Problem solving

Till last Thursday

Non-science

Never objective

or

MCQ ← Most subjective

Special office hour ~~S~~aturday

SSJ + TFs + GSV

HA (single bit) and FA (multi-bit):

$x_i \cdot y_i \cdot (c_i + \bar{c}_i)$ $c_i y_i$ $c_i x_i$

$$c_{i+1} = \bar{c}_i \cdot x_i \cdot y_i + c_i \cdot \bar{x}_i \cdot y_i + c_i \cdot x_i \cdot \bar{y}_i + c_i \cdot x_i \cdot y_i \longrightarrow \text{CSOP}$$

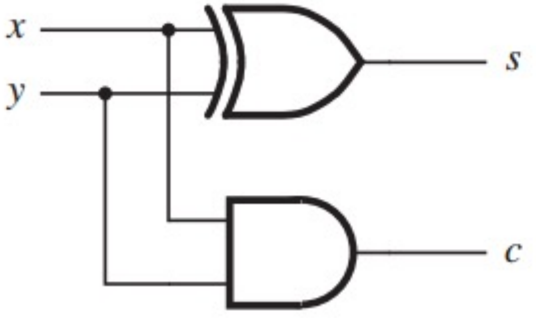
$$c_{i+1} = x_i \cdot y_i + c_i \cdot (\bar{x}_i \cdot y_i + x_i \cdot \bar{y}_i)$$

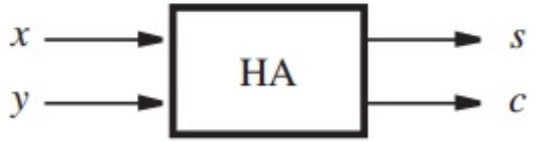
$x_i \cdot y_i + c_i x_i + c_i y_i$

XOR

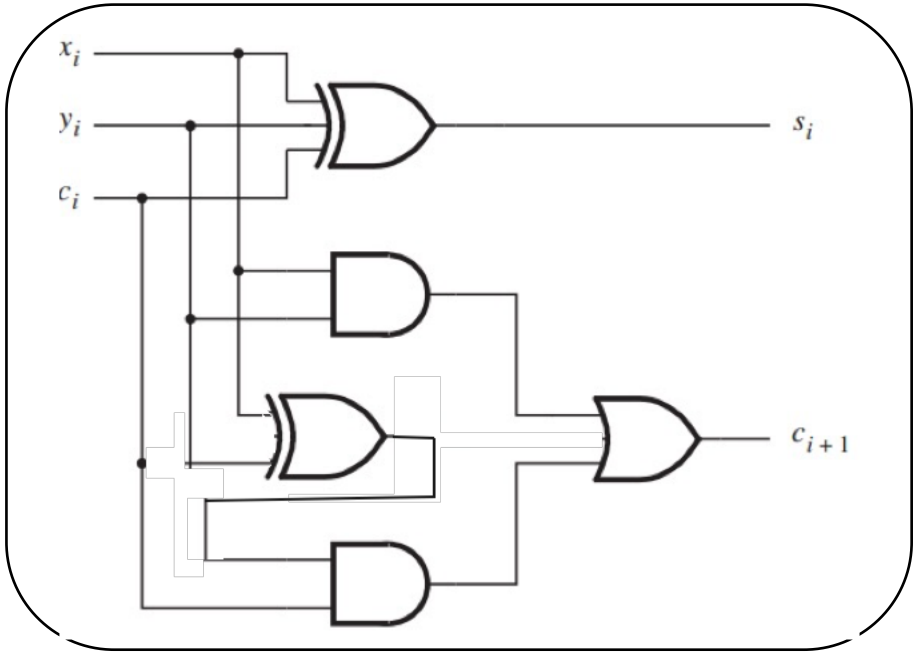
$$c_{i+1} = x_i \cdot y_i + c_i \cdot (x_i \oplus y_i)$$

x	y	Carry c	Sum s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0





c_i	x_i	y_i	c_{i+1}	s_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1



Signed Numbers: Addition/ Subtraction:

positive +3   negative −3 ← number   sign symbol   symbol Lab → ahead / in time

logic circuits (digital) 0,1   sign symbol position   left most is sign

1 compute   overflow error   7−7 ⇒ −14

11110   4 bits   8   1 → sign   7 bit → overflow (2^7-1) 7,128

Decimal	Signed Magnitude
+7	0111
+6	0110
+5	0101
+4	0100
+3	0011
+2	0010
+1	0001
+0	0000
−0	1000
−1	1001
−2	1010
−3	1011
−4	1100
−5	1101
−6	1110
−7	1111
−8	—

sign bit +   0 0111   8   sign 1000   10001   checks and insertion

1. If both the numbers have the same sign, drop the sign bit, add the two numbers and reinsert the sign bit.
2. If one is positive and the other is negative, remove the sign bit, subtract the ~~larger~~ smaller ~~magnitude~~ number from the ~~smaller~~ larger magnitude ← −ve number and then reintroduce the sign of the number with larger magnitude. ---- A very laborious process, hence not preferred.

