numeric_lib

vhdl_lib

2021-08-17

目次

1 numeric_lib

signed, unsgined, std_logic_vector の演算のライブラリです。
- 四則演算
- ビット演算 - クリップ
- 丸め
- 他

unsigned/signed は関数名共通です。
std_logic_vector 用は、signed であれば関数名の末尾に”_s“, unsigned では末尾に”_u“が付きます。
また、unsigned と signed の演算であれば、関数名の末尾に”_us” と付きます。
(例: f_add_us(): unsigned + signed)
また 2 個の引数がある場合、ビット幅は同じである必要はありません。

Function/Procedure	description
f_add()	+
f_sub()	-
f_mul()	*
f_div()	/
f_or_reduce()	bit or
f_and_reduce()	bit and
f_clip()	clip
f_truncate()	round
f_round_to_ward_zero()	round
f_round_half_up()	round
f_round_to_even()	round
f_round()	round
clog2()	log2(x) for cal bit width
f_increment()	Increment for Counter

1.1 f_add()

```
function f_add(a,b: in unsigned) return unsigned;
function f_add_u(a,b: in std_logic_vector) return std_logic_vector;
function f_add(a: in unsigned; b: in signed) return signed;
function f_add_us(a,b: in std_logic_vector) return std_logic_vector;
function f_add(a: in signed; b: in unsigned) return signed;
function f_add_su(a,b: in std_logic_vector) return std_logic_vector;
function f_add(a,b: in signed) return signed;
function f_add_s(a,b: in std_logic_vector) return std_logic_vector;
```

a と b の加算を行い、ビット拡張した値を返します。出力ビット幅は、a と b のビット幅の大きな方+1 ビットです。

1.2 f_sub()

```
function f_sub(a,b: in unsigned) return signed;
function f_sub_u(a,b: in std_logic_vector) return std_logic_vector;
function f_sub(a: in unsigned; b: in signed) return signed;
function f_sub_us(a,b: in std_logic_vector) return std_logic_vector;
function f_sub(a: in signed; b: in unsigned) return signed;
function f_sub_su(a,b: in std_logic_vector) return std_logic_vector;
function f_sub(a,b: in signed) return signed;
function f_sub_s(a,b: in std_logic_vector) return std_logic_vector;
‘
```

a と b の減算を行い、ビット拡張した値を返します。出力ビット幅は、a と b のビット幅の大きな方+1 ビットです。

1.3 f_mul()

```
function f_mul(a, b: in unsigned) return unsigned;
function f_mul_u(a,b: in std_logic_vector) return std_logic_vector;
function f_mul(a: in unsigned; b: in signed) return signed;
function f_mul_us(a,b: in std_logic_vector) return std_logic_vector;
function f_mul(a: in signed; b: in unsigned) return signed;
function f_mul_su(a,b: in std_logic_vector) return std_logic_vector;
function f_mul(a,b: in signed) return signed;
function f_mul_s(a,b: in std_logic_vector) return std_logic_vector;
```

a と b の乗算を行い、ビット拡張した値を返します。出力ビット幅は、a と b のビット幅を足したビット幅です。

1.4 f_div()

```
function f_div(a: in unsigned; b: in unsigned) return unsigned;
function f_div_u(a,b: in std_logic_vector) return std_logic_vector;
function f_div(a: in unsigned; b: in signed) return signed;
function f_div_us(a,b: in std_logic_vector) return std_logic_vector;
function f_div(a: in signed; b: in unsigned) return signed;
function f_div_su(a,b: in std_logic_vector) return std_logic_vector;
function f_div(a: in signed; b: in signed) return signed;
function f_div_s(a,b: in std_logic_vector) return std_logic_vector;
```

a と b の除算を行い、ビット拡張した値を返します。出力ビット幅は、b(除数) が unsigned の場合 a(被除数) のビット幅、b(除数) が signed の場合は a(被除数) のビット幅+1 ビットです。

1.5 f_or_reduce()

```
function f_or_reduce(a: in unsigned) return std_logic;
function f_or_reduce(a: in signed) return std_logic;
function f_or_reduce(a: in std_logic_vector) return std_logic;
```

全ビットの or を取った値を返します。

1.6 f_and_reduce()

```
function f_and_reduce(a: in unsigned) return std_logic;
function f_and_reduce(a: in signed) return std_logic;
function f_and_reduce(a: in std_logic_vector) return std_logic;
```

全ビットの and を取った値を返します。

1.7 f_clip()

```
function f_clip(a: in unsigned; constant n: in natural) return unsigned;
function f_clip_u(a: in std_logic_vector; constant n: in natural) return
std_logic_vector;
function f_clip(a: in signed; constant n: in natural) return signed;
```

```
function f_clip_s(a: in std_logic_vector; constant n: in natural) return
std_logic_vector;
```

n ビットでクリップした a を返します。出力ビット幅は、n ビットとなります。

1.8 f_truncate()

```
function f_truncate(a: unsigned; constant len: natural) return unsigned;
function f_truncate(a: signed; constant len: natural) return signed;
```

a の丸めを行い、len で指定したビット幅を出力します。丸め方法は、切り捨てです。

1.9 f_round_toward_zero()

```
function f_round_toward_zero(a: signed; constant len: natural) return
signed;
```

a の丸めを行い、len で指定したビット幅を出力します。丸め方法は、0 への丸めです。

1.10 f_round_half_up()

```
function f_round_half_up(a: unsigned; constant len: natural) return unsigned
;
function f_round_half_up(a: signed; constant len: natural) return signed;
```

a の丸めを行い、len で指定したビット幅を出力します。丸め方法は、0 捨 1 入です。

1.11 f_round_to_even()

```
function f_round_to_even(a: unsigned; constant len: natural) return unsigned
;
function f_round_to_even(a: signed; constant len: natural) return signed;
```

a の丸めを行い、len で指定したビット幅を出力します。丸め方法は、偶数への丸めです。

1.12 f_round()

```
alias f_round is f_round_half_up[unsigned, natural return unsigned];
alias f_round is f_round_half_up[signed, natural return signed];
```

f_round_half_up() への alias です。

1.13 clog2()

```
function clog2(a: positive) return positive;
```

log2 の計算を行います。
小数点以下は切り上げを行い、自然数を返します。
RAM の Word 数からアドレスビット幅の計算等で使用します。
Verilog の$clog2() 相当。
(real 使っているため、合成用回路での使用は非推奨)
Example

```
constant DEPTH: positive:= 256;
constant ADDRESS_WIDTH: positive:= clog2(DEPTH); -- 8
```

1.14 f_increment()

```
function f_increment(slv: std_logic_vector) return std_logic_vector;
```

std_logic_vector に 1 を加算し、返します。
オーバフローの考慮はしません (0xFF は 0x00 を返す)。
カウンターで使用します。