Summary

Binary Addition

The rules for binary addition are

$0 + 0 = 0$	Sum = 0, carry = 0
$0 + 1 = 1$	Sum = 1, carry = 0
$1 + 0 = 1$	Sum = 1, carry = 0
$1 + 1 = 10$	Sum = 0, carry = 1

When an input carry = 1 due to a previous result,
example: 10101 + 10111 = 101100

$1 + 1 = 10$	Sum = **0**, carry = 1
$1 + 1 + 0 = 10$	Sum = **0**, carry = 1
$1 + 1 + 1 = 11$	Sum = **1**, carry = 1
$1 + 0 + 0 = 01$	Sum = **1**, carry = 0
$0 + 1 + 1 = 10$	Sum = **0**, carry = **1**

Binary Addition

Example Add the binary numbers 00111 and 10101 and show the equivalent decimal addition.

Solution

Summary

Binary Addition

Example Add the binary numbers 00111 and 10101 and show the equivalent decimal addition.

Solution

$$
\begin{array}{rcr}
{\scriptstyle 0\,1\,1\,1} & & \\
00111 & & 7 \\
10101 & & 21 \\
\hline
11100 & = & 28
\end{array}
$$

Binary Subtraction

The rules for binary subtraction are

$$0 - 0 = 0$$
$$1 - 1 = 0$$
$$1 - 0 = 1$$
$$10 - 1 = 1 \quad \text{with a borrow of 1}$$

Example Subtract the binary number 00111 from 10101 and show the equivalent decimal subtraction.

Solution

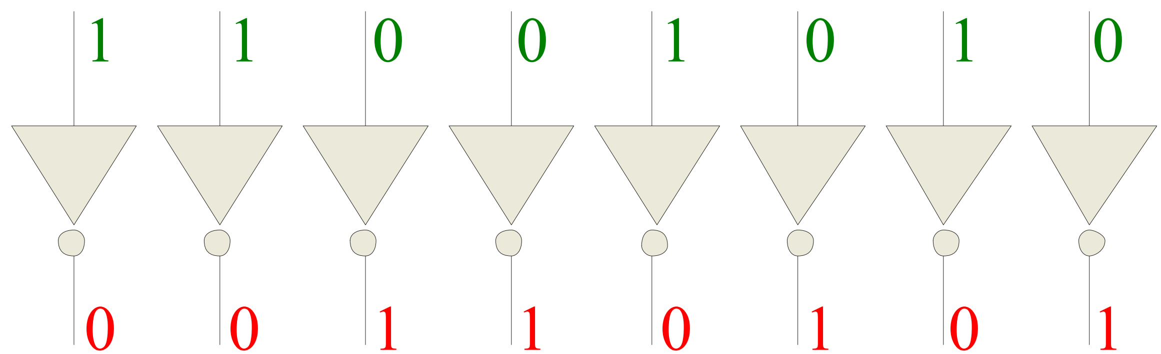
Summary

1's Complement

The 1's complement of a binary number is just the inverse of the digits. To form the 1's complement, change all 0's to 1's and all 1's to 0's.

For example, the 1's complement of 11001010 is
00110101

In digital circuits, the 1's complement is formed by using inverters:



Summary

2's Complement

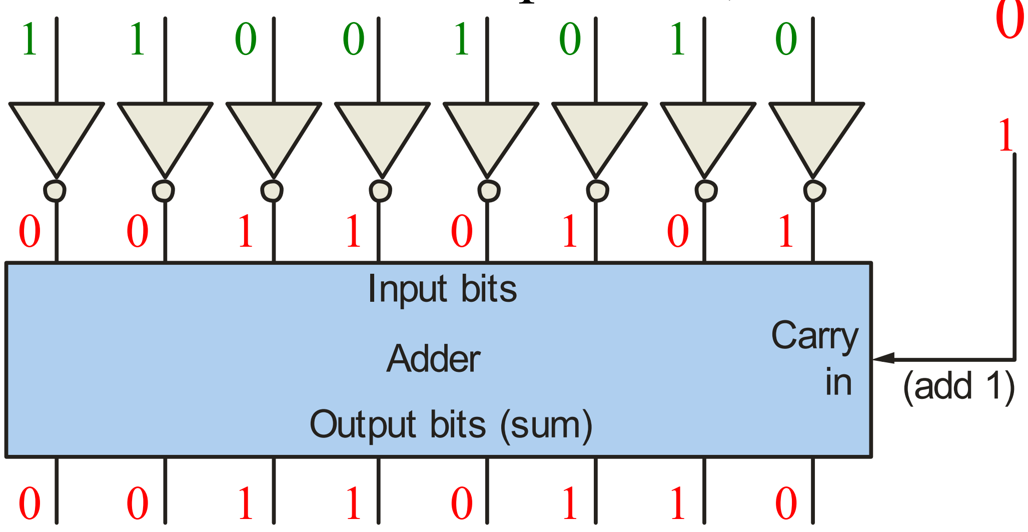
The 2's complement of a binary number is found by adding 1 to the LSB of the 1's complement.

Recall that the 1's complement of 11001010 is

00110101 (1's complement)

To form the 2's complement, add 1: +1

00110110 (2's complement)



Summary

Signed Binary Numbers

There are several ways to represent signed binary numbers. In all cases, the MSB in a signed number is the sign bit, that tells you if the number is positive or negative.

Computers use a modified 2's complement for signed numbers. Positive numbers are stored in *true* form (with a 0 for the sign bit) and negative numbers are stored in *complement* form (with a 1 for the sign bit).

