CS 2110 - Lab 02

Datatypes

Monday, May 23, 2022

Lab Assignment: Canvas Quiz

1. Go to Quizzes on Canvas
2. Select Lab 02, password: **byte**
3. Get 100% to get attendance!
    a) Unlimited attempts
    b) Collaboration is **allowed**!
    c) Ask your TAs for help :)

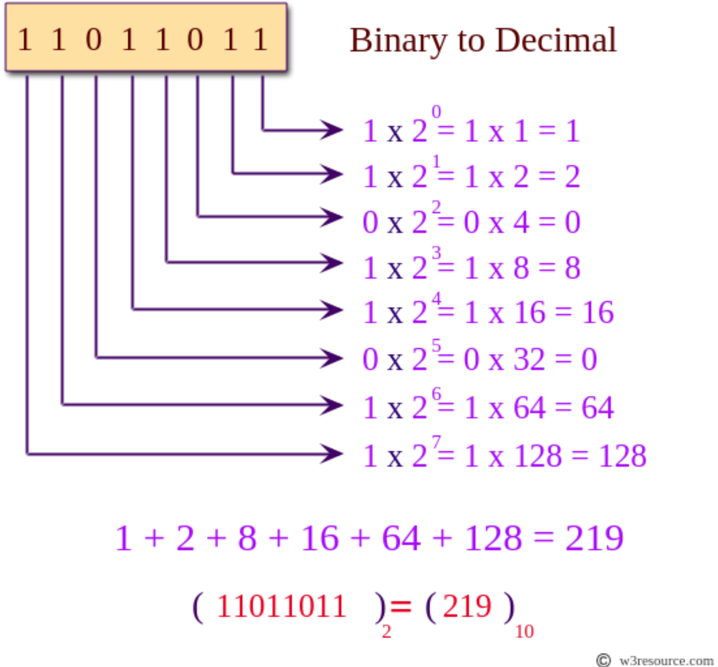
Homework 1

- Released!
- **Due 06/02 (Thursday) at 11:59 PM** (standard 24 hr grace period)
- Files available on Canvas
- Submit on Gradescope
  - **Double check that your grade on Gradescope is your desired grade!**

Topics

- Binary
- Octal
- Hexadecimal
- Signed integers
  - 2's complement
  - Negation
- Logical Operations
  - NOT, AND, OR, XOR, NAND, NOR
  - Shift
- BitVectors
  - Mask, Set, Clear
- ASCII

