Number Systems

- Decimal numbers

1000's column
100's column
10's column
1's column

$5374_{10} =$

- Binary numbers

8's column
4's column
2's column
1's column

$1101_2 =$

Number Systems

- Decimal numbers

1000's column
100's column
10's column
1's column

$$5374_{10} = 5 \times 10^3 + 3 \times 10^2 + 7 \times 10^1 + 4 \times 10^0$$

five thousands, three hundreds, seven tens, four ones

- Binary numbers

8's column
4's column
2's column
1's column

$$1101_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 13_{10}$$

one eight, one four, no two, one one

ELSEVIER

Unsigned Numbers Representation

- An n-bit binary number $A = \overline{a_{n-1}a_{n-2} \ldots a_2a_1a_0}$ has a value of:

$$\sum_{i=0}^{n-1} a_i \times 2^i$$

General Number System

- Decimal, binary and hexadecimal numbers are "**fixed-radix positional number systems**":

  position ***i*** has a value of $\boldsymbol{r^i}$ (r= 10, 2, 16)

- Non positional system:

  Roman or Abjad numerals: I, II, III, IV, V, VI

- Non-radix positional number systems:

  time in DDHHMMSS format

Powers of Two

- $2^0 =$
- $2^1 =$
- $2^2 =$
- $2^3 =$
- $2^4 =$
- $2^5 =$
- $2^6 =$
- $2^7 =$
- $2^8 =$
- $2^9 =$
- $2^{10} =$
- $2^{11} =$
- $2^{12} =$
- $2^{13} =$
- $2^{14} =$
- $2^{15} =$

Powers of Two

- $2^0 = 1$
- $2^1 = 2$
- $2^2 = 4$
- $2^3 = 8$
- $2^4 = 16$
- $2^5 = 32$
- $2^6 = 64$
- $2^7 = 128$
- $2^8 = 256$
- $2^9 = 512$
- $2^{10} = 1024$
- $2^{11} = 2048$
- $2^{12} = 4096$
- $2^{13} = 8192$
- $2^{14} = 16384$
- $2^{15} = 32768$
- Handy to memorize up to 2^9

Number Conversion

- Decimal to binary conversion:
  - Convert 10011_2 to decimal

- Decimal to binary conversion:
  - Convert 47_{10} to binary

Number Conversion

- Decimal to binary conversion:
  - Convert 10011_2 to decimal
  - $16\times1 + 8\times0 + 4\times0 + 2\times1 + 1\times1 = 19_{10}$

- Decimal to binary conversion:
  - Convert 47_{10} to binary
  - $32\times1 + 16\times0 + 8\times1 + 4\times1 + 2\times1 + 1\times1 = 101111_2$

Binary Values and Range

- *N*-digit decimal number
  - How many values?
  - Range?
  - Example: 3-digit decimal number:

- *N*-bit binary number
  - How many values?
  - Range:
  - Example: 3-digit binary number:

Binary Values and Range

- *N*-digit decimal number
  - How many values? **10^N**
  - Range? **$[0, 10^N - 1]$**
  - Example: 3-digit decimal number:
    - **10^3 = 1000 possible values**
    - **Range: [0, 999]**

- *N*-bit binary number
  - How many values? **2^N**
  - Range: **$[0, 2^N - 1]$**
  - Example: 3-digit binary number:
    - **2^3 = 8 possible values**
    - **Range: [0, 7] = [000_2 to 111_2]**

Hexadecimal Numbers

Hex Digit	Decimal Equivalent	Binary Equivalent
0	0	
1	1	
2	2	
3	3	
4	4	
5	5	
6	6	
7	7	
8	8	
9	9	
A	10	
B	11	
C	12	
D	13	
E	14	
F	15	

Hexadecimal Numbers

Hex Digit	Decimal Equivalent	Binary Equivalent
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
A	10	1010
B	11	1011
C	12	1100
D	13	1101
E	14	1110
F	15	1111

Hexadecimal Numbers

- Base 16
- Shorthand for binary

Hexadecimal to Binary Conversion

- Hexadecimal to binary conversion:
  - Convert $4AF_{16}$ (also written 0x4AF) to binary

- Hexadecimal to decimal conversion:
  - Convert 0x4AF to decimal

Hexadecimal to Binary Conversion

- Hexadecimal to binary conversion:
  - Convert $4AF_{16}$ (also written 0x4AF) to binary
  - $0100\ 1010\ 1111_2$
- Hexadecimal to decimal conversion:
  - Convert $4AF_{16}$ to decimal
  - $16^2 \times 4 + 16^1 \times 10 + 16^0 \times 15 = 1199_{10}$

