Number systems

- Decimal
- Octal
- Hexadecimal
- Binary
  - Negative numbers
  - Two's complement
  - Overflow
  - Sign extension
- References:
  - Section 2.1 to 2.3 in Bryant and O'Hallaron
  - Appendix A in Tanenbaum
  - Related links on Web site

Decimal

- Base 10: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
- $103_{10} = 1 \times 100 + 0 \times 10 + 3 \times 1$
  $\quad = 1 \times 10^2 + 0 \times 10^1 + 3 \times 10^0$
- $5.702_{10} = 5 \times 10^0 + 7 \times 10^{-1} + 0 \times 10^{-2} + 2 \times 10^{-3}$
- $\ldots\ 10^3\ \ 10^2\ \ 10^1\ \ 10^0\ .\ 10^{-1}\ \ 10^{-2}\ \ 10^{-3}\ \ldots$

Octal

- Base 8: 0, 1, 2, 3, 4, 5, 6, 7
- $\ldots\ 8^3\ \ 8^2\ \ 8^1\ \ 8^0\ .\ 8^{-1}\ \ 8^{-2}\ \ 8^{-3}\ \ldots$
- Octal $\rightarrow$ Decimal:

$$\begin{aligned} 103_8 &= 1 \times 8^2 + 0 \times 8^1 + 3 \times 8^0 \\ &= 1 \times 64 + 0 \times 8 + 3 \times 1 \\ &= 67_{10} \end{aligned}$$

- $42_8 = x_{10}$?

Decimal $\rightarrow$ Octal

- $132_{10} =\ ?_8$

$132/8 = 16$ reminder 4

$16/8 = 2$ reminder 0

$2/8 = 0$ reminder 2

$\rightarrow 132_{10} = 204_8$

- $42_{10} = x_8$?

Hexadecimal

- Base 16: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
- $\ldots\ 16^3\ \ 16^2\ \ 16^1\ \ 16^0\ .\ 16^{-1}\ \ 16^{-2}\ \ 16^{-3}\ \ldots$
- Hexadecimal → Decimal:

$$\begin{aligned} 10\mathsf{E}_{16} &= 1 \times 16^2 + 0 \times 16^1 + \mathsf{E} \times 16^0 \\ &= 1 \times 256 + 0 \times 16 + 14 \times 1 \\ &= 270_{10} \end{aligned}$$

- $42_{16} = x_{10}$?

Decimal → Hexadecimal

- $174_{10} = \ ?_{16}$

$174/16 = 10$ reminder 14 (E)

$10/16 = 0$ reminder 10 (A)

$\rightarrow 174_{10} = \mathsf{AE}_{16}$

- $42_{10} = x_{16}$?

Binary

- Base 2: 0, 1 (true/false) (on/off)
- $\ldots\ 2^3\ \ 2^2\ \ 2^1\ \ 2^0\ \ .\ \ 2^{-1}\ \ 2^{-2}\ \ 2^{-3}\ \ldots$
  $\ldots\ 8\ \ 4\ \ 2\ \ 1\ \ .\ \ 1/2\ \ 1/4\ \ 1/8\ \ldots$
- Binary → Decimal:

$$\begin{aligned} 1001_2 &= 1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 \\ &= 1 \times 8 + 0 \times 4 + 0 \times 2 + 1 \times 1 \\ &= 9_{10} \end{aligned}$$

- $1101_2 = x_{10}$?

Decimal → Binary

- $19_{10} = \ ?_2$

$19/2 = 9$ reminder 1

$9/2 = 4$ reminder 1

$4/2 = 2$ reminder 0

$2/2 = 1$ reminder 0

$1/2 = 0$ reminder 1

$\rightarrow 19_{10} = 10011_2$

- $42_{10} = x_2$?

