Logic Design I (17.341)

Fall 2011

Lecture Outline

Class # 01

September 12, 2011

Dohn Bowden

Today’s Lecture

- Administrative
- General Course Overview
- Main Logic Topic
- Lab
- Homework

Course
Admin

Administrative

- Admin for tonight ...
  - Attendance/Introductions/Backgrounds
  - Syllabus
  - Textbook
  - 17.341 Web Site
  - Email List creation
  - Course Objectives

Attendance / Introductions / Backgrounds

- Attendance ...
  - When called ... please introduce yourself
    - Include the following
      - Knowledge of electronics/logic design
      - Education
      - Work Experience
      - Other notable work/engineering/hobbies
      - Future Plans

My Background

- Education
- Work Experience
- Other notable work/engineering/hobbies
- Future Plans

Syllabus

- Syllabus
  - Hard copies available
  - Electronic copy available on the class website
    - Web Address on syllabus
  - Updates will be posted on the class website

Syllabus Review

Week	***Date***	***Topics***	***Chapter***	***Lab Report Due***
1	09/12/11	Introduction to digital systems, number systems, and Binary Codes	1	
2	09/19/11	Boolean algebra	2	
3	09/28/11	Boolean algebra (continued)	3	
4	10/03/11	**Examination 1**		
X	**10/10/11**	**No Class - Holiday**		
5	10/17/11	Application of Boolean Algebra. Lab lecture	4	
6	10/24/11	Karnaugh Maps	5	
7	10/31/11	Multi-Level Gate Circuits. NAND and NOR Gates	7	1
8	11/07/11	**Examination 2**		
9	11/14/11	Combinational Circuit Design and Simulation Using Gates	8	2
10	11/23/11	Multiplexers, Decoders. Encoder, and PLD	9	
11	11/28/11	Introduction to VHDL	10	3
12	12/05/11	**Examination 3**		
13	12/12/11	Review		4
14	12/19/11	**Final Exam**		

Class Hours

- Monday evenings ... 6 – 9 PM
  - Lectures ... BL-313

  - Access to Software (for homework assignments)
    - Computer Labs ... BL-420

    ... *OR* ...

    - Engineering Lab ... EB-321

- Call/email if you will not be in class ...

- I am available for extra help ***Before*** / ***After*** class
    - If possible ... please schedule in advance so I will ensure that I am available

Textbook

- Roth & Kinney, "Fundamentals of Logic Design, 6th Ed", CENGAGE Learning, 2010
  - Also available as an eBook
- Anh Tran, "Experiments in Logic Design” (*To be handed out in class*)

Software (*Optional*)

- If you want to purchase the software ...
  - Capilano Computing Systems Ltd., “LogicWorks 5: Interactive Circuit Design Software”, Addison Wesley, 2004

    ... *OR* ...

  - Capilano Computing Systems Ltd., “LogicWorks 4: Interactive Circuit Design Software”, Addison Wesley, 1999
- Otherwise either LogicWorks 4 or 5 is available in the computer laboratory (BL - 420) ... *OR* ... Engineering Laboratory (EB - 321)

Software (*included in the text*)

- Computer-aided Logic Design Program …
  - LogicAid
    - Included on the text’s CD
- Logic Simulator
  - SimUaid
    - Also included on the text’s CD

Access to Computer Lab and/or Engineering Lab

- To gain access to the Computer Lab (BL-420) ... *and/OR* ... the Engineering Lab (EB-321) ...
  - You will need to have an access cards
  - Access cards are given out at Access Services
    - Arrangements are done through the Continuing Education Office
  - Then I will need your UMS# (will be on your Access Card)
  - I will then send your name and UMS# to the Room POC

Course Web Site

- The Course Homepage is at:

  http://faculty.uml.edu/dbowden

- This website will contain the following:
  - Syllabus
  - Reference documents
    - Such as the textbook material
    - Links
  - Class lectures (will be placed on the web site *AFTER* the lecture)
  - Homework

Email Distribution List

- I will be creating a class email list
- Email me at:

Dohn_Bowden@uml.edu

- This will ensure that your correct email address or addresses are included
- The email list will allow me to provide information to each of you
- Let me know if you have a problem with your email in the “To” block of email messages

Course Objectives

- What do you want to get out of this class?

- My goals for the course ...

Course Evaluations

- How they are used

Questions?

Course Objectives and Introduction

Course Objectives

Course Objectives

- Design Logic circuits that ...
  - Are efficient
    - Minimal components to perform the operations required
  - At minimal cost
- Efficient and minimal cost go hand and hand
  - Efficient circuits will cost less due to a reduction in components

Course Objectives

- Logic Design I ...
  - Will center on combinational logic circuits
- Logic Design II ...
  - Will be geared towards sequential logic circuits
- Differences between combinational and sequential circuits ...
  - Will be discussed shortly

Course Objectives

- Towards the end of the semester we will start looking at ...
  - VHDL ...
    - *V*HSIC *H*ardware *D*escriptive *L*anguage
  - VHSIC ...
    - *V*ery *H*igh *S*peed *I*ntegrated *C*ircuit
- Logic Design II will explore VHDL in greater detail

Course Objectives

- VHDL hardware description language will be used for ...
  - The design of combinational logic
  - Sequential logic
  - And ... simple digital systems

Course Objectives

- We will be looking at ...
  - Boolean algebra ...
    - The basic mathematical tool needed to analyze and synthesize switching circuits
- We will also design circuits of logic gates that ... have ...
  - A specified relationship between signals at the input and output terminals

Course Objectives

- In Logic Design II ...
  - We will evaluate sequential switching circuits
  - Look at the Logical properties of ...
    - Flip-flops ... which ...
      - Serve as memory devices in sequential switching circuits

Course Objectives

- By combining flip-flops with circuits of logic gates ... we will ...
  - Design counters
  - Adders
  - Sequence detectors
  - And similar circuits

Chapter 1 ...

Introduction
and
Number Systems/Conversions

Introduction
to

Digital Systems
and
Switching Circuits

Introduction to Logic Design

- Fundamentals principles of Logic Design ...
  - Has not changed over the years ... however ...
    - There have been advances in technology used to ...
      - Implement digital systems

Introduction to Logic Design

- Design of digital systems ... three components ...
  - System design
  - Logic design ... and ...
  - Circuit design
- This course will focus in on the *Logic Design* component
- The three components in a little more detail ...

Introduction to Logic Design

- System design ...
  - Breaking the over-all system into subsystems ... and ...
  - Specifying the characteristics of each subsystem
- *Example* ... system design of a digital computer ...
  - Number and type of memory units
  - arithmetic units
  - Input - output devices ... as well as ...
  - The interconnection and control of these subsystems

Introduction to Logic Design

- Logic design ... what we will center on this semester ...
  - Determining how to interconnect basic logic building blocks to perform a specific function
- *Example* ...
  - Determine the interconnection of logic gates and flip-flops required to perform binary addition

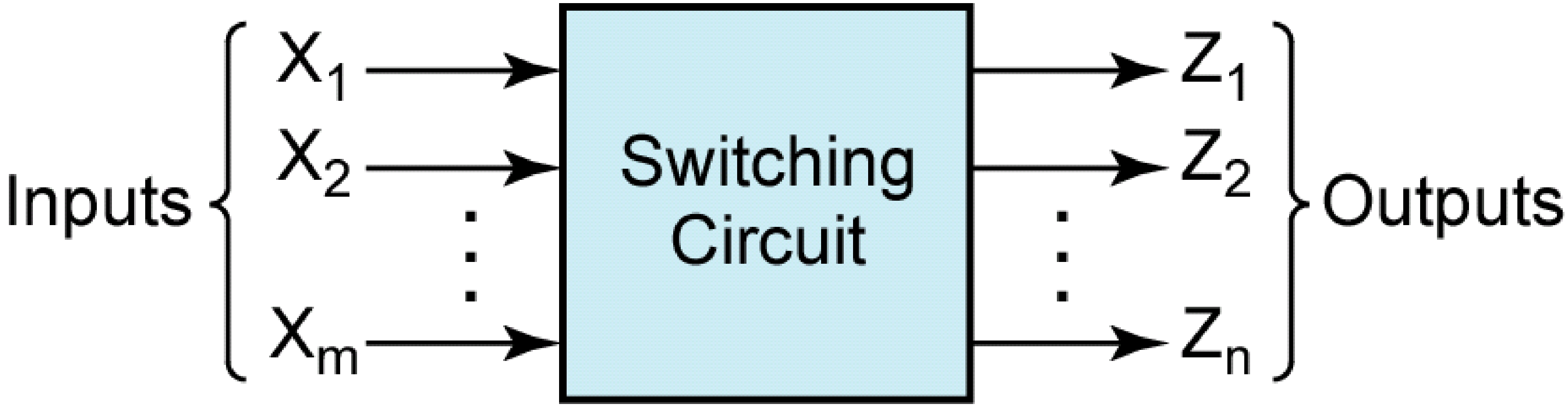
Introduction to Logic Design

- Circuit design ...
  - The interconnection of specific components such as ...
    - Resistors
    - Diodes
    - Transistors
  - To form a gate, flip-flop, or other logic building blocks

Introduction to Logic Design

- Digital systems ... utilize switching circuits
- Switching circuits ...
  - Have one or more inputs ... and ...
  - One or more outputs ...
    - Which take on *discrete values*

Introduction to Logic Design



Introduction to Logic Design

- Logic circuits have ...
  - A specified relationship between signals at the input terminals of the circuit ... and ...
  - Signals at the output terminals of the circuit

Introduction to Logic Design

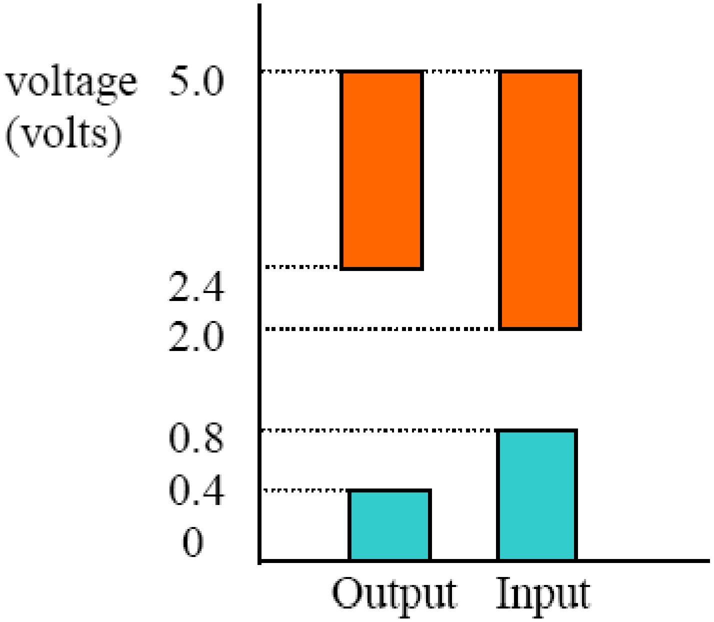
- We shall apply switching theory ...to ...
  - The solution of logic design problems
- We will learn both ...
  - The basic theory of switching circuits ... and ...
  - How to apply it

Digital and Analog Systems …

Digital Systems

- Digital systems ...
  - Signals can assume only *discrete values*
    - The output voltage of a digital system might be constrained to take on only two values such as ...
      - 0 volts ... and ...
      - 5 volts

Digital Systems



Analog Systems

- Analog systems ...
  - Signals may vary *continuously* over a specified range
    - The output voltage from an analog system might be allowed to assume any value in the ...
      - Range -10 volts to +10 volts

Switching Circuit

- There are two types of switching circuits ...
  - Combinational ... and ...
  - Sequential

Combinational Circuits

- Combinational circuit ...
  - The output values depend only on the present value of the inputs and not on past values

Sequential Circuits

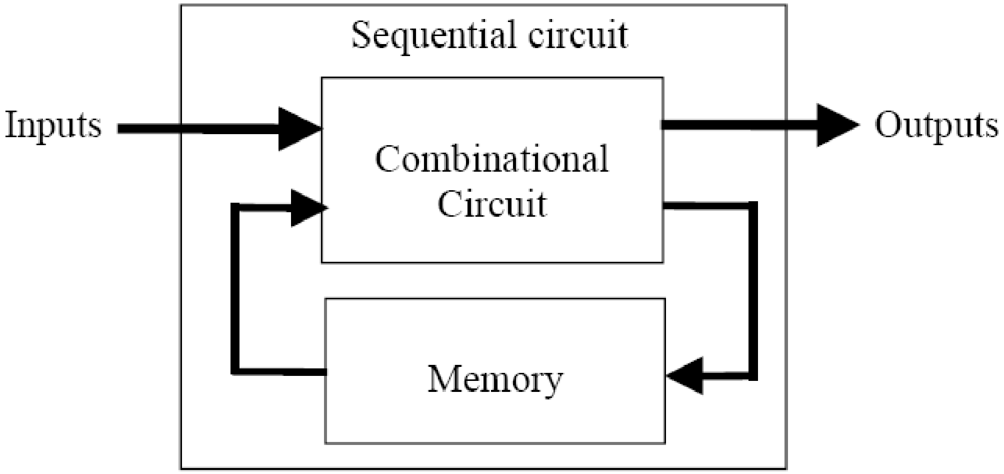
- Sequential circuits ...
  - The outputs depend on both ...
    - The present and past input values
- Sequential circuit have memory ... because ...
  - It “ remembers” something about the past sequence of inputs
- Combinational circuit has no memory

