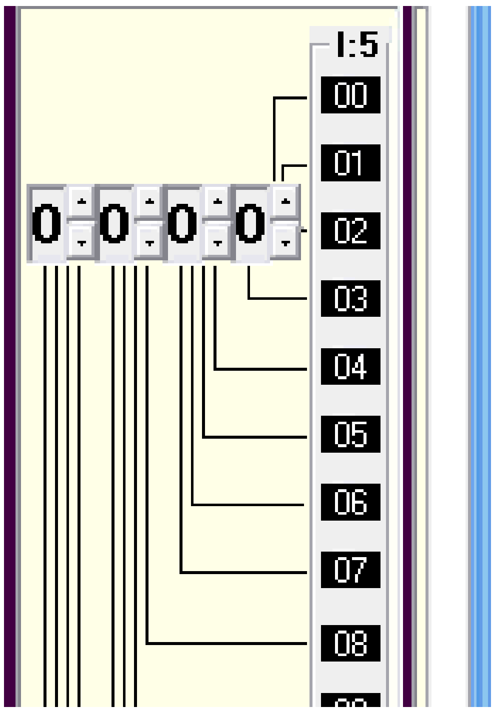
A *thumbwheel switch* is an input device that makes use of the BCD numbering system.

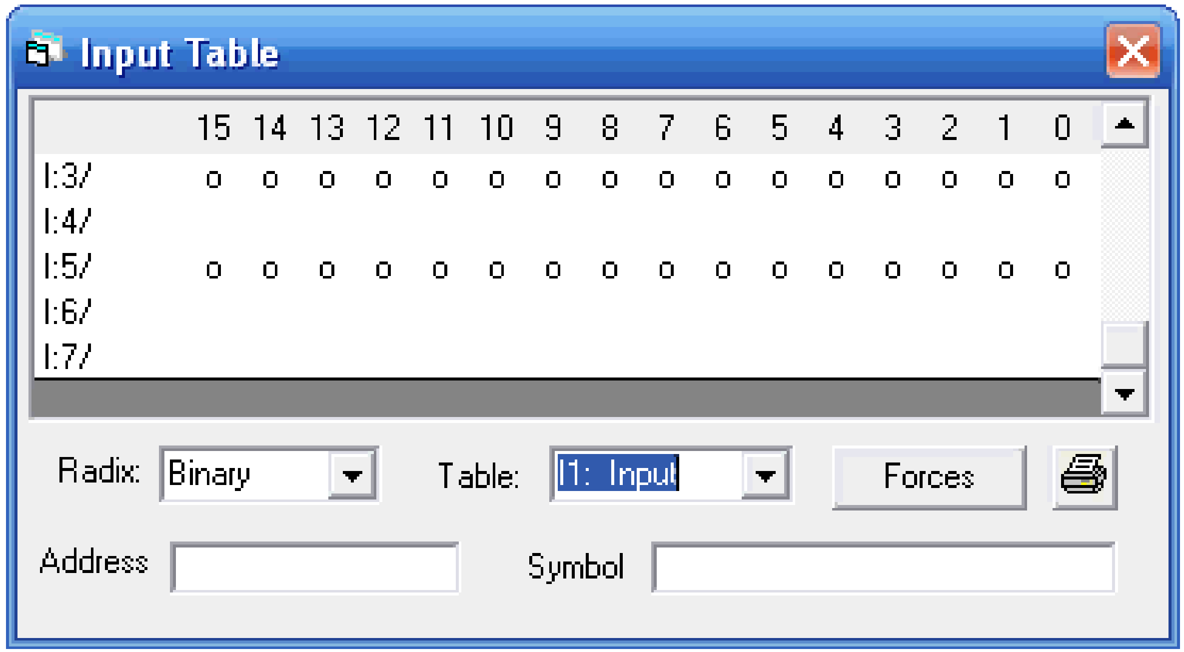
The input module attached to the thumbwheel has one connection for each bit's weighted value.