Binary addition and subtraction

- Addition:

$$\begin{array}{ll} 11010_2 + & 26_{10} + \\ \underline{01011_2} = & \underline{11_{10}} = \\ 100101_2 & 37_{10} \end{array}$$

- Subtraction:

$$\begin{array}{ll} 10101_2 - & 21_{10} - \\ \underline{01011_2} = & \underline{11_{10}} = \\ 01010_2 & 10_{10} \end{array}$$

Negative numbers

- Positive numbers (one byte):

$$\begin{array}{ll} 0000\ 0000 & 0 \\ 0000\ 0001 & 1 \\ 0000\ 0010 & 2 \\ \ldots & \ldots \\ 1111\ 1111_2 & 255_{10} \end{array}$$

- How can we represent negative numbers? E.g. -42_{10} ?

  $42_{10} = 0001\ 0101_2$, but the negative?

Negative numbers (cont.)

- Reserve one bit for sign (highest bit, most significant bit)
- Positive numbers:

$$\begin{array}{ll} 0\ 000\ 0000 & 0 \\ 0\ 000\ 0001 & 1 \\ \ldots & \ldots \\ 0\ 111\ 1111_2 & 127_{10} \end{array}$$

- Negative numbers:

$$\begin{array}{ll} 1\ 000\ 0000 & -0 \\ 1\ 000\ 0001 & -1 \\ \ldots & \ldots \\ 1\ 111\ 1111_2 & -127_{10} \end{array}$$

- Problems: Two zeros (!), addition is not simple: $4 + (-1) = -5$

Two's complement

- Most common system to represent negative integers in binary
  - Only one zero
  - Works with normal binary addition
  - Left-most bit indicates sign
- The rule: *To negate a number, flip the bits to the left of the right-most 1*
- Alternative: *flip all the bits and add 1 to the result*
- Examples:

0000 0101	5	1111 1000	-8
1111 1011	-5	0000 1000	8

Two's complement examples

$$\begin{array}{rr} 0100 & 4\ + \\ \underline{1111} & -1\ = \\ 10011 & 3 \end{array} \qquad \begin{array}{rr} 1101 & -3\ + \\ \underline{1110} & -2\ = \\ 11011 & -5 \end{array}$$

$$\begin{array}{rl} 0111 & 7\ + \\ \underline{0010} & 2\ = \\ 1001 & -7 \text{ or } 9\ ? \end{array} \qquad \begin{array}{rl} 1010 & -6\ + \\ \underline{1010} & -6\ = \\ 10100 & 4 \text{ or } -12\ ? \end{array}$$

Overflow

- Overflow occurs when the result cannot be represented in the given number of bits, because the magnitude is too great
- Rules:

$$\begin{array}{llll} +x & + & -y & \rightarrow \text{ No overflow} \\ -x & + & +y & \rightarrow \text{ No overflow} \\ +x & + & +y & \rightarrow \text{ Overflow if result is } -z \\ -x & + & -y & \rightarrow \text{ Overflow if result is } +z \end{array}$$

- Example: $\begin{array}{rr} 0\ 0001\ + & 1\ + \\ \underline{0\ 1111}\ = & 15\ = \\ 1\ 0000 & 0 \end{array}$

Sign extension

- Example: We want to convert a 10-bit number into a 16-bit number (needed later for PeANUt operands/arguments)
- Easy for positive case:

$$\begin{array}{rr} 0110 & 6 \\ 0000\ 0110 & 6 \end{array}$$

- Negative case?

$$\begin{array}{rr} 1010 & -6 \\ 0000\ 1010 & 10 \end{array}$$

Sign extension (cont.)

- Solution: Fill the extra digits with copies of the sign bit (the most left bit)
- Positive case:

$$\begin{array}{rr} 0111 & 7 \\ 0000\ 0111 & 7 \end{array}$$

- Negative case:

$$\begin{array}{rr} 1001 & -7 \\ 1111\ 1001 & -7 \end{array}$$

Gates

- Discussion of basic gates
- Show slides from Tanenbaum
- Introduce concepts of full and half adder