VHDL Language

Application in the programmable components of type FPGA

Contents :

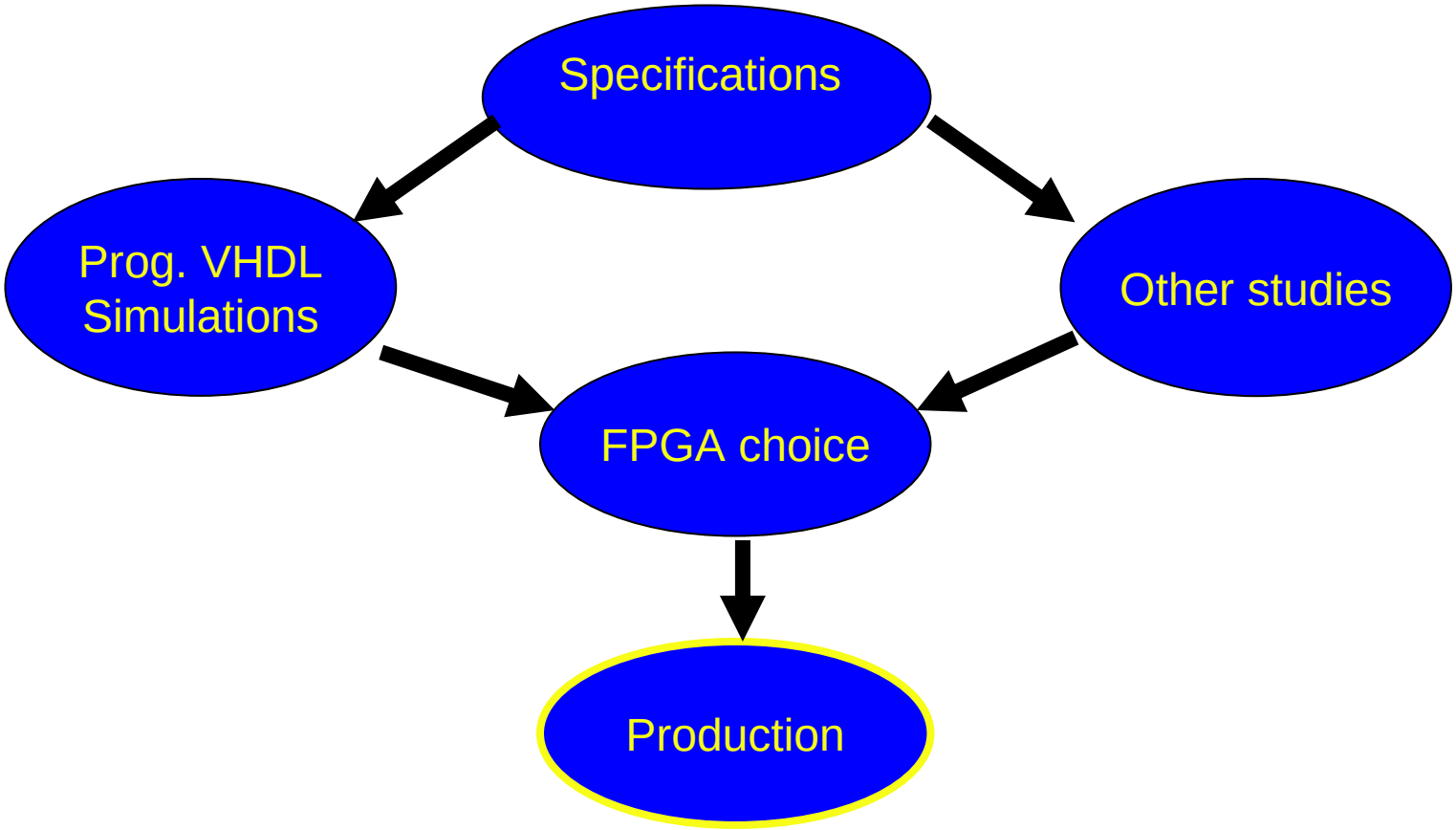
Laurent FERRIER

VHDL Language

I – Constitution of a VHDL program

- Standardized language IEEE-1076 in December, 1987
- Unique language allowing a behavioral description of a system without taking into account the characteristics of the component in which the system will be implanted. It is thus independent from processes and technologies.
- The choice of the component is made at the last stage of the study of the system