Selected digit of decimal 8 outputs the equivalent 4 bits of BCD data – 1000_{BCD}



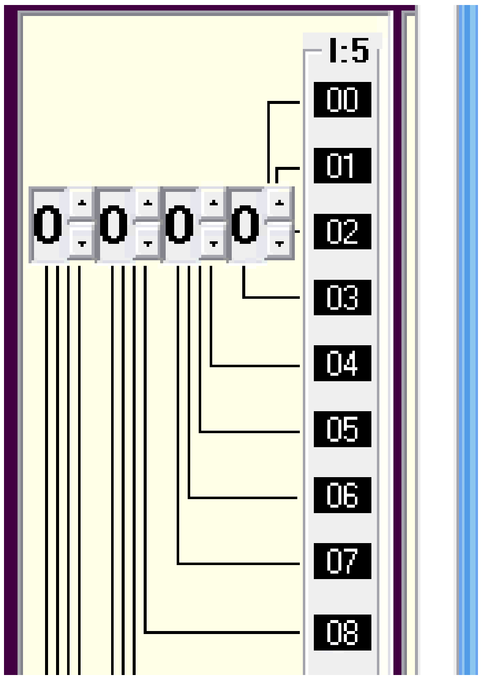
Single Digit Thumbwheel Switch Simulation

