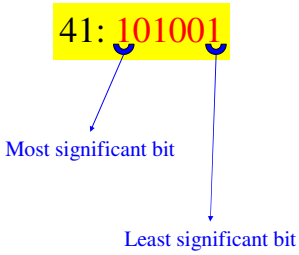
Coding Numbers

Coding Numbers

- Two possible values: V_{DD} and $V_{SS} \rightarrow 1/0$
- Radix 2
- Bit = **bi**nary digi**t**
- **n-bit word** : number of bits used in processor operations



Signed Numbers

- Find representation with little or no impact on circuit cost and latency → **transparent** key operators (addition).
- 1st solution:
  - Sign = most significant bit
  - **0**: >0, **1**: <0.

- Word = 4 bits
- Two representations for 0:
  - 0000
  - 1000
- Non-Transparent representation

(-1)		1	0	0	1
(1)	+	0	0	0	1
(-2)		**1**	**0**	**1**	**0**

Signed Numbers

- 2nd solution:
  - Negative value = logic complement
    -A = A'
  - A + (-A) = 0.
  - «Complement» (or 1-complement)

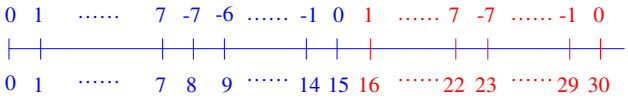
(-1)		1	1	1	0
(1)	+	0	0	0	1
(-0)		**1**	**1**	**1**	**1**

(-2)		1	1	0	1
(1)	+	0	0	0	1
(-1)		**1**	**1**	**1**	**0**

- Word = 4 bits.
- Two representations for 0:
  - 0000 (0)
  - 1111 (-0)
- Sign = most significant bit
- Non-Transparent representation

5		1 0	1	0	1
(-2)	+	1	1	0	1
(??)		**1**	**0**	**0**	**1**	**0**

Signed Numbers



- Complement = convention:
  - $0 \le x \le 7$: $X_{CPLT} = x$
  - $-7 \le x \le 0$: $X_{CPLT} = |x|' = (1-|x|_3)\,(1-|x|_2)\,(1-|x|_1)\,(1-|x|_0) = 1111 - |x|_3|x|_2|x|_1|x|_0 = 15-|x| = 15+x$
- $x \ge 0$, $y \le 0$: $X_{CPLT}+Y_{CPLT} = x + y + 15$
  - $x+y \le 0$: result in [0;15]
  - $x+y > 0$: result valid but in [16;22] (5 bits)
- $x \ge 0$, $y \ge 0$: if no overflow, $x+y \le 7$ and $X_{CPLT}+Y_{CPLT} = x+y$.
- $x \le 0$, $y \le 0$: if no overflow, $-7 \le x+y$ and $X_{CPLT}+Y_{CPLT} = 30 + x + y \Rightarrow$ result valid but in [23;30] (5 bits)

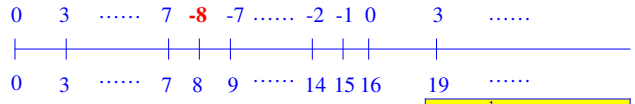
$\Rightarrow$ Result addition using complement representation:
$Z = (X_{CPLT}+Y_{CPLT})$ modulo 15.

Signed Numbers

- Non-Transparent representation. Impact on circuit ?
- Use 4 bits only (word size)
  - Result $30 > Z_{CPLT} > 15 \rightarrow$ with 5 bits: $1\,z_3\,z_2\,z_1\,z_0$ et $z_3\,z_2\,z_1\,z_0 \neq 1111$ et 1110.
  - Z_{CPLT} mod 15
    - $(10000 + z_3\,z_2\,z_1\,z_0)$ mod 15
    - $(15 + 1 + z_3\,z_2\,z_1\,z_0)$ mod 15
    - $(1 + z_3\,z_2\,z_1\,z_0)$ mod 15
    - $1 + z_3\,z_2\,z_1\,z_0$

$\Rightarrow$ Have to add carry: **two computations**, for one addition operation originally

Signed Numbers



- 3rd solution:
  - Modified convention:
    - $x \leq 0$: $X_{CPLT} = x + \mathbf{16}$.
  - Similar reasoning and $Z_{CPLT}=(X_{CPLT}+Y_{CPLT})$ *modulo* ***16***.
  - Modulo 16 ⇔ keep the 4 least significant bits
    → **No overhead cost**
  - 15th value (8↔1000):
    - either +8, or -8:
    - set to **-8** because most significant bit =1 ⇒ negative
  - Negative value: $x \geq 0$, $y=-x$, $Y_{CPLT} = -x + 16$
    $= 1 + 15 - x = \mathbf{1 + x'}$
    -0 = 0 but -(-8) = -8

	1				
5		0	1	0	1
(-2)	+	1	1	1	0
(19)	1	0	0	1	1

3 ← Modulo 16

- **Twos-Complement** (n bits):
  - $-2^{n-1} \leq x \leq 2^{n-1}-1$
  - Fixed-Point with n bits:
    $x=x_{n-1}x_{n-2}x_{n-3}\cdots x_0$
    Negative value: $-x + 16$
    $= x_{n-1},x_{n-2}x_{n-3}\ldots x_0 + 10,00\ldots0$
    $= x_{n-1},x_{n-2}x_{n-3}\ldots x_0 + 2,00\ldots0$

Hexadecimal

- More a representation than a coding
- Enables condensed representation of a binary number
- Radix 16
- One hexadecimal digit = 4 bits

Example:
$210_{10\ ou\ d}$
$11010010_{2\ ou\ b}$
$D2_{16\ ou\ x}$

Decimal	Binary	**Hexadecimal**
⋮	⋮	⋮
9	1001	**9**
10	1010	**A**
11	1011	**B**
12	1100	**C**
13	1101	**D**
14	1110	**E**
15	1111	**F**