INEL4206
Microprocessors

Lectures 1 & 2
Overview of
Number Systems, Boolean Algebra

Representation of Data

- Integers are written using a positional numbering system, where each digit represents the coefficient in a power series:

$$N = a_{n-1}b^{n-1} + a_{n-2}b^{n-2} + \ldots + a_1 b^1 + a_0 b^0$$

where n is the number of digits, b is the base, and a_i are the coefficients where each is an integer in the range $0 < a_i < b$.

From Dr. Itthisek Nilkhamhang digital circuits lecture slides 2011.

Numbering Systems

- Number systems
  - Three main characteristics:
    - Number of independent digits
      - Base or radix (b)
    - Place values of different digits
      - $b^{n-1} \ldots b^2 b^1 b^0$ with 0 known as the least significant digit and n – 1 the most significant digit
      - Fractional part (if present) is represented as $b^{-1}b^{-2}b^{-3}\ldots$
    - Maximum number of values that can be represented given a fixed number of digits (n)
      - b^n

Base 10 system (decimal)

- 10 different digit values
  - 0,1,2,3,4,5,6,7,8,9
- Example:
  - 9138.504

    $9138 = 8x10^0 + 3x10^1 + 1x10^2 + 9x10^3$

    $504 = 5x10^{-1} + 0x10^{-2} + 4x10^{-3}$
- With 10 digits, what are maximum number of values we can represent?
  - 10^{10} ranging from 0 to 10^{10} - 1

Base 2 system (binary)

- 2 different digits known as binary digits or “bits”
  - 0,1
- Example: (first 16 binary numbers, 4 bits)

  0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111,

  1000, 1001, 1010, 1011, 1100, 1101, 1110, 1111
- With 10 digits, what are maximum number of values we can represent?
  - 2^{10} =1024 values ranging from 0000000000 to 1111111111

Binary System Advantages

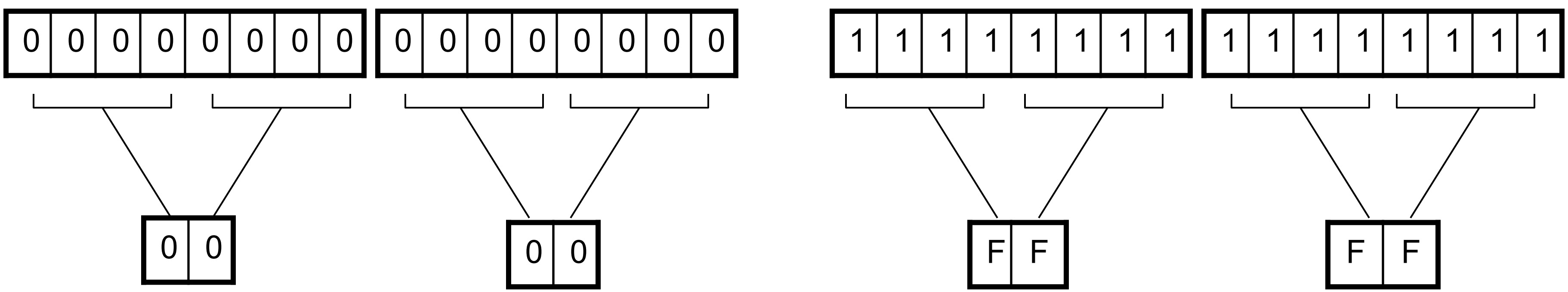
- George Boole (1854) mathematics of logic (Boolean Algebra)
  - truth values (false, true) represented by 0,1
- All data can be represented as a sequence of 1s and 0s
- Claude Shannon (1937) use of boolean algebra and binary arithmetic to create logic gates that simplified electromechanical relays used in the phone system switches
  - Foundation of digital circuit design
- Basic electronic devices can be operated in 2 different modes: (e.g. BJTs cut-off / saturation)
- Circuit implementation of 0s & 1s arithmetic easy to implement

Other popular base systems

- Base 8 (Octal)
  - Similar to decimal, just remove digits 8 & 9
- Base 16 (Hexadecimal or hex)
  - 16 different digits
  - 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
  - Short form to represent large binary numbers
    - Bits are arranged in group of 8 (byte)
    - Working with large binary numbers consume a large amount of digits, (e.g. memory addresses)

