

CS 211 - Digital Logic Design 211 عال - تصميم المنطق الرقمي

First Term - 1439/1440
Lecture #2

Dr. Hazem Ibrahim Shehata

Assistant Professor

College of Computing and Information Technology

Administrivia

➤ Course website:

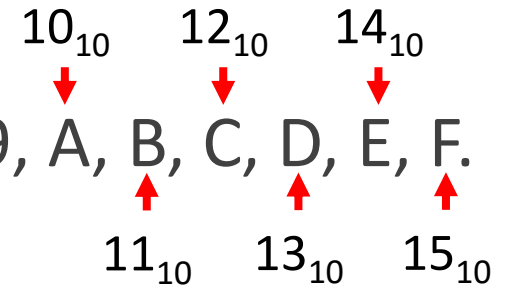
- Up and running but not fully functional yet!!
- URL: <http://hshehata.github.io/courses/su/cs211/>

Hexadecimal Numbers

➤ Radix/base of system is **16**.

◦ **Sixteen** possible values for each digit: 0, 1, 2, ..., 9, A, B, C, D, E, F.

◦ **Example**: The number $3D57.F0E_{16}$



➤ Column weights of hexadecimal num's are **powers of 16**:

... 16^3 16^2 16^1 16^0 . 16^{-1} 16^{-2} 16^{-3} 16^{-4} ...

Conversion: Hexadecimal → Decimal

➤ Method: **Sum of weights**

➤ **Example:** Convert $3FA.4_{16}$ to decimal.

➤ **Solution:**

16^2	16^1	16^0	.	16^{-1}	
256	16	1	.	1/16	
*	*	*		*	
3	F	A	.	4	
768	+240	+10		+0.25	= 1018.25_{10}

Conversion: Decimal → Hexadecimal

➤ Method:

- Repeated division-by-16 (for integer)
- Repeated multiplication-by-16 (for fraction)

➤ Example: Convert 943_{10} to Hexadecimal.

➤ Solution:

	Quotient	Remainder
◦ $943 \div 16 =$	58	15
◦ $58 \div 16 =$	3	10
◦ $3 \div 16 =$	0	3

STOP → 0

3 A F

Conversion: Binary \rightarrow Hexadecimal

- Method: **4-bit grouping** (Break binary num. into 4-bit groups starting from radix point; replace each 4-bit group with the equivalent hexadecimal symbol)
- **Example:** Convert 10011101010.101111_2 to hexadecimal.

- **Solution:**

$$\begin{array}{ccccccc} 0100 & 1110 & 1010 & . & 1011 & 1100 & \\ \downarrow & \downarrow & \downarrow & & \downarrow & \downarrow & \\ 4 & E & A & . & B & C & \end{array} = 4EA.BC_{16}$$

NOTE: A zero added to the right of a fraction or the left of an integer, doesn't change its value!!

Conversion: Hexadecimal → Binary

- Method: **4-bit replacement** (Replace each hexadecimal symbol with the appropriate four bits)
- Example: Convert $2B9.3A_{16}$ to binary.

➤ Solution:

$$\begin{array}{ccccccc} 2 & B & 9 & . & 3 & A & \\ \downarrow & \downarrow & \downarrow & & \downarrow & \downarrow & \\ 0010 & 1011 & 1001 & . & 0011 & 1010 & \\ & & & & & & = 1010111001.0011101_2 \end{array}$$

NOTE: A zero removed from the right of a fraction or the left of an integer doesn't change its value!!

Octal Numbers

- Radix/base of system is **8**.
 - **Eight** possible values for each digit: 0, 1, 2, ..., 7.
 - **Example**: The number 2407.321_8
- Column weights of octal numbers are **powers of 8**:
$$\dots 8^3 \ 8^2 \ 8^1 \ 8^0 . 8^{-1} \ 8^{-2} \ 8^{-3} \ 8^{-4} \dots$$

Conversion: Octal \leftrightarrow Dec./Hex./Bin.