For example, the positive number 58 is written using 8-bits as 00111010 (true form).

Sign bit

Magnitude bits

Summary

Signed Binary Numbers

Negative numbers are written as the 2's complement of the corresponding positive number.

The negative number –58 is written as:

–58 = 11000110 (complement form)

Sign bit   Magnitude bits

An easy way to read a signed number that uses this notation is to assign the sign bit a column weight of –128 (for an 8-bit number). Then add the column weights for the 1's.

Example Assuming that the sign bit = –128, show that 11000110 = –58 as a 2's complement signed number:

Solution Column weights: –128 64 32 16 8 4 2 1.

1	1	0	0	0	1	1	0	
–128	+64				+4	+2		= –58

Summary

Arithmetic Operations with Signed Numbers

Using the signed number notation with negative numbers in 2's complement form simplifies addition and subtraction of signed numbers.

Rules for **addition**: Add the two signed numbers. Discard any final carries. The result is in signed form.

Examples:

$$\begin{array}{r} 00011110 = +30 \\ 00001111 = +15 \\ \hline 00101101 = +45 \end{array} \qquad \begin{array}{r} 00001110 = +14 \\ 11101111 = -17 \\ \hline 11111101 = -3 \end{array} \qquad \begin{array}{r} 11111111 = -1 \\ 11111000 = -8 \\ \hline \cancel{1}\,11110111 = -9 \end{array}$$

Discard carry

Summary

Arithmetic Operations with Signed Numbers

Note that if the number of bits required for the answer is exceeded, overflow will occur. This occurs only if both numbers have the same sign. The overflow will be indicated by an incorrect sign bit.

Two examples are:

$$\begin{array}{r l}
01000000 & = +128 \\
01000001 & = +129 \\
\hline
10000001 & = \cancel{-126}
\end{array}
\qquad
\begin{array}{r l}
10000001 & = -127 \\
10000001 & = -127 \\
\hline
\text{Discard carry} \rightarrow \cancel{1}00000010 & = \cancel{+2}
\end{array}$$

Wrong! The answer is incorrect and the sign bit has changed.

Summary

Arithmetic Operations with Signed Numbers

Rules for **subtraction**: 2's complement the subtrahend and add the numbers. Discard any final carries. The result is in signed form.

Repeat the examples done previously, but subtract:

$$\begin{array}{rl} 00011110 & (+30) \\ -\ 00001111 & -(+15) \\ \hline \end{array} \qquad \begin{array}{rl} 00001110 & (+14) \\ -\ 11101111 & -(-17) \\ \hline \end{array} \qquad \begin{array}{rl} 11111111 & (-1) \\ -\ 11111000 & -(-8) \\ \hline \end{array}$$

2's complement subtrahend and add:

$$\begin{array}{rl} 00011110 & = +30 \\ 11110001 & = -15 \\ \hline \cancel{1}00001111 & = +15 \end{array} \qquad \begin{array}{rl} 00001110 & = +14 \\ 00010001 & = +17 \\ \hline 00011111 & = +31 \end{array} \qquad \begin{array}{rl} 11111111 & = -1 \\ 00001000 & = +8 \\ \hline \cancel{1}00000111 & = +7 \end{array}$$

Discard carry

Discard carry

Summary

Hexadecimal Numbers

Hexadecimal uses sixteen characters to represent numbers: the numbers 0 through 9 and the alphabetic characters A through F.

Large binary number can easily be converted to hexadecimal by grouping bits 4 at a time and writing the equivalent hexadecimal character.

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

Example Express $1001\ 0110\ 0000\ 1110_2$ in hexadecimal:

Solution Group the binary number by 4-bits starting from the right. Thus, 960E

Summary

Hexadecimal Numbers

Hexadecimal is a weighted number system. The column weights are powers of 16, which increase from right to left.

Column weights $\begin{cases} 16^3 & 16^2 & 16^1 & 16^0. \\ 4096 & 256 & 16 & 1\ . \end{cases}$

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

Example Express $1A2F_{16}$ in decimal.

Solution Start by writing the column weights:

4096 256 16 1

1 A 2 F_{16}

$1(4096) + 10(256) + 2(16) + 15(1) = 6703_{10}$

Summary

Octal Numbers

Octal uses eight characters the numbers 0 through 7 to represent numbers. There is no 8 or 9 character in octal.

Binary number can easily be converted to octal by grouping bits 3 at a time and writing the equivalent octal character for each group.

Decimal	Octal	Binary
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	10	1000
9	11	1001
10	12	1010
11	13	1011
12	14	1100
13	15	1101
14	16	1110
15	17	1111

Example Express $1\,001\,011\,000\,001\,110_2$ in octal:

Solution Group the binary number by 3-bits starting from the right. Thus, 113016_8

Summary

Octal Numbers

Octal is also a weighted number system. The column weights are powers of 8, which increase from right to left.

Column weights $\begin{cases} 8^3 & 8^2 & 8^1 & 8^0 \\ 512 & 64 & 8 & 1 \end{cases}$.

Example Express 3702_8 in decimal.

Solution Start by writing the column weights:

512 64 8 1
3 7 0 2_8

$3(512) + 7(64) + 0(8) + 2(1) = 1986_{10}$

Decimal	Octal	Binary
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	10	1000
9	11	1001
10	12	1010
11	13	1011
12	14	1100
13	15	1101
14	16	1110
15	17	1111