Binary addition

$$\begin{array}{r} 0 \\ +0 \\ \hline 0 \end{array} \quad \begin{array}{r} 0 \\ +1 \\ \hline 1 \end{array} \quad \begin{array}{r} 1 \\ +1 \\ \hline (1)\,0 \end{array} \rightarrow \text{carry } 1$$

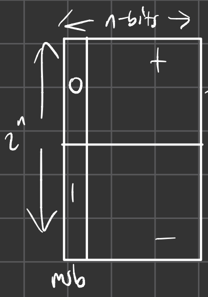
Sign Magnitude

$$+6 \quad \underset{+}{0} \; \underset{6}{0110}$$

$$-6 \quad \underset{-}{1} \; \underset{6}{0110}$$

Two's Compliment

- Assume an n-bit word



→ unsigned/sign-magnitude

→ $2^n - N$ ← # you want to make negative

ex: $-9_{2\text{s comp}}$?

$N = 9$

$n = 5$

$2^n - N = 2^5 - 9 = 32 - 9$

$= 23_{10}$

$= 10111_{2\text{s comp}}$

Two's compliment(N) = 1's compliment(N) + 1

$= \overline{N} + 1$

$+9: 01001$

$= (\overline{01001}) + 1$

$= 10110 + 1$

$= 10111$

Addition Rules

- add numbers and discard (final) carry
- if result is negative, its msb = 1 and its in 2's-comp form

$$\begin{array}{r} 9 \\ -6 \\ \hline 3 \end{array}$$

$+9: 01001$

$-6: 11010$

$$\begin{array}{rl} 00110 & (+6) \\ 11001 & (1\text{'s comp}) \\ + \quad\; 1 & \\ \hline 11010 & (2\text{'s comp}) \end{array}$$

$$\begin{array}{r} 01001 \\ +\,11010 \\ \hline \cancel{(1)}\,00011 \end{array}$$

(+) ∴ +3

$$\begin{array}{r} 6 \\ -9 \\ \hline -3 \end{array}$$

$+6: 00110$

$-9: 10111$

$$\begin{array}{r} 00110 \\ +\,10111 \\ \hline (0)\;11101 \end{array}$$

↓ (−) 3

$$\begin{array}{rl} 11101 & (2\text{'s comp}) \\ 00010 & (1\text{'s comp}) \\ + \quad\; 1 & \\ \hline 00011 & (2\text{'s comp}) \end{array}$$

(+) 3

2's Compliment Rules

- leave all least-significant 0's as is
- leave first least-significant 1 as is
- flip all remaining bits

0100011101111100111000_2

$1011100010000011001000_{2\text{s-comp}}$

Range

$k = 3$ bits

k-bits	unsigned	sign-magnitude	2's-comp
0 0 0	0	+0	+0
0 0 1	1	+1	+1
0 1 0	2	+2	+2
0 1 1	3	+3	+3
1 0 0	4	−0	−4
1 0 1	5	−1	−3
1 1 0	6	−2	−2
1 1 1	7	−3	−1

unsigned: 0 to $2^k - 1$
sign-magnitude: $-(2^{k-1} - 1)$ to $+(2^{k-1} - 1)$
2's-compliment: $-(2^{k-1})$ to $+(2^{k-1} - 1)$

Overflow

- assume a word size of $k = 6$ bits

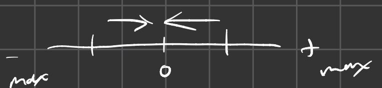
$$\begin{array}{r} 30 \\ +30 \\ \hline 60 \end{array} \qquad \begin{array}{rr} & 011110 \\ + & 011110 \\ \hline (0) & 111100 \end{array}$$

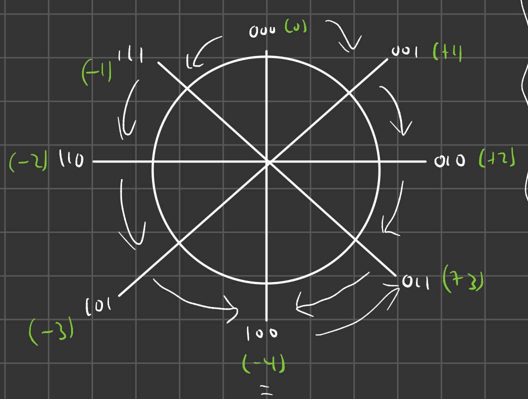
$\uparrow$ C_{out}

unsigned: 0 to $2^6 - 1$
0 to 63 ✓ $\Rightarrow$ no overflow has occured

2's-comp: -32 to $+31$ ✗ $\Rightarrow$ overflow has occured

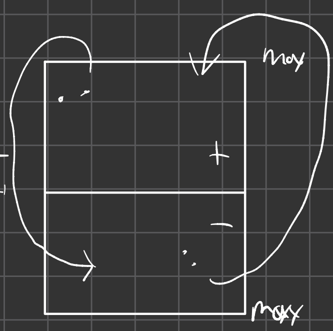
Unsigned: if(C_{out} == 0)
No overflow
else
Over flow





2's-comp:

$$V = \overline{A_{n-1}} \cdot \overline{B_{n-1}} \cdot R_{n-1} + A_{n-1} \cdot B_{n-1} \cdot \overline{R_{n-1}}$$
$$= \overline{0} \cdot \overline{0} \cdot 1 \qquad 1 \cdot 1 \cdot \overline{0}$$
$$= 1 \cdot 1 \cdot 1 \qquad 1 \cdot 1 \cdot 1$$
$$= 1\ (T) \qquad 1\ (T)$$



$A_{n-1} \mid A_{n-2} \ldots A_0$ (binary)
$B_{n-1} \mid B_{n-2} \ldots B_0$
$R_{n-1} \mid R_{n-2} \ldots R_0$
sign-bit

- Single expressions with a mix of signed and unsigned
  - signed values are implicitly cast to unsigned
  - includes comparison operations ==, >, <, >=, and <=
- Example: K = 16 bits, U_{max} = 65,535, S_{min} = -32,768, S_{max} = +32,767

Constant1	Constant2	Relation	Evaluation
0	0U	==	unsigned
-1	0	<	signed
-1	0U	>	unsigned
32767	-32768	>	signed
32767U	-32768	<	unsigned
-1	-2	>	signed
(unsigned) -1	-2	>	unsigned
32767	32768U	<	unsigned
32767	(int) 32768U	>	signed

Parity ⇒ One of the most common ways of achieving error detection

- extra "parity" bit included with data to make total 1's odd or even

Original Data	Even Parity	Odd Parity
000	0	1
001	1	0
010	1	0
011	0	1
100	1	0
101	0	1
110	0	1
111	1	0