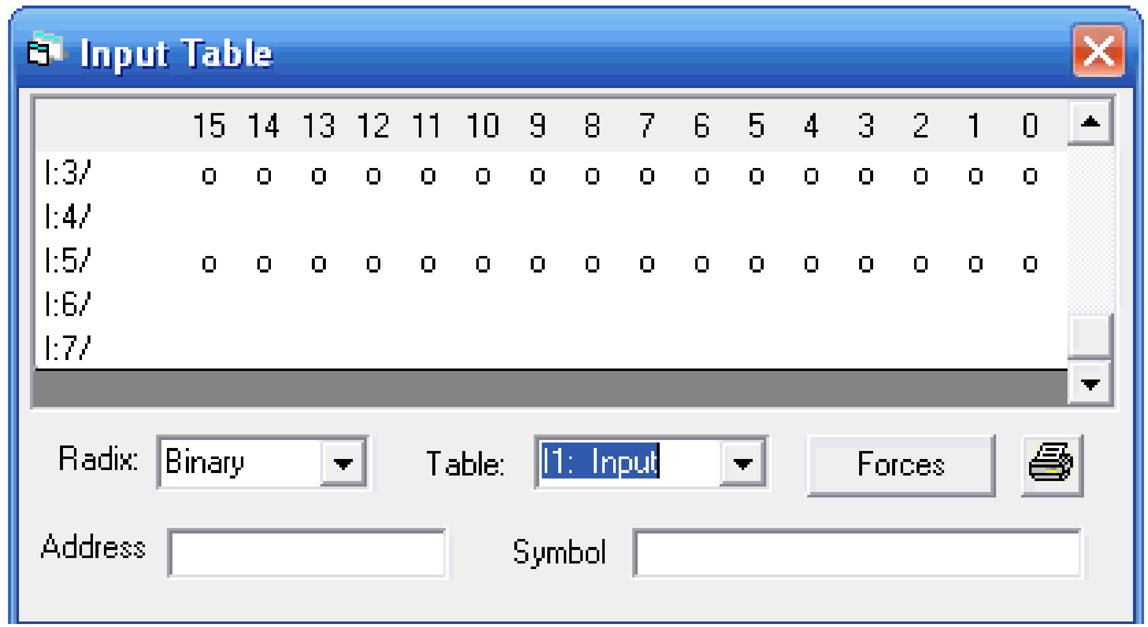




Decimal Settings **Binary Input**

Double Digit Thumbwheel Switch Simulation

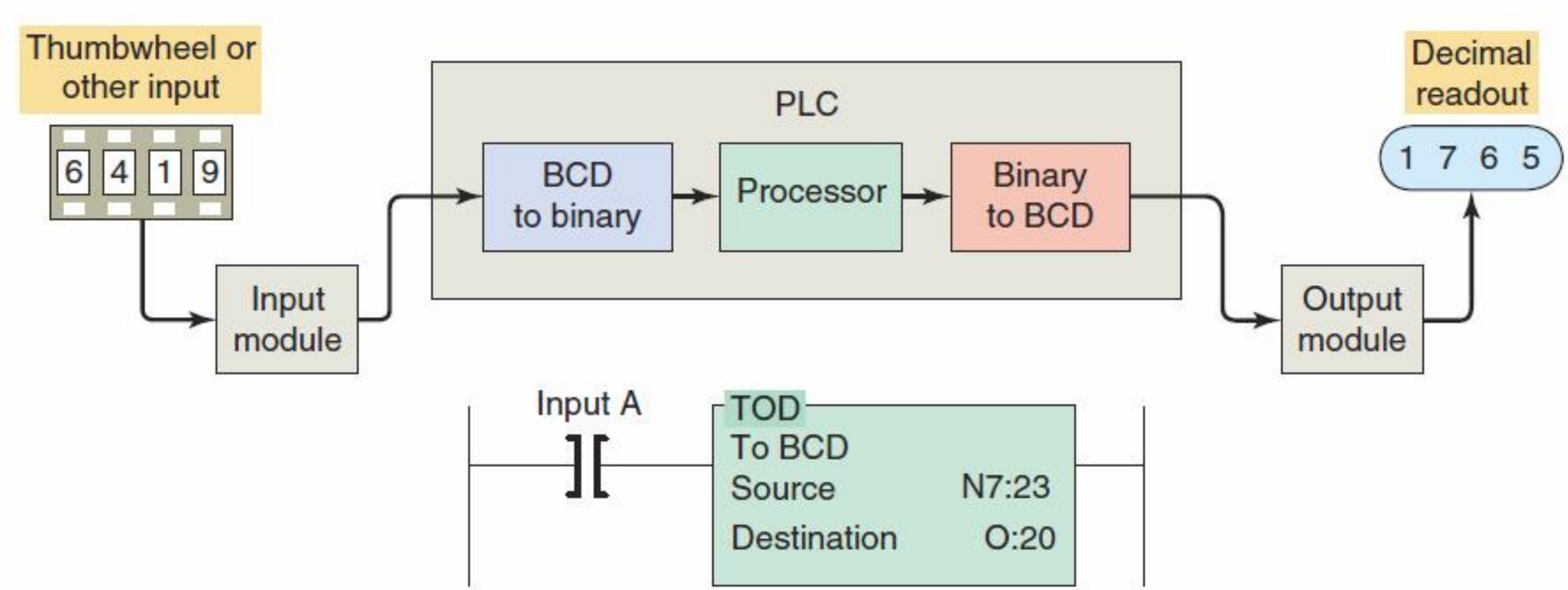




Decimal Settings

Binary Input

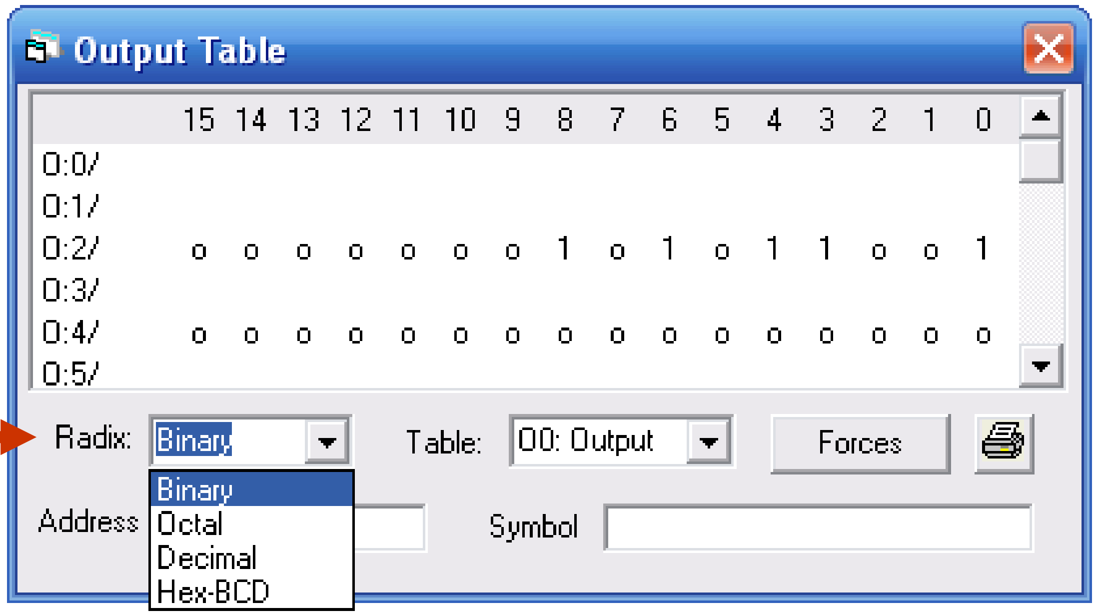
PLCs contain *number conversion* functions.



