Paquete STANDARD

```
-- This is Package STANDARD as defined in the VHDL Language Reference Manual.
--
--   NOTE: VCOM and VSIM will not work properly if these declarations
--         are modified.

-- Version information: @(#)standard.vhd   1.1 91/05/21

package standard is
        type boolean is (false,true);
        type bit is ('0', '1');
TYPE  character IS (
        NUL,    SOH,    STX,    ETX,    EOT,    ENQ, ACK,       BEL,
        BS,     HT,     LF,     VT,     FF,     CR,     SO,     SI,
        DLE,    DC1,    DC2,    DC3,    DC4,    NAK,    SYN,    ETB,
        CAN, EM,        SUB,    ESC,    FSP,    GSP,    RSP,    USP,
        ' ',    '!',    '"',    '#',    '$',    '%',    '&',    ''',
        '(',    ')',    '*',    '+',    ',',    '-',    '.',    '/',
        '0',    '1',    '2',    '3',    '4',    '5',    '6',    '7',
        '8',    '9',    ':',    ';',    '<',    '=',    '>',    '?',
        '@',    'A',    'B',    'C',    'D',    'E',    'F',    'G',
        'H',    'I',    'J',    'K',    'L',    'M',    'N',    'O',
        'P',    'Q',    'R',    'S',    'T',    'U',    'V',    'W',
        'X',    'Y',    'Z',    '[',    '\',    ']',    '^',    '_',
        '`',    'a',    'b',    'c',    'd',    'e',    'f',    'g',
        'h',    'i',    'j',    'k',    'l',    'm',    'n',    'o',
        'p',    'q',    'r',    's',    't',    'u',    'v',    'w',
        'x',    'y',    'z',    '{',    '|',    '}',    '~',    DEL);
        type severity_level is (note, warning, error, failure);
        type integer is range -2147483648 to 2147483647;
        type real is range -1.0E38 to 1.0E38;
        type time is range -2147483647 to 2147483647
                units
                        fs;
                        ps = 1000 fs;
                        ns = 1000 ps;
                        us = 1000 ns;
                        ms = 1000 us;
                        sec = 1000 ms;
                        min = 60 sec;
                        hr = 60 min;
                end units;
        function now return time;
        subtype natural is integer range 0 to integer'high;
        subtype positive is integer range 1 to integer'high;
        type string is array (positive range <>) of character;
        type bit_vector is array (natural range <>) of bit;
end standard;
```

Paquete TEXTIO

```
-- Package TEXTIO as defined in Chapter 14 of the IEEE Standard VHDL
--   Language Reference Manual (IEEE Std. 1076-1987), as modified
--   by the Issues Screening and Analysis Committee (ISAC), a subcommittee
--   of the VHDL Analysis and Standardization Group (VASG) on
--   10 November, 1988.  See "The Sense of the VASG", October, 1989.

-- Version information: @(#)textio.vhd        1.6 12/19/91

package TEXTIO is
   type LINE is access string;
   type TEXT is file of string;
   type SIDE is (right, left);
   subtype WIDTH is natural;

   file input : TEXT is in "STD_INPUT";
   file output : TEXT is out "STD_OUTPUT";

   procedure READLINE(variable f:in TEXT; L: inout LINE);

   procedure READ(L:inout LINE; VALUE: out bit; GOOD : out BOOLEAN);
   procedure READ(L:inout LINE; VALUE: out bit);

   procedure READ(L:inout LINE; VALUE: out bit_vector; GOOD : out BOOLEAN);
   procedure READ(L:inout LINE; VALUE: out bit_vector);

   procedure READ(L:inout LINE; VALUE: out BOOLEAN; GOOD : out BOOLEAN);
   procedure READ(L:inout LINE; VALUE: out BOOLEAN);

   procedure READ(L:inout LINE; VALUE: out character; GOOD : out BOOLEAN);
   procedure READ(L:inout LINE; VALUE: out character);

   procedure READ(L:inout LINE; VALUE: out integer; GOOD : out BOOLEAN);
   procedure READ(L:inout LINE; VALUE: out integer);

