Addition of 1-bit Numbers

$$\begin{array}{r} 0 \\ +\ 0 \\ \hline 0 \end{array} \qquad \begin{array}{r} 1 \\ +\ 0 \\ \hline 1 \end{array} \qquad \begin{array}{r} 0 \\ +\ 1 \\ \hline 1 \end{array} \qquad \begin{array}{r} 1 \\ +\ 1 \\ \hline 1\ 0 \end{array}$$

Carry-out

1.4 Three Representation Schemes

- 4-bit Binary

1. Sign and magnitude
2. One's (1's) complement
3. Two's (2's) complement

B	Values represented		
$b_3b_2b_1b_0$	Sign and magnitude	1's complement	2's complement
0 1 1 1	+7	+7	+7
0 1 1 0	+6	+6	+6
0 1 0 1	+5	+5	+5
0 1 0 0	+4	+4	+4
0 0 1 1	+3	+3	+3
0 0 1 0	+2	+2	+2
0 0 0 1	+1	+1	+1
0 0 0 0	+0	+0	+0
1 0 0 0	−0	−7	−8
1 0 0 1	−1	−6	−7
1 0 1 0	−2	−5	−6
1 0 1 1	−3	−4	−5
1 1 0 0	−4	−3	−4
1 1 0 1	−5	−2	−3
1 1 1 0	−6	−1	−2
1 1 1 1	−7	−0	−1

All 3 Representations: Positive Integer

- To represent a positive integer :

1. Most Significant Bit (b_3) being 0
   - Least Significant Bit (b_0)

MSB

LSB

2. Base$_2$ of (Integer-Base$_{10}$) for remaining bits
   - ➢ e.g., $+5_{10}$ = **0** Base$_2$ (5_{10}) = 0101 (4-bit)
   - ➢ e.g., $+5_{10}$ = **0** Base$_2$ (5_{10}) = 0000 0101 (8-bit)

B	Values represented		
$b_3b_2b_1b_0$	Sign and magnitude	1's complement	2's complement
0 1 1 1	+7	+7	+7
0 1 1 0	+6	+6	+6
0 1 0 1	+5	+5	+5
0 1 0 0	+4	+4	+4
0 0 1 1	+3	+3	+3
0 0 1 0	+2	+2	+2
0 0 0 1	+1	+1	+1
0 0 0 0	+0	+0	+0
1 0 0 0	−0	−7	−8
1 0 0 1	−1	−6	−7
1 0 1 0	−2	−5	−6
1 0 1 1	−3	−4	−5
1 1 0 0	−4	−3	−4
1 1 0 1	−5	−2	−3
1 1 1 0	−6	−1	−2
1 1 1 1	−7	−0	−1

Positive Integer Representation

- Convert a positive integer into binary: MSB/Binary
  - e.g., $+5_{10}$ = <u>0</u>101 (4-bit)
  - e.g., $+5_{10}$ = <u>0</u>000 0101 (8-bit)
  - e.g., $+5_{10}$ = <u>0</u>000 0000 0101 (12-bit)
  - e.g., $+5_{10}$ = <u>0</u>000 0000 0000 0101(16-bit)

Sign & Magnitude: Negative Integer

- To represent a negative integer in Sign & Magnitude:

1. Most Significant Bit being 1

2. $Base_2$ of ($Integer\text{-}Base_{10}$) for remaining bits

   - e.g., -5_{10} = **1** $Base_2$ (5_{10}) = **1** 101 (4-bit)

     $+5_{10}$ = **<u>0</u>**101 (4-bit)

   - e.g., -5_{10} = **1** $Base_2$ (5_{10}) = **1** 000 0101 (8-bit)

     $+5_{10}$ = **<u>0</u>**000 0101 (8-bit)

Compare to their + representation?

Sign & Magnitude

For Positive Integer

1. MSB = 0
2. Base_{10} to Base_2 for remaining bits

For Negative Integer

1. MSB = 1
2. Base_{10} to Base_2 for remaining bits

B	Values represented		
$b_3 b_2 b_1 b_0$	Sign and magnitude	1's complement	2's complement
0 1 1 1	+7	+7	+7
0 1 1 0	+6	+6	+6
0 1 0 1	+5	+5	+5
0 1 0 0	+4	+4	+4
0 0 1 1	+3	+3	+3
0 0 1 0	+2	+2	+2
0 0 0 1	+1	+1	+1
0 0 0 0	+0	+0	+0
1 0 0 0	−0	−7	−8
1 0 0 1	−1	−6	−7
1 0 1 0	−2	−5	−6
1 0 1 1	−3	−4	−5
1 1 0 0	−4	−3	−4
1 1 0 1	−5	−2	−3
1 1 1 0	−6	−1	−2
1 1 1 1	−7	−0	−1