Bits, Bytes, Nibbles…

- Bits

10010110

most significant bit

least significant bit

- Bytes & Nibbles

byte

10010110

nibble

- Bytes

CEBF9AD7

most significant byte

least significant byte

Large Powers of Two

- 2^{10} = 1 kilo ≈ 1000 (1024)
- 2^{20} = 1 mega ≈ 1 million (1,048,576)
- 2^{30} = 1 giga ≈ 1 billion (1,073,741,824)

Decimal Prefix	Value	Binary Prefix	Value
K: Kilo	10^3	Ki: Kibi	2^{10}
M: Mega	10^6	Mi: Mebi	2^{20}
G: Giga	10^9	Gi: Gibi	2^{30}

Estimating Powers of Two

- What is the value of 2^{24}?

- How many values can a 32-bit variable represent?

Estimating Powers of Two

- What is the value of 2^{24}?

  - **$2^4 \times 2^{20} \approx$ 16 million**

- How many values can a 32-bit variable represent?

  -$2^2 \times 2^{30} \approx$ 4 billion

Addition

- Decimal

$$\begin{array}{r} \color{blue}{11} \;\leftarrow \text{carries} \\ 3734 \\ +\;\; 5168 \\ \hline 8902 \end{array}$$

- Binary

$$\begin{array}{r} \color{blue}{11} \;\leftarrow \text{carries} \\ 1011 \\ +\;\; 0011 \\ \hline 1110 \end{array}$$

Binary Addition Examples

- Add the following 4-bit binary numbers

$$\begin{array}{r} 1001 \\ +\ 0101 \\ \hline \end{array}$$

- Add the following 4-bit binary numbers

$$\begin{array}{r} 1011 \\ +\ 0110 \\ \hline \end{array}$$

Binary Addition Examples

- Add the following 4-bit binary numbers

$$\begin{array}{r} {\color{blue}1} \\ 1001 \\ +\ \ 0101 \\ \hline {\color{blue}1110} \end{array}$$

- Add the following 4-bit binary numbers

$$\begin{array}{r} {\color{blue}111} \\ 1011 \\ +\ \ 0110 \\ \hline {\color{blue}10001} \end{array}$$

Overflow!

Overflow

- Digital systems operate on a **fixed number of bits**
- **Overflow**: when result is too big to fit in the available number of bits
- See previous example of 11 + 6
- In unsigned numbers, C_{out} of the adder indicates an overflow in addition
- Signed numbers have a different overflow indicator (comes later)

Signed Binary Numbers

- Sign/Magnitude Numbers
- Two's Complement Numbers

- Not covered here:

  Biased Numbers

  One's Complement Numbers

Sign/Magnitude Numbers

- 1 sign bit, *N*-1 magnitude bits
- Sign bit is the most significant (left-most) bit
  - Positive number: sign bit = 0
  - Negative number: sign bit = 1

$$A:\{a_{N-1},a_{N-2},\cdots a_2,a_1,a_0\}$$

$$A=(-1)^{a_{n-1}}\sum_{i=0}^{n-2}a_i 2^i$$

- Example, 4-bit sign/mag representations of ± 6:

  +6 =

  - 6 =

- Range of an *N*-bit sign/magnitude number:

Sign/Magnitude Numbers

- 1 sign bit, *N*-1 magnitude bits
- Sign bit is the most significant (left-most) bit
  – Positive number: sign bit = 0
  – Negative number: sign bit = 1

$$A:\{a_{N-1},a_{N-2},\cdots a_2,a_1,a_0\}$$

$$A=(-1)^{a_{n-1}}\sum_{i=0}^{n-2}a_i 2^i$$

- Example, 4-bit sign/mag representations of ± 6:

  +6 = **0110**

  - 6 = **1110**

- Range of an *N*-bit sign/magnitude number:

  $[-(2^{N-1}-1), 2^{N-1}-1]$

ELSEVIER

Sign/Magnitude Numbers

- Problems:
  - Binary addition doesn't work, for example -6 + 6:

```
   1110
 + 0110
 ------
  10100 (wrong!)
```

  - Two representations of 0 (± 0):

```
  1000
  0000
```

Two's Complement Numbers

- Don't have same problems as sign/magnitude numbers:
  - Binary addition works
  - Single representation for 0

Two's Complement Numbers

- MSB has value of -2^{N-1}

$$A = a_{n-1}\left(-2^{n-1}\right) + \sum_{i=0}^{n-2} a_i 2^i$$

- Most positive 4-bit number:
- Most negative 4-bit number:
- The most significant bit still indicates the sign (1 = negative, 0 = positive)
- Range of an N-bit 2's complement number :