Sequential Circuits

- Sequential circuits are composed of ...
  - A combinational circuit ... with ...
  - Added memory elements

Combinational versus Sequential Circuits

Combinational circuits:	Outputs depend on present inputs
Sequential circuits:	Outputs depend on present inputs as well as past inputs



Combinational / Sequential Circuits

- Combinational circuits are easier to design than sequential circuits

- Therefore we will start with those circuits

Designing Combinational Circuits

- Combinational circuits are constructed by using logic gates
- We must determine how to interconnect these gates in order to ...
  - Convert the circuit input signals to the desired output

Designing Combinational Circuits

- The relationship between the input and output ...
  - Can be described mathematically
    - Using Boolean algebra
- We shall explore ...
  - The basic laws and theorems of Boolean algebra ... and ...
  - The behavior of circuits of logic gates

Designing Combinational Circuits

- Basic steps in designing combinational circuits ...
  - Starting with a problem statement
  - Derive a table or the algebraic logic equations which ...
    - Describes the circuit outputs as a function of the circuit inputs
  - Simplify the logic equations using algebraic methods and other methods such as ...
    - Karnaugh map and Quine-McCluskey procedure

Designing Combinational Circuits

- Basic steps ... continued ...
  - Implementation of the simplified logic equations using ...
    - Gates
    - Programmable logic devices

Designing Sequential Circuits

- Designing sequential circuits ...
  - Will be explored next semester in Logic Design II

Switching Devices

- The switching devices are two-state devices ...
  - The output can assume only two different discrete values
- Examples of switching devices ...
  - Relays
  - Diodes
  - Transistors
    - When used in cut-off or saturated state

Binary Numbers

- We will use binary numbers ...
  - Because the outputs of most switching devices assume only two different values
- Binary numbers and number systems will be discussed ...
  - Before proceeding to the design of switching circuits

Number Systems
and
Conversions

Positional Notation

Positional Notation

- Positional notation ...
  - For base 10 numbers ...
    - Each digit is multiplied by an appropriate power of 10 depending on its position in the number
- For example ...

$$
\begin{aligned}
953.78_{10} &= 9\text{x}10^{2} + 5\text{x}10^{1} + 3\text{x}10^{0} + 7\text{x}10^{-1} + 8\text{x}10^{-2} \\
&= 900 + 50 + 3 + 0.7 + 0.08 \\
&= 953.78_{10}
\end{aligned}
$$

Positional / Polynomial Representations

Number representation

$$751.36 = 7\times10^{2} + 5\times10^{1} + 1\times10^{0} + 3\times10^{-1} + 6\times10^{-2}$$

Positional representation

Polynomial representation

Positional Notation

- Positional notation ...
  - For base 2 numbers ...
    - Each digit is multiplied by an appropriate power of 2 depending on its position in the number
- For example ...

$$
\begin{aligned}
1011.11_2 &= 1\text{x}2^3 + 0\text{x}2^2 + 1\text{x}2^1 + 1\text{x}2^0 + 1\text{x}2^{-1} + 1\text{x}2^{-2} \\
&= 8 + 0 + 2 + 1 + {}^1/_2 + {}^1/_4 \\
&= 11.75_{10}
\end{aligned}
$$

Positional Notation

- The decimal and binary points separates the ...
  - Positive and negative powers

$$953.78_{10} = 9x10^{2} + 5x10^{1} + 3x10^{0} + 7x10^{-1} + 8x10^{-2}$$

$$1011.11_{2} = 1x2^{3} + 0x2^{2} + 1x2^{1} + 1x2^{0} + 1x2^{-1} + 1x2^{-2}$$

Positional Notation

- The *base* of a number system ... such as ...
  - Base 10

    953.78_{10}

  - ... or ...
  - Base 2

    1011.11_2

- Is also called the *Radix*

Positional Notation

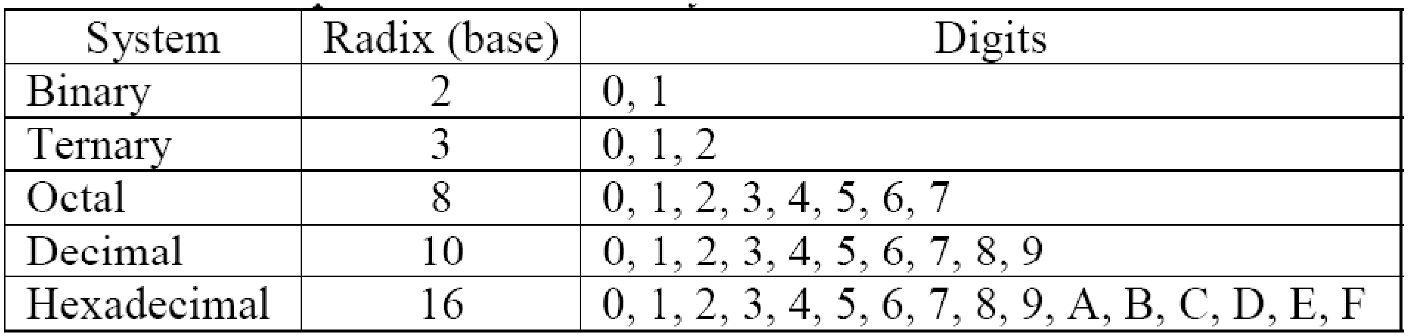
- Any positive integer ... R (R > 1) ... can be ...
  - Chosen as the *radix* or *base* of a number system
- If the *base* is R ...then ... R digits (0, 1, . . . , R - 1) are used
- For example ...
  - If R = 8 ... then ... the required digits are ...
    - » 0, 1, 2, 3, 4, 5, 6, and 7

Positional Notation

- For bases greater than 10 ...
  - More than 10 symbols are needed to represent the digits
- For example ... in hexadecimal (base 16) ...
  - A represents 10_{10}
  - B represents 11_{10}
  - C represents 12_{10}
  - D represents 13_{10}
  - E represents 14_{10}
  - F represents 15_{10}

Numbering Systems

- Examples of five number systems ...



System	Radix (base)	Digits
Binary	2	0, 1
Ternary	3	0, 1, 2
Octal	8	0, 1, 2, 3, 4, 5, 6, 7
Decimal	10	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
Hexadecimal	16	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

Numbering Systems

Base	10	2	4	8	16
	0	0	0	0	0
	1	1	1	1	1
	2	10	2	2	2
	3	11	3	3	3
	4	100	10	4	4
	5	101	11	5	5
	6	110	12	6	6
	7	111	13	7	7
	8		20	10	8
	9		21	11	9
	10		22	12	A
	11		23	13	B
	12		30	14	C
	13		31	15	D
	14		32	16	E
	15		33	17	F
	16			20	10

Power Series Expansion

Power Series Expansion

- Power series expansion ...
  - If the arithmetic indicated in the power series expansion is done in base 10 ... then ... the result is the decimal equivalent

$$147.3_8 = 1 \text{ x } 8^2 + 4 \text{ x } 8^1 + 7 \text{ x } 8^0 + 3 \text{ x } 8^{-1} = 103.375_{10}$$

base 10 **decimal**

Power Series Expansion

- The power series expansion can be used to convert to *any base* ...
  - Converting 147_{10} to base 3 would be written as ...

$$147_{10} = 1 \text{ x } (101)^2 + (11) \text{ x } (101)^1 + (21) \text{ x } (101)^0$$

base 3

» Where all the numbers on the right hand side are base 3 numbers ... **MUST USE base 3 arithmetic**

  - Then ... the result is base 3 equivalent
- However ... this is not very convenient if the arithmetic is being done by hand

Example

- Convert Base 8 to Base 10

$$(356.1)_8 = (\ ? \ )_{10}$$

$$(356.1)_8 = (3 \times 10^2 + 5 \times 10^1 + 6 \times 10^0 + 1 \times 10^{-1})_8$$

$$(3 \times 8^2 + 5 \times 8^1 + 6 \times 8^0 + 1 \times 8^{-1})_{10} = (238.125)_{10}$$

Conversion Table

ANY BASE

Power Series Expansion

BASE 10

Division Method

Division Method

- Division Method ...
  - The conversion of a decimal integer N to base R

$$N = (a_0a_1a_2....a_n)_R$$

Division Method

- The base R equivalent of a decimal integer N can be determined by …
  - Dividing N by R ... we get ...
    - Quotient Q_1 and remainder is a_0
  - Then we divide Q_1 by R ... we get ...
    - Quotient Q_2 and remainder a_1
  - The process is continued until we finally obtain a_n

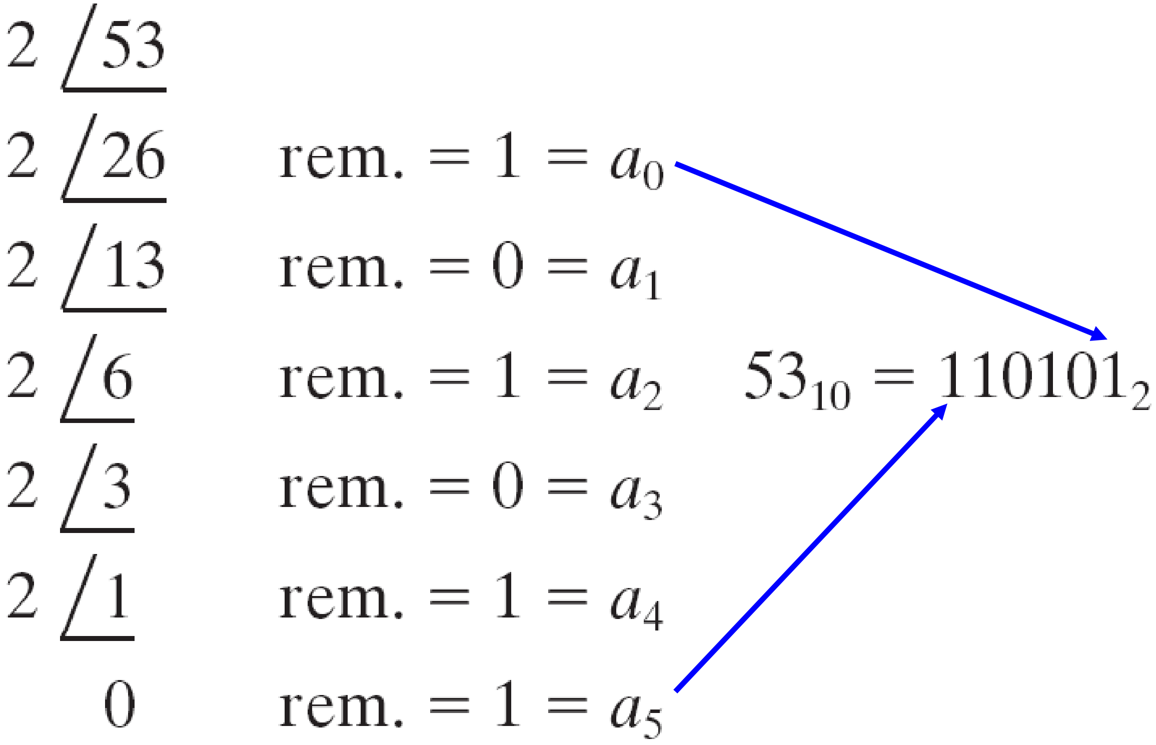
Division Method

- The remainder obtained at each step is one of the desired digits
  - The least significant digit is determined first ... a_0
  - The most significant digit is determined last ... a_n

$$N = (a_n...a_2a_1a_0)_R$$

Division Method

- Example ... Convert 53_{10} to binary



$2\,\underline{/53}$

$2\,\underline{/26}$ $\quad \text{rem.} = 1 = a_0$

$2\,\underline{/13}$ $\quad \text{rem.} = 0 = a_1$

$2\,\underline{/6}$ $\quad \text{rem.} = 1 = a_2$

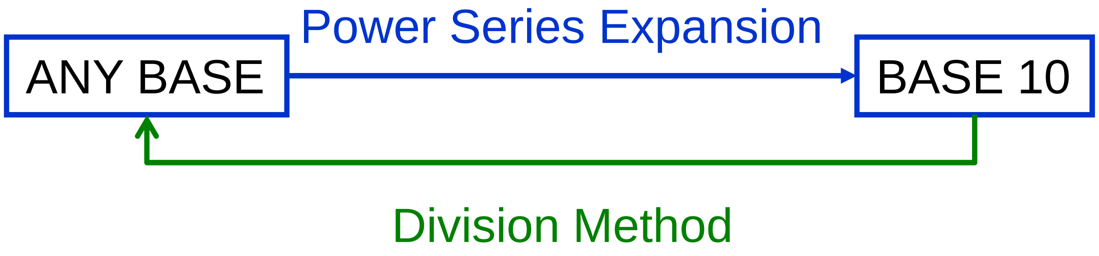
$2\,\underline{/3}$ $\quad \text{rem.} = 0 = a_3$

