**מדינת ישראל**
משרד החינוך

סוג הבחינה: גמר לבתי־ספר לטכנאים ולהנדסאים
מועד הבחינה: אביב תשס"ח, 2008
נספח לשאלון: 711003

**אין להעביר את הנוסחאון לנבחן אחר**

*מקום למדבקת נבחן*

# נוסחאון בשפת תיאור חומרה (VHDL) לכיתה י"ד

**(12 עמודים)**

| בלוקים עיקריים בשפה | |
|---|---|
| **ENTITY**__entity_name **IS**<br>**GENERIC** (parameter_name : string := default_value;<br>parameter_name : integer := default_value**);**<br>**PORT** (<br>input_name : **IN STD_LOGIC**;<br>input_vector_name : **IN BIT_VECTOR** (high downto low);<br>bidir_name : **INOUT STD_LOGIC**;<br>output_name : **OUT STD_LOGIC** **);**<br>**END**__entity_name | **מבנה כללי של ישות תוכנית** |

המשך בעמוד 2

| בלוקים עיקריים בשפה | |
|---|---|
| **ARCHITECTURE** a **OF** __entity_name **IS**<br>הצהרה על משאבי התוכנית<br>**BEGIN**<br>-- Process Statement<br>-- Concurrent Procedure Call<br>-- Concurrent Signal Assignment<br>-- Conditional Signal Assignment<br>-- Selected Signal Assignment<br>-- Component Instantiation Statement<br>-- Generate Statement<br>**END** a; | **מבנה כללי של גוף תוכנית** |
| __process_label:<br>**PROCESS** (__signal_name, __signal_name)<br>**VARIABLE** __variable_name : **STD_LOGIC;**<br>**VARIABLE** __variable_name : **STD_LOGIC;**<br>**BEGIN**<br>-- Signal Assignment Statement<br>-- Variable Assignment Statement<br>-- Procedure Call Statement<br>-- If Statement<br>-- Case Statement<br>-- Loop Statement<br>**END PROCESS** __process_label; | **מבנה כללי של הליך טורי** |

המשך בעמוד 3

| שימוש במבניות בתכנון היררכי | |
|---|---|
| **COMPONENT** __component_name<br>**GENERIC (**__parameter_name : string := __default_value;<br>__parameter_name : integer := __default_value **);**<br>**PORT (**<br>input_name, input_name **: IN STD_LOGIC;**<br>bidir_name, bidir_name **: INOUT STD_LOGIC;**<br>output_name, output_name **: OUT**<br>**STD_LOGIC);**<br>**END COMPONENT;** | **הצהרה על מבנית שבשימוש בתוכנית האב** |
| __instance_name: __component_name<br>**GENERIC MAP** ( parameter_name => parameter_value,<br>parameter_name => parameter_value )<br>**PORT MAP** ( component_port => connect_port,<br>component_port => connect_port ); | **שימוש במבנית בתכנון היררכי** |
| __generate_label:<br>**FOR** __index_variable **IN** __range **GENERATE**<br>__statement;<br>__statement;<br>**END GENERATE** __generate_label; | **שרשור מבניות GENERATE בלולאת FOR** |
| __generate_label:<br>**IF** __expression **GENERATE**<br>__statement;<br>__statement;<br>**END GENERATE** __generate_label; | **שרשור מותנה IF GENERATE** |

המשך בעמוד 4

| **LOOPS לולאות** | |
|---|---|
| __loop_label:<br>**FOR** __index_variable **IN** __range **LOOP**<br>__statement;<br>__statement;<br>**END LOOP** __loop_label; | **FOR לולאת** |
| __loop_label:<br>**WHILE** __boolean_expression **LOOP**<br>__statement;<br>__statement;<br>**END LOOP** __loop_label; | **WHILE לולאת** |

המשך בעמוד 5

| התניות בגוף התוכנית – מחוץ ל-PROCESS | |
|---|---|
| **התניית**<br>**When ... Else** | __signal <= __expression **WHEN** __boolean_expression **ELSE**<br>__expression **WHEN** __boolean_expression **ELSE**<br>__expression _____<br>**דוגמא 1 :** מימוש שער AND באמצעות פקודת "WHEN" פשוטה.<br>Y <= '1' when (a = '1') and (b = '1') else '0' ;<br>**דוגמא 2 :** מימוש שער AND באמצעות פקודת "WHEN" המשכית.<br>Y <= '0' when (a = '0') and (b = '0') else<br>'0' when (a = '0') and (b = '1') else<br>'0' when (a = '1') and (b = '0') else<br>'1' ; |
| **התניית**<br>**With ... Select** | **WITH** __ expression **SELECT**<br>__signal <= __expression **WHEN** __constant_value;<br>__expression **WHEN** __constant_value;<br>__expression **WHEN** __constant_value;<br>__expression **WHEN** __constant_value;<br>_____<br>**דוגמא:** למימוש שער AND באמצעות פקודת התניה WITH .. SELECT<br>Signal ab : bit_vector (1 downto 0);<br>Begin<br>ab <= a & b;<br><br>with ab select<br>y <= '0' when "00" ,<br>'0' when "01" ,<br>'0' when "10",<br>'1' when others ; |