Convert Binary to Decimal



1 1 0 1 1 0 1 1

Binary to Decimal

$1 \times 2^0 = 1 \times 1 = 1$

$1 \times 2^1 = 1 \times 2 = 2$

$0 \times 2^2 = 0 \times 4 = 0$

$1 \times 2^3 = 1 \times 8 = 8$

$1 \times 2^4 = 1 \times 16 = 16$

$0 \times 2^5 = 0 \times 32 = 0$

$1 \times 2^6 = 1 \times 64 = 64$

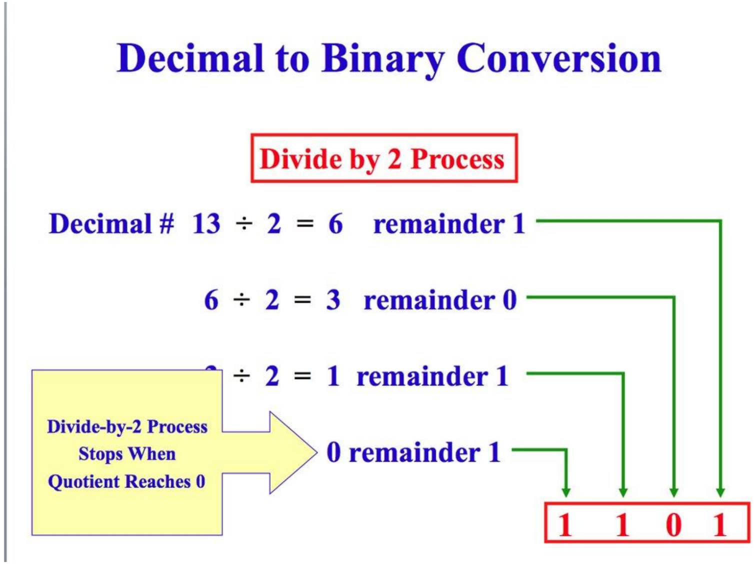
$1 \times 2^7 = 1 \times 128 = 128$

$1 + 2 + 8 + 16 + 64 + 128 = 219$

$(11011011)_2 = (219)_{10}$

© w3resource.com

Convert Decimal to Binary



Decimal to Binary Conversion

Divide by 2 Process

Decimal # 13 ÷ 2 = 6 remainder 1

6 ÷ 2 = 3 remainder 0

3 ÷ 2 = 1 remainder 1

0 remainder 1

1 1 0 1

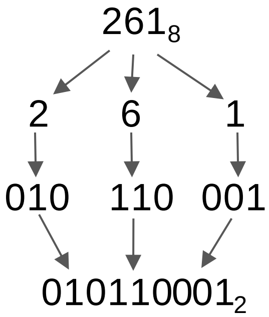
Divide-by-2 Process Stops When Quotient Reaches 0

Binary Addition

- Just like decimal addition
- 0 + 0 = 0, carry out = 0
- 0 + 1 = 1, carry out = 0
- 1 + 0 = 1, carry out = 0
- 1 + 1 = 0, carry out = 1

Octal to Binary

1. Split each digit up
2. Convert each digit to 3 bit binary
3. Combine each binary number
4. Think about how to do the reverse i.e Binary to Octal?



261_8

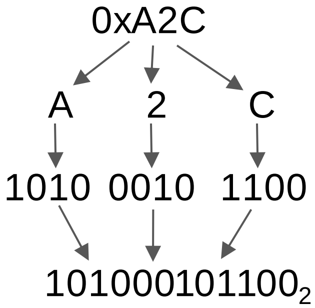
2 6 1

010 110 001

010110001_2

Hexadecimal to Binary

1. Split each digit up
2. Convert each digit to 4 bit binary
3. Combine each binary number
4. Reverse?



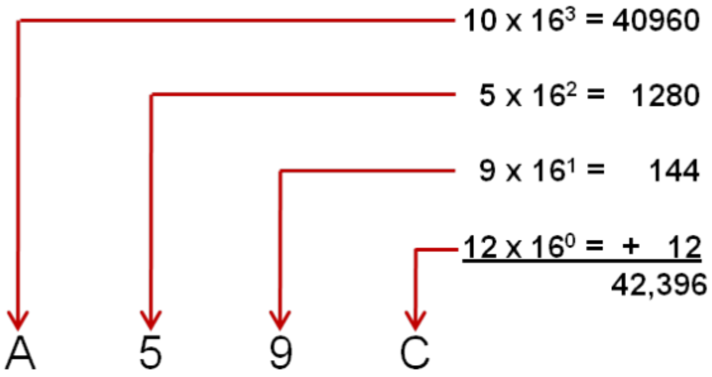
0xA2C

A 2 C

1010 0010 1100

101000101100_2

Hexadecimal to Decimal

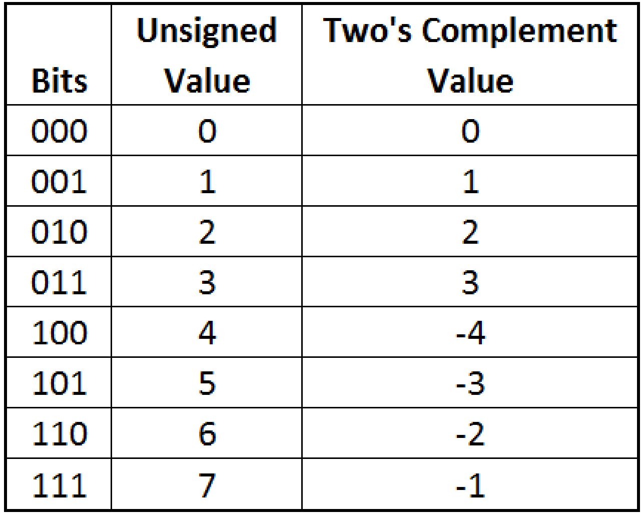


Signed vs Unsigned

Can unsigned binary represent negative numbers?