Large binary to Hex

- 64K memories have up to 2^{16} = 65,536 different values



- In general 4 binary digits = 1 hex digit
  – 3 binary digits = 1 hex digit

Common Powers

$2^{-3} = 0.125$
$2^{-2} = 0.25$
$2^{-1} = 0.5$
$2^{0} = 1$
$2^{1} = 2$
$2^{2} = 4$
$2^{3} = 8$
$2^{4} = 16$
$2^{5} = 32$
$2^{6} = 64$
$2^{7} = 128$
$2^{8} = 256$
$2^{9} = 512$
$2^{10} = 1024$
$2^{11} = 2048$
$2^{12} = 4096$

$16^{0} = 1 = 2^{0}$
$16^{1} = 16 = 2^{4}$
$16^{2} = 256 = 2^{8}$
$16^{3} = 4096 = 2^{12}$

$2^{10} = 1024 = 1$ K
$2^{20} = 1048576 = 1$ M (1 Mega) $= 1024$ K $= 2^{10} * 2^{10}$
$2^{30} = 1073741824 = 1$ G (1 Giga)

Binary to decimal conversion

- 1001.0101

  $1001 = 1x2^0 + 0x2^1 + 0x2^2 + 1x2^3$

  $= 1 + 0 + 0 + 8 = 9$

  $.0101 = 0x2^{-1} + 1x2^{-2} + 0x2^{-3} + 1x2^{-4}$

  $= 0 + .25 + 0 + .0625 = .3125$

  $1001.0101 = 9.3125$

Decimal to binary conversion

- 13.375

13 :

$$\frac{13}{2} = 6 \ \ R = 1$$

$$\frac{6}{2} = 3 \ \ R = 0$$

$$\frac{3}{2} = 1 \ \ R = 1$$

$$\frac{1}{2} = 0 \ \ R = 1 \ \ (\text{MSB})$$

.375

$$0.375 \times 2 = 0.75 \ \ C0$$

$$0.75 \times 2 = 0.5 \ \ C1$$

$$0.5 \times 2 = 0 \ \ C1$$

Thus $13.375_{10} = 1101.011_2$

Base Conversion

Convert 53 to binary

53/2 = 26, R. = 1 ← Least Significant Digit

26/2 = 13, R. = 0

13/2 = 6 , R. = 1

6 /2 = 3, R. = 0

3/2 = 1, R. = 1

1/2 = 0, R. = 1 ← Most Significant Digit

$$53 = 0b\ 110101$$
$$= 1*2^5 + 1*2^4 + 0*2^3 + 1*2^2 + 0*2^1 + 1*2^0$$
$$= 32 + 16 + 0 + 4 + 0 + 1 = 53$$

Base Conversion (2)

Convert 53 to Hex

$$53/16 = 3, \qquad R = 5$$

$$3/16 = 0, \qquad R = 3$$

$$53 = 0x35$$

$$= 3 * 16^1 + 5 * 16^0$$

$$= 48 + 5 = 53$$

Binary arithmetic

- Addition
  - Basic rules:

$$0 + 0 = 0$$

$$0 + 1 = 1$$

$$1 + 0 = 1$$

$1 + 1 = 0$ carry ‘1’ to the next more significant bit

$1 + 1 + 1 = 1$ carry ‘1’ to the next more significant bit

Binary arithmetic

- Subtraction
  - Basic rules

$$0 - 0 = 0$$

$$1 - 0 = 1$$

$$1 - 1 = 0$$

$$0 - 1 = 1$$ borrow ‘1’ form the next most significant bit

Binary Arithmetic

addition

$$
\begin{array}{r}
0111 \\
1001\ 0101 \\
+\ \underline{0100\ 0011} \\
1101\ 1000
\end{array}
$$

subtraction

$$
\begin{array}{cccccccccc}
 & 0\,2 & & & & & 0\,2 & & \\
 & \cancel{1}\,0 & 0 & 1 & & 0 & \cancel{1}\,0 & 1 \\
- & \underline{0\,1} & \underline{0} & \underline{0} & & \underline{0} & \underline{0\,1} & \underline{1} \\
 & 0\,1 & 0 & 1 & & 0 & 0\,1 & 0
\end{array}
$$