המשך בעמוד 6

**התניות בהליך טורי – בתוך PROCESS**

**התניית If .. Then .. Else**

```
IF      __boolean_expression THEN
        __signal <= __ expression;
ELSIF   __boolean_expression THEN
        __signal <= __expression;
ELSE
        __signal <= __expression;
END IF;
```

**התניית CASE**

```
CASE __expression IS
        WHEN __constant_value =>
                __statement;
                __statement;
        WHEN __constant_value =>
                __statement;
                __statement;
        WHEN OTHERS =>
                __statement;
                __statement;
END CASE;
```

---

**דוגמא** למימוש מפענח בהתניית CASE :

```
Process
Begin
    Case  s  is
            When "00" =>   y <= "0001" ;
            When "01" =>   y <= "0010" ;
            When "10" =>   y <= "0100" ;
            When "11" =>   y <= "1000" ;
    End case;
End process;
```

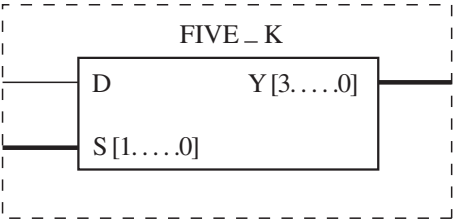
המשך בעמוד 7

## התניות בהליך טורי – בתוך PROCESS

התוכנית תבצע את טבלת האמת הבאה:

| S : BIT_VECTOR (1 DOWNTO 0) | | Y : BIT_VECTOR (3 DOWNTO 0) | | | |
|---|---|---|---|---|---|
| S1 | S0 | Y3 | Y2 | Y1 | Y0 |
| 0 | 0 | 0 | 0 | 0 | **1** |
| 0 | 1 | 0 | 0 | **1** | 0 |
| 1 | 0 | 0 | **1** | 0 | 0 |
| 1 | 1 | **1** | 0 | 0 | 0 |

**דוגמא לתוכנית שלמה המבצעת פעולת מפלג, תוך שימוש בהתניית CASE**




```
libraty ieee;
use ieee.std_logig_1164.all;
entity DEMUX is
port  (d  : in bit;
       s  : in bit_vector (1 downto 0);
       y  : out bit_vector (3 downto 0);
end;
architecture behave of DEMUX is
begin
process
begin
       case s is
              when "00" =>   y <= '0' & '0' & '0' & d;
              when "01" =>   y <= '0' & '0' & d & '0';
              when "10" =>   y <= '0' & d & '0' & '0';
              when "11" =>   y <= d & '0' & '0' & '0';
       end case;
end process;
end behave;
```

המשך בעמוד 8

**התניות בהליך טורי – בתוך PROCESS**

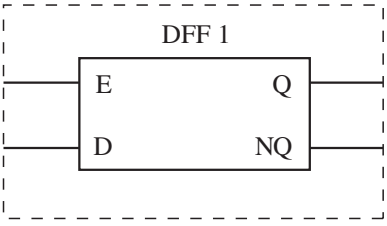
**פקודת השהייה WAIT**

| | |
|---|---|
| **wait;** | עצירה לעד אשר שימושית לעצירת סביבת בדיקה |
| **wait on** | רשימת משתנים ; |
| **wait until** | תנאי לוגי ; |
| **wait for** | מספר יחידות זמן ; |

**דוגמא למימוש DFF**

```
Libraty ieee;
Use ieee.std_logic_1164.all;
Entity DFF is
    Port ( e, d  : in std_logic;
           Q     : inout std_logic;
           Nq    : out std_logic );
End;
Architecture behave of DFF is
Begin
    Process ( e )
    Begin
        Wait until e'event and e = '1' ;
        Q <= d;
        Nq <= not d;
    End process;
End behave;
```

התוכנית יצרה סמל SYMBOL עפ"י הישות (ENTITY) בתוכנית.




