Adders / Subtractors

Last updated 8/23/24

Adders / Subtractors

- Addition and subtraction are common functions in digital designs
  - Long hand addition and subtraction is very tedious
  - VHDL syntheses vendors have provided the ability to do addition and subtraction behaviorally
    - Makes no sense for SLV – just a collection of wires - not supported
    - Supported for signed and unsigned signals
      - signed + signed
      - unsigned + unsigned
      - signed – signed
      - unsigned – unsigned
    - No overflow/underflow detection
      - 4 bit unsigned : 12 + 12 → 8

Adders / Subtractors

- 4 bit unsigned adder

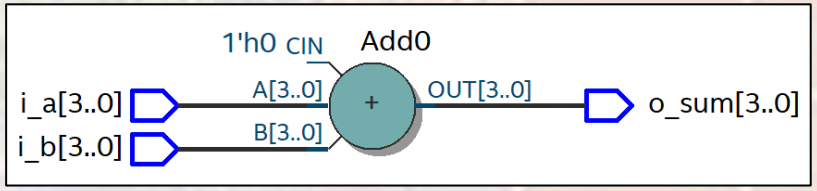
Explicit Type Conversions

```
--
-- adder_unsigned_4.vhdl
--
-- created 8/22/24
-- tj
--
----------------------------------------
--
-- Adder example for the notes
--
----------------------------------------
--
-- inputs:  2 4 bit inputs
-- outputs: 4 bit output, carry out
--
----------------------------------------
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity adder_unsigned_4 is
   port (
         i_a :          in std_logic_vector(3 downto 0);
         i_b :          in std_logic_vector(3 downto 0);

         o_sum :        out std_logic_vector(3 downto 0)
    );
end entity;

architecture behavioral of adder_unsigned_4 is
   -- type conversions
   signal a_sig      :  unsigned(3 downto 0);
   signal b_sig      :  unsigned(3 downto 0);
   signal sum_sig    :  unsigned(3 downto 0);

begin
   -- input conversion
   a_sig <= unsigned(i_a);
   b_sig <= unsigned(i_b);

   add : process(a_sig, b_sig)
   begin
      sum_sig <= a_sig + b_sig;
   end process;

   -- output conversion
   o_sum <= std_logic_vector(sum_sig);

end architecture;
```

In-line Type Conversions

```
--
-- adder_unsigned_4.vhdl
--
-- created 8/22/24
-- tj
--
----------------------------------------
--
-- Adder example for the notes
--
----------------------------------------
--
-- inputs:  2 4 bit inputs
-- outputs: 4 bit output, carry out
--
----------------------------------------
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity adder_unsigned_4 is
   port (
         i_a :          in std_logic_vector(3 downto 0);
         i_b :          in std_logic_vector(3 downto 0);

         o_sum :        out std_logic_vector(3 downto 0)
    );
end entity;

architecture behavioral of adder_unsigned_4 is

begin

   o_sum <= std_logic_vector(unsigned(i_a) + unsigned(i_b));

end architecture;
```