1's Complement: Negative Integers

- Complement its Positive Representation

1. Most Significant Bit being 0; $Base_2$ of ($Integer\text{-}Base_{10}$) for remaining bits
2. Complement (step 1): 0 → 1 and 1 → 0

- E.g., convert -5 to 1's complement

1. $+5_{10} = 0101$
2. -5_{10} = Complement (0101) = 1010 (4-bit)

   = Complement (0000 0101) = 1111 1010 (8-bit)

1's Complement

For Positive Integer

1. MSB = 0
2. $Base_{10}$ to $Base_2$ for remaining bits

For Negative Integer

a) Complement its positive representation

B	Values represented		
$b_3 b_2 b_1 b_0$	Sign and magnitude	1's complement	2's complement
0 1 1 1	+7	+7	+7
0 1 1 0	+6	+6	+6
0 1 0 1	+5	+5	+5
0 1 0 0	+4	+4	+4
0 0 1 1	+3	+3	+3
0 0 1 0	+2	+2	+2
0 0 0 1	+1	+1	+1
0 0 0 0	+0	+0	+0
1 0 0 0	−0	−7	−8
1 0 0 1	−1	−6	−7
1 0 1 0	−2	−5	−6
1 0 1 1	−3	−4	−5
1 1 0 0	−4	−3	−4
1 1 0 1	−5	−2	−3
1 1 1 0	−6	−1	−2
1 1 1 1	−7	−0	−1

2's Complement: Negative Integers

- 2's Complement for Negative Integer
  - Add 1 to its 1's complement
    - 1's complement: Complement its Positive Representation

- Example,

  -5_{10} = Complement (0101) + 1 = 1010 + 1 = 1011 (4-bit)

  = Complement (0000 0101) + 1 = 1111 1011 (8-bit)

2’s Complement

For Positive Integer

1. **MSB = 0**
2. **Base_{10} to Base_2 for remaining bits**

For Negative Integer

- **Add 1 to its 1’s complement**
  - **1’s complement: Complement its Positive Representation**

B	Values represented		
$b_3 b_2 b_1 b_0$	Sign and magnitude	1’s complement	2’s complement
0 1 1 1	+7	+7	+7
0 1 1 0	+6	+6	+6
0 1 0 1	+5	+5	+5
0 1 0 0	+4	+4	+4
0 0 1 1	+3	+3	+3
0 0 1 0	+2	+2	+2
0 0 0 1	+1	+1	+1
0 0 0 0	+0	+0	+0
1 0 0 0	−0	−7	−8
1 0 0 1	−1	−6	−7
1 0 1 0	−2	−5	−6
1 0 1 1	−3	−4	−5
1 1 0 0	−4	−3	−4
1 1 0 1	−5	−2	−3
1 1 1 0	−6	−1	−2
1 1 1 1	−7	−0	−1