Convert-to-BCD instruction

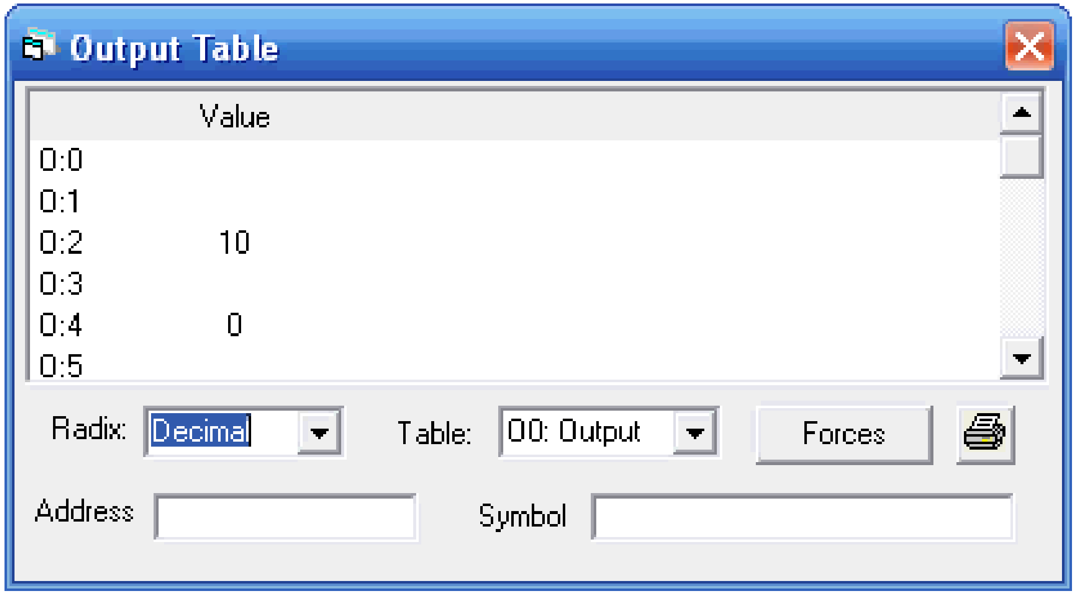
The Convert-to-BCD instruction will convert the binary bit pattern at the source address, N7:23, into a BCD bit pattern of the same decimal value and store it at the destination address, O:20.

Many PLCs allow you to *change the format* of the data that the data monitor displays.



The change radix function found on Allen-Bradley controllers allows you to change the display format of data to binary, octal, decimal, or hexadecimal.

Change Radix Function Simulation



Decimal	Binary	Octal	Hexadecimal
10	1010	12	A

3.7

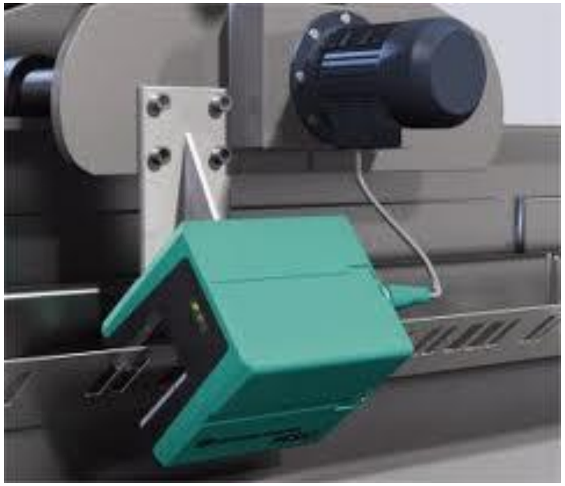
Gray Code

The Gray code is set up so that as we progress from one number to the next, *only one bit* changes.