$2\,\underline{/1}$ $\quad \text{rem.} = 1 = a_4$

0 $\quad \text{rem.} = 1 = a_5$

$$53_{10} = 110101_2$$

Conversion Table



Power Series Expansion

ANY BASE

BASE 10

Division Method

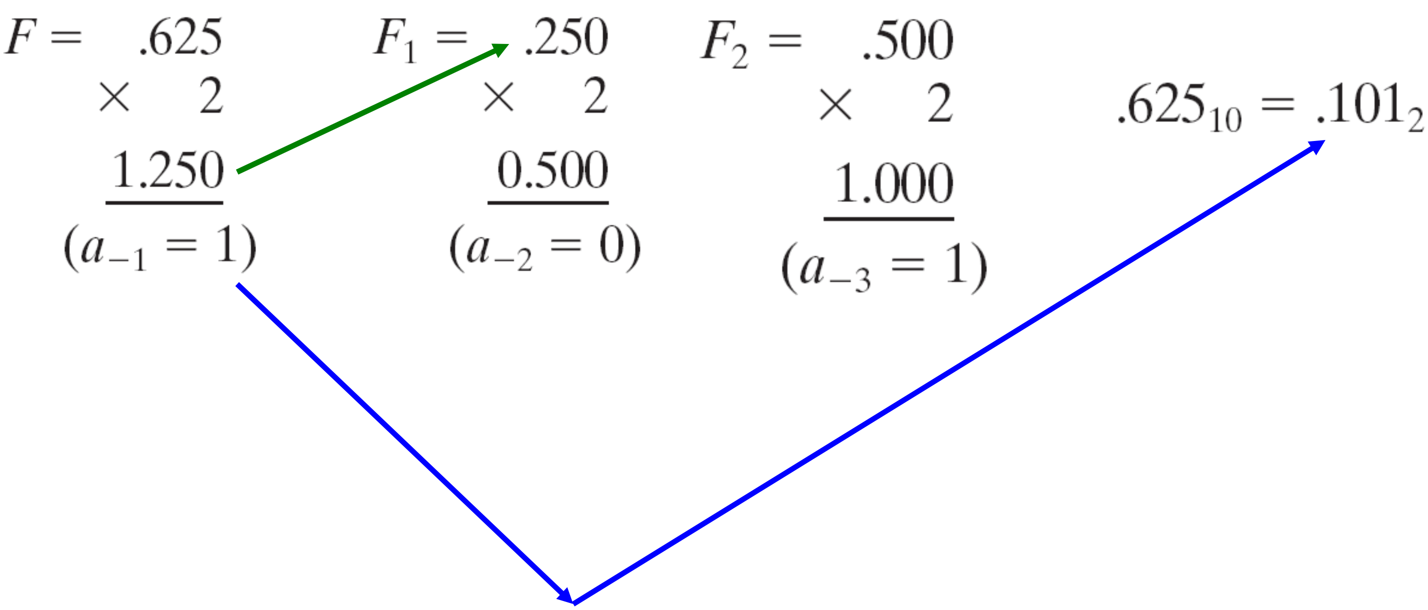
Successive Multiplications

Successive Multiplications

- Successive Multiplications ...
    - Conversion of a decimal fraction F to base R
        - Accomplished by using successive multiplications by R
- The integer part obtained at each step is one of the desired digits
- The most significant digit is obtained first

Successive Multiplications

- EXAMPLE ... Convert 0.625_{10} to binary



$$\begin{array}{r} F = \quad .625 \\ \times \quad 2 \\ \hline 1.250 \\ \hline (a_{-1} = 1) \end{array} \qquad \begin{array}{r} F_1 = \quad .250 \\ \times \quad 2 \\ \hline 0.500 \\ \hline (a_{-2} = 0) \end{array} \qquad \begin{array}{r} F_2 = \quad .500 \\ \times \quad 2 \\ \hline 1.000 \\ \hline (a_{-3} = 1) \end{array}$$

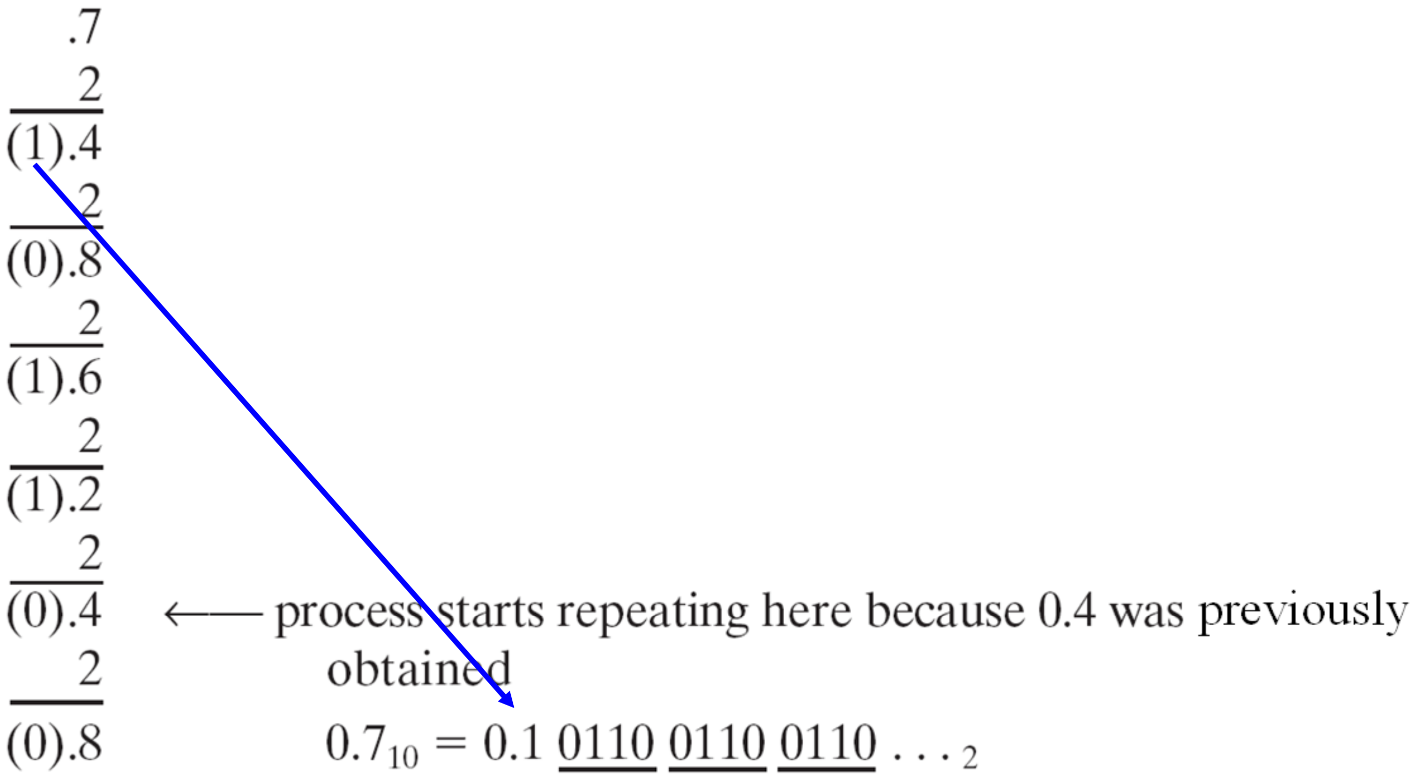
$$.625_{10} = .101_2$$

Successive Multiplications

- Successive Multiplications ...
  - This process does not always terminate ...
    - If it does not terminate ...
      - The result is a repeating fraction

Successive Multiplications

- EXAMPLE ... Convert 0.7_{10} to binary

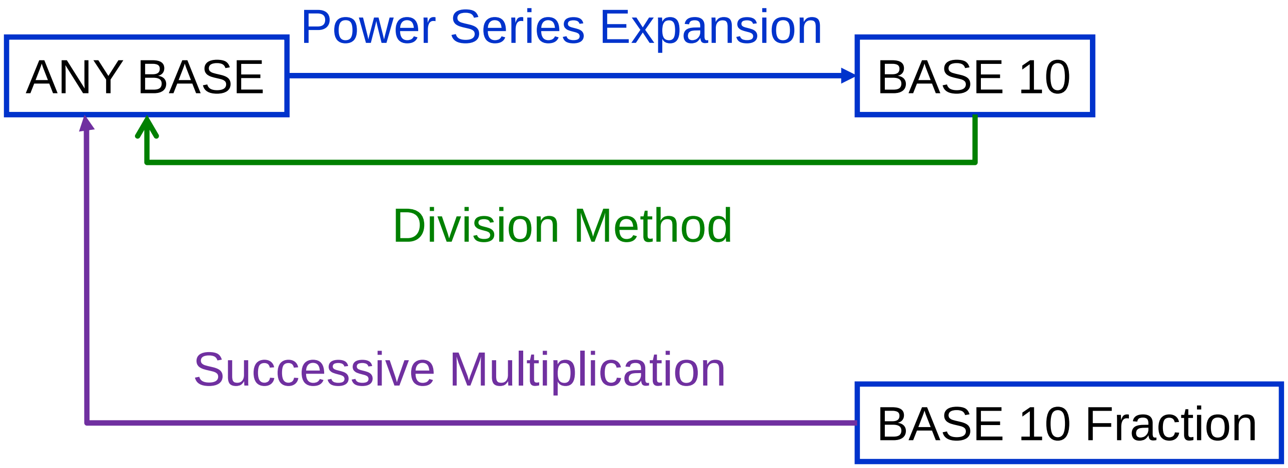


$$\begin{array}{r} .7 \\ \underline{\quad 2} \\ (1).4 \\ \underline{\quad 2} \\ (0).8 \\ \underline{\quad 2} \\ (1).6 \\ \underline{\quad 2} \\ (1).2 \\ \underline{\quad 2} \\ (0).4 \\ \underline{\quad 2} \\ (0).8 \end{array}$$

←— process starts repeating here because 0.4 was previously obtained

$$0.7_{10} = 0.1\ \underline{0110}\ \underline{0110}\ \underline{0110}\ \ldots_2$$

Conversion Table



Power Series Expansion

ANY BASE

BASE 10

Division Method

Successive Multiplication

BASE 10 Fraction

Conversions Between

Two Bases

Conversions Between Two Bases

- Conversion between two bases other than decimal can be done directly by ...
  - Using the procedures provided thus far ...
    - However ... the arithmetic operations would have to be carried out using *a base other than 10*
  - It is generally easier to convert to decimal first ... and then ... convert the decimal number to the new base
- Therefore ... to Convert between two bases other than decimal ...
  - convert to decimal first ... then ... convert the decimal number to the new base

Conversions Between Two Bases

- EXAMPLE ... Convert 231.34 to base 7

Base 4 to base 10

$$231.3_4 = 2 \times 16 + 3 \times 4 + 1 + \tfrac{3}{4} = 45.75_{10}$$

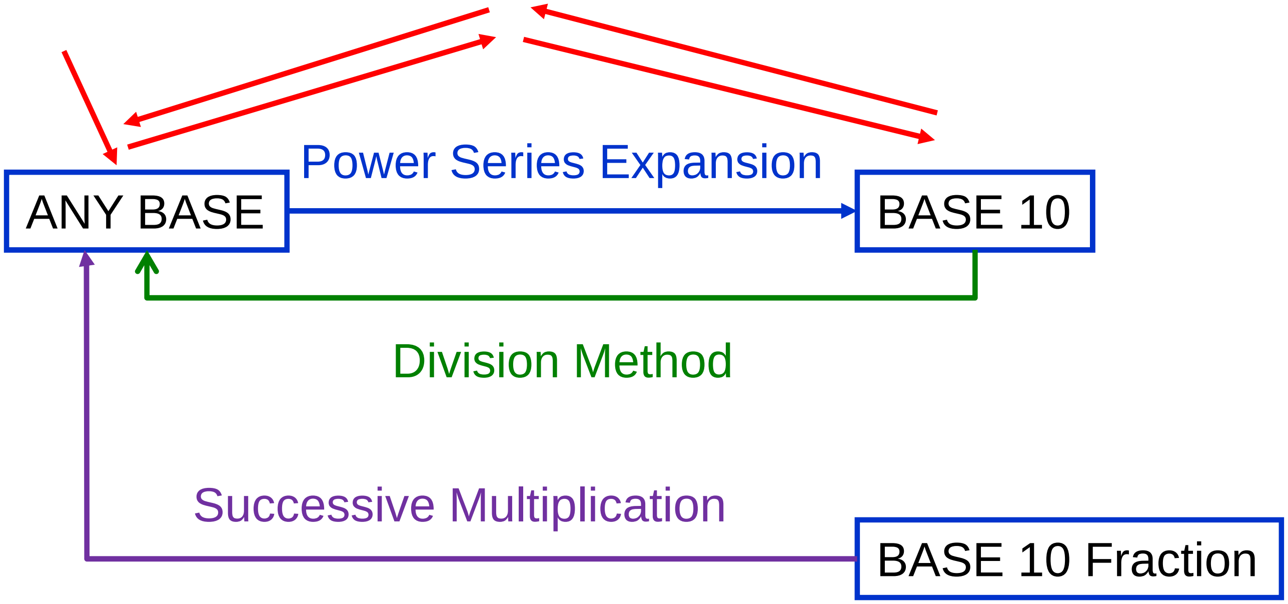
$$\begin{array}{rl} 7\,\big/\,\underline{45} & \\ 7\,\big/\,\underline{6} & \text{rem. } 3 \\ 0 & \text{rem. } 6 \end{array} \qquad \begin{array}{r} .75 \\ 7 \\ \hline (5)\ .25 \\ 7 \\ \hline (1)\ .75 \\ 7 \\ \hline (5)\ .25 \\ 7 \\ \hline (1)\ .75 \end{array}$$

$$45.75_{10} = 63.5151\ldots_7$$

Base 10 to base 7

Note that you need to
Convert the integer ...
Then the fraction ...
Then combine!