- Methods are similar to hexadecimal; with few changes.
 - Octal \rightarrow decimal: **sum of weights**
 - Decimal \rightarrow octal: **repeated division-by-8** and **repeated multiplication-by-8**
 - Binary \rightarrow octal: **3-bit grouping**
 - Octal \rightarrow binary: **3-bit replacement**

Unsigned Binary Addition

➤ Eight basic **rules for adding** binary digits (bits):

When there is no carry (or a carry of 0)	When there is a carry of 1
$0 + 0 \rightarrow \text{Sum} = 0, \text{carry} = 0$	$\underline{1} + 0 + 0 \rightarrow \text{Sum} = 1, \text{carry} = 0$
$0 + 1 \rightarrow \text{Sum} = 1, \text{carry} = 0$	$\underline{1} + 1 + 0 = 10 \rightarrow \text{Sum} = 0, \text{carry} = 1$
$1 + 0 \rightarrow \text{Sum} = 1, \text{carry} = 0$	$\underline{1} + 0 + 1 = 10 \rightarrow \text{Sum} = 0, \text{carry} = 1$
$1 + 1 \rightarrow \text{Sum} = 0, \text{carry} = 1$	$\underline{1} + 1 + 1 \rightarrow \text{Sum} = 1, \text{carry} = 1$

➤ **Example:**

$$\begin{array}{r}
 \begin{array}{c} \textcolor{red}{1} \\ 0 \end{array} \begin{array}{c} \textcolor{red}{1} \\ 1 \end{array} \begin{array}{c} \textcolor{red}{1} \\ 1 \end{array} \begin{array}{c} \textcolor{green}{0} \\ 1 \end{array} 0 \\
 + \begin{array}{c} 0 \\ 0 \end{array} \begin{array}{c} 0 \\ 0 \end{array} \begin{array}{c} 1 \\ 1 \end{array} \begin{array}{c} 1 \\ 1 \end{array} 1 \\
 \hline
 \begin{array}{c} 1 \\ 0 \end{array} \begin{array}{c} 1 \\ 0 \end{array} \begin{array}{c} 1 \\ 0 \end{array} \begin{array}{c} 0 \\ 1 \end{array} 1
 \end{array}$$

Unsigned Binary Subtraction

➤ Eight basic **rules for subtracting** binary digits (bits):

When there is no borrow (or a borrow of 0)	When there is a borrow of <u>1</u>
$0 - 0 \rightarrow \text{sum} = 0, \text{borrow} = 0$	$0 - 0 - \underline{1} \rightarrow \text{sum} = 1, \text{borrow} = 1$
$0 - 1 \rightarrow \text{sum} = 1, \text{borrow} = 1$	$0 - 1 - \underline{1} \rightarrow \text{sum} = 0, \text{borrow} = 1$
$1 - 0 \rightarrow \text{sum} = 1, \text{borrow} = 0$	$1 - 0 - \underline{1} \rightarrow \text{sum} = 0, \text{borrow} = 0$
$1 - 1 \rightarrow \text{sum} = 0, \text{borrow} = 0$	$1 - 1 - \underline{1} \rightarrow \text{sum} = 1, \text{borrow} = 1$

➤ **Example:**

$$\begin{array}{r}
 \begin{array}{cccccc}
 \underline{1} & \underline{1} & \underline{1} & 0 & & \\
 1 & 0 & 1 & 0 & 1 & \\
 - & 0 & 1 & 1 & 1 & 1 \\
 \hline
 0 & 0 & 1 & 1 & 0 &
 \end{array}
 \end{array}$$