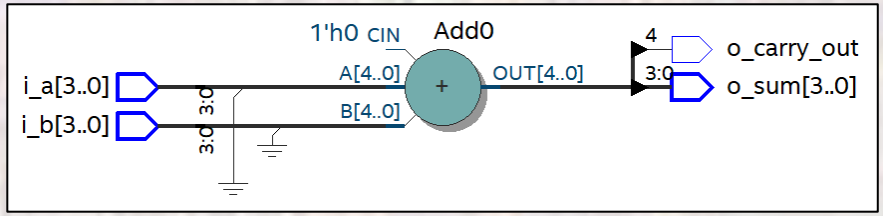
Identical Results



Adders / Subtractors

- 4 bit unsigned adder w/ carry out

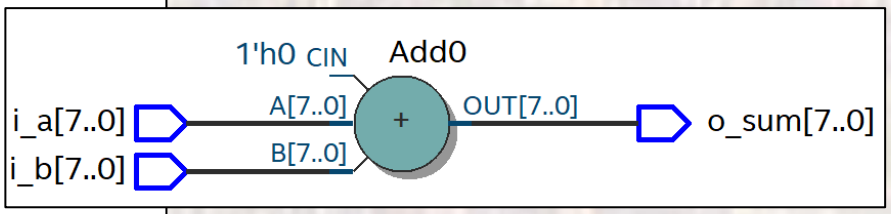
```
--
-- adder_unsigned_co_4.vhdl
--
-- created 8/22/24
-- tj
--
----------------------------------------
--
-- Adder example for the notes
--
----------------------------------------
--
-- inputs:  2 4 bit inputs
-- outputs: 4 bit output, carry out
--
----------------------------------------
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity adder_unsigned_co_4 is
    port (
          i_a :           in std_logic_vector(3 downto 0);
          i_b :           in std_logic_vector(3 downto 0);

          o_sum :         out std_logic_vector(3 downto 0);
          o_carry_out :   out std_logic
     );
end entity;

architecture behavioral of adder_unsigned_co_4 is
    -- type conversions
    signal a_sig       :  unsigned(4 downto 0);
    signal b_sig       :  unsigned(4 downto 0);
    signal carry_sig   :  unsigned(4 downto 0);

    -- temporary signal to overflow value
    signal tmp_sig :  unsigned(4 downto 0);

begin
    -- input conversio
    a_sig <= unsigned('0' & i_a);
    b_sig <= unsigned('0' & i_b);

    add : process(a_sig, b_sig)
    begin
       tmp_sig <= a_sig + b_sig;
    end process;

    -- output conversion
    o_sum <= std_logic_vector(tmp_sig(3 downto 0));
    o_carry_out <= tmp_sig(4);

end architecture;
```



Adders / Subtractors

- N bit unsigned adder

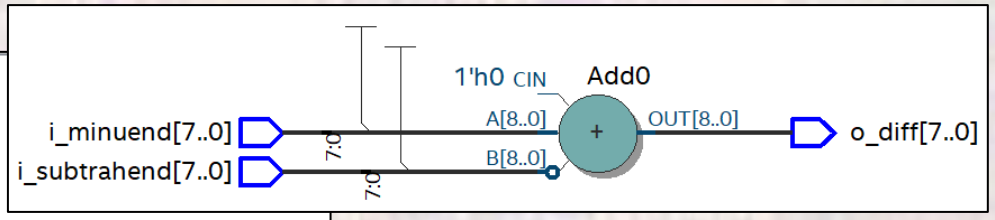
```
--
-- adder_unsigned_n.vhdl
--
-- created 8/22/24
-- tj
--
----------------------------------------
--
-- Adder example for the notes
--
----------------------------------------
--
-- inputs:  2 N bit inputs
-- outputs: N bit output
--
----------------------------------------
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity adder_unsigned_n is
   generic(
            N :   natural  := 8
          );
   port (
         i_a :          in std_logic_vector((N - 1) downto 0);
         i_b :          in std_logic_vector((N - 1) downto 0);

         o_sum :        out std_logic_vector((N - 1) downto 0)
    );
end entity;

architecture behavioral of adder_unsigned_n is

begin

   o_sum <= std_logic_vector(unsigned(i_a) + unsigned(i_b));

end architecture;
```



Adders / Subtractors

- N bit unsigned subtractor

```
--
-- subtractor_unsigned_n.vhdl
--
-- created 8/22/24
-- tj
--
----------------------------------------
--
-- subtractor example for the notes
--
----------------------------------------
--
-- inputs:  2 N bit inputs
-- outputs: N bit output
--
----------------------------------------
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity subtractor_unsigned_n is
   generic(
            N :   natural  := 8
          );
   port (
         i_minuend :       in std_logic_vector((N - 1) downto 0);
         i_subtrahend :    in std_logic_vector((N - 1) downto 0);

         o_diff :          out std_logic_vector((N - 1) downto 0)
    );
end entity;

architecture behavioral of subtractor_unsigned_n is

begin

   o_diff <= std_logic_vector(unsigned(i_minuend) - unsigned(i_subtrahend));

end architecture;
```



subtract

minuend & 1 → a bcde fgh1

~(subtrahend & 1) → + j klmn opq0

r stuv wxy1

diff

subtract

Shift and concatenate 1 → 2x + 1

Shift and concatenate 1 + ~(2y) + 1

2x – 2y

Only select upper bits → /2 x - y