Conversion Table



Power Series Expansion

ANY BASE

BASE 10

Division Method

Successive Multiplication

BASE 10 Fraction

Any non-Base 10 to another non-Base 10 …

Convert to Base 10 … then …

Convert to other non-Base 10

Binary
to
Hexadecimal
Conversions

Binary ⬄ Hexadecimal Conversion

- Conversion from binary to hexadecimal ... and ... conversely ...
  - Convert by inspection ...
    - Each hexadecimal digit corresponds to exactly four binary digits (bits)
    - Starting at the binary point ...
      - The bits are divided into groups of four
      - Each group is replaced by a hexadecimal digit
    - Extra 0’s are added at each end of the bit string as needed to fill out the groups of four bits

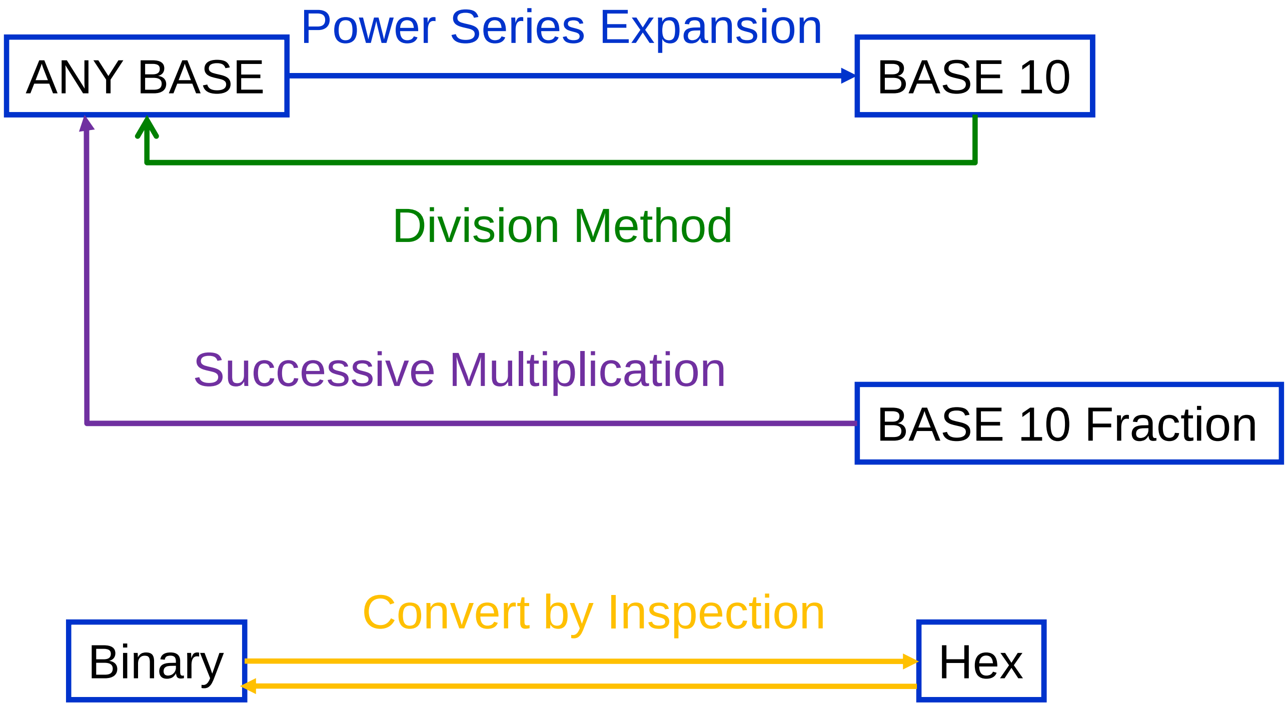
Binary ⇔ Hexadecimal Conversion

- Example ... Convert 1001101.010111_2 to hexadecimal

$$1001101.010111_2 = \underbrace{0100}_{4}\ \underbrace{1101}_{D} . \underbrace{0101}_{5}\ \underbrace{1100}_{C} = 4D.5C_{16}$$

Added zeros ... leading and trailing

Conversion Table



ANY BASE

Power Series Expansion

BASE 10

Division Method

Successive Multiplication

BASE 10 Fraction

Binary

Convert by Inspection

Hex

Summary …

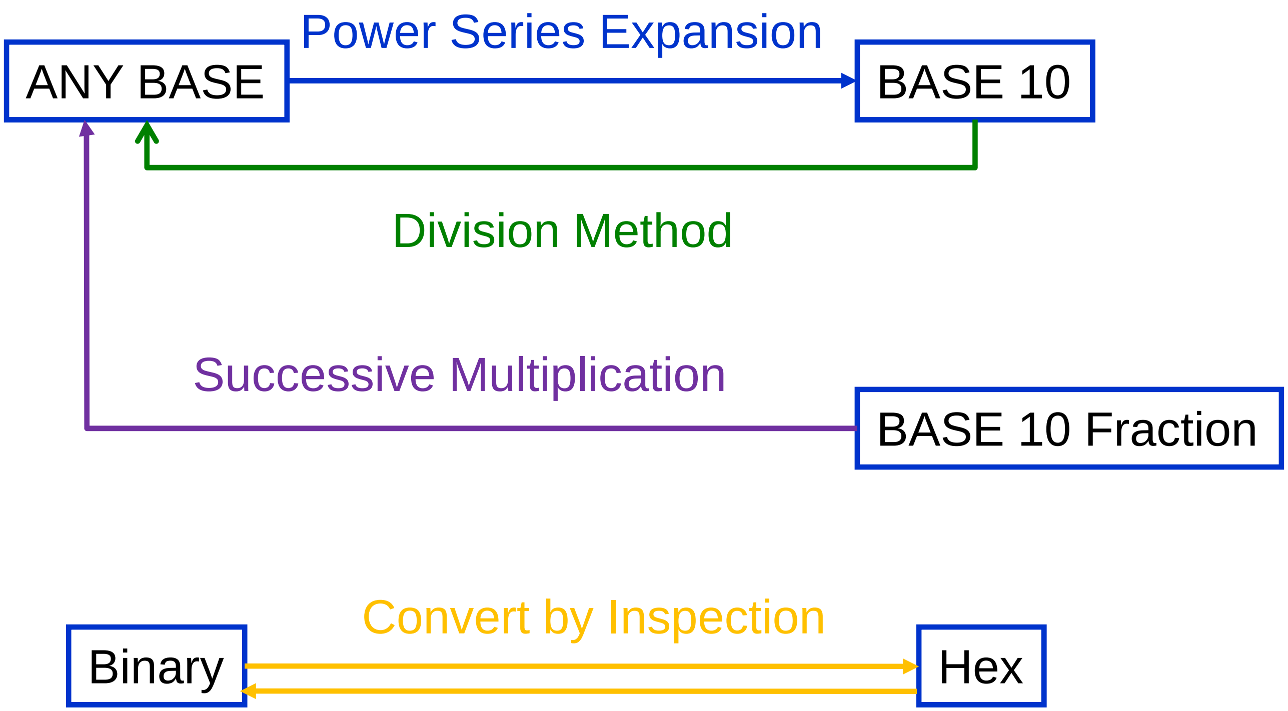
Summary

- The *Power Series Expansion* can be used to convert to *<u>any base</u>*
- *Division Method* ... the conversion of a decimal integer N to base R

  $$N = (a_0 a_1 a_2 \ldots . a_n)_R$$

- *Successive Multiplications* ... Conversion of a decimal fraction F to base R
- Conversion *between two bases other than decimal* ... convert to decimal first ... then ... convert the decimal number to the new base
- Conversion from *binary to hexadecimal ... and ... conversely* ... convert by inspection ... Each hexadecimal digit corresponds to exactly four binary digits (bits)

Conversion Table



ANY BASE

Power Series Expansion

BASE 10

Division Method

Successive Multiplication

BASE 10 Fraction

Binary

Convert by Inspection

Hex

Binary Arithmetic

Binary Arithmetic

- Arithmetic operations in digital systems are ...
  - Usually performed in binary
    - The design of logic circuits to perform binary arithmetic is much easier than for decimal

Binary Addition

Binary Arithmetic - Addition

- The addition table for binary numbers is ...

  0 + 0 = 0

  0 + 1 = 1

  1 + 0 = 1

  1 + 1 = 0 and carry 1 to next column

- Carrying 1 to a column is equivalent to adding 1 to that column

Binary Arithmetic - Addition

- Example ... Add 13_{10} and 11_{10} in binary

$$\begin{array}{rrl} & 1111 & \longleftarrow \text{carries} \\ 13_{10} = & 1101 & \\ 11_{10} = & \underline{\ 1011} & \\ & 11000 & = 24_{10} \end{array}$$

Binary Subtraction

Binary Arithmetic - Subtraction

- The subtraction table for binary numbers is ...

0 - 0 = 0

0 - 1 = 1 and borrow 1 from the next column

1 - 0 = 1

1 - 1 = 0

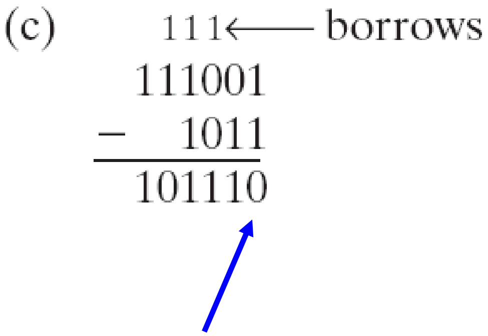
- Borrowing 1 from a column is equivalent to subtracting a 1 from that column

Binary Arithmetic - Subtraction

- Example ... of binary subtraction ...

$$\begin{array}{r} 111001 \\ \underline{-\quad 1011} \end{array}$$

Binary Arithmetic - Subtraction



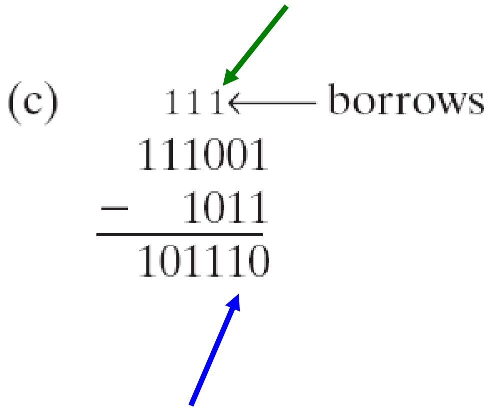
(c)

$$\begin{array}{rrl} & 111 & \longleftarrow \text{borrows} \\ & 111001 & \\ - & 1011 & \\ \hline & 101110 & \end{array}$$

- Starting with the rightmost column ...

1 - 1 = 0

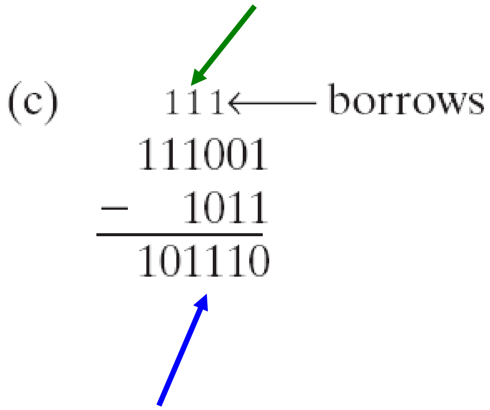
Binary Arithmetic - Subtraction



- Second column ... borrow from the third column
  - Rather than borrow immediately ... place a 1 over the third column to indicate that a borrow is necessary
    - Actual borrowing when we get to the third column
      - Similar to the way borrow propagate in a computer
  - Now because of borrowed 1 ... the second column becomes 10 ... and ...

10 - 1 = 1

Binary Arithmetic - Subtraction

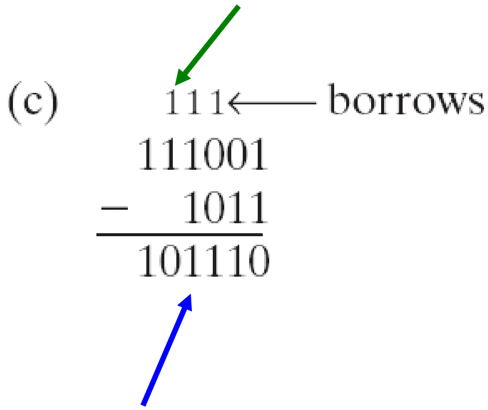


- In order to borrow 1 from the third column ... we must ...
  - Borrow 1 from the fourth column

  - Column 3 then becomes 10 ...
    - Subtracting off the borrow yields 1 ... and ...