Binary → unsigned 1st   insert −ve sign

universal convention { unless stated unsigned   D

Signed Numbers:

12×10^{6}

12.1×10^{5}

Mantissa | exponent

IEEE rep.

more details

Computer organization

useful

increases hardware

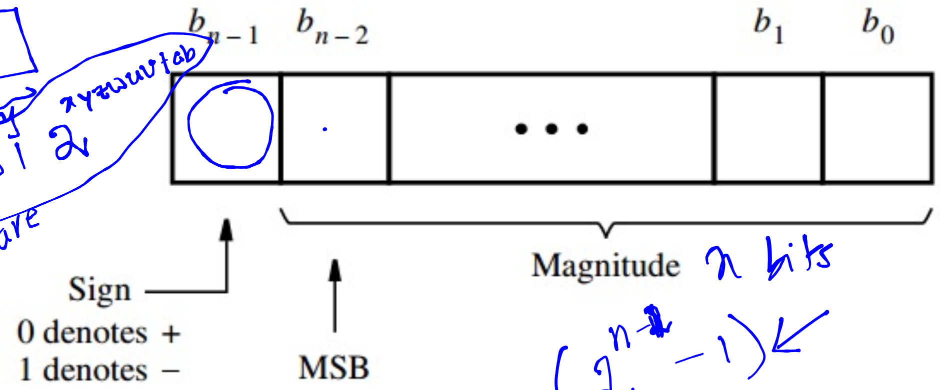
used

$10^{00} \to 10^{99}$

arbitrary

16 → 6 bits, 9

2^{256}



n bits

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

- Sign and Magnitude Representation

+5	00000101	-5	1101
+11	01011	-11	11011
+14	01110	-14	11110

bit extension Software

8 bits

+5 4 bits

sign bit

0101

00000101

64 bits → 128 bits

double precision mode

An easy-to-understand coding practice, but not comfortable for Hardware Realization.

Signed Numbers (for subtraction):

5 bit

−14 → 8 4 2 1 / 11110 sign, 14 −14 → code 1111110010 2's complement coding

1's complement

2's complement

- (r-1)'s Complement Representation

r = 10 decimal r → radix

In decimal 2 is the 9's compliment of 7 and 7 is the 9's compliment of 2,

0	1	2	3	4	5	6	7	8	9
↕	↕	↕	↕	↕	↕	↕	↕	↕	↕
9	8	7	6	5	4	3	2	1	0

999

$$176 - 048$$

3 digit digit

$$\downarrow$$

4 digit

$$176 + 951 - 999$$

$$\downarrow$$

carry out C_4 S_0

$$1127 - 999 - 1 + 1$$

remove

$$\downarrow$$

$$1128 - 1000$$

$$\downarrow$$

$$128$$

If there is a Carry add 1 to unit Digit

decimal

1248 − 0120

C_i

$x_i = 3$

y_i

C_{i+1} S_i

C_4

$$0176 - 1348$$

$$\downarrow$$

$$0176 + 8651 - 9999$$

$$\downarrow$$

No carry out

$$8827 - 9999$$

No Carry obtain 9's compliment

negative number

$$-1172$$

−1172

In binary, 0 is the 1's compliment of 1 and 1 is the 1's compliment of 0,

$$1101 - 0110$$

$$\downarrow$$

$$1101 + 1001 - 1111$$

$$\downarrow$$

$$10110 - 1111$$

$$\downarrow$$

$$10110 - 1111 - 1 + 1$$

$$\downarrow$$

$$10111 - 10000$$

$$\downarrow$$

$$\boxed{0111}$$

If there is a Carry drop the carry and add 1 to LSB

$$1101 - 1110$$

$$\downarrow$$

$$1101 + 0001 - 1111$$

$$\downarrow$$

$$1110 - 1111$$

No Carry take the 1's compliment and add a negative sign.

$$-1$$

4 bit signed

$0 \& 1 \rightarrow x_3 x_2 x_1 x_0$

+7 -7

1 001 $-7+1 = -6$

-14 →

sign bit has value

1 1 1 1 1 1 1 1 0 1 0 -14

1's comp. rep.

1001 → 9

9-15 = -6

position value

$2^3\ 2^2\ 2^1\ 2^0$

1001

$(1-2^4)-15+9=-6$

$-15\ (1-2^4) = -15$

0001 0+1=1

-0110

1111 0110 1001

8 bit

1's

+ve

0110 1001 → -6

$-(2^3-1)$

$-(2^4-1)$

10000

signed → stated

default is +ve

8

-8

+ -

Questions { Read next 2 slides; ask Qs on r's complement

Signed Numbers

- r's Complement Representation (shortcut)

- Given a number $B = b_{n-1}\, b_{n-2} \cdots b_1\, b_0$, its 2's complement, $K = k_{n-1}\, k_{n-2} \cdots k_1\, k_0$, can be found by examining the bits of B from right to left and taking the following action: copy all bits of B that are 0 and the first bit that is 1; then simply complement the rest of the bits

$$\longleftarrow - - 0110 ---\rightarrow 1001 + 0001 = \boxed{1010}$$

$$\longleftarrow - - 01101 ---\rightarrow 10010 + 00001 = 10011$$

Signed Numbers (for Base other than 2)

- r's Complement Representation: r's complement of any number *N*, can be formed by leaving all least significant 0's unchanged, subtracting the first nonzero least significant digit from r, and subtracting all higher significant digits from r-1.