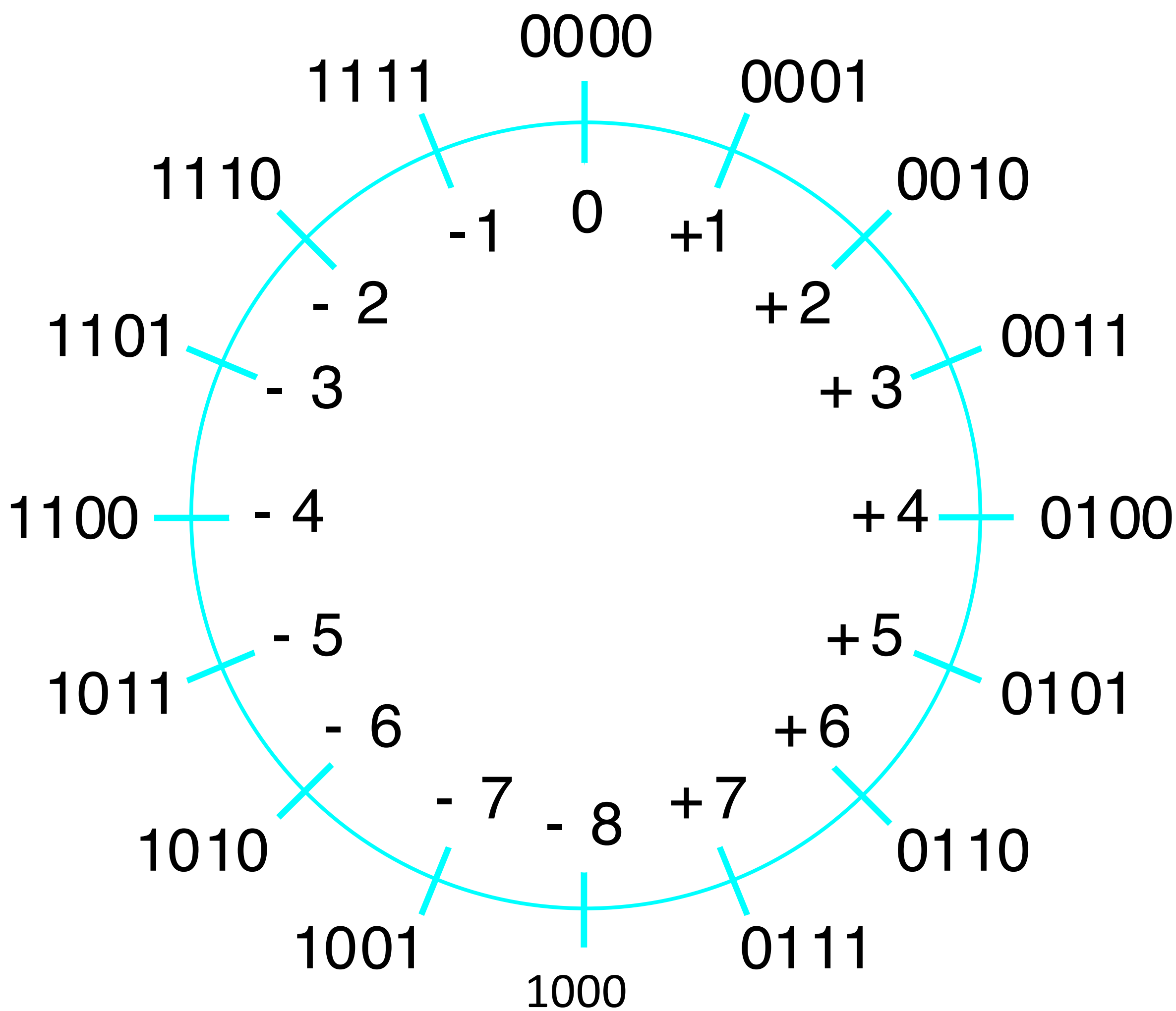
Negative Integer Summary

A. Sign and magnitude: Most Significant Bit being 1
   a) e.g., -5 = 1101 (4-bit) = 1000 0101 (8-bit)

B. 1’s complement: Complement the +ve representation
   a) e.g., -5 = complement (0101) = 1010
   b) e.g., -5 = complement (0000 0101) = 1111 1010

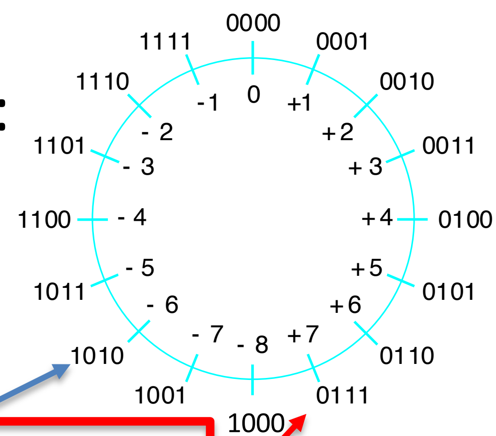
C. 2’s complement: Add 1 to its 1’s complement
   a) e.g., -5 = complement (0101) + 1 = 1010 + 1 = 1011
   b) e.g., -5 = complement (0000 0101) + 1 = 1111 1010 + 1 = 1111 1011

2's Complement for -8 to +7



Arithmetic Overflow

- When a number does not fit in the number range:
- 4-bit 2's complement in range of integer {-8..+7}
- Overflow examples (as the result of an operation)



0 1 1 0	+6
+ $0_1$1 0 0	+ +4
------------	------
1 0 1 0	+10

1 1 1 0	-2
$_1$1 0 0 1	+ -7
----------	------
0 1 1 1	-9

2’s-Complement Add & Subtract Examples

(a)	0010 + 0011	(+2) + (+3)	0101	(+5)
(b)	0100 + 1010	(+4) + (- 6)	1110	(- 2)
(c)	1011 + 1110	(- 5) + (- 2)	1001	(- 7)
(d)	0111 + 1101	(+7) + (- 3)	0100	(+4)

