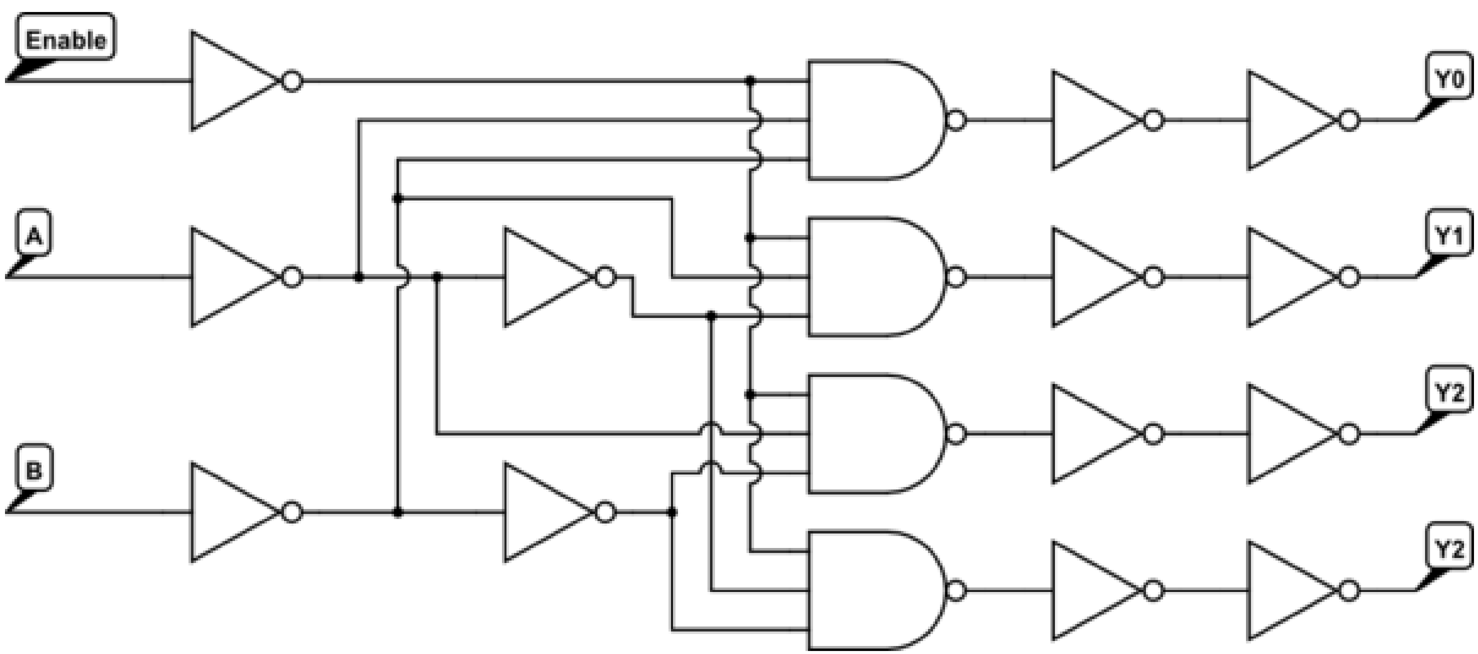
ECE 1551 Digital Logic

Fall 2024

TR 3:30 - 4:45 PM

Dr. S. Kozaitis

Introduction



This Course is About Hardware (dedicated circuits that do one thing)

Topics

Number systems

- Binary numbers (computers only understand binary)
- How to represent binary numbers in other number systems
- Difference between a number system and a code

Boolean algebra

- Used to describe the function of digital circuits
- Different representations of the same function

Combinational logic

- Basic and conventional devices to perform popular functions – adder ...

Sequential circuits (state machines)

- Those with small a number of memory elements - counter ...

Combinational and sequential circuits can be implemented with small processors such as Arduinos (general purpose) - ECE 1552

Syllabus

ECE 1551 Digital Logic
Fall 2024
Tuesday, Thursday 3:30-4:45pm
Location: 118 OEC

Instructor Name: Kozaitis

Office Location: 344EC

Office Hours: MW 2-4pm

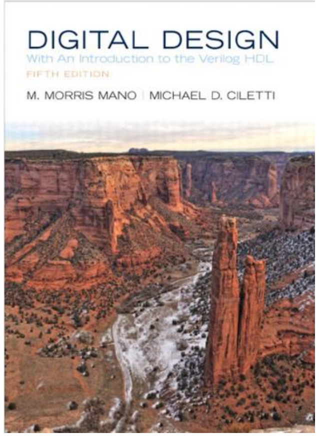
Phone: x7312

Email: kozaitis@fit.edu Use "ECE 1551" somewhere in the subject line.

Course Objectives

1. The student will be able to use different number systems.
2. The student will be able to analyze and design digital circuits.
3. The student will be able to design logic circuits and perform simulations.