   procedure READ(L:inout LINE; VALUE: out real; GOOD : out BOOLEAN);
   procedure READ(L:inout LINE; VALUE: out real);

   procedure READ(L:inout LINE; VALUE: out string; GOOD : out BOOLEAN);
   procedure READ(L:inout LINE; VALUE: out string);

   procedure READ(L:inout LINE; VALUE: out time; GOOD : out BOOLEAN);
   procedure READ(L:inout LINE; VALUE: out time);

   procedure WRITELINE(f : out TEXT; L : inout LINE);
```

```
    procedure WRITE(L : inout LINE; VALUE : in bit;
                    JUSTIFIED: in SIDE := right;
                    FIELD: in WIDTH := 0);

    procedure WRITE(L : inout LINE; VALUE : in bit_vector;
                    JUSTIFIED: in SIDE := right;
                    FIELD: in WIDTH := 0);

    procedure WRITE(L : inout LINE; VALUE : in BOOLEAN;
                    JUSTIFIED: in SIDE := right;
                    FIELD: in WIDTH := 0);

    procedure WRITE(L : inout LINE; VALUE : in character;
                    JUSTIFIED: in SIDE := right;
                    FIELD: in WIDTH := 0);

    procedure WRITE(L : inout LINE; VALUE : in integer;
                    JUSTIFIED: in SIDE := right;
                    FIELD: in WIDTH := 0);

    procedure WRITE(L : inout LINE; VALUE : in real;
                    JUSTIFIED: in SIDE := right;
                    FIELD: in WIDTH := 0;
                    DIGITS: in NATURAL := 0);

    procedure WRITE(L : inout LINE; VALUE : in string;
                    JUSTIFIED: in SIDE := right;
                    FIELD: in WIDTH := 0);

    procedure WRITE(L : inout LINE; VALUE : in time;
                    JUSTIFIED: in SIDE := right;
                    FIELD: in WIDTH := 0;
                    UNIT: in TIME := ns);

    -- function ENDLINE(variable L : in LINE) return BOOLEAN;
    --
    -- Function ENDLINE as declared cannot be legal VHDL, and
    --   the entire function was deleted from the definition
    --   by the Issues Screening and Analysis Committee (ISAC),
    --   a subcommittee of the VHDL Analysis and Standardization
    --   Group (VASG) on 10 November, 1988.  See "The Sense of
    --   the VASG", October, 1989, VHDL Issue Number 0032.
end;
```

Paquete STD_LOGIC_1164

```
-- ---------------------------------------------------------------------
```

```
--
--   Title     :  std_logic_1164 multi-value logic system
--   Library   :  This package shall be compiled into a library
--             :  symbolically named IEEE.
--             :
--   Developers:  IEEE model standards group (par 1164)
--   Purpose   :  This packages defines a standard for designers
--             :  to use in describing the interconnection data types
--             :  used in vhdl modeling.
--             :
--   Limitation:  The logic system defined in this package may
--             :  be insufficient for modeling switched transistors,
--             :  since such a requirement is out of the scope of this
--             :  effort. Furthermore, mathematics, primitives,
--             :  timing standards, etc. are considered orthogonal
--             :  issues as it relates to this package and are therefore
--             :  beyond the scope of this effort.
--             :
--   Note      :  No declarations or definitions shall be included in,
--             :  or excluded from this package. The "package declaration"
--             :  defines the types, subtypes and declarations of
--             :  std_logic_1164. The std_logic_1164 package body shall be
--             :  considered the formal definition of the semantics of
--             :  this package. Tool developers may choose to implement
--             :  the package body in the most efficient manner available
--             :  to them.