1 - 0 = 1

Binary Arithmetic - Subtraction



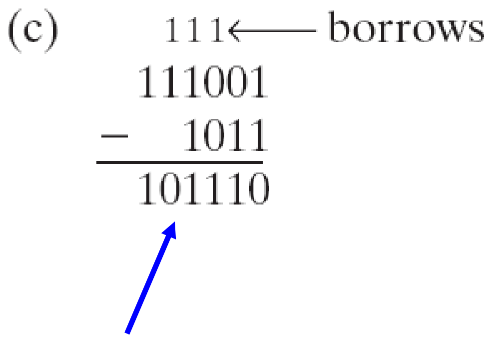
(c)

```
    111<—— borrows
  111001
−   1011
  ------
  101110
```

- Column 4 ... subtract off the borrow leaving 0
- In order to complete the subtraction ... we must ...
- Borrow from column 5 ... which gives 10 in column 4 ... and ...

10 - 1 = 1

Binary Arithmetic - Subtraction



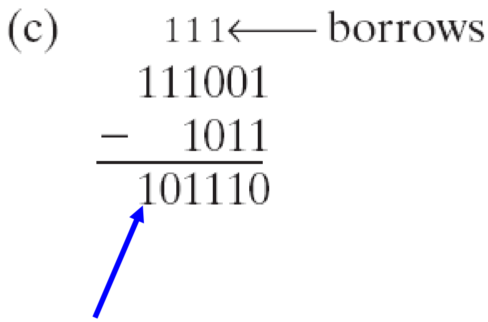
(c)

$$\begin{array}{r} 111 \longleftarrow \text{borrows} \\ 111001 \\ -\quad 1011 \\ \hline 101110 \end{array}$$

- Column 5 ... subtract off the borrow leaving 0

0 - 0 = 0

Binary Arithmetic - Subtraction



(c)

$$\begin{array}{r} 111\longleftarrow \text{borrows} \\ 111001 \\ -\quad 1011 \\ \hline 101110 \end{array}$$

- Column 6 ... subtract 0 from 1

1 - 0 = 0

- Resulting in ...

111001 – 1011 = 101110

Binary Arithmetic - Subtraction

- Two other examples of binary subtractions ...

(a)

$$\begin{array}{rl}
1\longleftarrow & \text{(indicates}\\
11101 & \text{a borrrow}\\
\underline{-10011} & \text{from the}\\
1010 & \text{3rd column)}
\end{array}$$

(b)

$$\begin{array}{rl}
1111\longleftarrow & \text{borrows}\\
10000 & \\
\underline{-\qquad 11} & \\
1101 &
\end{array}$$

Binary Multiplication

Binary Arithmetic - Multiplication

- The multiplication table for binary numbers is ...

0 x 0 = 0
0 x 1 = 0
1 x 0 = 0
1 x 1 = 1

Binary Arithmetic - Multiplication

- Example ... multiply 13_{10} by 11_{10} in binary ...

$13_{10} = 1101_2$
$11_{10} = 1011_2$ ... therefore ...

$$
\begin{array}{r}
1101 \\
\text{X} \quad 1011 \\
\hline
1101 \\
1101 \\
0000 \\
1101 \\
\hline
10001111
\end{array}
= 143_{10}
$$

Binary Arithmetic - Multiplication

- Note that each partial product is either ...
  - The multiplicand (1101) shifted over the appropriate number of places ... or ...
  - Is zero

$$\begin{array}{r}
1101 \\
\text{X} \quad 1011 \\
\hline
1101 \\
1101 \\
0000 \\
1101 \\
\hline
10001111
\end{array} = 143_{10}$$

Binary Arithmetic - Multiplication

- When adding up long columns of binary numbers ...
  - The sum of the bits in a single column can exceed 11_2
    - Therefore ... the carry to the next column can be greater than 1
- To avoid carries greater than 1 ...
  - Add the partial products one at a time

Binary Arithmetic - Multiplication

- Example ... multiplication adding partial products one at a time

1111	multiplicand
x 1101	multiplier
1111	1st partial product
0000	2nd partial product
(01111)	sum of first two partial products
1111	3rd partial product
(1001011)	sum after adding 3rd partial product
1111	4th partial product
11000011	final product (sum after adding 4th partial product)

Binary Division

Binary Arithmetic - Division

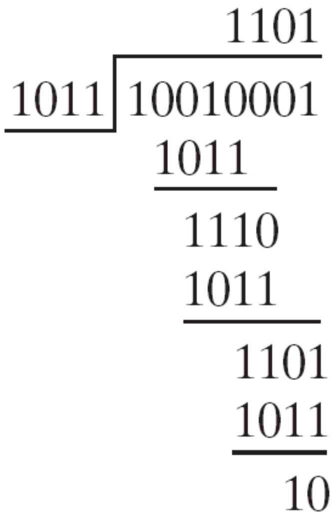
- Binary division is similar to decimal division ... except ...
  - There are only two possible quotient digits ...
    - » 0 and 1

Binary Arithmetic - Division

- Start division by ...
  - Comparing the divisor with the upper bits of the dividend
    - If we cannot subtract without getting a negative result ...
      - We move one place to the right and try again
    - If we can subtract ...
      - We place a 1 for the quotient above the number we subtracted from ... and ...
        - Append the next dividend bit to the end of the difference ... and ...
        - Repeat this process with this modified difference until we run out of bits in the dividend

Binary Arithmetic - Division

- Example ... Divide 145_{10} (10010001_2) by 11_{10} (1011_2)



The quotient is 1101 with a remainder of 10.

Representation of Negative Numbers

Negative Numbers

- Thus far ... we have been working with unsigned positive numbers
- Need to be able to work with negative numbers as well
- Three (3) Systems for representing negative numbers in binary ...
  - Sign and Magnitude system
  - 1’s complement
  - 2’s complement

Negative Numbers - Sign and Magnitude

Negative Numbers - Sign and Magnitude

- Sign and Magnitude system ...
  - Most significant bit is the sign
- Example ...

$$-5_{10} = 1101_2$$

Negative Numbers - 1’s Complement

Negative Numbers - 1’s Complement

- 1’s Complement ...
  - A negative number ... – N
  - Word length of ... n bits
  - 1’s complement ... $\overline{N}$
- 1’s complement is defined as ...

$$\overline{N} = (2^n - 1) - N$$

Negative Numbers - 1’s Complement

- Example ... 1’s Complement ... $\overline{N} = (2^n - 1) - N$ ... for $n = 4$

$-5_{10} = (2^4 - 1) - 5 = 16 - 1 - 5 = 10_{10}$

Recall ...

$= 2\underline{/10}$

$= 2\underline{/5\;\;}$ rem $= 0 = a_0$

$= 2\underline{/2\;\;}$ rem $= 1 = a_1$

$= 2\underline{/1\;\;}$ rem $= 0 = a_2$

$= \;\; 0$ rem $= 1 = a_3$

Therefore ...

$10_{10} = 1010_2$

... So ...

$-5_{10} = 1010_2$ using 1’s complement method

Negative Numbers - 1's Complement

- For 1's complement in the 4 bit system ...
  - 1111 represents a minus 0 ... and ...
  - Minus 8 has no representation

			Negative Integers		
$+N$	Positive Integers (all systems)	$-N$	Sign and Magnitude	2's Complement N^*	1's Complement $\bar{N}$
+0	0000	−0	1000	——	1111
+1	0001	−1	1001	1111	1110
+2	0010	−2	1010	1110	1101
+3	0011	−3	1011	1101	1100
+4	0100	−4	1100	1100	1011
+5	0101	−5	1101	1011	1010
+6	0110	−6	1110	1010	1001
+7	0111	−7	1111	1001	1000
		−8	——	1000	——

Negative Numbers - 1’s Complement

- An alternate way to form the 1’ s complement is to ...
  - Complement N bit-by-bit by replacing 0’ s with 1’ s and 1’ s with 0’ s
- So ... for N = 5 ... the 1’s complement of N is found ...

N = 0101 by using the alternate method above

replacing 0’ s with 1’ s and 1’ s with 0’ s

$\overline{N}$ = 1010 ... which is the same as previous example

Negative Numbers - 2’s Complement

Negative Numbers - 2’s Complement

- 2’s Complement ...
  - A negative number ... – N
  - Word length of ... n bits
  - 2’s complement ... N*
- 2’s complement is defined as ...

$$N^* = 2^n - N$$

Negative Numbers - 2's Complement

- Example ... 2's Complement ... $N^* = 2^n - N$ ... for $n = 4$

$- 5_{10} = 2^4 - 5 = 16 - 5 = 11_{10}$

Recall ...

$= 2\underline{/11}$

$= 2\underline{/5\ \ }$ rem $= 1 = a_0$

$= 2\underline{/2\ \ }$ rem $= 1 = a_1$

$= 2\underline{/1\ \ }$ rem $= 0 = a_2$

$= \ \ 0$ rem $= 1 = a_3$

Therefore ...

$11_{10} = 1011_2$

... So ...

$- 5_{10} = 1011_2$ using 2's complement method

Negative Numbers - 2’s Complement

- Alternate methods obtaining 2’ s complement ...
  - Complementing N bit-by-bit and then adding 1
  - Or ...
  - Starting at the right ... complement all bits to the left of the first 1

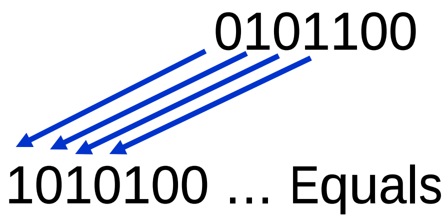
Negative Numbers - 2’s Complement

- Example ... if ... N = 0101100 … then … N* = 1010100 …
- By either …

0101100 … Complement to get … 1010011

1010011 ... Then add 1
+ 0000001
1010100 … Equals N*

- Or … complement all bits to the left of the first 1 ... and get ...



0101100

1010100 … Equals N*

Signed Binary Integers

(Word Length n = 4)

Signed Binary Integers (Word Length n = 4)

			Negative Integers		
$+N$	Positive Integers (all systems)	$-N$	Sign and Magnitude	2's Complement N^*	1's Complement $\bar{N}$
+0	0000	−0	1000	——	1111
+1	0001	−1	1001	1111	1110
+2	0010	−2	1010	1110	1101
+3	0011	−3	1011	1101	1100
+4	0100	−4	1100	1100	1011
+5	0101	−5	1101	1011	1010
+6	0110	−6	1110	1010	1001
+7	0111	−7	1111	1001	1000
		−8	——	1000	——

Determine Magnitude of Negative Numbers

Magnitude of Negative Numbers

- Sign and Magnitude system ...
  - Most significant bit is the sign
  - Remainder is the magnitude
- 1's Complement system ...
  - Take the 1' s complement of $\overline{N}$

    $$N = (2^n - 1) - \overline{N}$$
- 2's Complement system ...
  - Take the 2' s complement of N*

    $$N = 2^n - N^*$$

Addition of 2’s Complement Numbers

Addition of 2's Complement Numbers

- The addition is carried out just as if all the numbers were positive ...
  - Any carry from the sign position is ignored
    - This will always yield the correct result except when an *overflow* occurs
- *Overflow* ... has occurred if ...
  - The correct representation of the sum (including sign) requires more than n bits (for word length of n bits)

Addition of 2's Complement Numbers

- A general rule for detecting overflow when adding two n-bit signed binary numbers (1' s or 2' s complement) to get an n-bit sum is ...

- An *overflow* occurs if ...

  - Adding two positive numbers gives a negative answer ... or ...

