Digital Systems - 5

Pere Palà - Alexis López

iTIC http://itic.cat

February 2016

Substracting Unsigned Integers

```
signal u1,u2,r1,r2    : unsigned(3 downto 0);
...
u1 <= "0011";   -- 3
u2 <= "0100";   -- 4

r1 <= u1-u2;    --> "1111"
r2 <= u2-u1;    --> "0001"
```

- "1111" --> -1
- "1110" --> -2
- ...

Signed Integers

Definition

- $x = -x_{n-1} \times 2^{n-1} + x_{n-2} \times 2^{n-2} + \cdots + x_0 \times 2^0$.
- Only difference: sign of x_{n-1}
- It is called *2's complement*

Use

- Usable range: $-2^{n-1} \cdots 2^{n-1} - 1$
- Not symmetric around zero!

```
signal my_signed   : signed   (3 downto 0);
signal my_unsigned : unsigned (3 downto 0);

my_signed <= "0000";
my_unsigned <= my_signed;      -- Illegal: different types!
my_unsigned <= unsigned(my_signed); --Ok: Type conversion
my_signed   <= signed(my_unsigned); --Ok: Type conversion
```

Two's Complement

code	n
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	-8
1001	-7
1010	-6
1011	-5
1100	-4
1101	-3
1110	-2
1111	-1

Sign change

- Convert 5 to -5
  1. Invert 0101 $\rightarrow$ 1010
  2. Add 1 $\rightarrow$ 1011
- Convert -5 to 5
  1. Invert 1011 $\rightarrow$ 0100
  2. Add 1 $\rightarrow$ 0101
- `x_neg <= not x + 1;`

Resizing Signed Integers

Expand

```
signal a         : signed(3 downto 0);
signal b1,b2,b3 : signed(7 downto 0);

b1 <= a(3) & a(3) & a(3) & a(3) & a;

b2(3 downto 0) <= a;
b2(7 downto 4) <=(others => a(3));

b3 <= resize(a,8);
b3 <= resize(a,b3'length);
```

Truncate

```
signal a1,a2 : signed(3 downto 0);
signal b     : signed(7 downto 0);

a1 <= b(3 downto 0);

a2 <= resize(b,4);
a2 <= resize(b,a2'length);
```

Adding Signed Integers

	0	0	0	0	0	0	0	0
72:	0	1	0	0	1	0	0	0
49:	0	0	1	1	0	0	0	1
121:	0	1	1	1	1	0	0	1

no overflow

	1	1	0	0	0	0	0	0
−63:	1	1	0	0	0	0	0	1
−32:	1	1	1	0	0	0	0	0
−95:	1	0	1	0	0	0	0	1

no overflow

	0	0	0	0	0	0	0	0
−42:	1	1	0	1	0	1	1	0
8:	0	0	0	0	1	0	0	0
−34:	1	1	0	1	1	1	1	0

no overflow

	0	1	0	0	1	0	0	0
72:	0	1	0	0	1	0	0	0
105:	0	1	1	0	1	0	0	1
	1	0	1	1	0	0	0	1

positive overflow

	1	0	0	0	0	0	0	0
−63:	1	1	0	0	0	0	0	1
−96:	1	0	1	0	0	0	0	0
	0	1	1	0	0	0	0	1

negative overflow

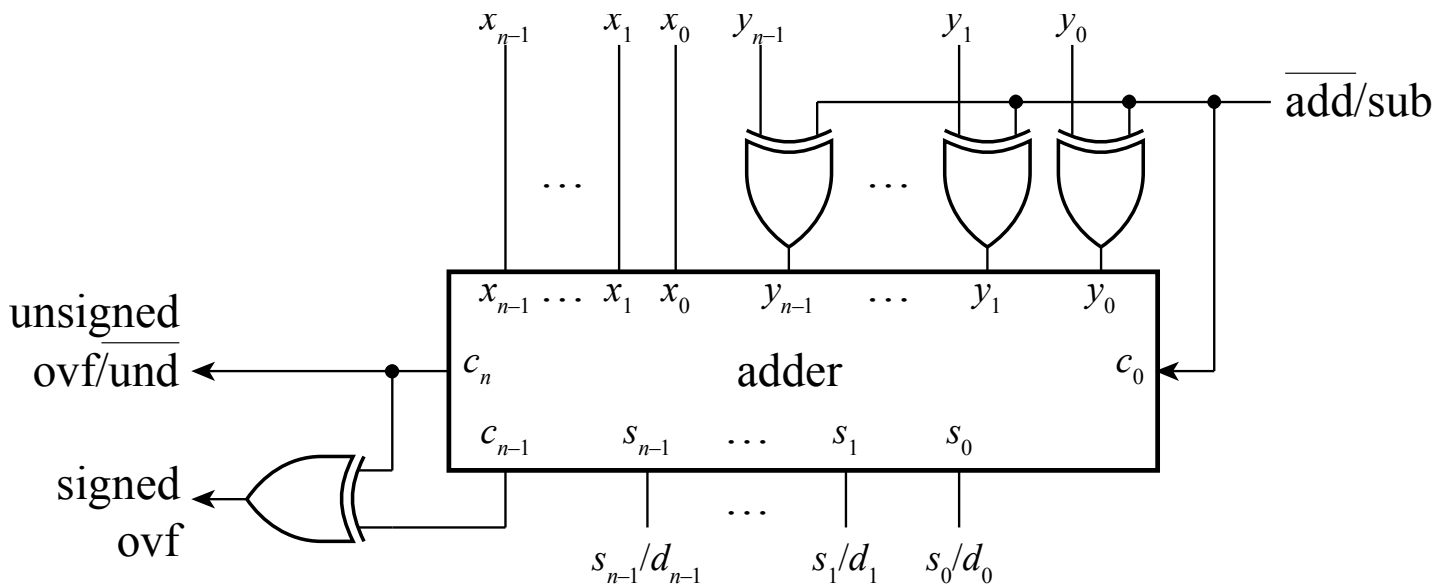
	1	1	1	1	1	0	0	0
42:	0	0	1	0	1	0	1	0
−8:	1	1	1	1	1	0	0	0
34:	0	0	1	0	0	0	1	0

no overflow

- The same hardware as for unsigned integers.

Substracting Signed Integers

- $x - y = x + (-y)$
- `rslt <= x + (not y + 1);`



Abstract Numeric Types

- Take care of the numeric aspects of data...
- ... and *not* the binary encoding.
- `natural range 0 to 2**N-1`

```
signal my_nat : natural range 0 to 255;
signal my_uns : unsigned (7 downto 0);
```

- `integer range -2**(N-1) to 2**(N-1)-1`

```
signal my_int : integer range -128 to 127;
signal my_sig : signed (7 downto 0);
```

Conversions

to → / from ↓	std_logic _vector	signed	unsigned	integer
std_logic _vector	=	signed()	unsigned()	×
signed	std_logic _vector()	=	unsigned()	to_integer()
unsigned	std_logic _vector()	signed()	=	to_integer()
integer	×	to_signed (,length)	to_unsigned (,length)	=