Each position does not have a definite weight. This can be quite confusing for counting circuits, but it is ideal for encoder circuits.

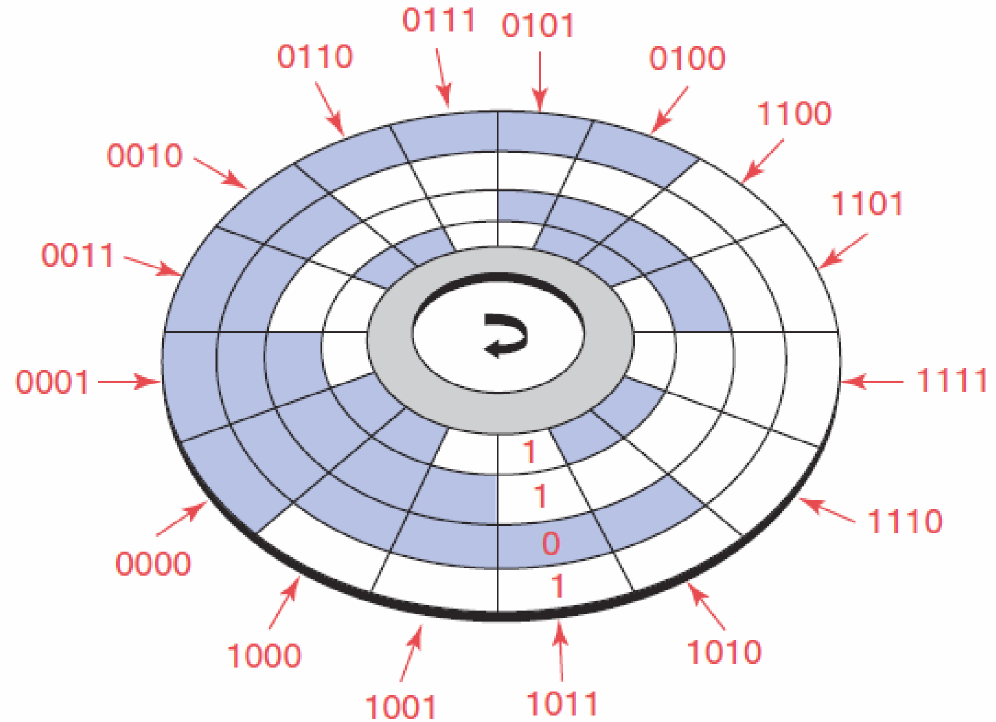
Gray Code	Binary
0000	0000
0001	0001
0011	0010
0010	0011
0110	0100
0111	0101
0101	0110
0100	0111
1100	1000
1101	1001
1111	1010
1110	1011
1010	1100
1011	1101
1001	1110
1000	1111

Gray codes are used with *position encoders* for accurate control of the motion of robots, machine tools, and servomechanisms.



Since only one bit changes at a time, the Gray code is considered to be an *error-minimizing* code.

Optical encoder disk that uses a *4-bit* Gray code.



The encoder disk is attached to a rotating shaft and outputs a digital Gray code signal that is used to determine the position of the shaft.

3.8

ASCII Code

ASCII stands for American Standard Code for Information Interchange.

It is an **alphanumeric** code because it includes letters as well as numbers.

Partial Listing of ASCII Code

Character	7-Bit ASCII	Character	7-Bit ASCII
A	100 0001	X	101 1000
B	100 0010	Y	101 1001
C	100 0011	Z	101 1010
D	100 0100	0	011 0000
E	100 0101	1	011 0001
F	100 0110	2	011 0010
G	100 0111	3	011 0011
H	100 1000	4	011 0100
I	100 1001	5	011 0101
J	100 1010	6	011 0110
K	100 1011	7	011 0111
L	100 1100	8	011 1000
M	100 1101	9	011 1001
N	100 1110	blank	010 0000
O	100 1111	.	010 1110
P	101 0000	,	010 1100
Q	101 0001	+	010 1011

The *keystrokes* on the *keyboard* of a computer are converted directly into ASCII for processing by the computer.

