1164 PACKAGES QUICK REFERENCE CARD

Revision 2.2

()	Grouping	[ ]	Optional
{}	Repeated	|	Alternative
bold	As is	CAPS	User Identifier
italic	VHDL-93	c	commutative

b	::= BIT
bv	::= BIT_VECTOR
u/l	::= STD_ULOGIC/STD_LOGIC
uv	::= STD_ULOGIC_VECTOR
lv	::= STD_LOGIC_VECTOR
un	::= UNSIGNED
sg	::= SIGNED
in	::= INTEGER
na	::= NATURAL
sm	::= SMALL_INT (subtype INTEGER range 0 to 1)

1. IEEE's STD_LOGIC_1164

1.1 Logic Values

'U'	Uninitialized
'X'/'W'	Strong/Weak unknown
'0'/'L'	Strong/Weak 0
'1'/'H'	Strong/Weak 1
'Z'	High Impedance
'-'	Don't care

1.2 Predefined Types

STD_ULOGIC	Base type
Subtypes:	
STD_LOGIC	Resolved STD_ULOGIC
X01	Resolved X, 0 & 1
X01Z	Resolved X, 0, 1 & Z
UX01	Resolved U, X, 0 & 1
UX01Z	Resolved U, X, 0, 1 & Z

STD_ULOGIC_VECTOR(na **to | downto** na**)**
Array of STD_ULOGIC

STD_LOGIC_VECTOR(na **to | downto** na**)**
Array of STD_LOGIC

1.3 Overloaded Operators

Description	Left	Operator	Right
bitwise-and	u/l,uv,lv	**and**, **nand**	u/l,uv,lv
bitwise-or	u/l,uv,lv	**or**, **nor**	u/l,uv,lv
bitwise-xor	u/l,uv,lv	**xor**, **xnor**	u/l,uv,lv
bitwise-not		**not**	u/l,uv,lv

1.4 Conversion Functions

From	To	Function
u/l	b	**TO_BIT(**from[**,** xmap]**)**
uv,lv	bv	**TO_BITVECTOR(**from[**,** xmap]**)**
b	u/l	**TO_STDULOGIC(**from**)**
bv,uv	lv	**TO_STDLOGICVECTOR(**from**)**
bv,lv	uv	**TO_STDULOGICVECTOR(**from**)**

2. IEEE's NUMERIC_STD

2.1 Predefined Types

UNSIGNED(na **to | downto** na**)**	Array of STD_LOGIC
SIGNED(na **to | downto** na**)**	Array of STD_LOGIC

2.2 Overloaded Operators

Left	Op	Right	Return
	abs	sg	sg
	-	sg	sg
un	**+,-,*,/,rem,mod**	un	un
sg	**+,-,*,/,rem,mod**	sg	sg
un	**+,-,*,/,rem,mod** c	na	un
sg	**+,-,*,/,rem,mod** c	in	sg
un	**<,>,<=,>=,=,/=**	un	bool
sg	**<,>,<=,>=,=,/=**	sg	bool
un	**<,>,<=,>=,=,/=** c	na	bool
sg	**<,>,<=,>=,=,/=** c	in	bool

2.3 Predefined Functions

SHIFT_LEFT(un, na**)**	un
SHIFT_RIGHT(un, na**)**	un
SHIFT_LEFT(sg, na**)**	sg
SHIFT_RIGHT(sg, na**)**	sg
ROTATE_LEFT(un, na**)**	un
ROTATE_RIGHT(un, na**)**	un
ROTATE_LEFT(sg, na**)**	sg
ROTATE_RIGHT(sg, na**)**	sg
RESIZE(sg, na**)**	sg
RESIZE(un, na**)**	un
STD_MATCH(u/l, u/l**)**	bool
STD_MATCH(uv, uv**)**	bool
STD_MATCH(lv, lv**)**	bool
STD_MATCH(un, un**)**	bool
STD_MATCH(sg, sg**)**	bool

2.4 Conversion Functions

From	To	Function
un,lv	sg	**SIGNED(**from**)**
sg,lv	un	**UNSIGNED(**from**)**
un,sg	lv	**STD_LOGIC_VECTOR(**from**)**
un,sg	in	**TO_INTEGER(**from**)**
na	un	**TO_UNSIGNED(**from, size**)**
in	sg	**TO_SIGNED(**from, size**)**

3. IEEE's NUMERIC_BIT

3.1 Predefined Types

UNSIGNED(na **to | downto** na**)**	Array of BIT
SIGNED(na **to | downto** na**)**	Array of BIT

