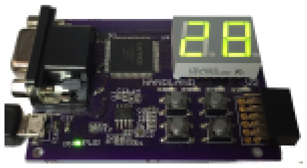
Source : https://www.nandland.com/vhdl/tips/tip-convert-numeric-std-logic-vector-to-integer.html

NANDLAND



Only $65

Now Shipping!

Search nandland.com:

Examples of VHDL Conversions

Using both Numeric_Std and Std_Logic_Arith Package Files

Below are the most common conversions used in VHDL. The page is broken up into two sections. The first half of the page shows conversions using the Numeric_Std package file. The second half of the page shows conversions using the Std_Logic_Arith package file. It is good practice to use the Numeric_Std package as you should not use Std_Logic_Arith. Since many people still insist on using it, both examples are demonstrated below.

Note that many of the below examples use the **'length** VHDL attribute. This attribute makes your code more portable and versatile, so it should be used.

Example Conversions using Numeric Std

Integer to Signed

Integer to Std_Logic_Vector

Integer to Unsigned

Std_Logic_Vector To Integer

Std_Logic_Vector To Signed

Std_Logic_Vector To Unsigned

Signed to Integer

Signed to Std_Logic_Vector

Signed to Unsigned

Example Conversions using Std_Logic_Arith

Convert from Integer to Signed using Numeric_Std

The below example uses the to_signed conversion, which requires two input parameters. The first is the signal that you want to convert, the second is the length of the resulting vector.

```
signal input_3  : integer;
signal output_3 : signed(3 downto 0);

output_3 <= to_signed(input_3, output_3'length);
```

Convert from Integer to Std_Logic_Vector using Numeric_Std

First you need to think about the range of values stored in your integer. Can your integer be positive *and* negative? If so, you will need to use the **to_signed()** conversion. If your integer is only positive, you will need to use the **to_unsigned()** conversion.

Both of these conversion functions require two input parameters. The first is the signal that you want to convert, the second is the length of the resulting vector.

```
signal input_1   : integer;
signal output_1a : std_logic_vector(3 downto 0);
signal output_1b : std_logic_vector(3 downto 0);

-- This line demonstrates how to convert positive integers
output_1a <= std_logic_vector(to_unsigned(input_1, output_1a'length));

-- This line demonstrates how to convert positive or negative integers
output_1b <= std_logic_vector(to_signed(input_1, output_1b'length));
```

Convert from Integer to Unsigned using Numeric_Std

The below example uses the to_unsigned conversion, which requires two input parameters. The first is the signal that you want to convert, the second is the length of the resulting vector.

```
signal input_2  : integer;
signal output_2 : unsigned(3 downto 0);

output_2 <= to_unsigned(input_2, output_2'length);
```

Convert from Std_Logic_Vector to Integer using Numeric_Std

First you need to think about the data that is represented by your std_logic_vector. Is it signed data or is it unsigned data? Signed data means that your std_logic_vector can be a positive *or* negative number. Unsigned data means that your std_logic_vector is *only* a positive number. The example below uses the **unsigned()** typecast, but if your data can be negative you need to use the **signed()** typecast. Once you cast your input std_logic_vector as unsigned or signed, then you can convert it to integer as shown below:

```
signal input_4   : std_logic_vector(3 downto 0);
signal output_4a : integer;
signal output_4b : integer;

-- This line demonstrates the unsigned case
output_4a <= to_integer(unsigned(input_4));

-- This line demonstrates the signed case
output_4b <= to_integer(signed(input_4));
```

Convert from Std_Logic_Vector to Signed using Numeric_Std

This is an easy conversion, all you need to do is cast the std_logic_vector as signed as shown below:

```
signal input_6  : std_logic_vector(3 downto 0);
signal output_6 : signed(3 downto 0);

output_6 <= signed(input_6);
```

Convert from Std_Logic_Vector to Unsigned using Numeric_Std

This is an easy conversion, all you need to do is cast the std_logic_vector as unsigned as shown below:

```
signal input_5  : std_logic_vector(3 downto 0);
signal output_5 : unsigned(3 downto 0);

output_5 <= unsigned(input_5);
```

Convert from Signed to Integer using Numeric_Std

This is an easy conversion, all you need to do is use the to_integer function call from numeric_std as shown below:

```

signal input_10  : signed(3 downto 0);
signal output_10 : integer;

output_10 <= to_integer(input_10);
```

Convert from Signed to Std_Logic_Vector using Numeric_Std

This is an easy conversion, all you need to do is use the std_logic_vector cast as shown below:

```
signal input_11  : signed(3 downto 0);
signal output_11 : std_logic_vector(3 downto 0);

output_11 <= std_logic_vector(input_11);
```

Convert from Signed to Unsigned using Numeric_Std

This is an easy conversion, all you need to do is use the unsigned cast as shown below:

```
signal input_12  : signed(3 downto 0);
signal output_12 : unsigned(3 downto 0);

output_12 <= unsigned(input_12);
```

Convert from Unsigned to Integer using Numeric_Std

This is an easy conversion, all you need to do is use the to_integer function call from numeric_std as shown below:

```
signal input_7  : unsigned(3 downto 0);
signal output_7 : integer;

output_7 <= to_integer(input_7);
```

Convert from Unsigned to Signed using Numeric_Std

This is an easy conversion, all you need to do is use the signed cast as shown below:

```
signal input_9  : unsigned(3 downto 0);
signal output_9 : signed(3 downto 0);

output_9 <= signed(input_9);
```