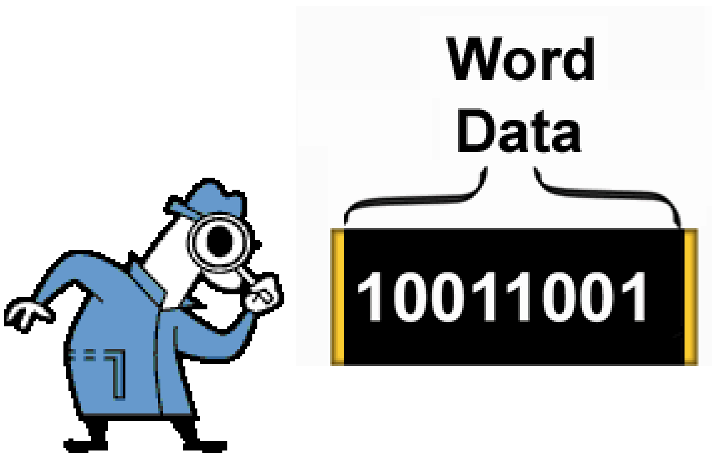
The communication interfacing is done through either an RS-232 or RS-422 protocol.

ASCII input modules convert ASCII code input information from an external device to alphanumeric information that the PLC can process.

3.9

Parity Bit

Some PLC communication systems use a *binary bit* to check the accuracy of data transmission.



When data are transferred between PLCs, one of the binary digits may be accidentally changed from a 1 to a 0.

A parity bit is used to detect errors that may occur while a word is moved.

Parity is a system in which each character transmitted contains *one additional parity bit.*

The parity bit may be a 0 or 1, depending on the number of 1s and 0s in the character itself.

Odd parity means that the total number of binary 1 bits in the character, including the parity bit, is odd.

Character	Even Parity Bit	Odd Parity Bit
0000	0	1
0001	1	0
0010	1	0
0011	0	1
0100	1	0
0101	0	1
0110	0	1
0111	1	0
1000	1	0
1001	0	1

Even parity means that the number of binary 1 bits in the character, including the parity bit, is even.

3.10

Binary Arithmetic

When *adding* with binary numbers, there are only four conditions that can occur:

$$\begin{array}{r} 0 \\ +0 \\ \hline 0 \end{array} \qquad \begin{array}{r} 1 \\ +0 \\ \hline 1 \end{array} \qquad \begin{array}{r} 0 \\ +1 \\ \hline 1 \end{array} \qquad \begin{array}{r} 1 \\ +1 \\ \hline 0 \end{array} \text{ carry } 1$$

Decimal	Equivalent binary
$\begin{array}{r} 5 \\ +2 \\ \hline 7 \end{array}$	$\begin{array}{r} 101 \\ +\ \ 10 \\ \hline 111 \end{array}$

$$
\begin{array}{r}
0 \\
+0 \\ \hline
0
\end{array}
\qquad
\begin{array}{r}
1 \\
+0 \\ \hline
1
\end{array}
\qquad
\begin{array}{r}
0 \\
+1 \\ \hline
1
\end{array}
\qquad
\begin{array}{r}
1 \\
+1 \\ \hline
0
\end{array}
\text{ carry 1}
$$

Decimal

$$
\begin{array}{r}
10 \\
+\;\;3 \\ \hline
13
\end{array}
$$

Equivalent binary

carry

$$
\begin{array}{rr|r}
 & 1 & \\
 & 10 & 10 \\
+ & & 11 \\ \hline
 & 11 & 01
\end{array}
$$

$$
\begin{array}{r}
0 \\
+0 \\
\hline
0
\end{array}
\qquad
\begin{array}{r}
1 \\
+0 \\
\hline
1
\end{array}
\qquad
\begin{array}{r}
0 \\
+1 \\
\hline
1
\end{array}
\qquad
\begin{array}{r}
1 \\
+1 \\
\hline
0
\end{array}
\ \text{carry } 1
$$

Decimal	Equivalent binary
	carry carry
	1 1
26	1 1010
+12	+ 1100
38	1 0 0110

To *subtract* from larger binary numbers, subtract column by column, borrowing from the adjacent column when necessary. When borrowing from the adjacent column, there are now two digits, i.e., 0 borrow 1 gives 10.

