QUALIS DESIGN CORPORATION

1164 PACKAGES QUICK REFERENCE CARD

REVISION 1.0

()	Grouping	[ ]	Optional
{}	Repeated	|	Alternative
bold	As is	CAPS	User Identifier

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

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**	u/l,uv,lv
bitwise-or	u/l,uv,lv	**or**	u/l,uv,lv
bitwise-xor	u/l,uv,lv	**xor**	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,ul	lv	**TO_STDLOGICVECTOR(**from**)**
bv,lv	uv	**TO_STDULOGICVECTOR(**from**)**

1.5. PREDICATES

RISING_EDGE(SIGID**)**	Rise edge on signal ?
FALLING_EDGE(SIGID**)**	Fall edge on signal ?
IS_X(OBJID**)**	Object contains 'X' ?

2. IEEE'S NUMERIC_STD

2.1. PREDEFINED TYPES

UNSIGNED(na **to** | **downto** na**)**
SIGNED(na **to** | **downto** na)
Arrays 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

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**)**
in	sg	**TO_SIGNED(**from**)**

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

4.2. OVERLOADED OPERATORS

Left	Op	Right	Return
	abs	sg	sg,lv
	+	un	un,lv
	+,-	sg	sg,lv
un	**+,-,*,/**	un	un,lv
sg	**+,-,*,/**	sg	sg,lv
sg	**+,-,*,/** $_c$	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-exten

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_MISC

5.1. PREDEFINED FUNCTIONS

AND_REDUCE(lv | uv**)**	u/l
OR_REDUCE(lv | uv**)**	u/l
XOR_REDUCE(lv | uv**)**	u/l

6. SYNOPSYS' STD_LOGIC_UNSIGNED

6.1. 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

6.2. CONVERSION FUNCTIONS

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

7. SYNOPSYS' STD_LOGIC_SIGNED

7.1. 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

7.2. CONVERSION FUNCTIONS

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

8. SYNOPSYS' STD_LOGIC_TEXTIO

Read/write binary values
READ(line**,** u/l**,** [good]**);**
READ(line**,** uv**,** [good]**);**
READ(line**,** lv**,** [good]**);**
WRITE(line**,** u/l**,** [justify]**,** [width]**);**
WRITE(line**,** uv**,** [justify]**,** [width]**);**
WRITE(line**,** lv**,** [justify]**,** [width]**);**

Read/write octal values
OREAD(line**,** uv**,** [good]**);**
OREAD(line**,** lv**,** [good]**);**
OWRITE(line**,** uv**,** [justify]**,** [width]**);**
OWRITE(line**,** lv**,** [justify]**,** [width]**);**

Read/write hexadecimal values
HREAD(line**,** uv**,** [good]**);**
HREAD(line**,** lv**,** [good]**);**
HWRITE(line**,** uv**,** [justify]**,** [width]**);**
HWRITE(line**,** lv**,** [justify]**,** [width]**);**

9. CADENCE'S STD_LOGIC_ARITH

9.1. OVERLOADED OPERATORS

Left	Op	Right	Return
	+	uv	uv
	+	lv	lv
u/l	**+,-,*,/**	u/l	u/l
lv	**+,-,*,/**	lv	lv
lv	**+,-,*,/** $_c$	u/l	lv
lv	**+,-** $_c$	in	lv
uv	**+,-,***	uv	uv
uv	**+,-,*** $_c$	u/l	uv
uv	**+,-** $_c$	in	uv
lv	**<,>,<=,>=,=,/=** $_c$	in	bool
uv	**<,>,<=,>=,=,/=** $_c$	in	bool

9.2. PREDEFINED FUNCTIONS

C-like ?: replacements:

COND_OP(bool**,** lv**,** lv**)**	lv
COND_OP(bool**,** uv**,** uv**)**	uv
COND(bool**,** u/l**,** u/l**)**	u/l

Shift operations:

SH_LEFT(lv**,** na**)**	lv
SH_LEFT(uv**,** na**)**	uv
SH_RIGHT(lv**,** na**)**	lv
SH_RIGHT(uv**,** na**)**	uv

Resize functions:

ALIGN_SIZE(lv**,** na**)**	lv
ALIGN_SIZE(uv**,** na**)**	uv
ALIGN_SIZE(u/l**,** na**)**	lv
ALIGN_SIZE(u/l**,** na**)**	uv

9.3. CONVERSION FUNCTIONS

From	To	Function
lv,uv,u/l	in	**TO_INTEGER(**from**)**
in	lv	**TO_STDLOGICVECTOR(**from**,** size**)**
in	uv	**TO_STDULOGICVECTOR(**from**,** size**)**

Qualis Design Corporation
Beaverton, OR USA
Phone: +1-503-531-0377 FAX: +1-503-629-5525
E-mail: info@qualis.com