Required Texts / Materials:
M. M. Mano, and M. D. Ciletti, Digital Design, (5th ed.) Pearson Prentice-Hall: New Jersey (2013). (Most students have found it free online)



Required Training (if applicable):
None

Grading Policy:

Tests	40%
Lab	20%
Homework	10%
Final	30%

No make-up tests

Attendance Policy:
None

Where to Find Extra Help:
https://www.allaboutcircuits.com/textbook/digital/

Academic Honesty Definitions & Procedures: Is located in the student handbook at https://www.fit.edu/policies/student-handbook/standards-and-policies/academic-honesty/

Title IX Statement: The university's Title IX policy is available at https://www.fit.edu/policies/title-ix/

Title IX of the Education Amendments of 1972 is a federal civil rights law that prohibits discrimination on the basis of sex in federally funded education programs and activities. Florida Institute of Technology policy also prohibits discrimination on the basis of sex.

Florida Tech faculty are committed to helping create a safe learning environment for all students that is free from all forms of discrimination and sexual harassment, including sexual assault, domestic violence, dating violence, and stalking. If you, or someone you know, have experienced or is experiencing any of these behaviors, know that help and support are available.

Florida Tech strongly encourages all members of the community to take action, seek support, and report any incident of sexual harassment or gender discrimination to Fanak Baarmand, Title IX Coordinator at 321-674-8885 or fbaarman@fit.edu.

Please note that as your professor, I am required to report any incidents to the Title IX Coordinator. If you wish to speak to an employee who does not have this reporting responsibility, please contact the Student Counseling Center at 321-674-8050.

Academic Accommodations: Florida Tech is committed to equal opportunity for persons w/disabilities in the participation of activities operated/sponsored by the university. Therefore, students w/documented disabilities are entitled to reasonable educational accommodations. The Office of Accessibility Resources (OAR) supports students by assisting w/accommodations, providing recommended interventions, and engaging in case management services. It is the student's responsibility to make a request to OAR before any accommodations can be approved/implemented. Also, students w/approved accommodations are encouraged to speak w/the course instructor to discuss any arrangements and/or concerns relating to their accommodations for the class. Office of Accessibility Resources (OAR): Telephone: 321-674-8285 / Email: accessibilityresources@fit.edu Website: https://www.fit.edu/accessibility-resources

Recording Disclosure: Portions of this course will be recorded and available to course participants. Students are subject to having their images and voices recorded during certain activities for the purpose of self-evaluation, peer-evaluation, presentations, assessments, and/or discussions. Course participants should have no expectation of privacy regarding their participation in classroom presentations, discussions, or peer activities.

Syllabus

Weekly Subject Matter and Assignment Schedule (subject to change):

	Weekly Topic	Assignment
Week 1	Number Systems Secs. 1.2-1.4	
Week 2	Addition/subtraction Sec. 1.6-1.7	HW Chap 1/Lab1
Week 3	Boolean algebra Sec. 2.4,2.5,2.6,2.8	HW Chap 2/Lab2
Week 4	DeMorgan's Theorem/K-maps Secs. 2.5,3.2,3.3	HW Chap 3/Lab3
Week 5	K-maps Secs,3.3,3.5/review	
Week 6	Test 1/VHDL	Lab 4
Week 7	Multiplexers/Comparators Secs. 4.11, 4.8	
Week 8	Adders Sec. 4.5/ALU	Lab 5
Week 9	Decoders/Encoders/Parity Sec. 4.9, 4.4, 4.10, 3.8	HW Chap 4, Lab 6
Week 10	Combinational Logic Problems/Test 2	
Week 11	Flip-flops/ Analysis of flip-flop circuits Secs. 5.1-5.4	Lab 7
Week 12	Analysis/Design of flip-flop circuits Secs. 5.4, 5.5	HW Chap 5
Week 13	Counters Secs. 5.8, 6.4	Lab 8
Week 14	Test3/Registers Secs. 6.1, 6.2, 6.5	(Lab 9)
Week 15	Review	

FINAL Information:

Cumulative

Thursday December 11th , 3:30-5:30pm

Material Covered (subject to change):

- Number systems (Secs. 1.2 – 1.6, sections are from 5th ed. of reference text)
  - Decimal, binary, octal and hexadecimal numbers
  - Signed numbers, addition and subtraction with signed numbers
- Boolean algebra and logic gates (Secs. 2.2-2.9)
  - Definitions, basic theorems, Boolean functions
  - DeMorgan's Theorem
  - Other logic operations, gates, integrated circuits
  - Minterms
- Function minimization, K-maps (Secs. 3.2, 3.3, 3.5)
  - Three and four variable K-map method to minimize functions
  - "Don't care" variables
- Combinational logic (Secs. 4.11, 4.8, 4.5, 4.9,4.10,3.8,4.4)
  - Multiplexers
  - Magnitude comparators
  - Adders
  - Decoders
  - Encoders
  - Parity checker
- Introduction to VHDL (not in textbook)
  - Levels of simulation
  - Defining input and outputs
  - Components
- Sequential Circuits (Secs., 5.1, 5.2, 5.3, 5.4, 5.5, 5.8)
  - Flip-flops and latches (SR, D and JK)
  - Characteristic tables
  - State table
  - State diagram
  - Flip-flop input equations
  - Analysis of flip-flop circuits
  - Design of flip-flop circuits
- Registers and counters (Secs., 6.1, 6.2, 6.4)
  - Shift registers, SISO, PIPO, SIPO, PISO
  - Synchronous counters

Lab

The lab section has three objectives.