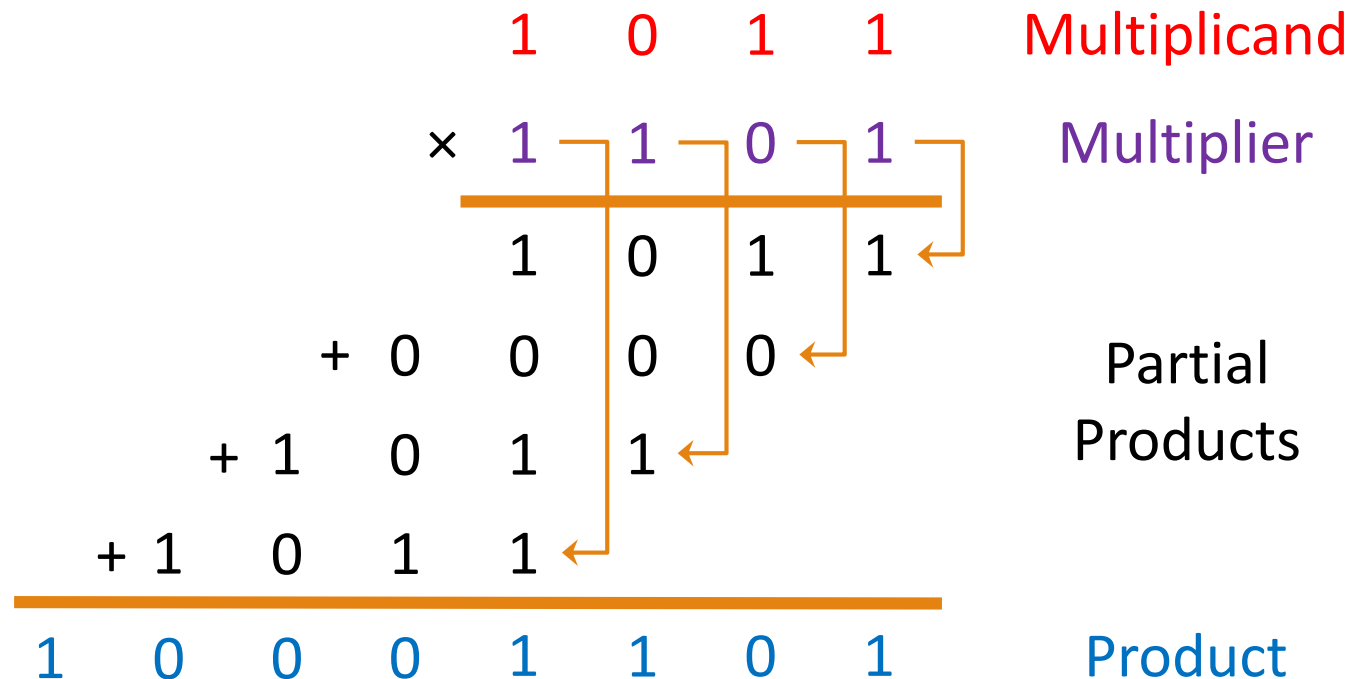
The diagram illustrates the subtraction process with borrow propagation. The top row shows the minuend bits (1, 0, 1, 0, 1) and the borrow-in (1) for the first bit. The second row shows the subtrahend bits (0, 1, 1, 1, 1). The third row shows the resulting difference bits (0, 0, 1, 1, 0). Arrows indicate the borrow propagation from right to left: from the 1st bit to the 2nd, 2nd to 3rd, 3rd to 4th, and 4th to 5th.

Unsigned Binary Multiplication

➤ Four basic **rules for multiplying** binary digits (bits):

- $0 \times 0 = 0$, $0 \times 1 = 0$, $1 \times 0 = 0$, $1 \times 1 = 1$

➤ **Example:**



Unsigned Binary Division

- Same procedure as decimal division ➔ **Long Division**

➤ **Example:**

The diagram illustrates the binary long division process. The Divisor is 1011 and the Dividend is 00001101. The Quotient is 00001101 and the Remainder is 0100. The steps shown are:

 - Divisor: 1011
 - Dividend: 00001101
 - Partial Remainders: 0011, 1011, 0011, 1011
 - Quotient: 00001101
 - Remainder: 0100

Reading Material

➤ Floyd, Chapter 2:

- Pages 54 – 57
- Pages 72 – 82