--             :
-- ---------------------------------------------------------------------
--   modification history :
-- ---------------------------------------------------------------------
--  version | mod. date:|
--   v4.200 | 01/02/92  |
-- ---------------------------------------------------------------------

PACKAGE std_logic_1164 IS

   ---------------------------------------------------------------------
   -- logic state system  (unresolved)
   ---------------------------------------------------------------------
   TYPE std_ulogic IS ( 'U',  -- Uninitialized
                        'X',  -- Forcing  Unknown
                        '0',  -- Forcing  0
                        '1',  -- Forcing  1
                        'Z',  -- High Impedance
                        'W',  -- Weak     Unknown
                        'L',  -- Weak     0
                        'H',  -- Weak     1
```

```
                '-'   -- Don't care
               );
  ----------------------------------------------------------------
  -- unconstrained array of std_ulogic for use with the resolution function
  ----------------------------------------------------------------
  TYPE std_ulogic_vector IS ARRAY ( NATURAL RANGE <> ) OF std_ulogic;

  ----------------------------------------------------------------
  -- resolution function
  ----------------------------------------------------------------
  FUNCTION resolved ( s : std_ulogic_vector ) RETURN std_ulogic;


  ----------------------------------------------------------------
  -- *** industry standard logic type ***
  ----------------------------------------------------------------
  SUBTYPE std_logic IS resolved std_ulogic;

  ----------------------------------------------------------------
  -- unconstrained array of std_logic for use in declaring signal arrays
  ----------------------------------------------------------------
  TYPE std_logic_vector IS ARRAY ( NATURAL RANGE <>) OF std_logic;

  ----------------------------------------------------------------
  -- common subtypes
  ----------------------------------------------------------------
  SUBTYPE X01     IS resolved std_ulogic RANGE 'X' TO '1'; -- ('X','0','1')
  SUBTYPE X01Z    IS resolved std_ulogic RANGE 'X' TO 'Z'; -- ('X','0','1','Z')
  SUBTYPE UX01    IS resolved std_ulogic RANGE 'U' TO '1'; -- ('U','X','0','1')
  SUBTYPE UX01Z   IS resolved std_ulogic RANGE 'U' TO 'Z'; -- ('U','X','0','1','Z')

  ----------------------------------------------------------------
  -- overloaded logical operators
  ----------------------------------------------------------------

  FUNCTION "and"  ( l : std_ulogic; r : std_ulogic ) RETURN UX01;
  FUNCTION "nand" ( l : std_ulogic; r : std_ulogic ) RETURN UX01;
  FUNCTION "or"   ( l : std_ulogic; r : std_ulogic ) RETURN UX01;
  FUNCTION "nor"  ( l : std_ulogic; r : std_ulogic ) RETURN UX01;
  FUNCTION "xor"  ( l : std_ulogic; r : std_ulogic ) RETURN UX01;
--  function "xnor" ( l : std_ulogic; r : std_ulogic ) return ux01;
  FUNCTION "not"  ( l : std_ulogic                 ) RETURN UX01;

  ----------------------------------------------------------------
  -- vectorized overloaded logical operators
  ----------------------------------------------------------------
  FUNCTION "and"  ( l, r : std_logic_vector  ) RETURN std_logic_vector;
```

```
    FUNCTION "and"  ( l, r : std_ulogic_vector ) RETURN std_ulogic_vector;

    FUNCTION "nand" ( l, r : std_logic_vector  ) RETURN std_logic_vector;
    FUNCTION "nand" ( l, r : std_ulogic_vector ) RETURN std_ulogic_vector;

    FUNCTION "or"   ( l, r : std_logic_vector  ) RETURN std_logic_vector;
    FUNCTION "or"   ( l, r : std_ulogic_vector ) RETURN std_ulogic_vector;

    FUNCTION "nor"  ( l, r : std_logic_vector  ) RETURN std_logic_vector;
    FUNCTION "nor"  ( l, r : std_ulogic_vector ) RETURN std_ulogic_vector;

    FUNCTION "xor"  ( l, r : std_logic_vector  ) RETURN std_logic_vector;
    FUNCTION "xor"  ( l, r : std_ulogic_vector ) RETURN std_ulogic_vector;

--  ------------------------------------------------------------------------
--  Note : The declaration and implementation of the "xnor" function is
--  specifically commented until at which time the VHDL language has been
--  officially adopted as containing such a function. At such a point,
--  the following comments may be removed along with this notice without
--  further "official" ballotting of this std_logic_1164 package. It is
--  the intent of this effort to provide such a function once it becomes
--  available in the VHDL standard.