- Make you familiar with the process of designing modern integrated circuits.
- Have you use a professional-level design tool.
- Introduce the language VHDL.

Software

- We will use a software package called Quartus Prime (Lite Edition) made by Intel Inc. It is free and does not require a license, and you can download it to a laptop or home computer.

- Quartus Prime is a package someone would use in industry to design a large-scale digital system. It has much functionality, far more than we need for this course. The Lite Edition is the same as the professional version except much of that functionality is not available.

- Keep in mind you are only using Quartus Prime and VHDL to expose yourself to modern tools.

- Unfortunately, we have had several students experience problems installing or simulating circuits. If you can't things to work properly, you can use FIT's remote server from anywhere, where Quartus Prime is installed and works properly.

Number Systems

- Decimal numbers (1.2)
- Binary numbers (1.2)
- Binary-to-decimal conversion (1.3)
- Decimal-to-binary conversion (1.3)
- Octal numbers (1.4)
- Hexadecimal numbers (1.4)
- Binary addition and multiplication (1.2)

1.2 Decimal Numbers

The decimal number system has only 10 symbols that represent quantity: 0 through 9. If you want to represent a value greater than 9, you must use more than one digit.

The column weights of decimal numbers are powers of ten that increase from right to left beginning with $10^0 = 1$:

$$\ldots 10^5 \ \ 10^4 \ \ 10^3 \ \ 10^2 \ \ 10^1 \ \ 10^0 .$$

For fractional decimal numbers, the column weights are negative powers of ten that decrease from left to right:

$$\ldots \ 10^2 \ \ 10^1 \ \ 10^0 . \ 10^{-1} \ \ 10^{-2} \ \ 10^{-3} \ \ldots$$

1.2 Decimal Numbers

Decimal numbers can be expressed as the sum of the products of each digit times the column value for that digit. Thus, the number 9240 can be expressed as

$$(9 \times 10^3) + (2 \times 10^2) + (4 \times 10^1) + (0 \times 10^0)$$

Express the number 480.52 as the sum of values of each digit.

$$480.52 = (4 \times 10^2) + (8 \times 10^1) + (0 \times 10^0) + (5 \times 10^{-1}) + (2 \times 10^{-2})$$

1.2 Binary Numbers

The binary number system is used in digital systems. Binary has only two symbols that are used to represent quantities: 0 and 1.

The column weights of binary numbers are **powers of two** that increase from right to left beginning with $2^0 = 1$:

$$\ldots\ 2^5\ \ 2^4\ \ 2^3\ \ 2^2\ \ 2^1\ \ 2^0.$$

For fractional binary numbers, the column weights are negative powers of two that decrease from left to right:

$$\ldots\ 2^2\ \ 2^1\ \ 2^0\,.\ 2^{-1}\ \ 2^{-2}\ \ 2^{-3}\ \ 2^{-4}\ \ldots$$

1.2 Binary Numbers

The positional weights for binary numbers are assigned as shown below.

TABLE 2–2 • Binary weights.

POSITIVE POWERS OF TWO (WHOLE NUMBERS)									NEGATIVE POWERS OF TWO (FRACTIONAL NUMBER)					
2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}
256	128	64	32	16	8	4	2	1	1/2 0.5	1/4 0.25	1/8 0.125	1/16 0.625	1/32 0.03125	1/64 0.015625

1.2 Binary Numbers

A binary counting sequence for numbers from zero to fifteen is shown.

Notice the pattern of zeros and ones in each column.

TABLE 2–1

DECIMAL NUMBER	BINARY NUMBER			
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1
	2^3	2^2	2^1	2^0

1.3 Binary-to-Decimal Conversion

The decimal equivalent of a binary number can be determined by adding the column values of all of the bits that are 1 and discarding all of the bits that are 0.

Convert the binary number 100101.01 to decimal.

Start by writing the column weights; then add the weights that correspond to each 1 in the number.

2^5	2^4	2^3	2^2	2^1	$2^0.$	2^{-1}	2^{-2}	
32	16	8	4	2	1	0.5	0.25	
1	**0**	**0**	**1**	**0**	**1.**	**0**	**1**	
32			4		1		0.25	= **37.25**

1.3 Binary-to-Decimal Conversion

10011011_2

$2^7 \quad 2^6 \quad 2^5 \quad 2^4 \quad 2^3 \quad 2^2 \quad 2^1 \quad 2^0$

128, 64, 32, 16, 8, 4, 2, 1

128	64	32	16	8	4	2	1
1	0	0	1	1	0	1	1

$128 + 0 + 0 + 16 + 8 + 0 + 2 + 1$

$= 155$

1.3 Binary-to-Decimal Conversion

The decimal equivalent of a binary number can be determined by adding the column values of all of the bits that are 1 and discarding all of the bits that are 0.

Convert the binary number 100101.01 to decimal.

Start by writing the column weights; then add the weights that correspond to each 1 in the number.