  - If adding two negative numbers gives a positive answer

Addition of 2’s Complement Numbers

1. Addition of two positive numbers, sum $< 2^{n-1}$

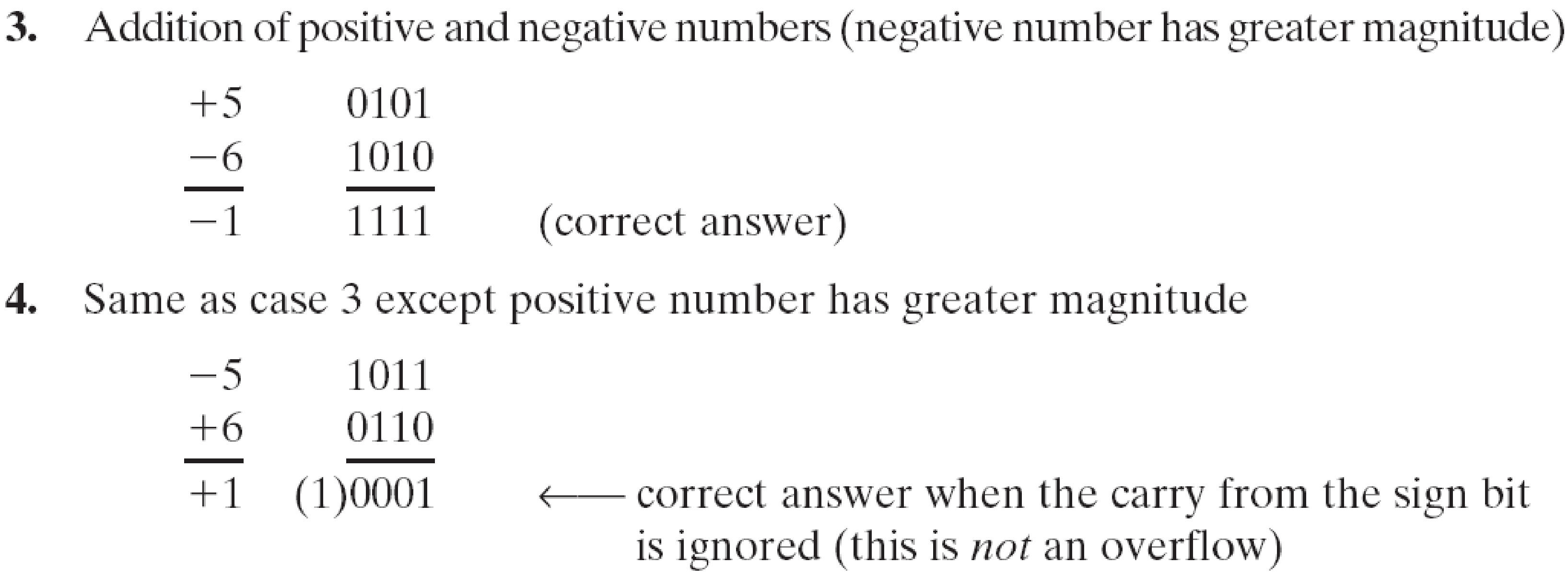
$$\begin{array}{rr} +3 & 0011 \\ +4 & 0100 \\ \hline +7 & 0111 \end{array} \qquad \text{(correct answer)}$$

2. Addition of two positive numbers, sum $\geq 2^{n-1}$

$$\begin{array}{rr} +5 & 0101 \\ +6 & 0110 \\ \hline & 1011 \end{array} \qquad \longleftarrow \text{wrong answer because of overflow } (+11 \text{ requires}$$

5 bits including sign)

Addition of 2’s Complement Numbers



3. Addition of positive and negative numbers (negative number has greater magnitude)

$$\begin{array}{rrl} +5 & 0101 & \\ \underline{-6} & \underline{1010} & \\ -1 & 1111 & \text{(correct answer)} \end{array}$$

4. Same as case 3 except positive number has greater magnitude

$$\begin{array}{rrl} -5 & 1011 & \\ \underline{+6} & \underline{0110} & \\ +1 & (1)0001 & \longleftarrow \end{array}$$

correct answer when the carry from the sign bit is ignored (this is *not* an overflow)

Addition of 2’s Complement Numbers

5. Addition of two negative numbers, $|\text{sum}| \le 2^{n-1}$

$$\begin{array}{rr} -3 & 1101 \\ -4 & 1100 \\ \hline -7 & (1)1001 \end{array}$$

⟵ correct answer when the last carry is ignored
(this is *not* an overflow)

6. Addition of two negative numbers, $|\text{sum}| > 2^{n-1}$

$$\begin{array}{rr} -5 & 1011 \\ -6 & 1010 \\ \hline & (1)0101 \end{array}$$

⟵ wrong answer because of overflow
(−11 requires 5 bits including sign)

Addition of 1’s Complement Numbers

Addition of 1’s Complement Numbers

- Addition of 1’ s complement numbers is similar to 2’ s complement except ...
  - Instead of discarding the last carry ...
    - It is added to the n-bit sum in the position furthest to the right
      - This is referred to as an end-around carry

Addition of 1's Complement Numbers

- RECALL ... A general rule for detecting overflow when adding two n-bit signed binary numbers (1' s or 2' s complement) to get an n-bit sum is ...

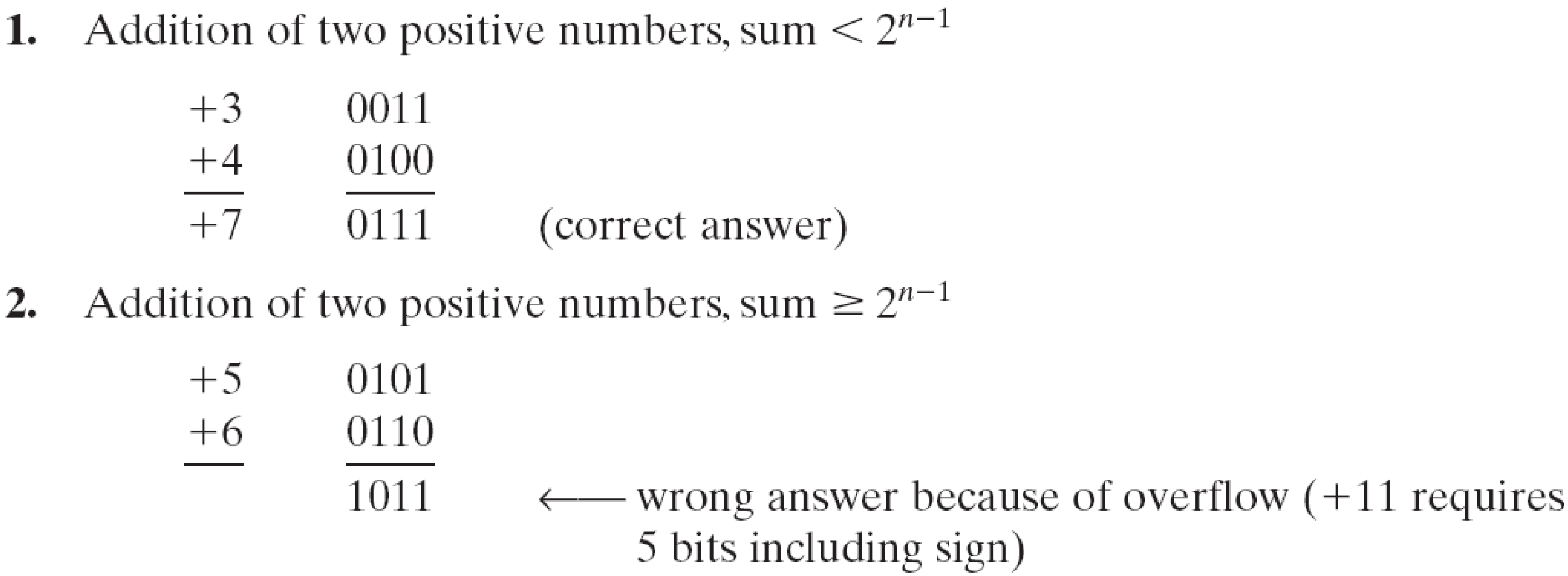
- An *overflow* occurs if ...

  - Adding two positive numbers gives a negative answer ... or ...

  - If adding two negative numbers gives a positive answer

Addition of 1’s Complement Numbers

- Same as 2’s complement!



1. Addition of two positive numbers, sum $< 2^{n-1}$

$$\begin{array}{ll} +3 & 0011 \\ \underline{+4} & \underline{0100} \\ +7 & 0111 \end{array}$$

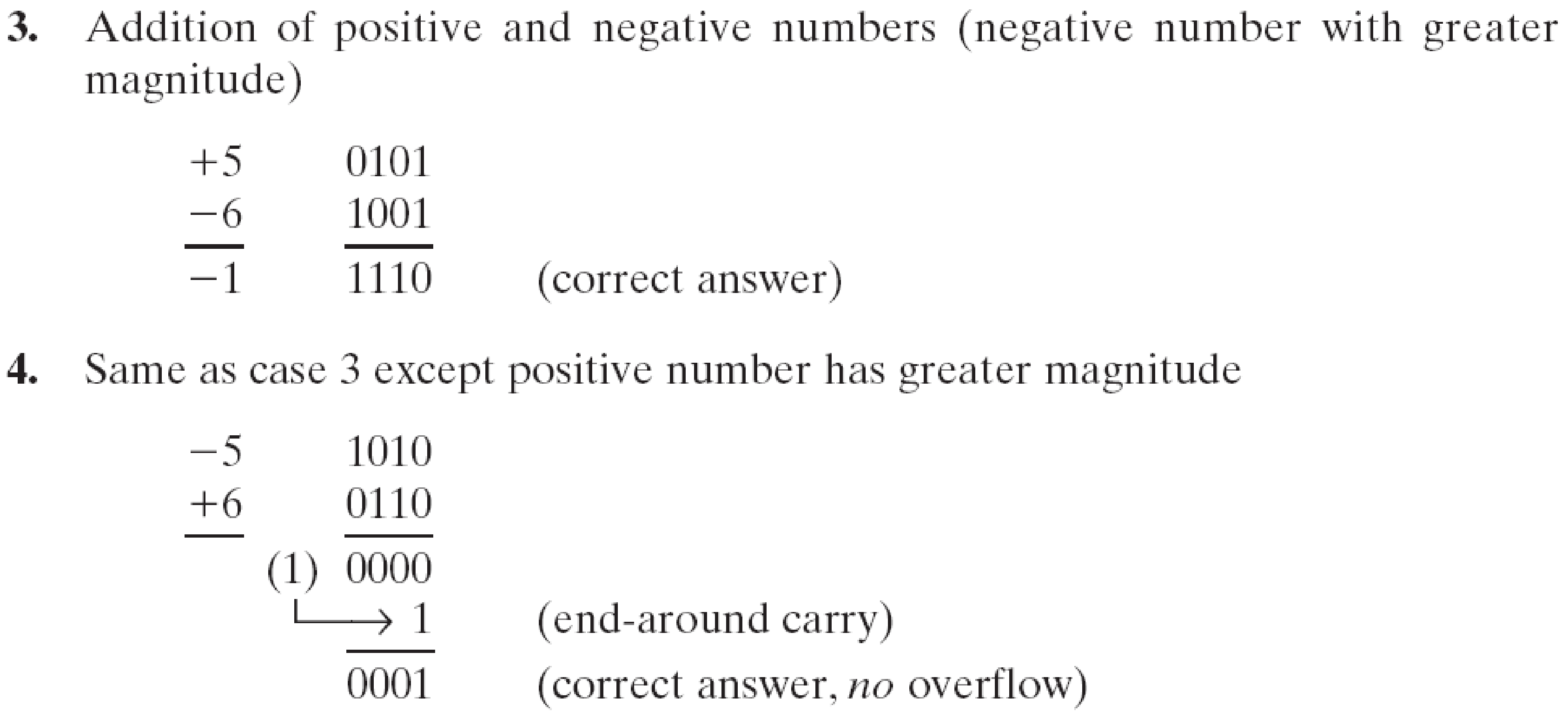
(correct answer)

2. Addition of two positive numbers, sum $\geq 2^{n-1}$

$$\begin{array}{ll} +5 & 0101 \\ \underline{+6} & \underline{0110} \\ & 1011 \end{array}$$

⟵ wrong answer because of overflow (+11 requires 5 bits including sign)

Addition of 1’s Complement Numbers



3. Addition of positive and negative numbers (negative number with greater magnitude)

$$\begin{array}{rrrl} +5 & & 0101 & \\ \underline{-6} & & \underline{1001} & \\ -1 & & 1110 & \text{(correct answer)} \end{array}$$

4. Same as case 3 except positive number has greater magnitude

$$\begin{array}{rrrl} -5 & & 1010 & \\ \underline{+6} & & \underline{0110} & \\ & (1) & 0000 & \\ & \llcorner\!\!\longrightarrow & \underline{1} & \text{(end-around carry)} \\ & & 0001 & \text{(correct answer, } \textit{no} \text{ overflow)} \end{array}$$