--  ------------------------------------------------------------------------
--  function "xnor" ( l, r : std_logic_vector  ) return std_logic_vector;
--  function "xnor" ( l, r : std_ulogic_vector ) return std_ulogic_vector;

    FUNCTION "not"  ( l : std_logic_vector  ) RETURN std_logic_vector;
    FUNCTION "not"  ( l : std_ulogic_vector ) RETURN std_ulogic_vector;

    ---------------------------------------------------------------------
    -- conversion functions
    ---------------------------------------------------------------------
    FUNCTION To_bit       ( s : std_ulogic;       xmap : BIT := '0') RETURN BIT;
    FUNCTION To_bitvector ( s : std_logic_vector ; xmap : BIT := '0') RETURN BIT_VECTOR;
    FUNCTION To_bitvector ( s : std_ulogic_vector; xmap : BIT := '0') RETURN BIT_VECTOR;

    FUNCTION To_StdULogic      ( b : BIT             ) RETURN std_ulogic;
    FUNCTION To_StdLogicVector  ( b : BIT_VECTOR       ) RETURN std_logic_vector;
    FUNCTION To_StdLogicVector  ( s : std_ulogic_vector ) RETURN std_logic_vector;
    FUNCTION To_StdULogicVector ( b : BIT_VECTOR       ) RETURN std_ulogic_vector;
    FUNCTION To_StdULogicVector ( s : std_logic_vector  ) RETURN std_ulogic_vector;

    ---------------------------------------------------------------------
    -- strength strippers and type convertors
    ---------------------------------------------------------------------

    FUNCTION To_X01  ( s : std_logic_vector  ) RETURN  std_logic_vector;
```

```
    FUNCTION To_X01  ( s : std_ulogic_vector ) RETURN  std_ulogic_vector;
    FUNCTION To_X01  ( s : std_ulogic        ) RETURN  X01;
    FUNCTION To_X01  ( b : BIT_VECTOR        ) RETURN  std_logic_vector;
    FUNCTION To_X01  ( b : BIT_VECTOR        ) RETURN  std_ulogic_vector;
    FUNCTION To_X01  ( b : BIT               ) RETURN  X01;

    FUNCTION To_X01Z ( s : std_logic_vector  ) RETURN  std_logic_vector;
    FUNCTION To_X01Z ( s : std_ulogic_vector ) RETURN  std_ulogic_vector;
    FUNCTION To_X01Z ( s : std_ulogic        ) RETURN  X01Z;
    FUNCTION To_X01Z ( b : BIT_VECTOR        ) RETURN  std_logic_vector;
    FUNCTION To_X01Z ( b : BIT_VECTOR        ) RETURN  std_ulogic_vector;
    FUNCTION To_X01Z ( b : BIT               ) RETURN  X01Z;

    FUNCTION To_UX01  ( s : std_logic_vector  ) RETURN  std_logic_vector;
    FUNCTION To_UX01  ( s : std_ulogic_vector ) RETURN  std_ulogic_vector;
    FUNCTION To_UX01  ( s : std_ulogic        ) RETURN  UX01;
    FUNCTION To_UX01  ( b : BIT_VECTOR        ) RETURN  std_logic_vector;
    FUNCTION To_UX01  ( b : BIT_VECTOR        ) RETURN  std_ulogic_vector;
    FUNCTION To_UX01  ( b : BIT               ) RETURN  UX01;

    --------------------------------------------------------------------
    -- edge detection
    --------------------------------------------------------------------
    FUNCTION rising_edge  (SIGNAL s : std_ulogic) RETURN BOOLEAN;
    FUNCTION falling_edge (SIGNAL s : std_ulogic) RETURN BOOLEAN;

    --------------------------------------------------------------------
    -- object contains an unknown
    --------------------------------------------------------------------
    FUNCTION Is_X ( s : std_ulogic_vector ) RETURN  BOOLEAN;
    FUNCTION Is_X ( s : std_logic_vector  ) RETURN  BOOLEAN;
    FUNCTION Is_X ( s : std_ulogic        ) RETURN  BOOLEAN;

END std_logic_1164;
```