2^5	2^4	2^3	2^2	2^1	2^0.	2^{-1}	2^{-2}	
32	16	8	4	2	1	0.5	0.25	
1	**0**	**0**	**1**	**0**	**1.**	**0**	**1**	
32			4		1		0.25	= **37.25**

1.3 Binary-to-Decimal Conversions
Decimal-to-Binary Conversions

2^6	2^5	2^4	2^3	2^2	2^1	2^0
64	32	16	8	4	2	1
1	**0**	**0**	**0**	**0**	**0**	**1**

64+0 + 0 + 0 + 0 + 0 + 1 = 65

2^6	2^5	2^4	2^3	2^2	2^1	2^0
64	32	16	8	4	2	1
1	**0**	**1**	**1**	**0**	**0**	**0**

64+0 + 16 + 8 + 0 + 0 + 0 = 88

2^6	2^5	2^4	2^3	2^2	2^1	2^0
64	32	16	8	4	2	1
0	**0**	**0**	**1**	**1**	**0**	**0**

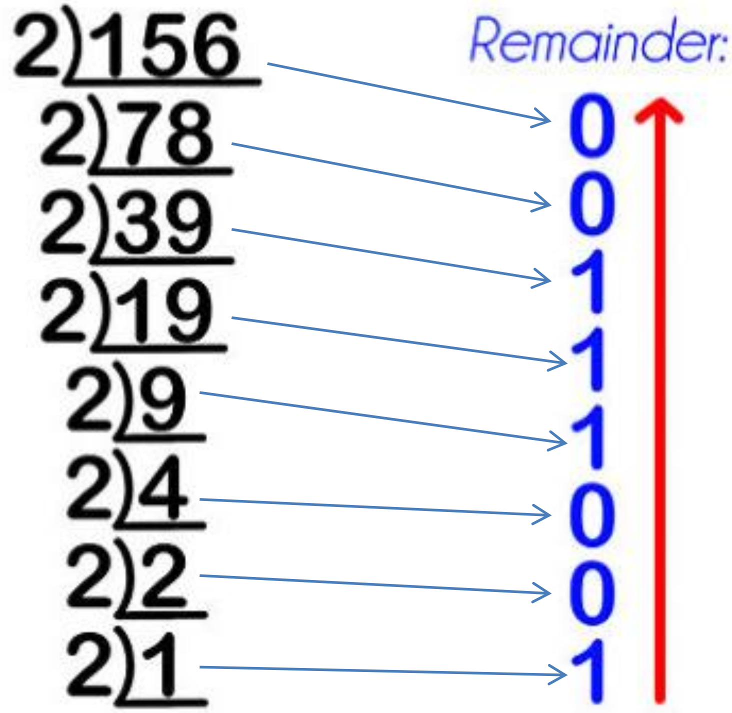
0 + 0 + 0 + 8 + 4 + 0 + 0 = 12

2^6	2^5	2^4	2^3	2^2	2^1	2^0
64	32	16	8	4	2	1
1	**1**	**0**	**0**	**1**	**1**	**0**

64+ 32+ 0 + 0 + 4 + 2 + 0 = 102

1.3 Decimal-to-Binary Conversions

Example: 156_{10}



$$\boxed{156_{10} = 10011100_2}$$

1.3 Decimal-to-Binary Conversions

Decimal	2^4	2^3	2^2	2^1	2^0	
	16	8	4	2	1	
6			1	1	0	$(6)_{10} = (110)_2$
11		1	0	1	1	$(11)_{10} = (1011)_2$
30	1	1	1	1	0	$(30)_{10} = (11110)_2$
23	1	0	1	1	1	$(23)_{10} = (10111)_2$

more formal way (useful for larger numbers)

$$\begin{array}{r|l l}
2 & 6 & 0 \\
2 & 3 & 1 \\
2 & 1 & 1 \\
 & 0 &
\end{array}
\qquad
\begin{array}{r|l l}
2 & 11 & 1 \\
2 & 5 & 1 \\
2 & 2 & 0 \\
2 & 1 & 1 \\
 & 0 &
\end{array}
\qquad \ldots\ldots$$

$$\begin{array}{r|l l}
2 & 211 & 1 \\
2 & 105 & 1 \\
2 & 52 & 0 \\
2 & 26 & 0 \\
2 & 13 & 1 \\
2 & 6 & 0 \\
2 & 3 & 1 \\
2 & 1 & 1 \\
 & 0 &
\end{array}$$

$(211)_2 = (11010011)_2$

1.4 Octal Numbers

Octal is also a weighted number system. The column weights are powers of 8, which increase from right to left.

Column weights	8^3	8^2	8^1	8^0
	512	64	8	1

Express 3702_8 in decimal (base 10).

Start by writing the column weights:

512	64	8	1
3	7	0	2

(3x512) + (7x64) +(0x8) +(2x1) = $\mathbf{1986_{10}}$

24_8 = (2x8) + (4x1) = $\mathbf{20_{10}}$

217_8 = (2x64) + (1x8) + (7x1) = $\mathbf{143_{10}}$

Decimal	Octal	Binary
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	10	1000
9	11	1001
10	12	1010
11	13	1011
12	14	1100
13	15	1101
14	16	1110
15	17	1111

1.4 Octal Numbers (binary to octal)

Binary number can easily be converted to octal by grouping bits 3 at a time and writing the equivalent octal character for each group.