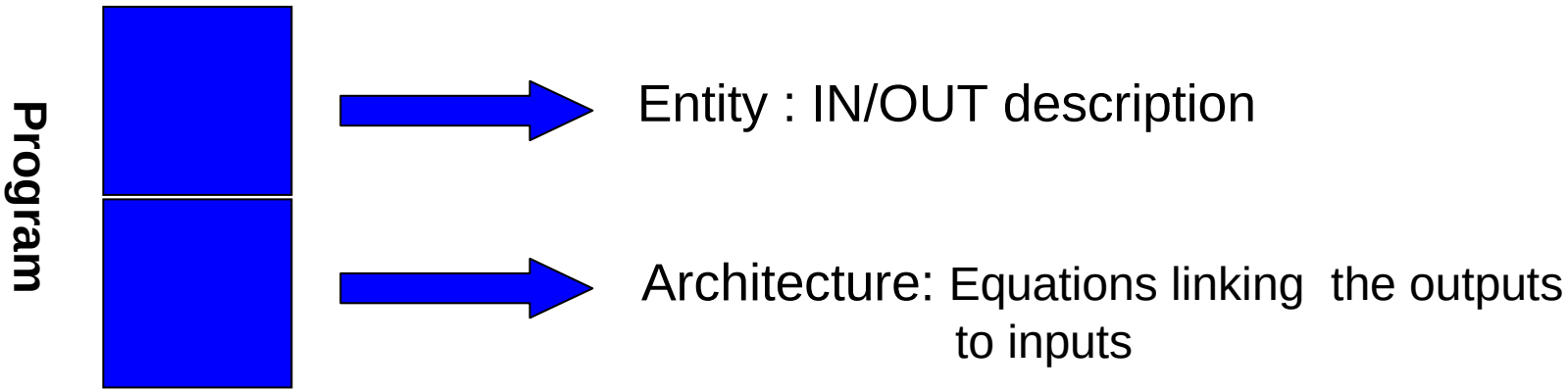
VHDL Language



VHDL Language

Entity : IN /OUT descriptions

The program is comprised of two parts:



VHDL Language

II – Entity declaration

- It contains libraries or packages for the the program syntax

Package examples :

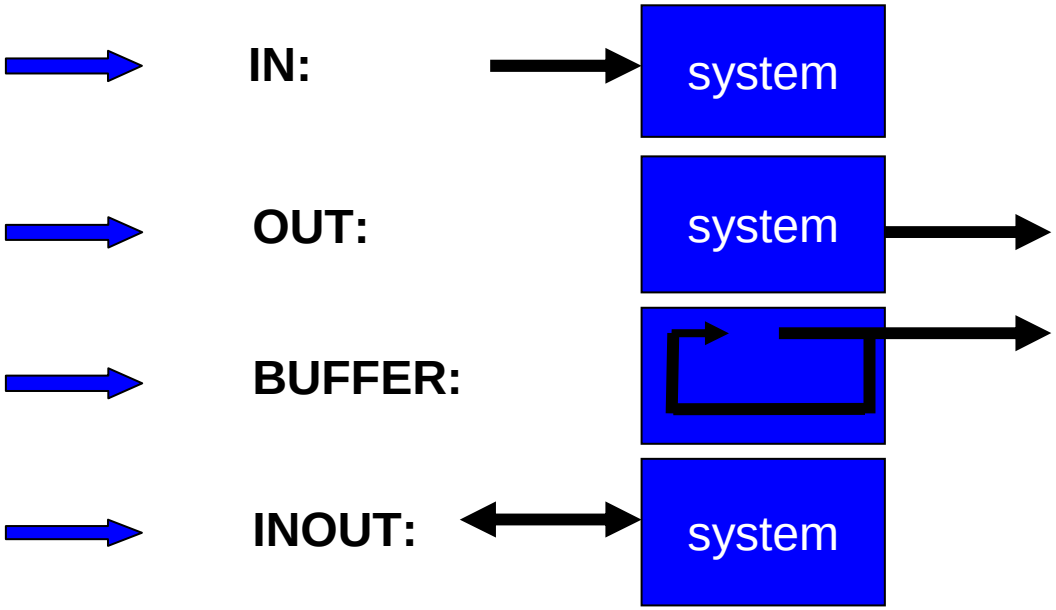
- ieee.std_logic-1164.all → Libraries of the logical functions (or,and,xor,…..)
- ieee.std_arith.all → Mathematical library (+,-)

- It defines inputs and outputs: the ports of system

VHDL Language

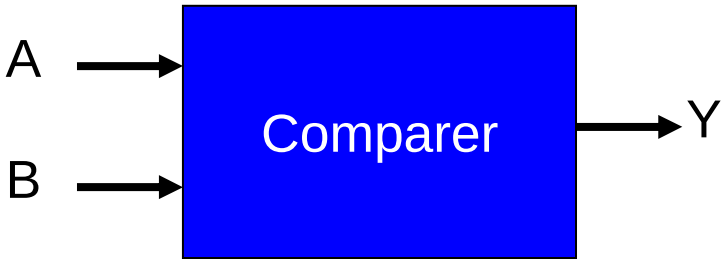
Definitions of the ports of the system:

Inputs or outputs can have several "directions"