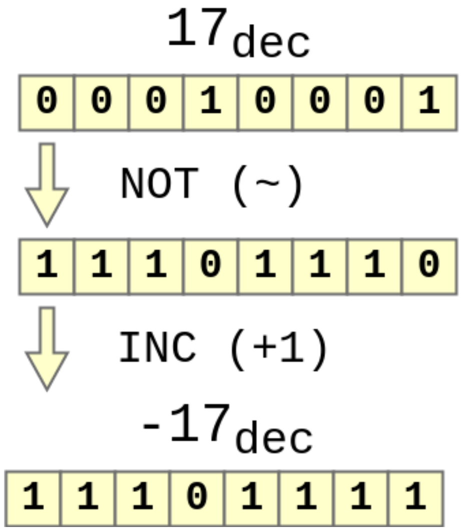
How do we represent negative numbers in binary?

2's Complement



Bits	Unsigned Value	Two's Complement Value
000	0	0
001	1	1
010	2	2
011	3	3
100	4	-4
101	5	-3
110	6	-2
111	7	-1

2's Complement Negation



17_{dec}

0	0	0	1	0	0	0	1

NOT (~)

1	1	1	0	1	1	1	0

INC (+1)

-17_{dec}

1	1	1	0	1	1	1	1

Two's complement negation:

```
-n == (~n)+1
```

Detecting Overflow

Unsigned overflow – did the addition carry out?

Example: 6 + 12 = 0110 + 1100 = (1)0010 = 2 (carry out dropped)

Signed overflow – did I get the wrong sign?

Two methods of detecting:

1. Adding two numbers with the same sign results in the opposite sign. Adding two numbers with opposite sign never overflows—why?
2. If the carry in and the carry out of the last bit are **different**

Example: 6 + 3 = 0110 + 0011 = 1001 = -7

Logical Operators

- NOT (Symbol : ~)
- AND (Symbol : &)
- OR (Symbol : |)
- XOR (Symbol : ^)

Note: A NOR B is NOT (A OR B) and NAND is NOT (A AND B)

Masking

Using the operators we just learned, how do we isolate or remove a specific part of a binary integer?

- To **SET** a bit means to make it a **1**
- To **CLEAR** a bit means to make it a **0**

Masking

Using the operators (OR, NOT, XOR, AND) we just went over, how can we...

- Extract the specified bits (make the rest all 0)
- Clear the specified bits (make *them* all 0)
- Set the specified bits
- Flip the specified bits

1101 0110 0100 1001

Shifting

- Left shifting (<<)
- Right shifting (signed and unsigned) (>>, >>>)

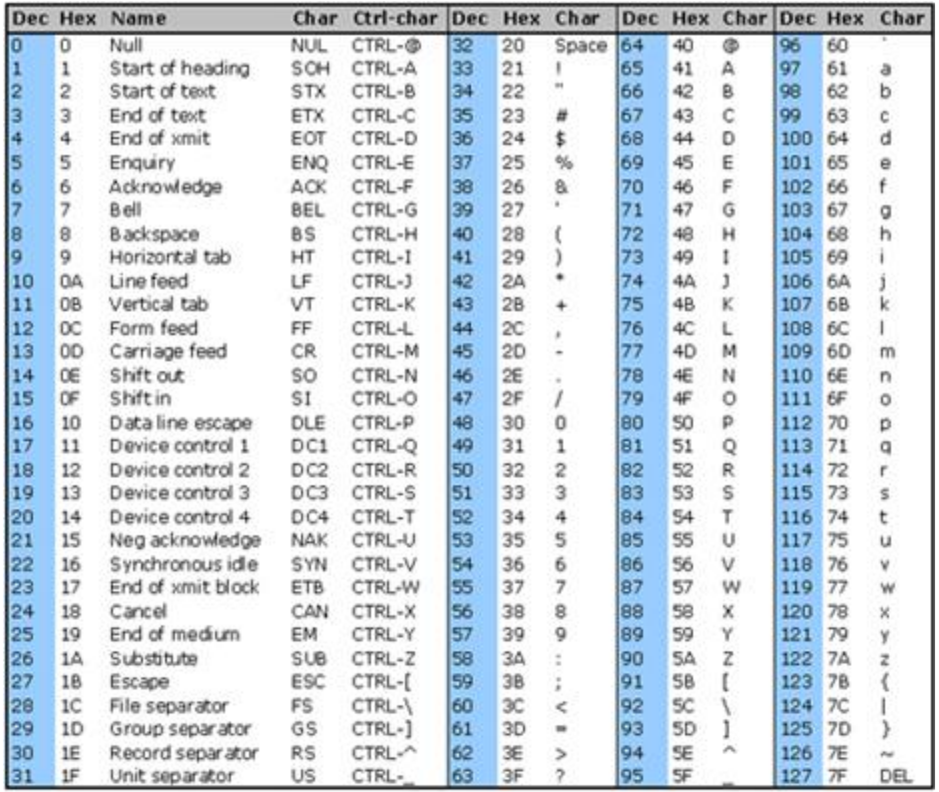
What are these operations equivalent to (in terms of addition, subtraction, multiplication, division...)?