Express 1001011000001110_2 in octal:

Group the binary number by 3-bits starting from left to right. Thus, 001 011 000 001 110 = **13016_8**

101 111 000 111 100 = **57074_8**

1 001 011 000 001 110 = **113016_8**

11 000 010 000 000 100 = **302004_8**

Decimal	Octal	Binary
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	10	1000
9	11	1001
10	12	1010
11	13	1011
12	14	1100
13	15	1101
14	16	1110
15	17	1111

1.4 Hexadecimal Numbers

Hexadecimal is a weighted number system. The column weights are powers of 16, which increase from right to left.

Column weights	16^3	16^2	16^1	16^0
	4096	256	16	1

Express $1A2F_{16}$ in decimal.

Start by writing the column weights:

4096	256	16	1
1	A	2	F_{16}

$(1x4096) + (10x256) + (2x16) + (15x1) = \mathbf{6703_{10}}$

$24_{16} = (2x16) + (4x1) = \mathbf{36_{10}}$

$2C7_{16} = (2x256) + (12x16) + (7x1) = \mathbf{711_{10}}$

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

1.4 Hexadecimal Numbers

(binary to hex)

Large binary number can easily be converted to hexadecimal by dividing it into 4-bit groups and converting each into its equivalent hexadecimal character.

Express 1001011000001110_2 in hexadecimal:

Group the binary number by 4-bits starting from the right. Thus, 1001 0110 0000 1110 = 960E

1000 1111 1111 1100 = **$8FFC_{16}$**

1111 1111 1110 1110 = **$FFEE_{16}$**

11 1111 0001 1111 1111 1110 1101 1111= **$3F1FFEDF_{16}$**

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

1.4 Hexadecimal Numbers

(binary to hex)

Group Binary in sets of 4

0010 0010

2 2

can add leading 0's

$(00100010)_2 = (22)_{16}$

0001 1011 1101 0011 $= (1BD3)_{16}$

1 B D 3

10	A
11	B
12	C
13	D
14	E
15	F

Hex to Binary

10FF = 0001 0000 1111 1111

23CA = 0010 0011 1100 1010

Number Systems

Decimal – base 10, good for humans

Binary - base 2, good for digital circuits

Hexadecimal - base 16, good for humans to represent binary

Should be able to convert between these systems

Codes

Gray code – only one bit changes per digit, used in industrial settings

ASCII – code for keyboard

BCD – binary coded decimal, used for displays

Binary to decimal conversion

2^5	2^4	2^3	2^2	2^1	2^0		
			1	0	0	$\to 4$	$(100)_2 = (4)_{10}$
		1	0	1	0	$\to 8+2$	$(1010)_2 = (10)_{10}$
	1	0	1	1	0	$\to 16+4+2$	$(10110)_2 = (22)_{10}$

Decimal to binary

	2^4	2^3	2^2	2^1	2^0	
$8 \to$		1	0	0	0	$(8)_{10} = (1000)_2$
$11 \to$		1	0	1	1	$(11)_{10} = (1011)_2$
$9 \to$		1	0	0	1	$(9)_{10} = (1001)_2$

$2 \,|\, 8 \to 0$
$2 \,|\, 4 \to 0$
$2 \,|\, 2 \to 0$
$2 \,|\, 1 \to 1$
0

$2 \,|\, 11 \to 1$
$2 \,|\, 5 \to 1$
$2 \,|\, 2 \to 0$
$2 \,|\, 1 \to 1$
0

hexadecimal to binary

$(1\ A)_{16}$ → convert digits independently with 4 binary digits

1 A
0001 1010

$(7\ F)_{16}$ →

7 F
0111 1111

Binary to hex

1001110 group 4 bits at a time → $(4\ D)_{16}$

10100011 → $(A\ 3)_{16}$

decimal to hex

de decimal → binary → hex

$(26)_{10}$ → 11010 → $(1A)_{16}$

$(31)_{10}$ → 11111 → $(1\ F)_{16}$

Number Systems

- 1’s complement (1.6)
- 2’s complement (1.6)
- Signed numbers(1.6)
- Addition with signed numbers (1.6)

1.6 Binary Addition

The rules for binary addition are

0 + 0 = 0	Sum = 0, carry = 0
0 + 1 = 1	Sum = 1, carry = 0
1 + 0 = 1	Sum = 1, carry = 0
1 + 1 = 10	Sum = 0, carry = 1

When an input carry = 1 due to a previous result, the rules are

1 + 0 + 0 = 01	Sum = 1, carry = 0
1 + 0 + 1 = 10	Sum = 0, carry = 1
1 + 1 + 0 = 10	Sum = 0, carry = 1
1 + 1 + 1 = 11	Sum = 1, carry = 1