3.2 Overloaded Operators

Left	Op	Right	Return
	abs	sg	sg
	-	sg	sg
un	**+,-,*,/,rem,mod**	un	un
sg	**+,-,*,/,rem,mod**	sg	sg
un	**+,-,*,/,rem,mod** c	na	un
sg	**+,-,*,/,rem,mod** c	in	sg
un	**<,>,<=,>=,=,/=**	un	bool
sg	**<,>,<=,>=,=,/=**	sg	bool
un	**<,>,<=,>=,=,/=** c	na	bool
sg	**<,>,<=,>=,=,/=** c	in	bool

3.3 Predefined Functions

SHIFT_LEFT(un, na**)**	un
SHIFT_RIGHT(un, na**)**	un
SHIFT_LEFT(sg, na**)**	sg
SHIFT_RIGHT(sg, na**)**	sg
ROTATE_LEFT(un, na**)**	un
ROTATE_RIGHT(un, na**)**	un
ROTATE_LEFT(sg, na**)**	sg
ROTATE_RIGHT(sg, na**)**	sg
RESIZE(sg, na**)**	sg
RESIZE(un, na**)**	un

3.4 Conversion Functions

From	To	Function
un,bv	sg	**SIGNED(**from**)**
sg,bv	un	**UNSIGNED(**from**)**
un,sg	bv	**BIT_VECTOR(**from**)**
un,sg	in	**TO_INTEGER(**from**)**
na	un	**TO_UNSIGNED(**from**)**
in	sg	**TO_SIGNED(**from**)**

See reverse side for additional information.

4. Synopsys’ STD_LOGIC_ARITH

4.1 Predefined Types

UNSIGNED(na **to | downto** na**)**	Array of STD_LOGIC
SIGNED(na **to | downto** na**)**	Array of STD_LOGIC
SMALL_INT	Integer subtype, 0 or 1

4.2 Overloaded Operators

Left	Op	Right	Return
	abs	sg	sg,lv
	-	sg	sg,lv
un	**+,-,***	un	un,lv
sg	**+,-,***	sg	sg,lv
sg	**+,-,***	un	sg,lv
un	**+,-** $_{c}$	in	un,lv
sg	**+,-** $_{c}$	in	sg,lv
un	**+,-** $_{c}$	u/l	un,lv
sg	**+,-** $_{c}$	u/l	sg,lv
un	**<,>,<=,>=,=,/=**	un	bool
sg	**<,>,<=,>=,=,/=**	sg	bool
un	**<,>,<=,>=,=,/=** $_{c}$	in	bool
sg	**<,>,<=,>=,=,/=** $_{c}$	in	bool

4.3 Predefined Functions

SHL(un, un**)**	un	**SHR(**un, un**)**	un
SHL(sg, un**)**	sg	**SHR(**sg, un**)**	sg
EXT(lv, in**)**	lv	zero-extend	
SEXT(lv, in**)**	lv	sign-extend	

4.4 Conversion Functions

From	To	Function
un,lv	sg	**SIGNED(**from**)**
sg,lv	un	**UNSIGNED(**from**)**
sg,un	lv	**STD_LOGIC_VECTOR(**from**)**
un,sg	in	**CONV_INTEGER(**from**)**
in,un,sg,u	un	**CONV_UNSIGNED(**from, size**)**
in,un,sg,u	sg	**CONV_SIGNED(**from, size**)**
in,un,sg,u	lv	**CONV_STD_LOGIC_VECTOR(**from, size**)**

5. Synopsys’ STD_LOGIC_UNSIGNED

5.1 Overloaded Operators

Left	Op	Right	Return
	+	lv	lv
lv	**+,-,***	lv	lv
lv	**+,-**$_{c}$	in	lv
lv	**+,-** $_{c}$	u/l	lv
lv	**<,>,<=,>=,=,/=**	lv	bool
lv	**<,>,<=,>=,=,/=** $_{c}$	in	bool

5.2 Conversion Functions

From	To	Function
lv	in	**CONV_INTEGER**(from)

6. Synopsys’ STD_LOGIC_SIGNED

6.1 Overloaded Operators

Left	Op	Right	Return
	abs	lv	lv
	+,**-**	lv	lv
lv	**+**,**-**,*****	lv	lv
lv	**+**,**-**$_{c}$	in	lv
lv	**+**,**-** $_{c}$	u/l	lv
lv	**<,>,<=,>=,=,/=**	lv	bool
lv	**<,>,<=,>=,=,/=** $_{c}$	in	bool