How do we multiply 3 by 4 without using the multiply operator?
How do we divide 8 by 2 without using the divide operator?

ASCII

- ASCII : American standard code for information interchange
- Each character on your computer (a, z, Z, 1, $ etc.) is associated with a unique binary integer dictated by this code

ASCII Table:



Dec	Hex	Name	Char	Ctrl-char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	0	Null	NUL	CTRL-@	32	20	Space	64	40	@	96	60	`
1	1	Start of heading	SOH	CTRL-A	33	21	!	65	41	A	97	61	a
2	2	Start of text	STX	CTRL-B	34	22	"	66	42	B	98	62	b
3	3	End of text	ETX	CTRL-C	35	23	#	67	43	C	99	63	c
4	4	End of xmit	EOT	CTRL-D	36	24	$	68	44	D	100	64	d
5	5	Enquiry	ENQ	CTRL-E	37	25	%	69	45	E	101	65	e
6	6	Acknowledge	ACK	CTRL-F	38	26	&	70	46	F	102	66	f
7	7	Bell	BEL	CTRL-G	39	27	'	71	47	G	103	67	g
8	8	Backspace	BS	CTRL-H	40	28	(	72	48	H	104	68	h
9	9	Horizontal tab	HT	CTRL-I	41	29	)	73	49	I	105	69	i
10	0A	Line feed	LF	CTRL-J	42	2A	*	74	4A	J	106	6A	j
11	0B	Vertical tab	VT	CTRL-K	43	2B	+	75	4B	K	107	6B	k
12	0C	Form feed	FF	CTRL-L	44	2C	,	76	4C	L	108	6C	l
13	0D	Carriage feed	CR	CTRL-M	45	2D	-	77	4D	M	109	6D	m
14	0E	Shift out	SO	CTRL-N	46	2E	.	78	4E	N	110	6E	n
15	0F	Shift in	SI	CTRL-O	47	2F	/	79	4F	O	111	6F	o
16	10	Data line escape	DLE	CTRL-P	48	30	0	80	50	P	112	70	p
17	11	Device control 1	DC1	CTRL-Q	49	31	1	81	51	Q	113	71	q
18	12	Device control 2	DC2	CTRL-R	50	32	2	82	52	R	114	72	r
19	13	Device control 3	DC3	CTRL-S	51	33	3	83	53	S	115	73	s
20	14	Device control 4	DC4	CTRL-T	52	34	4	84	54	T	116	74	t
21	15	Neg acknowledge	NAK	CTRL-U	53	35	5	85	55	U	117	75	u
22	16	Synchronous idle	SYN	CTRL-V	54	36	6	86	56	V	118	76	v
23	17	End of xmit block	ETB	CTRL-W	55	37	7	87	57	W	119	77	w
24	18	Cancel	CAN	CTRL-X	56	38	8	88	58	X	120	78	x
25	19	End of medium	EM	CTRL-Y	57	39	9	89	59	Y	121	79	y
26	1A	Substitute	SUB	CTRL-Z	58	3A	:	90	5A	Z	122	7A	z
27	1B	Escape	ESC	CTRL-[	59	3B	;	91	5B	[	123	7B	{
28	1C	File separator	FS	CTRL-\	60	3C	<	92	5C	\	124	7C	|
29	1D	Group separator	GS	CTRL-]	61	3D	=	93	5D	]	125	7D	}
30	1E	Record separator	RS	CTRL-^	62	3E	>	94	5E	^	126	7E	~
31	1F	Unit separator	US	CTRL-_	63	3F	?	95	5F	_	127	7F	DEL