Two's Complement Numbers

- MSB has value of -2^{N-1}

$$A = a_{n-1}\left(-2^{n-1}\right) + \sum_{i=0}^{n-2} a_i 2^i$$

- Most positive 4-bit number: **0111**
- Most negative 4-bit number: **1000**
- The most significant bit still indicates the sign (1 = negative, 0 = positive)
- Range of an *N*-bit 2's complement number:

$[-(2^{N-1}), 2^{N-1}-1]$

Two's Complement Numbers

- A different formula, very similar to unsigned representation, useful in some occasions

$$A = \sum_{i=0}^{n-1} a_i \times 2^i$$

$$a_{o\ldots n-2} \in \{0, 1\}$$

$$a_{n-1} \in \{0, -1\}$$

Taking the Two's Complement

- Flip the sign of a two's complement number
- Method:
    1. Invert the bits
    2. Add 1
- Example: Flip the sign of $6_{10} = 0110_2$

Taking the Two's Complement

- Flip the sign of a two's complement number
- Method:
    1. Invert the bits
    2. Add 1
- Example: Flip the sign of $6_{10} = 0110_2$
    1. **1001**
    2. **+ 1**

    $1010 = -6_{10}$

Taking the Two's Complement

- Flip the sign of a two's complement number
- Method:
    1. Starting from right, don't touch 0's to first 1
    2. Don't touch first 1 (from right)
    3. Invert the rest

    Proof?

- Example: Flip the sign of $6_{10} = 0110_2$

1010

Increasing Bit Width

- **Extend number from *N* to *M* bits (*M* > *N*) :**
  - Zero Extension

    Used for unsigned numbers

  - Sign Extension

    Used for signed numbers

Zero-Extension

- Zeros copied to MSB's
- Number value is same for unsigned numbers
- *Warning: Invalid operation on signed numbers*

- **Example 1:**
  - 4-bit value = $0011_2 = 3_{10}$
  - 8-bit zero-extended value: $00000011 = 3_{10}$
- **Example 2:**
  - 4-bit value = $1011 = 11_{10}$
  - 8-bit zero-extended value: $00001011 = 11_{10}$

Sign-Extension

- Sign (=MSB) bit copied to new MSB's
- Number value is same for signed numbers
- *Warning: Invalid operation on unsigned integer*

- **Example 1:**
  - 4-bit representation of 3 = 0011
  - 8-bit sign-extended value: 00000011
- **Example 2:**
  - 4-bit representation of -5 = 1011
  - 8-bit sign-extended value: 11111011

Number System Comparison

Number System	**Range**
Unsigned	$[0, 2^N-1]$
Sign/Magnitude	$[-(2^{N-1}-1), 2^{N-1}-1]$
Two's Complement	$[-2^{N-1}, 2^{N-1}-1]$

For example, 4-bit representation:

	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Unsigned									0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
Two's Complement	1000	1001	1010	1011	1100	1101	1110	1111	0000	0001	0010	0011	0100	0101	0110	0111								
Sign/Magnitude		1111	1110	1101	1100	1011	1010	1001	0000 1000	0001	0010	0011	0100	0101	0110	0111								

Two's Complement Addition

- Add 6 + (-6) using two's complement numbers in a binary adder

$$\begin{array}{r} 0110 \\ +\ 1010 \\ \hline \end{array}$$

- Add -2 + 3 using two's complement numbers

$$\begin{array}{r} 1110 \\ +\ 0011 \\ \hline \end{array}$$

Two's Complement Addition

- Add 6 + (-6) using two's complement numbers

```
   111
    0110
+   1010
--------
   10000
```

- Add -2 + 3 using two's complement numbers

```
   111
    1110
+   0011
--------
   10001
```

Two's Complement Addition

- C_{out} of binary adder does not indicate anything in a two's complement addition and/or subtraction
- Still we have to be aware of overflow
  - 6 + 6: 0110 + 0110 = 1100 = -4

Wrong: 12 can not be represented in a 4-bit signed number

  - -6 + -6 : 1010 + 1010 = 0100 = 8

Wrong: -12 can not be shown in a 4-bit signed number

ELSEVIER

Overflow in 2's Complement Addition

- In a two's complement addition, overflow occurs when:

  Sum of two positive numbers is negative

  0110 + 0110 = 1100

  Sum of two negative numbers is positive

  1010 + 1010 = 0100