Addition of 1’s Complement Numbers

5. Addition of two negative numbers, $|\text{sum}| < 2^{n-1}$

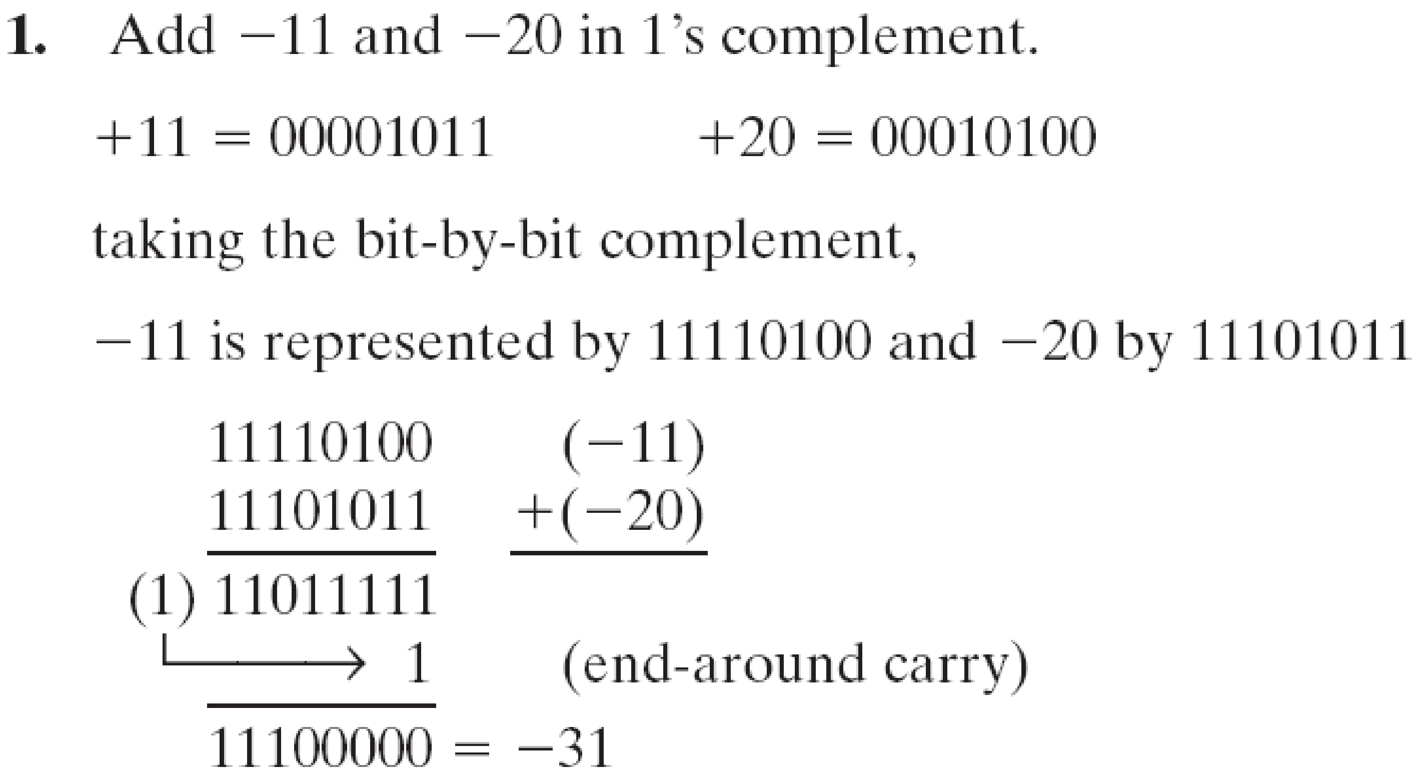
$$\begin{array}{rrrl} -3 & & 1100 & \\ \underline{-4} & & \underline{1011} & \\ & (1) & 0111 & \\ & \llcorner\!\!\longrightarrow & \underline{1} & \text{(end-around carry)} \\ & & 1000 & \text{(correct answer, } \textit{no} \text{ overflow)} \end{array}$$

6. Addition of two negative numbers, $|\text{sum}| \geq 2^{n-1}$

$$\begin{array}{rrrl} -5 & & 1010 & \\ \underline{-6} & & \underline{1001} & \\ & (1) & 0011 & \\ & \llcorner\!\!\longrightarrow & \underline{1} & \text{(end-around carry)} \\ & & 0100 & \text{(wrong answer because of overflow)} \end{array}$$

Examples of Addition of 1's Complement and 2's Complement Numbers

1's Complement … n = 8



1. Add −11 and −20 in 1's complement.

+11 = 00001011 +20 = 00010100

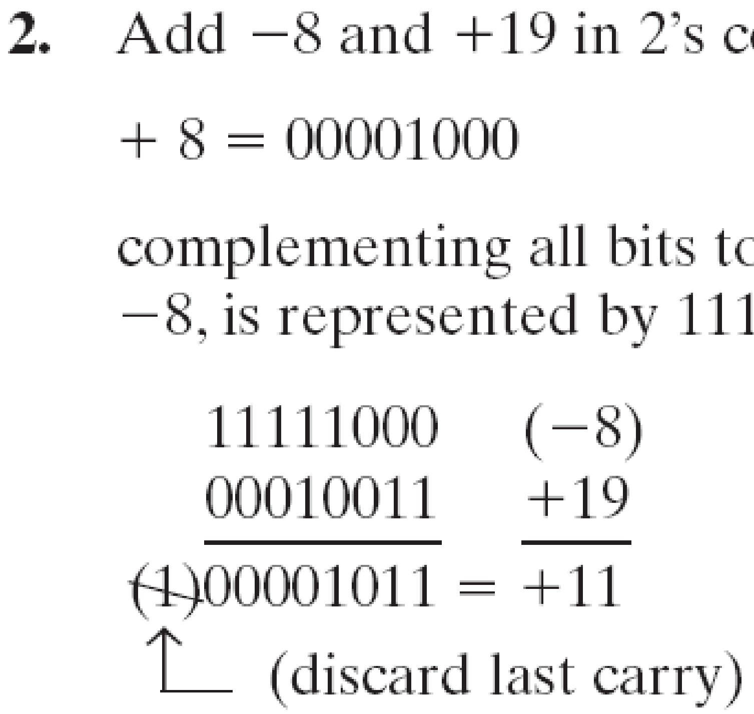
taking the bit-by-bit complement,

−11 is represented by 11110100 and −20 by 11101011

$$\begin{array}{rl} 11110100 & \quad (-11) \\ \underline{11101011} & \quad \underline{+(-20)} \\ (1)\ 11011111 & \\ \longrightarrow 1 & \quad \text{(end-around carry)} \\ \hline 11100000 = -31 & \end{array}$$

The addition produced a carry out of the furthest left bit position, but there is no overflow because the answer can be correctly represented by eight bits (including sign)

2’s Complement … n = 8



2. Add −8 and +19 in 2’s complement

+ 8 = 00001000

complementing all bits to the left of the first 1,
−8, is represented by 11111000

$$\begin{array}{rr} 11111000 & (-8) \\ \underline{00010011} & \underline{+19} \\ \cancel{(1)}00001011 = & +11 \end{array}$$

↑ (discard last carry)

The addition produced a carry out of the furthest left bit position, but there is no overflow because the answer can be correctly represented by eight bits (including sign)

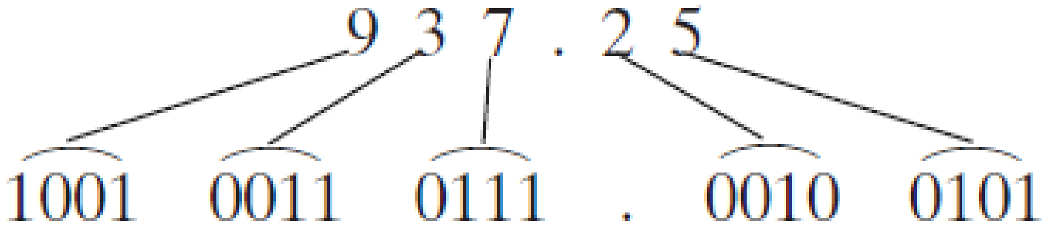
Binary Codes

Binary Code

- Most computers work internally with binary numbers ... however ...
  - Input-output equipment generally use decimal numbers
- Most logic circuits only accept two-valued signals ... therefore ...
  - Decimal numbers must be coded in terms of binary signals
  - In the simplest form of binary code, each decimal digit is replaced by its binary equivalent. For example, 937.25 is represented by:

Binary Code

- In the simplest form of binary code ...
  - Each decimal digit is replaced by its binary equivalent
- Example ... 937.25 is represented by ...

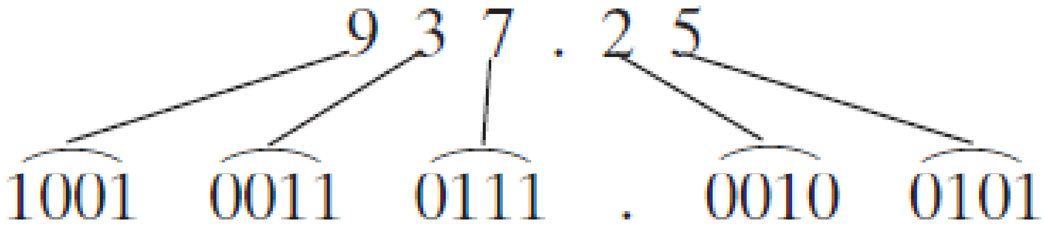


- The result is quite different than that obtained by converting the number as a whole into binary

Binary-Coded Decimal (BCD)

Binary-Coded Decimal - (BCD)

- The following representation is referred to as binary-coded-decimal (BCD) ... or ...
  - 8-4-2-1 BCD



  - There are only ten decimal digits ... therefore ...
    - 1010 through 1111 are not valid BCD codes

Other Binary Codes

Binary Codes

- Binary-coded-decimal (BCD) ... or ... 8-4-2-1 BCD
- 6-3-1-1 Code
- Excess-3 Code
- 2-out-of-5 Code
- Gray Code

Decimal Digit	8-4-2-1 Code (BCD)	6-3-1-1 Code	Excess-3 Code	2-out-of-5 Code	Gray Code
0	0000	0000	0011	00011	0000
1	0001	0001	0100	00101	0001
2	0010	0011	0101	00110	0011
3	0011	0100	0110	01001	0010
4	0100	0101	0111	01010	0110
5	0101	0111	1000	01100	1110
6	0110	1000	1001	10001	1010
7	0111	1001	1010	10010	1011
8	1000	1011	1011	10100	1001
9	1001	1100	1100	11000	1000

6-3-1-1 Code

6-3-1-1 Code

- The 8-4-2-1 (BCD) code and the 6-3-1-1 code are weighted codes
- 4-bit weighted codes have weights that are ...
  - w_3 ... w_2 ... w_1 ... and ... w_0
- The code $a_3a_2a_1a_0$ represents a decimal number N ...
  - Where $N = w_3a_3 + w_2a_2 + w_1a_1 + w_0a_0$

6-3-1-1 Code

Decimal Digit	8-4-2-1 Code (BCD)	6-3-1-1 Code	Excess-3 Code	2-out-of-5 Code	Gray Code
0	0000	0000	0011	00011	0000
1	0001	0001	0100	00101	0001
2	0010	0011	0101	00110	0011
3	0011	0100	0110	01001	0010
4	0100	0101	0111	01010	0110
5	0101	0111	1000	01100	1110
6	0110	1000	1001	10001	1010
7	0111	1001	1010	10010	1011
8	1000	1011	1011	10100	1001
9	1001	1100	1100	11000	1000

6-3-1-1 Code

- Example ... the weights for the 6-3-1-1 code are ...

  $w_3 = 6$, $w_2 = 3$, $w_1 = 1$, and $w_0 = 1$

  The binary code 1011 thus represents the decimal digit ...

  $N = 6*1 + 3*0 +1*1 +1*1 = 8_{10}$

Excess-3 Code

Excess-3 Code

- Excess-3 code is ….
  - Obtained from the 8-4-2-1 code by …
    - Adding 3 (0011) to each of the codes

Decimal Digit	8-4-2-1 Code (BCD)	6-3-1-1 Code	Excess-3 Code	2-out-of-5 Code	Gray Code
0	0000	0000	0011	00011	0000
1	0001	0001	0100	00101	0001
2	0010	0011	0101	00110	0011
3	0011	0100	0110	01001	0010
4	0100	0101	0111	01010	0110
5	0101	0111	1000	01100	1110
6	0110	1000	1001	10001	1010
7	0111	1001	1010	10010	1011
8	1000	1011	1011	10100	1001
9	1001	1100	1100	11000	1000

2-out-of-5 Code

2-out-of-5 Code

- 2-out-of-5 code ...
  - 2 out of the 5 bits are 1 for every valid code combination
- This code has useful error-checking properties ...
  - If any one of the bits in a code combination is changed due to a malfunction of the logic circuitry ...
    - The number of 1 bits is no longer exactly two

2-out-of-5 Code

Decimal Digit	8-4-2-1 Code (BCD)	6-3-1-1 Code	Excess-3 Code	2-out-of-5 Code	Gray Code
0	0000	0000	0011	00011	0000
1	0001	0001	0100	00101	0001
2	0010	0011	0101	00110	0011
3	0011	0100	0110	01001	0010
4	0100	0101	0111	01010	0110
5	0101	0111	1000	01100	1110
6	0110	1000	1001	10001	1010
7	0111	1001	1010	10010	1011
8	1000	1011	1011	10100	1001
9	1001	1100	1100	11000	1000

2-out-of-5 Code

- 2-out-of-5 code is not a weighted code
  - The decimal value of a coded digit cannot ...
    - Be computed by a simple formula when a non-weighted code is used

Gray Code

Gray Code

- Gray code ...
  - Codes for successive decimal digits differ in exactly one bit
  - A Gray code is often used when translating an analog quantity, such as a shaft position, into digital form
    - A small change in the analog quantity will change only one bit in the code ... which ...
      - Gives more reliable operation than if two or more bits changed at a time
- Gray Code is not a weighted code

Gray Code

Decimal Digit	8-4-2-1 Code (BCD)	6-3-1-1 Code	Excess-3 Code	2-out-of-5 Code	Gray Code
0	0000	0000	0011	00011	0000
1	0001	0001	0100	00101	0001
2	0010	0011	0101	00110	0011
3	0011	0100	0110	01001	0010
4	0100	0101	0111	01010	0110
5	0101	0111	1000	01100	1110
6	0110	1000	1001	10001	1010
7	0111	1001	1010	10010	1011
8	1000	1011	1011	10100	1001
9	1001	1100	1100	11000	1000