EXAMPLE

Subtract 1001 from 1101.

$$\begin{array}{r} 1101 \\ -1001 \\ \hline 0100 \end{array}$$

Subtract 0111 from 1011.

$$\begin{array}{r} 1011 \\ -0111 \\ \hline 0100 \end{array}$$

Binary numbers can also be *negative*. The procedure for this calculation is identical to that of decimal numbers because the smaller value is subtracted from the larger value and a negative sign is placed in front of the result.

EXAMPLE

Subtract 111 from 100.

$$\begin{array}{r} 111 \\ -100 \\ \hline -011 \end{array}$$

Subtract 11011 from 10111.

$$\begin{array}{r} 11011 \\ -10111 \\ \hline -00100 \end{array}$$

There are other methods available for doing subtraction: *1's complement* and *2's complement*.

Using the 1's complement

Decimal **Binary**

$$\begin{array}{r} 10 \\ -\ \ 6 \\ \hline 4 \end{array} \qquad \begin{array}{r} 1010 \\ -0110 \\ \hline 100 \end{array} \xrightarrow{\text{1's complement}} \begin{array}{r} 1010 \\ +1001 \\ \hline 10011 \\ \text{End-around carry} \rightarrow +1 \\ \hline 100 \end{array}$$

When there is a carry at the end of the result, the result is positive. When there is no carry, then the result is negative and a minus sign has to be placed in front of it.

When there is a carry at the end of the result, the result is positive. When there is no carry, then the result is negative and a minus sign has to be placed in front of it.

EXAMPLE

Subtract 11011 from 01101.

$$\begin{array}{rl} 01101 & \\ \underline{+\ \cancel{0}\,00100} & \text{The 1's complement} \\ 10001 & \text{There is no carry,} \\ & \text{so we take the 1's} \\ & \text{complement and add} \\ & \text{the minus sign:} \\ & \underline{-01110} \end{array}$$

For subtraction using the *2's complement*, the 2's complement is added instead of subtracting the numbers. In the result, if the carry is a 1, then the result is positive; if the carry is a 0, then the result is negative and requires a minus sign.

EXAMPLE

Subtract 101 from 111.

$$
\begin{array}{r l}
111 & \\
+\ \cancel{0}011 & \text{The 2's complement} \\
\hline
1010 & \text{The first 1 indicates that the result is positive, so it is disregarded:} \\
 & \underline{010}
\end{array}
$$

EXAMPLE

Subtract 11011 from 01101.

$$
\begin{array}{rl}
01101 & \\
+\ \not{0}\ 00101 & \text{The 2's complement} \\
\hline
10010 & \text{There is no carry, so}
\end{array}
$$

the result is negative; therefore a 1 has to be subtracted and the 1's complement taken to give the result:

subtract 1 $\quad 10010 - 1 = 10001$

1's complement $\underline{-01110}$

When *multiplying* binary numbers, there are only four conditions that can occur:

$$0 \times 0 = 0$$
$$0 \times 1 = 0$$
$$1 \times 0 = 0$$
$$1 \times 1 = 1$$

Decimal	Equivalent binary
$\begin{array}{r} 5 \\ \times 6 \\ \hline 30 \end{array}$	$\begin{array}{r} 101 \\ \times 110 \\ \hline 000 \\ 101 \\ 101 \\ \hline 11110 \end{array}$

The process for *dividing* one binary number by another is the same for both binary and decimal numbers.

Decimal	Equivalent binary
$\begin{array}{r} 7 \\ 2\overline{)14} \end{array}$	$\begin{array}{r} 111 \\ 10\overline{)1110} \\ \underline{10} \\ 11 \\ \underline{10} \\ 10 \\ \underline{10} \\ 00 \end{array}$

PLC data *comparison* instructions are used to compare the data stored in two words.

$$A = B \text{ (}A\text{ equals }B\text{)}$$

$$A > B \text{ (}A\text{ is greater than }B\text{)}$$

$$A < B \text{ (}A\text{ is less than }B\text{)}$$

At times, devices may need to be controlled when they are less than, equal to, or greater than other data values or set points.