המשך בעמוד 9

## דוגמאות נוספות

**דוגמא למימוש DFF עם מבואות ישירים**

```
Process (enable)
Begin
    If      pre = '1'   then    q <= '1' ;
    Elsif   clr = '1'   then    q <= '0' ;
    Elsif   enable 'event and enable = '1'   then   q <= d;
End process;
```

ההליך בדוגמא יבצע את הפעולות המתוארות בטבלת האמת הבאה:

| PRE | CLR | ENABLE | D | Q |
|---|---|---|---|---|
| 1 | 0 | Φ | Φ | 1 |
| 0 | 1 | Φ | Φ | 0 |
| 0 | 0 | אין עליה | Φ | נשמר מצב קודם |
| 0 | 0 | עליה | 0 | 0 |
| 0 | 0 | עליה | 1 | 1 |

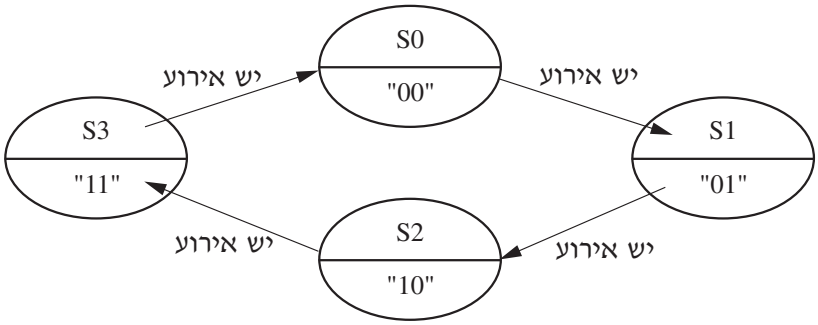
**דוגמאות ליצירה והצבה במערכים**

```
architecture EXAMPLE1 of ARRAY is

    type INTEGER_VECTOR is array (1 to 8) of integer;
    -- 1 --
    type MATRIX_A is array (1 to 3) of INTEGER_VECTOR;
    -- 2 --
    type MATRIX_B is array (1 to 4 , 1 to 8) of integer;

    signal MATRIX_3x8 : MATRIX_A;

    signal MATRIX_4x8 : MATRIX_B;

begin

    MATRIX_3x8 (3) (5)  <= 10; -- מערך בתוך מערך

    MATRIX_4x8 (4 , 5)  <= 17; -- מערך דו־ממדי

end EXAMPLE1;
```

המשך בעמוד 10

## דוגמאות נוספות

**דוגמא לכתיבת מונה במכונת מצבים**




```
use ieee.std_logic_1164.all;
entity meely_state_machine is
port  (clk          :   in bit;
       count_out  :   out bit_vector (1 downto 0));
end;
architecture behave of meely_state_machine is
type state_type is (s0 , s1 , s2 , s3); -- names of states
signal state : state_type;

begin
counting : process (clk)
           begin
               if clk'event and clk = '1' then -- positive edge event
                   case state is
                       when s0 =>   state <= s1;
                       when s1 =>   state <= s2;
                       when s2 =>   state <= s3;
                       when s3 =>   state <= s0;
                   end case;
           end if;
end process counting;
with state select
           count_out <=      "00" when s0,
                             "01" when s1,
                             "10" when s2,
                             "11" when s3;
end behave;
```

המשך בעמוד 11

| דוגמאות נוספות | |
|---|---|
| **דוגמא לתוכנית המשתמשת בפונקציה המבצעת חיבור בין מספרים**<br>entity ADDER_FUNC is<br>generic (max : integer := 15);<br>port (<br>A , B : in integer range 0 to max;<br>SUM : out integer range 0 to MAX + MAX);<br>end;<br>architecture behave of ADDER_FUNC is<br>function ADD (x , y : integer) return integer is<br>variable s : integer;<br>begin<br>s := x + y;<br>return s;<br>end function ADD;<br>begin<br>sum <= ADD (a , b); -- הקריאה לפונקציה<br>end behave; | |
| **דוגמא לתוכנית המשתמשת בפרוצדורה המבצעת חיבור בין מספרים**<br>libraty ieee;<br>use ieee.std_logic_1164.all;<br>entity adder_proc is<br>port ( A , B : in integer;<br>SUM : out integer);<br>end;<br>architecture behave of adder_proc is<br>**procedure ADD (signal x , y : in integer;**<br>**signal s : out integer) is**<br>**begin**<br>**s <= x + y;**<br>**end procedure ADD;**<br>begin<br>**ADD (a , b , sum);**<br>end behave; | |

המשך בעמוד 12

| דוגמאות נוספות | |
|---|---|
| **PACKAGE** __package_name **IS**<br>-- Type Declaration<br>-- Subtype Declaration (FUNCTIONS , PROCEDURES)<br>-- Constant Declaration<br>-- Signal Declaration<br>-- Component Declaration<br>**END** __package_name;<br>**package body** package_name **is**<br>declarations<br>deferred constant declaration<br>subprogram bodies (תיאור פעולות הפונקציות והפרוצדורות)<br>**end package body** package_name; | **הגדרת PACKAGE** |

| Mode | קריאה | כתיבה |
|---|---|---|
| in | כן | לא |
| out | לא | כן |
| inout | כן | כן |
| buffer | כן | כן |

**בהצלחה!**