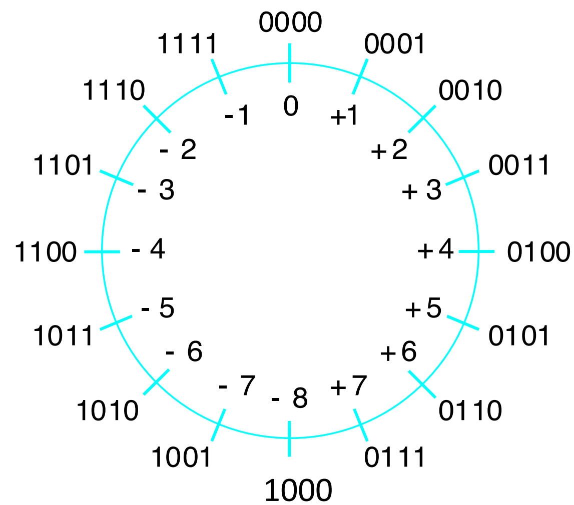
(e)	1101 - 1001	(- 3) - (- 7)	⇒	1101 + 0111	0100	(+4)
(f)	0010 - 0100	(+2) - (+4)	⇒	0010 + 1100	1110	(- 2)
(g)	0110 - 0011	(+6) - (+3)	⇒	0110 + 1101	0011	(+3)
(h)	1001 - 1011	(- 7) - (- 5)	⇒	1001 + 0101	1110	(- 2)
(i)	1001 - 0001	(- 7) - (+1)	⇒	1001 + 1111	1000	(- 8)
(j)	0010 - 1101	(+2) - (- 3)	⇒	0010 + 0011	0101	(+5)

Adding Two n-bit 2's-Complement Numbers

- Add the numbers (in 2's already)
  a) If result in range $\{-2^{n-1} \ldots +(2^{n-1}-1)\}$:

(d)

	0 1 1 1	(+7)
+	$_1$1 1 0 1	(- 3)
	0 1 0 0	(+4)



  b) If result not in range $\{-2^{n-1} \ldots +(2^{n-1}-1)\}$:
     - ***Arithmetic Overflow!***

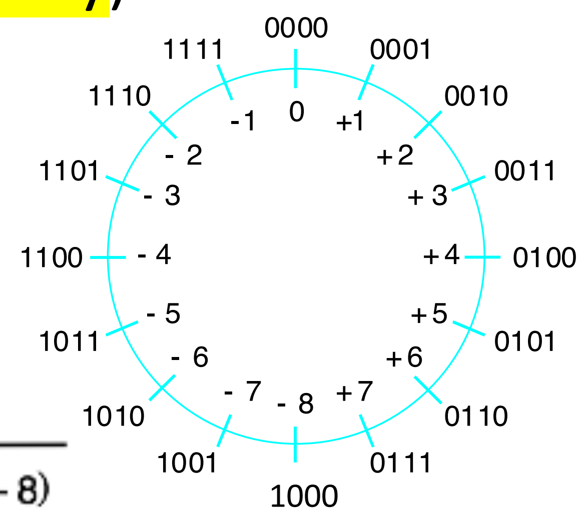
  - e.g., add +3 (0011) and +5 (0101) = +8 not in {-8…+7}
  - 0011 + 0101 = 1000 is 2's complement of -8!

Subtract 2 n-bit 2's-Complement Numbers

- To subtract Y from X (i.e., X – Y; X and Y in 2's already)
  1. Form the 2's complement of Y
  2. Add to X
  a) If result in $\{-2^{n-1} \ldots +(2^{n-1} - 1)\}$; yes

(i)

1001	(- 7)	⇒	1001	
- 0001	(+1)		+11111	
			1000	(- 8)



  b) If result not in $\{-2^{n-1} \ldots +(2^{n-1} - 1)\}$
  - ***Arithmetic Overflow!***
  - e.g., subtract -2 (1110) from +6 (0110) = +8
  - 2's (1110) + 0110 = (0001+1) + 0110 = 0010 + 0110

  = 1000 is 2's complement of -8!