6.2 Conversion Functions

From	To	Function
lv	in	**CONV_INTEGER(**from**)**

7. Synopsys’ STD_LOGIC_MISC

7.1 Predefined Functions

AND_REDUCE(lv | uv**)**	u/l
[X]OR_REDUCE(lv | uv**)**	u/l
[N]AND_REDUCE(lv | uv**)**	UX01
OR_REDUCE(lv | uv**)**	UX01
NOR_REDUCE(lv | uv**)**	UX01
XOR_REDUCE(lv | uv**)**	UX01
XNOR_REDUCE(lv | uv**)**	UX01

8. Exemplar’s STD_LOGIC_ARITH

8.1 Overloaded Operators

Left	Op	Right	Return
	+,-,*,	u/l	u/l
	abs	u/l	u/l

8.2 Predefined Functions

sl(u/l**,** in**)**	u/l
sl2(u/l**,** in**)**	u/l
sr(u/l**,** in**)**	u/l
sr2(u/l**,** in**)**	u/l
add(u/l**)**	u/l
add2(u/l**)**	u/l
sub(u/l**)**	u/l
sub2(u/l**)**	u/l
mult(u/l**)**	u/l
mult2(u/l**)**	u/l
extend(u/l, in**)**	u/l
extend2(u/l, in**)**	u/l
comp2(u/l**)**	u/l

8.3 Conversion Functions

From	To	Function
bool	uv	**bool2elb**
uv	bool	**elb2bool**
u/l	na	**evec2int**
in	u/l	**int2evec (**size**)**
uv	na	**elb2int**

9. Mentor’s STD_LOGIC_ARITH

9.1 Predefined Types

UNSIGNED(na **to** | **downto** na**)**	Array of STD_LOGIC
SIGNED(na **to** | **downto** na)	Array of STD_LOGIC

9.2 Overloaded Operators

Left	Op	Right	Return
	abs	sg	sg
	-	sg	sg
u/l	**+,-**	u/l	u/l
uv	**+,-,*,/,mod**,**rem**,******	uv	uv
lv	**+,-,*,/,mod**,**rem**,******	lv	lv
un	**+,-,*,/,mod**,**rem**,******	un	un
sg	**+,-,*,/,mod**,**rem**,******	sg	sg
un	**<**,**>**,**<=**,**>=**,**=**,**/=**	un	bool
sg	**<**,**>**,**<=**,**>=**,**=**,**/=**	sg	bool
	not	un	un
	not	sg	sg
un	**and**,**nand**,**or,nor**,**xor**	un	un
sg	**and**,**nand**,**or,nor**,**xor**,***xnor***	sg	sg
uv	***sla***,***sra***,***sll,srl***,***rol***,***ror***	uv	uv
lv	***sla***,***sra***,***sll,srl***,***rol***,***ror***	lv	lv
un	***sla***,***sra***,***sll,srl***,***rol***,***ror***	un	un
sg	***sla***,***sra***,***sll,srl***,***rol***,***ror***	sg	sg

9.3 Predefined Functions

ZERO_EXTEND(uv | lv | un**,** na**)**	same
ZERO_EXTEND(u/l**,** na**)**	lv
SIGN_EXTEND(sg**,** na**)**	sg
AND_REDUCE(uv | lv | un | sg**)**	u/l
OR_REDUCE(uv | lv | un | sg**)**	u/l
XOR_REDUCE(uv | lv | un | sg**)**	u/l

9.4 Conversion Functions

From	To	Function
u/l,uv,lv,un,sg	in	**TO_INTEGER(**from**)**
u/l,uv,lv,un,sg	in	**CONV_INTEGER(**from**)**
bool	u/l	**TO_STDLOGIC(**from**)**
na	un	**TO_UNSIGNED(**from,size**)**
na	un	**CONV_UNSIGNED(**from,size**)**
in	sg	**TO_SIGNED(**from,size**)**
in	sg	**CONV_SIGNED(**from,size**)**
na	lv	**TO_STDLOGICVECTOR(**from,size**)**
na	uv	**TO_STDULOGICVECTOR(**from,size**)**

Qualis Design Corporation

Elite Training and Consulting in Reuse and Methodology

Phone: +1.503.670.7200 FAX: +1.503.670.0809
Email: info@qualis.com Web: http://www.qualis.com

Also available: VHDL Quick Reference Card
Verilog Quick Reference Card