3. More numbers

Sept. 10, 2018

- Review
  - Number representations
  - Addition, Subtraction
  - Sign extension
- Overflow
- Hexadecimal representation
- Fractions, large numbers
  - Floating point representation
  - Examples
- Representing characters
  - ASCII
- LC3 Data Types

Review

SIGN MAGNITUDE

SIGN

01110 = ? = 14

−14 : 11110

2's COMPLEMENT <u>REPRESENTATION</u> OF 14_{10}

FLIP 011|10 ← COPY

-14_{10} : 10010

ALSO COMP: 10001

+1

10010

BINARY TO DECIMAL

$$\begin{array}{cccccccc} 2^7 & 2^6 & 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \end{array} \leftarrow \text{2's COMP.}$$

$$2^6 + 2^5 + 2^3 = 104$$

2's COMP:

$$X = 11100110 = ?$$

$$-X = 00011010 = 26$$

$$X = -26$$

DECIMAL TO BINARY

$x = 104_{10}$

		REM	
$104/2$	$= 52$	0	0 — LSB
$52/2$	$= 26$	0	0
$26/2$	$= 13$	0	0
$13/2$	$= 6$	1	1
$6/2$	$= 3$	0	0
$3/2$	$= 1$	1	1
$1/2$	$= 0$	1	1

$104_{10} = 01101000$

↑
SIGN

SIGN EXTENSION

	4 BITS	8 BITS
+4	0100	00000100
−4	1100	11111100

8 BITS → 4 BITS

$x = 10110111$ → TOO BIG

$-x = 0100\boxed{1001}$??

OVERFLOW

OVERFLOW

5 BITS (2's comp)

8_{10} : 01000

9 : 01001

+ 10001 → −15

⇓

OVERFLOW

CARRY

−8 : 11000

−9 : 10111

01111

CARRY

RULES:

1. SIGN OF RESULT IS DIFFERENT

"OR": 2. CARRY INTO SIGN BIT (MSB) ≠ CARRY OUT

EX: 11111

00001

100000

CARRY

Hexadecimal Notation

It is often convenient to write binary (base-2) numbers as hexadecimal (base-16) numbers instead.

- fewer digits -- four bits per hex digit
- less error prone -- easy to corrupt long string of 1's and 0's

Binary	Hex	Decimal	Binary	Hex	Decimal
0000	0	0	1000	8	8
0001	1	1	1001	9	9
0010	2	2	1010	A	10
0011	3	3	1011	B	11
0100	4	4	1100	C	12
0101	5	5	1101	D	13
0110	6	6	1110	E	14
0111	7	7	1111	F	15

Converting from Binary to Hexadecimal

Every four bits is a hex digit.

- start grouping from right-hand side

0|011|1010|1000|1111|0100|1101|0111

3 A 8 F 4 D 7

REPRESENTATIONS

WHAT IS 1000

= 8 UNSIGNED INT

= -8 2's COMP.

0100 0001

= 65 (INTEGER)

= A (ASCII CHARACTER)

$21_{10} = 010101_2$

IN TERNARY? {0, 1, 2}

$3^1 \; 3^0$

$21_3 = 7_{10}$

$4^1 \; 4^0$

$21_4 = 9$

$21_9 = 19$

TERNARY: 0, 01, 02, 10, 11, 12, 20,

.. 21, 22, 100

FRACTIONS

(FIXED POINT)

$$0.5_{10} = 1/2 = 2^{-1}$$

$$(2^{-1})$$

$$= 0.1_2$$

(BINARY POINT)

$$0.01_2 = (1/4)_{10}$$

$$3/4_{10} = 0.\overset{2^{-1}}{1}\overset{2^{-2}}{1}$$

SHIFT LEFT: = X 2

0.11 SHIFTED LEFT: 1.10 $= 1.5_{10}$

EXAMPLE $40.625_{10} - 1.25_{10} =$

$40.625_{10} = 00101000.101$

$\phantom{40.625_{10} =} 11111110.110$

$\phantom{40.625_{10} =} 00100111.011$

$1.25 = 01.010$

$-1.25 = 10.110$

EXAMPLE

0.3_{10} IN BINARY

$\times 2$

0	.3
0	.6
1	.2
0	.4
0	.8
1	.6
1	.2

.001001100110011 . . .

VERY LARGE AND VERY SMALL NUMBERS

$\Rightarrow$ FLOATING POINT

EXAMPLE

6.023×10^{23} : NEEDS 79 BITS

6.626×10^{-34} : NEEDS > 110 BITS

$F \times 2^{E}$

SCIENTIFIC NOTATION.

IEEE 754 Floating Point Standard (32 BITS)

Single Precision

1b	8b	23b
S	Exponent	Fraction

$$N = (-1)^S \times \underline{1.\text{fraction}} \times 2^{\text{exponent}-127},\ 1 \le \text{exponent} \le 254$$

$$N = (-1)^S \times 0.\text{fraction} \times 2^{-126},\ \text{exponent} = 0$$

$1.25_{10} = 1.01 \times 2^{0} = 10.1 \times 2^{-1}$