1.6 Binary Addition

carry

	1
2	010
+ 3	011
	101 = 5
	1 1 1
7	0111
+ 3	0011
	1010 = 10
12	1100
+ 10	1010
	10110 = 22
	1 1
7	111
+ 7	111
	1110 = 14

1.6 Binary Multiplication

The rules for binary multiplication are:

$0 \times 0 = 0$

$0 \times 1 = 0$

$1 \times 0 = 0$

$1 \times 1 = 1$

Perform the following binary multiplications:

(a) 11 × 11 **(b)** 101 × 111

SOLUTION

(a)

$$\begin{array}{lr} & 11 \\ & \times\, 11 \\ \hline \text{Partial} & 11 \\ \text{products} & +11 \\ \hline & \mathbf{1001} \end{array} \qquad \begin{array}{r} 3 \\ \times\, 3 \\ \hline 9 \end{array}$$

(b)

$$\begin{array}{lr} & 111 \\ & \times\, 101 \\ \hline & 111 \\ \text{Partial products} & 000 \\ & +111 \\ \hline & \mathbf{100011} \end{array} \qquad \begin{array}{r} 7 \\ \times\, 5 \\ \hline 35 \end{array}$$

1.6 Binary Multiplication

```
   2      010
 x 2      010
 ---     ----
   4      000
         010
        -----
         100
```

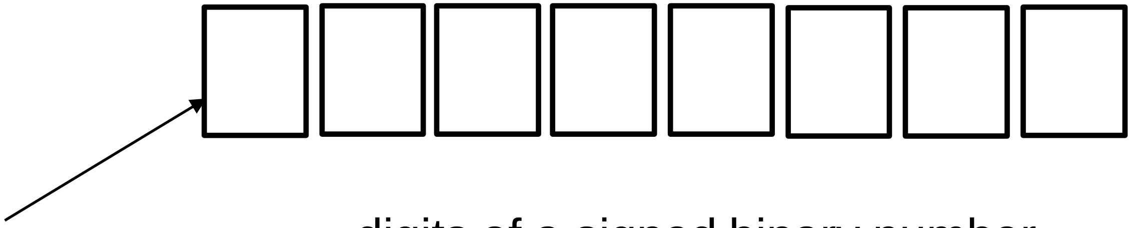
```
   3      011
 x3       011
 ---     ----
   9      011
         011
        -----
         1001
```

```
   7
 x 7         0111
 ---         0111
  49     1     1
            ------
         1  0  111
            111
          111
        --------
        110001
```

1.6 Signed numbers

Unsigned numbers – only positive numbers

Signed numbers – positive and negative



digits of a signed binary number

Sign bit
1 negative number
0 positive number

Negative numbers are stored in 2's complement notation

1.6 One's Complement

The 1's complement of a binary number is the number that is formed by inverting (complementing) all the digits.