Convert from Unsigned to Std_Logic_Vector using Numeric_Std

This is an easy conversion, all you need to do is use the std_logic_vector cast as shown below:

```
signal input_8  : unsigned(3 downto 0);
signal output_8 : std_logic_vector(3 downto 0);

output_8 <= std_logic_vector(input_8);
```

Convert from Integer to Signed using Std_Logic_Arith

The below example uses the conv_signed conversion, which requires two input parameters. The first is the signal that you want to convert, the second is the length of the resulting vector.

```
signal input_3  : integer;
signal output_3 : signed(3 downto 0);

output_3 <= conv_signed(input_3, output_3'length);
```

Convert from Integer to Std_Logic_Vector using Std_Logic_Arith

The below example uses the conv_std_logic_vector conversion, which requires two input parameters. The first is the signal that you want to convert, the second is the length of the resulting vector.

One thing to note here is that if you input a negative number into this conversion, then your output std_logic_vector will be represented in 2's complement signed notation.

```
signal input_1  : integer;
signal output_1 : std_logic_vector(3 downto 0);

output_1 <= conv_std_logic_vector(input_1, output_1'length);
```

Convert from Integer to Unsigned using Std_Logic_Arith

The below example uses the conv_unsigned conversion, which requires two input parameters. The first is the signal that you want to convert, the second is the length of the resulting vector.

```
signal input_2  : integer;
signal output_2 : unsigned(3 downto 0);

output_2 <= conv_unsigned(input_2, output_2'length);
```

Convert from Std_Logic_Vector to Integer using Std_Logic_Arith

First you need to think about the data that is represented by your std_logic_vector. Is it signed data or is it unsigned data? Signed data means that your std_logic_vector can be a positive *or* negative number. Unsigned data means that your std_logic_vector is *only* a positive number. The example below uses the **unsigned()** typecast, but if your data can be negative you need to use the **signed()** typecast. Once your input std_logic_vector is unsigned or signed, then you can convert it to integer as shown below:

```
signal input_4   : std_logic_vector(3 downto 0);
signal output_4a : integer;
signal output_4b : integer;

-- This line demonstrates the unsigned case
output_4a <= conv_integer(unsigned(input_4));

-- This line demonstrates the signed case
output_4b <= conv_integer(signed(input_4));
```

Convert from Std_Logic_Vector to Signed using Std_Logic_Arith

This is an easy conversion, all you need to do is cast the std_logic_vector as signed as shown below:

```
signal input_6  : std_logic_vector(3 downto 0);
signal output_6 : signed(3 downto 0);

output_6 <= signed(input_6);
```

Convert from Std_Logic_Vector to Unsigned using Std_Logic_Arith

This is an easy conversion, all you need to do is cast the std_logic_vector as unsigned as shown below:

```
signal input_5  : std_logic_vector(3 downto 0);
signal output_5 : unsigned(3 downto 0);

output_5 <= unsigned(input_5);
```

Convert from Signed to Integer using Std_Logic_Arith

This is an easy conversion, all you need to do is use the conv_integer function call from std_logic_arith as shown below:

```

signal input_10  : signed(3 downto 0);
signal output_10 : integer;

output_10 <= conv_integer(input_10);
```

Convert from Signed to Std_Logic_Vector using Std_Logic_Arith

This is an easy conversion, all you need to do is use the std_logic_vector cast as shown below:

```
signal input_11  : signed(3 downto 0);
signal output_11 : std_logic_vector(3 downto 0);

output_11 <= std_logic_vector(input_11);
```

Convert from Signed to Unsigned using Std_Logic_Arith

This is an easy conversion, all you need to do is use the unsigned cast as shown below:

```
signal input_12  : signed(3 downto 0);
signal output_12 : unsigned(3 downto 0);

output_12 <= unsigned(input_12);
```

Convert from Unsigned to Integer using Std_Logic_Arith

This is an easy conversion, all you need to do is use the conv_integer function call from std_logic_arith as shown below:

```
signal input_7  : unsigned(3 downto 0);
signal output_7 : integer;

output_7 <= conv_integer(input_7);
```

Convert from Unsigned to Signed using Std_Logic_Arith

This is an easy conversion, all you need to do is use the signed cast as shown below:

```
signal input_9  : unsigned(3 downto 0);
signal output_9 : signed(3 downto 0);

output_9 <= signed(input_9);
```

Convert from Unsigned to Std_Logic_Vector using Std_Logic_Arith

This is an easy conversion, all you need to do is use the std_logic_vector typecast as shown below:

```

signal input_8  : unsigned(3 downto 0);
signal output_8 : std_logic_vector(3 downto 0);

output_8 <= std_logic_vector(input_8);
```

Most Popular Nandland Pages

Most Popular Nandland Pages

Avoid Latches in your FPGA
Learn what is a latch and how they are created. Usually latches are created by accident. Learn the simple trick to avoid them.

Convert Binary to BCD
Binary Coded Decimal (BCD) is used to display digits on a 7-Segment display, but internal signals in an FPGA use binary. This module converts binary to BCD using a double-dabbler.

Example Code for UART
See example VHDL and Verilog code for a UART. See the basics of UART design and use this fully functional design to implement your own UART. Good example of a state machine.

What is a FIFO?
Learn the basics of a FIFO. There are two simple rules governing FIFOs: Never write to a full FIFO and Never Read from an Empty FIFO...