Gray Code

- Gray code is not a weighted code
  - The decimal value of a coded digit cannot ...
    - Be computed by a simple formula when a non-weighted code is used

ASCII Code

(American Standard Code for Information Interchange)

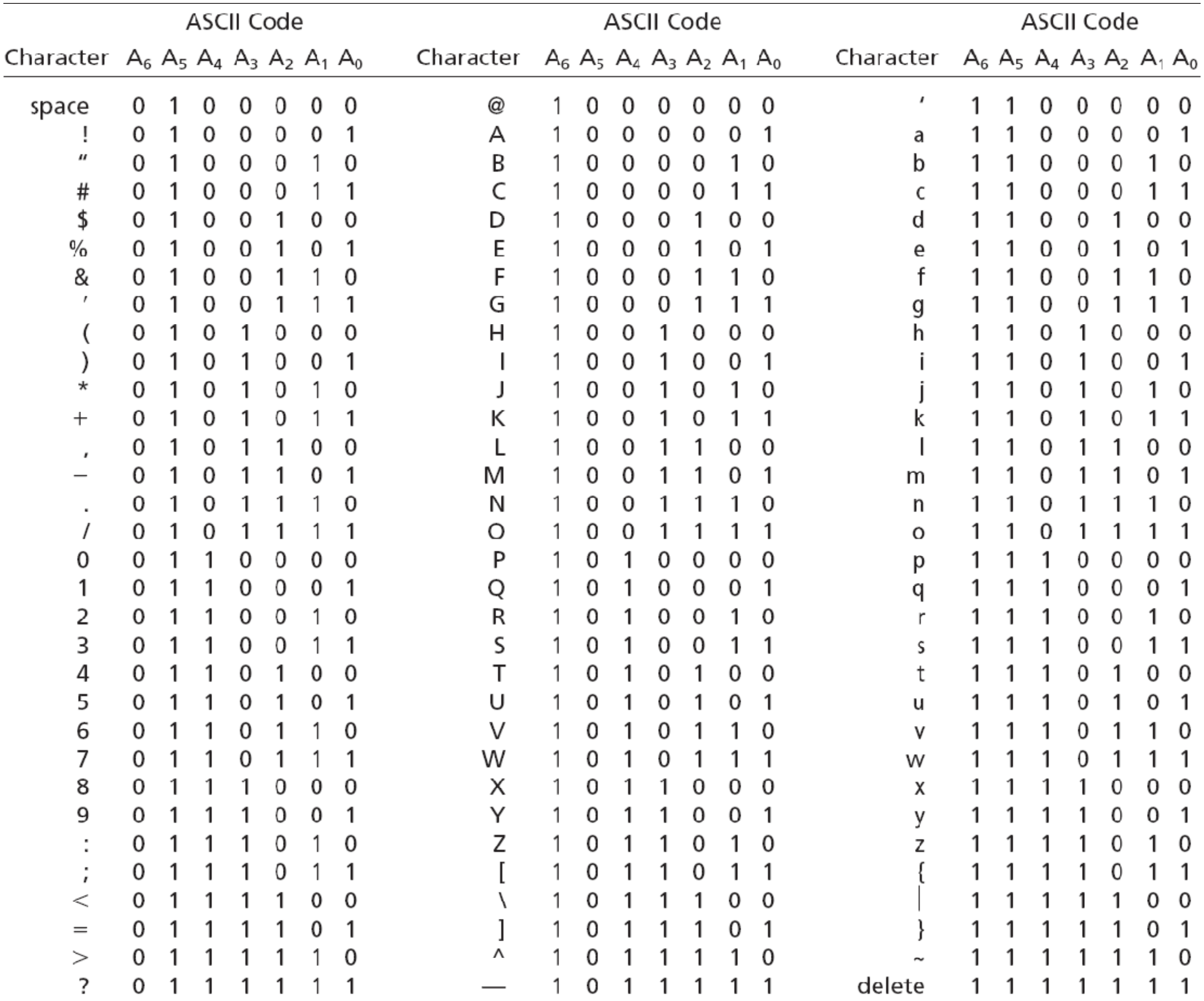
ASCII Code

- ASCII Code (American Standard Code for Information Interchange)
  - Many applications of computers require the processing of data which contains ...
    - Numbers, letters, and other symbols such as punctuation marks
  - In order to transmit such alphanumeric data to or from a computer or store it internally in a computer ...
    - Each symbol must be represented by a binary code

ASCII Code

- A 7-bit code ... therefore ...
  - 2^7 (128) different code combinations are available ...
    - They can represent ...
      - Letters
      - Numbers
      - And other symbols

ASCII Code



	ASCII Code								ASCII Code								ASCII Code						
Character	A_6	A_5	A_4	A_3	A_2	A_1	A_0	Character	A_6	A_5	A_4	A_3	A_2	A_1	A_0	Character	A_6	A_5	A_4	A_3	A_2	A_1	A_0
space	0	1	0	0	0	0	0	@	1	0	0	0	0	0	0	‘	1	1	0	0	0	0	0
!	0	1	0	0	0	0	1	A	1	0	0	0	0	0	1	a	1	1	0	0	0	0	1
“	0	1	0	0	0	1	0	B	1	0	0	0	0	1	0	b	1	1	0	0	0	1	0
#	0	1	0	0	0	1	1	C	1	0	0	0	0	1	1	c	1	1	0	0	0	1	1
$	0	1	0	0	1	0	0	D	1	0	0	0	1	0	0	d	1	1	0	0	1	0	0
%	0	1	0	0	1	0	1	E	1	0	0	0	1	0	1	e	1	1	0	0	1	0	1
&	0	1	0	0	1	1	0	F	1	0	0	0	1	1	0	f	1	1	0	0	1	1	0
′	0	1	0	0	1	1	1	G	1	0	0	0	1	1	1	g	1	1	0	0	1	1	1
(	0	1	0	1	0	0	0	H	1	0	0	1	0	0	0	h	1	1	0	1	0	0	0
)	0	1	0	1	0	0	1	I	1	0	0	1	0	0	1	i	1	1	0	1	0	0	1
*	0	1	0	1	0	1	0	J	1	0	0	1	0	1	0	j	1	1	0	1	0	1	0
+	0	1	0	1	0	1	1	K	1	0	0	1	0	1	1	k	1	1	0	1	0	1	1
,	0	1	0	1	1	0	0	L	1	0	0	1	1	0	0	l	1	1	0	1	1	0	0
−	0	1	0	1	1	0	1	M	1	0	0	1	1	0	1	m	1	1	0	1	1	0	1
.	0	1	0	1	1	1	0	N	1	0	0	1	1	1	0	n	1	1	0	1	1	1	0
/	0	1	0	1	1	1	1	O	1	0	0	1	1	1	1	o	1	1	0	1	1	1	1
0	0	1	1	0	0	0	0	P	1	0	1	0	0	0	0	p	1	1	1	0	0	0	0
1	0	1	1	0	0	0	1	Q	1	0	1	0	0	0	1	q	1	1	1	0	0	0	1
2	0	1	1	0	0	1	0	R	1	0	1	0	0	1	0	r	1	1	1	0	0	1	0
3	0	1	1	0	0	1	1	S	1	0	1	0	0	1	1	s	1	1	1	0	0	1	1
4	0	1	1	0	1	0	0	T	1	0	1	0	1	0	0	t	1	1	1	0	1	0	0
5	0	1	1	0	1	0	1	U	1	0	1	0	1	0	1	u	1	1	1	0	1	0	1
6	0	1	1	0	1	1	0	V	1	0	1	0	1	1	0	v	1	1	1	0	1	1	0
7	0	1	1	0	1	1	1	W	1	0	1	0	1	1	1	w	1	1	1	0	1	1	1
8	0	1	1	1	0	0	0	X	1	0	1	1	0	0	0	x	1	1	1	1	0	0	0
9	0	1	1	1	0	0	1	Y	1	0	1	1	0	0	1	y	1	1	1	1	0	0	1
:	0	1	1	1	0	1	0	Z	1	0	1	1	0	1	0	z	1	1	1	1	0	1	0
;	0	1	1	1	0	1	1	[	1	0	1	1	0	1	1	{	1	1	1	1	0	1	1
<	0	1	1	1	1	0	0	\	1	0	1	1	1	0	0	|	1	1	1	1	1	0	0
=	0	1	1	1	1	0	1	]	1	0	1	1	1	0	1	}	1	1	1	1	1	0	1
>	0	1	1	1	1	1	0	^	1	0	1	1	1	1	0	~	1	1	1	1	1	1	0
?	0	1	1	1	1	1	1	—	1	0	1	1	1	1	1	delete	1	1	1	1	1	1	1

Lab

LAB OVERVIEW

- Structure ...
  - There are four experiments in this course
  - Circuits analyzed/designed in each experiment are simulated using the software package ...
    - LogicWorks 4 ... OR ...
    - LogicWorks 5

LAB OVERVIEW

- Circuits can be analyzed/designed at ...
  - Home ... OR ... at UMass Labs
    - In the computer laboratory (BL - 420) ... OR ...
    - Engineering Laboratory (EB - 321)
      - Either LogicWorks 4 or 5 is available

Access to Labs

- To gain access to BL-420 and EB-321, which has LogicWorks4 and 5 installed
  - You will need to have an access cards
  - Access cards are given out on the South Campus at Access Services
    - Arrangements are done through the Continuing Education Office
  - Then I will need your UMS# (will be on your Access Card)
  - I will then send your name and UMS# to the Room POC

LAB OVERVIEW

- A report is required for each experiment
  - Detailed requirements will be provided
- You are also required to wire a given sequential circuit

LAB OVERVIEW

- Labs to be performed are ...
  - Experiment 1 ... Analysis of Digital Circuits
  - Experiment 2 ... Two-level Circuits
  - Experiment 3 ... Binary Code Converter
  - Experiment 4 ... Design with Decoders and Multiplexers

LAB OVERVIEW

- Lab Policies ...

1. All experiments in this course should be done independently
2. Reports should be submitted on the due date
   - Points will be deducted for late lab reports
3. Circuits that are not designed according to requirements will not be accepted
4. Additional report and design requirements are described in the laboratory notes

LAB OVERVIEW

- Mixture of breadboarding and Simulation software ... there are benefits to both

- Benefits of hands-on labs/breadboarding ...
  - Use of ...
    - Components
    - Test equipment
      - Knowledge of Test equipment is a foundation for hardware troubleshooting
  - ** Learn troubleshooting techniques
  - ** Will greatly enhance the class material
  - Solving Lab Problems will enforce the course material

LAB OVERVIEW

- Basic lab knowledge/techniques
  - Use of a breadboard
  - Learn the identification systems for components
    - Resistors
    - Capacitors
    - Integrated circuits
  - Application of data sheets

LAB OVERVIEW

- Benefits of LogicWorks Simulation software ...
  - An interactive circuit design tool intended for teaching and learning digital logic
  - Gives you the power, speed and flexibility to create and test an unlimited number of circuit elements on-screen
  - You can study advanced concepts much more quickly and clearly using on-screen simulation than you can by spending time wiring up expensive and damage-prone parts in a lab

- **Problems encountered during lab performance ...**
  - Knowledge gained from troubleshooting

LAB OVERVIEW

- Lab grade ...
  - Lab Report Content 100 points
    - Cover page, folder, disk
    - Conforms to requirements
    - Technical adequacy
  - Adjustment to grade (deductions ... up to 30 points)
    - Neatness and legibility (-10 points)
    - Templates (-20 points)
    - Late submitted reports (up to -30 points)

Additional Requirements - Laboratory Reports

1. Lab Reports
   - Report form for each lab will be available on the course web pages

   - Electronic report submission is an alternative to hard copies ...
     - PDF format only
     - Sent via email NLT than 11:59 PM on the due date

   - Send your schematics to me via email

2. Write your report using the given report template

3. A cover page (second page of report template) is required

Additional Requirements - Laboratory Reports

4. Include a schematic diagram for each of your designs in the report (see report template)

   - Use the given template for the schematic diagram to draw your diagram
   - Report will not be accepted if the schematic template is not used or if the template is changed in either contents or location, or in both
   - Twenty (out of 100) points will be deducted after the schematic diagram is redrawn using the template and without any change of the template
   - Remember to fill in your name

Additional Requirements - Laboratory Reports

5. Email your schematic diagram(s)
6. Include a grade sheet for each report

Lab Materials

- Lab requirements and booklets will be provided in the coming weeks

Questions?

Next Week ...

Next Week Topics

- Boolean Algebra ...
  - Chapter 2 ... Boolean Algebra
    - Pages 27 - 55

Home Work

Homework

1. Send an email with your email address or addresses (for class distribution list)

2. Send me your UMS# (will be on your Access Card) so I can get access to BL-420 and EB-321 (computer labs), if you currently do not have access

3. Read ...
    - Chapter 1 ... pages 1 - 23

    - Chapter 2 ... pages 27 – 55

Homework

4. Solve the following Chapter 1 problems ...

   - 1.1 ... (a)
   - 1.2 ... (b)
   - 1.5 ... (a), (b), (c), and (d)
   - 1.6 ... (a)
   - 1.7 ... (a) and (d)
   - 1.8
   - 1.9
   - 1.16 ... (b)

References

1. None