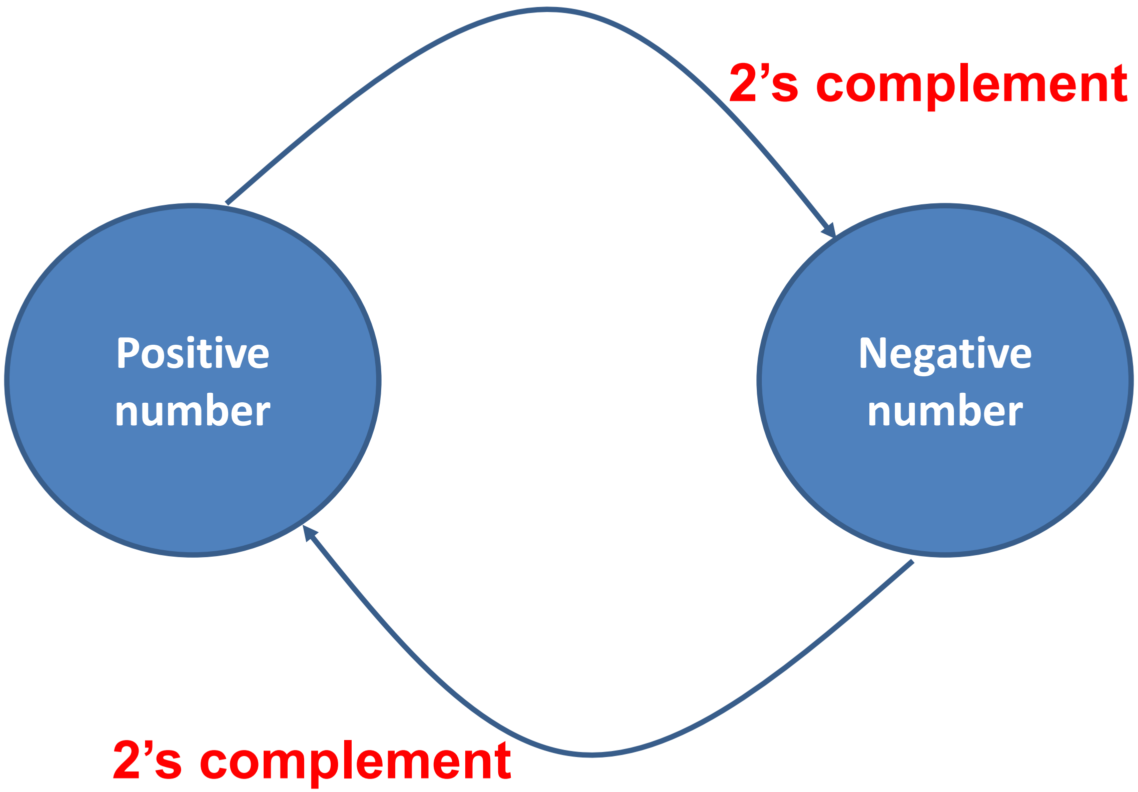
Binary number

1 1 0 0 1 0 1 0

1's complement

0 0 1 1 0 1 0 1

1.6 Two's Complement



1.6 Two's Complement

The 2's complement of a binary number is found by adding 1 to the 1's complement.

Easy way:

- Look at the binary number from right to left.
- Keep all digits the same until the first "1" is reached
- Complement all digits after the first "1"

Number -> 2's complement

- 01001 -> 10111
- 0100100000 -> 1011100000
- 0101010100 -> 1010101100
- 01110 -> 10010
- … 0001110 -> … 1110010

Adding leading 0's to a positive number doesn't change its value

Adding leading 1's to a 2's complement number doesn't change its value

1.6 Negative (Two's Complement) Binary Numbers

The most significant bit (MSB) in a signed number is the sign bit.

- Positive numbers are stored with a 0 for the sign bit)
- Negative numbers are stored with a 1 for the sign bit).

For example, the positive number 58 is written using 8-bits as 00111010

Sign bit

The number –58 is written as:

–58 = 11000110 (2's complement form)

Sign bit

2's complement notation is not equal to a 1 in front of the magnitude

1.6 Two's Complement Binary Numbers

0000011100100101 -> positive number

Sign bit
1 negative number
0 positive number

- Two's complement is found by adding 1 to the 1's complement.

- Look at the binary number from right to left.
- Keep all digits the same until the first "1" is reached
- Complement all digits after the first "1"

+ 1 = 00001
-1 = 11111

+16 = 010000
-16 = 110000

+12 = 01100 = 001100
-12 = 10100 = 110100

+15 = 01111 = 001111
-15 = 10001 = 110001

1.6 Negative (Two's Complement) Binary Numbers

Range of unsigned and signed numbers

unsigned integers

$(1111)_2 = (15)_{10}$

range of #'s is from

0 to $2^N - 1$

1111	15
⋮	
0010	2
0001	1
0000	0

$(0 \to 15)$

2's complement integers

largest # is 7 if we use a sign bit

range of #'s is from

-2^{N-1} to $2^{N-1} - 1$

0111	7
⋮	
0010	2
0001	1
0000	0
1111	-1
1110	-2
⋮	
1001	-7
1000	-8

$-8 \to +7$

↑ sign bit

1.6 Arithmetic Operations With Two's Complement Numbers

Rules for **subtraction**: form the 2's complement of the subtrahend and **add** the numbers. Discard any final carries. The result is in signed form.

$$
\begin{array}{rl} & 00011110 \quad (+30) \\ - & \underline{00001111} \quad -(15) \end{array}
\qquad
\begin{array}{rl} & 00001110 \quad (+14) \\ - & \underline{11101111} \quad -(-17) \end{array}
\qquad
\begin{array}{rl} & 11111111 \quad (-1) \\ - & \underline{11111000} \quad -(-8) \end{array}
$$

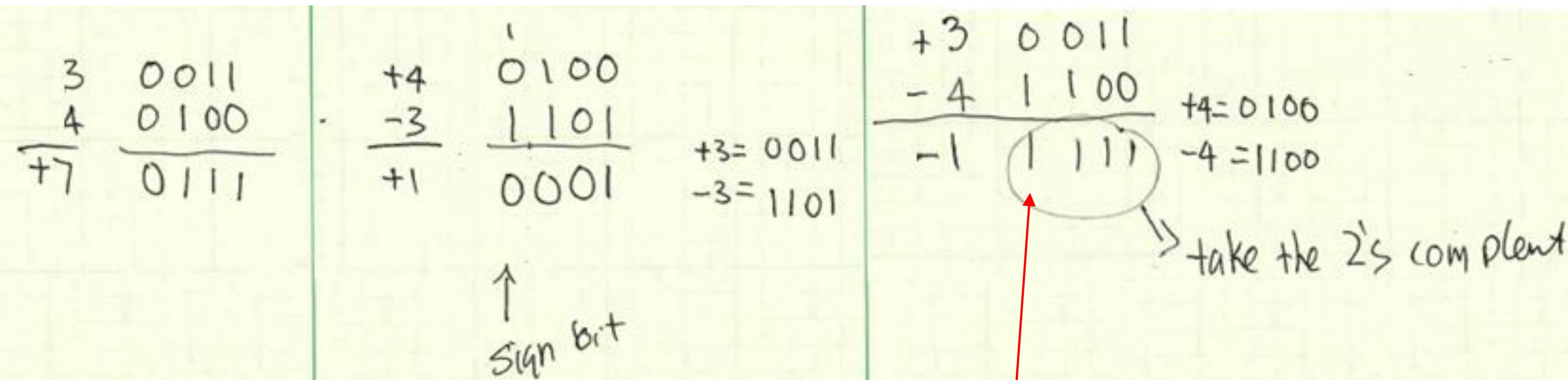
2's complement subtrahend and **add**:

$$
\begin{array}{r} 00011110 = +30 \\ \underline{11110001 = -15} \\ \cancel{1}00001111 = +15 \end{array}
\qquad
\begin{array}{r} 00001110 = +14 \\ \underline{00010001 = +17} \\ 00011111 = +31 \end{array}
\qquad
\begin{array}{r} 11111111 = -1 \\ \underline{00001000 = +8} \\ \cancel{1}00000111 = +7 \end{array}
$$

Discard carry

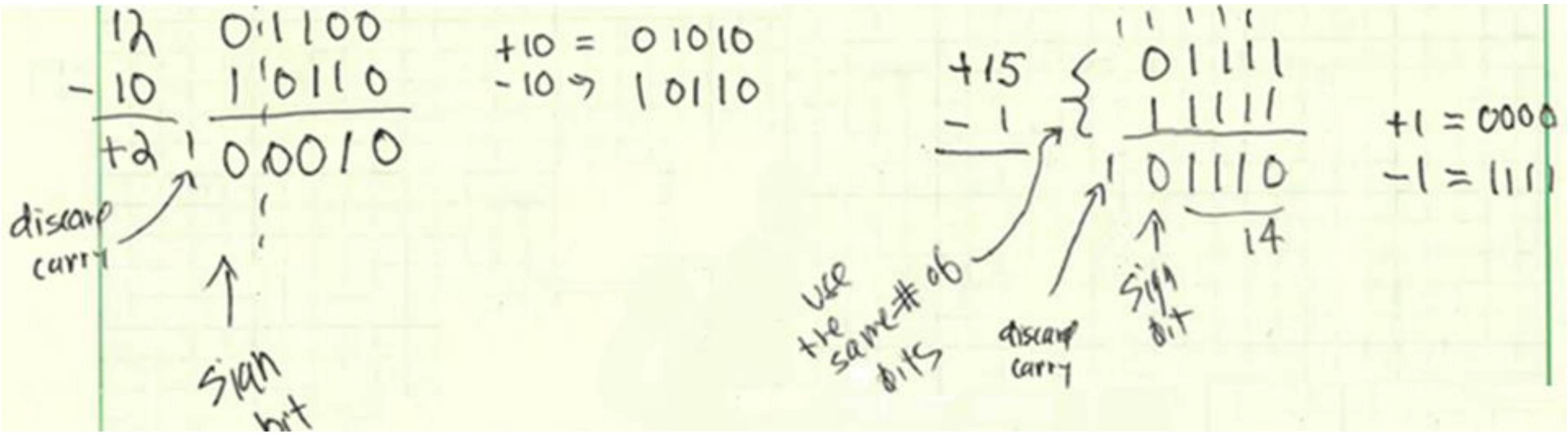
Discard carry

1.6 Subtraction using Two's Complement Binary Numbers



Make sure you use enough bits

1.6 Subtraction using Two's Complement Binary Numbers



1.6 Arithmetic Operations With Two's Complement Numbers

Note that if the number of bits required for the answer is exceeded, overflow will occur. This occurs only if both numbers have the same sign. The overflow is indicated by an incorrect sign bit.

Two examples are:

01000000 = +128	10000001 = –127
01000001 = +129	10000001 = –127
10000001 = ~~–126~~	Discard carry → ~~1~~00000010 = ~~+2~~

Wrong! The answer is incorrect because the sign bit has changed.

8-bit addition

8	00001000
−4	11111100
	$\boxed{00000100}$

↑ sign-bit

---

add 10 + 12 using binary 8-bit addition

10	sign bit 0:0001010
+12	+ 0:0001100
22	0:0010110

(Easy mistake) add 10 + 12 in binary

sign bit 0:1010

+ 0:1100

1:0110 $\neq 22$

↑ indicates a negative #

Didn't use enough bits

⇒ add leading zeros if not sure of the result

$$\begin{array}{r} 10 \\ -12 \\ \hline -2 \end{array}$$

sign bit

$$\begin{array}{r|l} 0 & 1010 \\ 1 & 0100 \\ \hline 1 & 1110 \end{array}$$

negative #

12 → 01100
-12 → 10100

11110 what # is it?
take the 2's complement

– 00010 = -2

11110 = - 00010 = - 2

don't write it this way

$$\begin{array}{r} -10 \\ +12 \\ \hline +2 \end{array}$$

$$\begin{array}{r|l} 1 & 0110 \\ 0 & 1100 \\ \hline 0 & 0010 \end{array} \rightarrow +2$$

discard carry

+10 = 01010
-10 = 10110

Hexadecimal Addition/Subtraction

Convert to binary

1A	→	00011010		
+2F		00101111	→	49
		01001001		

1A	→	00011010		
- 2F		11010001	→	EB
		11101011		(if negative)
				= -00010101
				= -15