ASCII

Example:

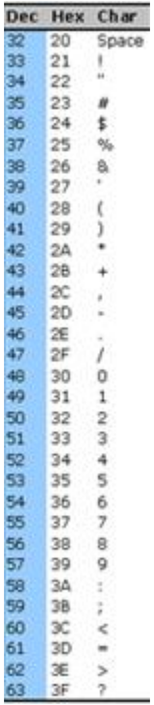
‘a’ is equivalent to 97, or 0110 0001

‘A’ is equivalent to 65, or 0100 0001

Notice that these differ by exactly 32… How can we use masking to convert from upper to lowercase?

ASCII Tips and Tricks

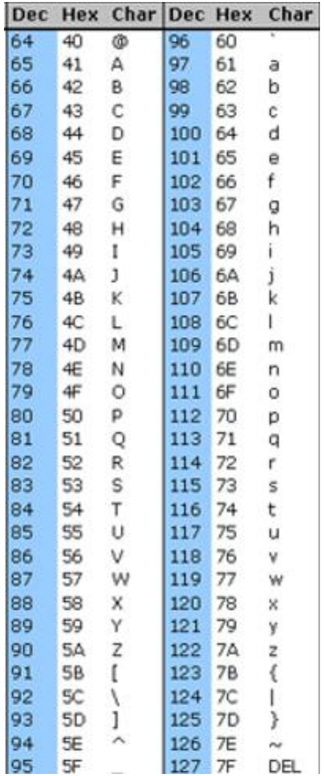
- Converting from numeric characters to numbers
    - ASCII numbers are ordered increasingly, starting with 48 == ‘0’
    - We can take advantage of this fact to convert from ASCII codes to numbers easily
    - `char c = ‘7’;       // any ‘0’ thru ’9’`
    - `int i = (int) c;`
    - What is the value of `i`? What should we do instead?



Dec	Hex	Char
32	20	Space
33	21	!
34	22	"
35	23	#
36	24	$
37	25	%
38	26	&
39	27	'
40	28	(
41	29	)
42	2A	*
43	2B	+
44	2C	,
45	2D	-
46	2E	.
47	2F	/
48	30	0
49	31	1
50	32	2
51	33	3
52	34	4
53	35	5
54	36	6
55	37	7
56	38	8
57	39	9
58	3A	:
59	3B	;
60	3C	<
61	3D	=
62	3E	>
63	3F	?

ASCII Tips and Tricks

- Converting between upper/lower case
    - Corresponding letters differ by exactly 32
    - To convert from upper to lower case, add 32
    - To convert from lower to upper case, subtract 32
    - Alternatively: note that all uppercase letters are <96 and lowercase are >96.
    - So, we can simply toggle bit 5, as it is always unset for capital letters and set for lowercase ones.



Dec	Hex	Char	Dec	Hex	Char
64	40	@	96	60	`
65	41	A	97	61	a
66	42	B	98	62	b
67	43	C	99	63	c
68	44	D	100	64	d
69	45	E	101	65	e
70	46	F	102	66	f
71	47	G	103	67	g
72	48	H	104	68	h
73	49	I	105	69	i
74	4A	J	106	6A	j
75	4B	K	107	6B	k
76	4C	L	108	6C	l
77	4D	M	109	6D	m
78	4E	N	110	6E	n
79	4F	O	111	6F	o
80	50	P	112	70	p
81	51	Q	113	71	q
82	52	R	114	72	r
83	53	S	115	73	s
84	54	T	116	74	t
85	55	U	117	75	u
86	56	V	118	76	v
87	57	W	119	77	w
88	58	X	120	78	x
89	59	Y	121	79	y
90	5A	Z	122	7A	z
91	5B	[	123	7B	{
92	5C	\	124	7C	|
93	5D	]	125	7D	}
94	5E	^	126	7E	~
95	5F	_	127	7F	DEL