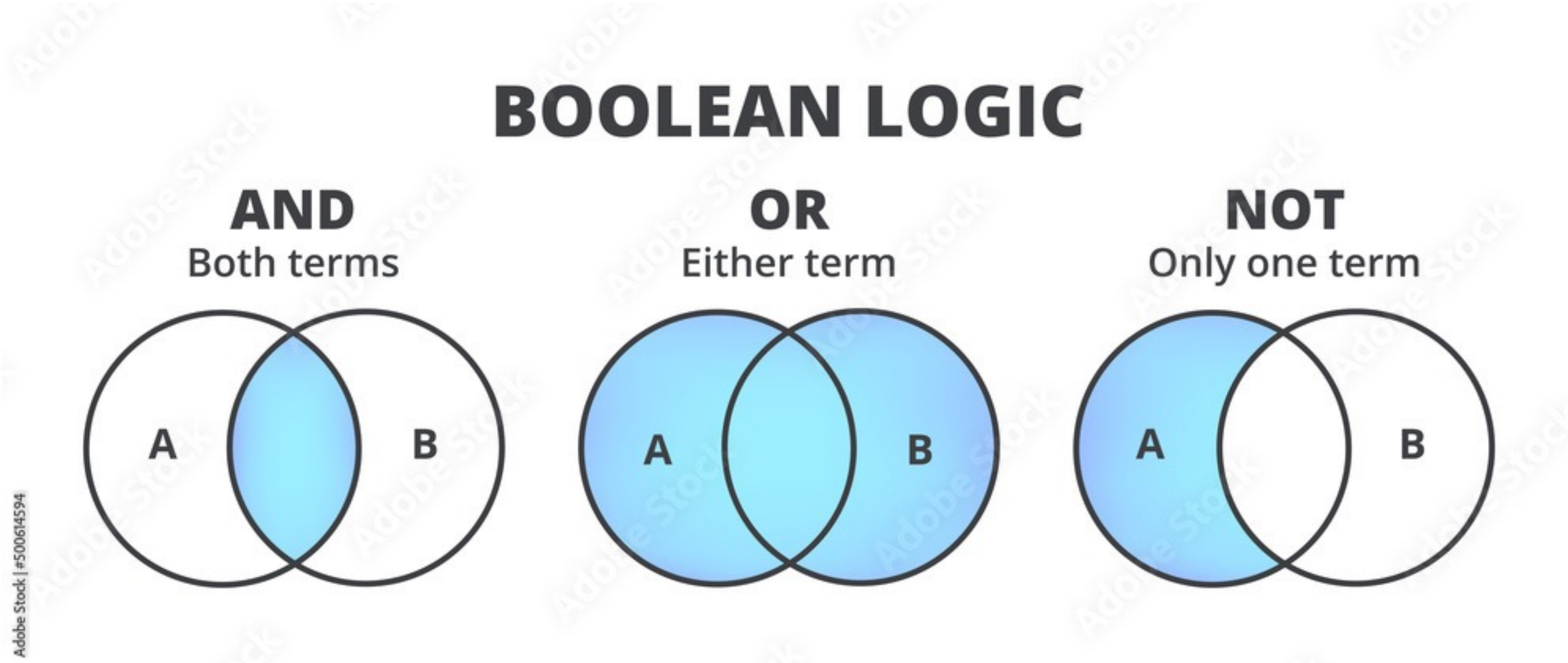
Sign Extension

- ***n***-bit 2's complement to an ***m***-bit format, where ***m*** > ***n***

  Repeat the sign bit to the left till the MSB

  - For both positive and negative numbers
    - Examples: 4-bit to 8-bit sign extension
      - +3 (0011) to +3 (0000 0011)
      - -3 (1101) to -3 (1111 1101)

Logic Operations



https://stock.adobe.com/ch_fr

Logic Operations (NOT)

INPUT	OUTPUT
A	X = A'
0	1
1	0

Logic Operations (OR)

INPUT		OUTPUT
A	B	$X = A + B$
0	0	0
0	1	1
1	0	1
1	1	1

Logic Operations (AND)

INPUT		OUTPUT
A	B	X = A . B
0	0	0
0	1	0
1	0	0
1	1	1

2.8.1 Logic Instructions

- AND, OR, and NOT operations on single binary bits

NOT (0101)? 1010

- Or R4, R2, R3
  - OR the individual bits in R2 and R3 and put the result in R4

(0110) OR (0101)? 0111

- AND R5,R6,#0xFF
  - AND the individual bits in R6 with #(0x(FF)) and put the result in R5

(0110) AND (0101)? 0100