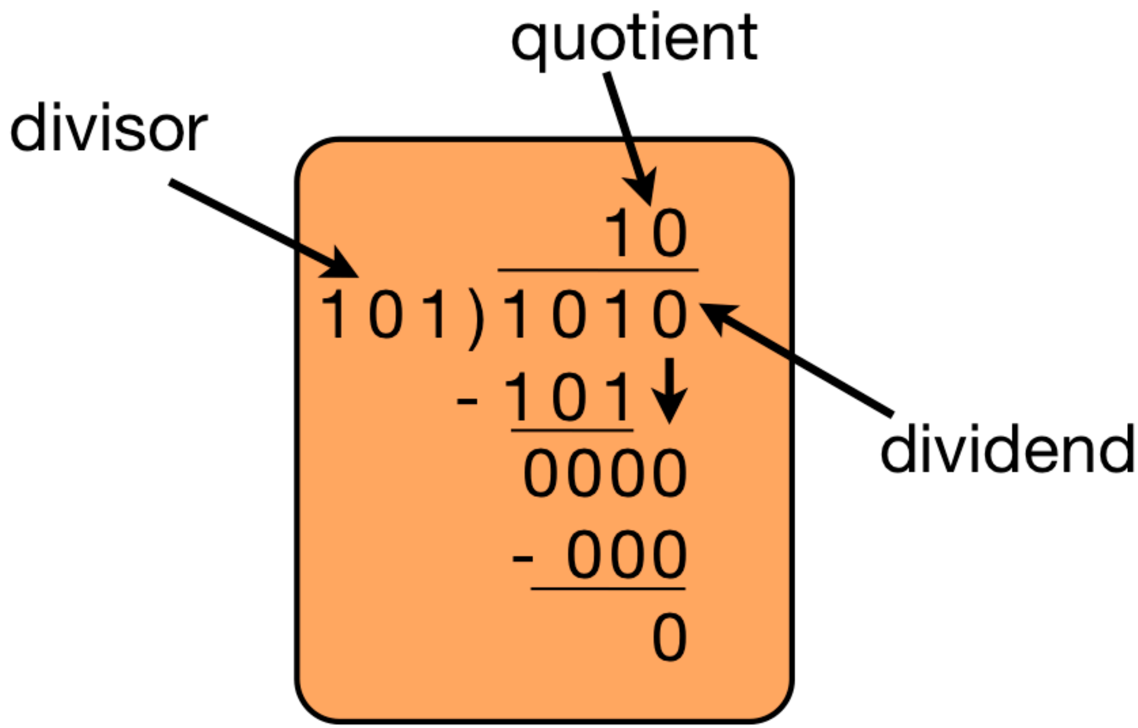
From Dr. Itthisek Nilkhamhang digital circuits lecture slides 2011.

Binary Arithmetic

multiplication

```
     1001 0101
x           10
     0000 0000
+ 1 0010 101
  1 0010 1010
```

division



From Dr. Itthisek Nilkhamhang digital circuits lecture slides 2011.

Complements

Binary	1’s	2’s	10010110	01101001	01101010
Decimal	9’s	10’s	2496	7503	7504
Octal	7’s	8’s	562	215	216
Hex	15’s	16’s	3BF	C40	C41

- Binary 1’s and 2s complements are important because they allow for easy arithmetic logic implementation.
- In 2s complement notation, the positive value is the same binary and the negative is the 2s complement of the positive value

  +9 : 00001001

  –9 : 11110111

- In 8-bit binary MSB provides sign, rest of the bits are the number representation, possible values range: $+(2^{(n-1)} - 1)$ to $-(2^{(n-1)} - 1)$

Addition using 2’s complement

- Final carry obtained while adding MSBs should be disregarded
  - Consider –18 and –37 addition:

2’s –18: 11101110

2’s –37: 11011011

Sum: 11001001 : 2’s –55

Subtraction using 2’s complement

- Similar to addition, add 2’s complement of subtrahend and disregard carry:
  - Consider +24 – +14

2’s +24:	00011000		00011000
2’s +14:	00001110	2’s:	<u>11110010</u>
		Sum:	00001010