VHDL Language

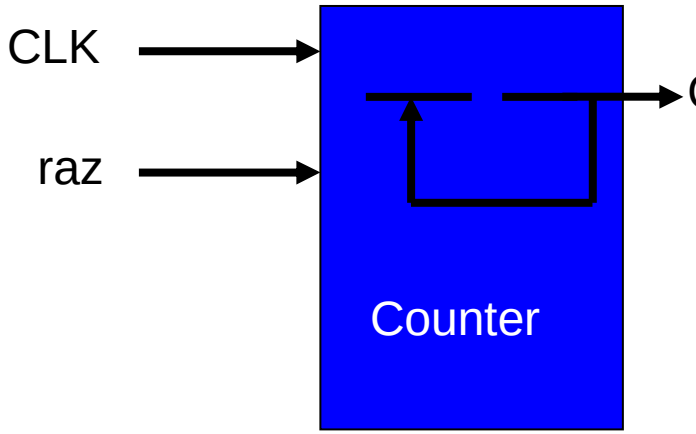
Example of description of an entity: comparator




```
library ieee;Use ieee.std_logic_1164.all;

entity comparer is
      port (A,B: in std_logic;
                  Y: out std_logic);
end comparer;
```

VHDL Language



1st possibility:

```
library ieee;
Use ieee.std_logic_1164.all;

entity counter is
port ( CLK,raz: in std_logic;
        Q:buffer std_logic_vector (2 downto 0));
end counter;
```

VHDL Language

2nd possibility:

```
library ieee;
Use ieee.std_logic_1164.all;

entity counter is
    port ( CLK,raz: in std_logic;
              Q:out std_logic_vector (2 downto 0));
end counter;
```

NB: The feedback (present state, future state) will be assured by one internal variable defined as an internal signal (of which we do what we wish) in the architecture (cf continuation))

VHDL Language

III - Declaration of the architecture

■The architecture defines the equation linking the outputs to the inputs of the system

It describes the functioning of the ENTITY

■The language VHDL authorizes several levels of description:

→ Low level (structural): description by use of primitive (gates AND, OR)

→ High level (behavioral): description of the feature in reference to the underlying material

IV – Types of descriptions

- The behavioral descriptions facilitate the mobility (independence of the technology and the component)

- The structural descriptions facilitate the efficiency (optimization of the produced function):
  = > optimization manual worker of the compilation
  = > description by blocks thus easier debugging

- At any time (including inside same architecture) it is the designer who decides on the level of description.

VHDL Language

Various architectures:

- **Behavioral description**


```
library ieee;
Use ieee.std_logic_1164.all;

entity comp is
      port (A,B: in std_logic;
                  Y: out std_logic);
end comp;
```

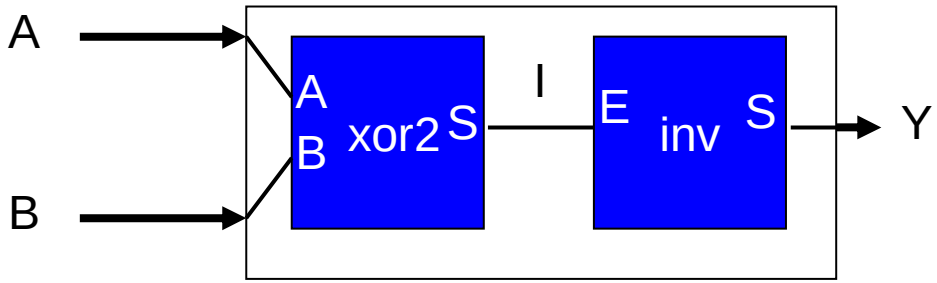
```
Architecture ARCH1 of comp is
begin
      Y<= '1' when (A=B) else '0';
End ARCH1;
```

```
Architecture ARCH2 of comp is
begin
      Y<= not (A xor B);
End ARCH2;
```

VHDL Language

- **Structural description:**

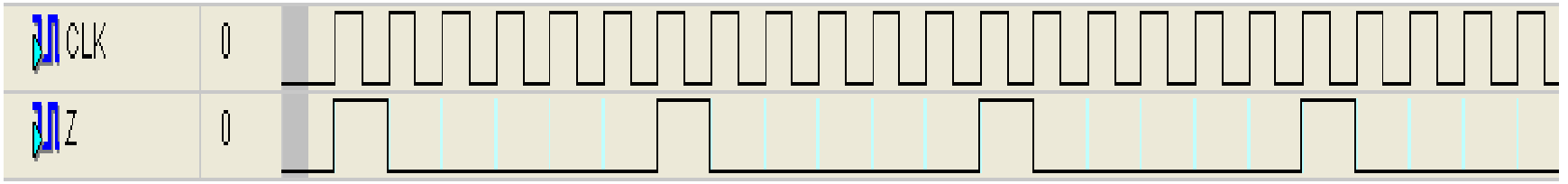
```
library ieee;
Use ieee.std_logic_1164.all;
entity comp is
      port (A,B: in std_logic;
                  Y: out std_logic);
end comp;
Architecture ARCH3 of comp is
Signal I:bit;
begin
      IC0: xor2 port map (A=>A,B=>B,S=>I);
      IC1: inv port map (I=>E,S=>Y);
End ARCH3;
```

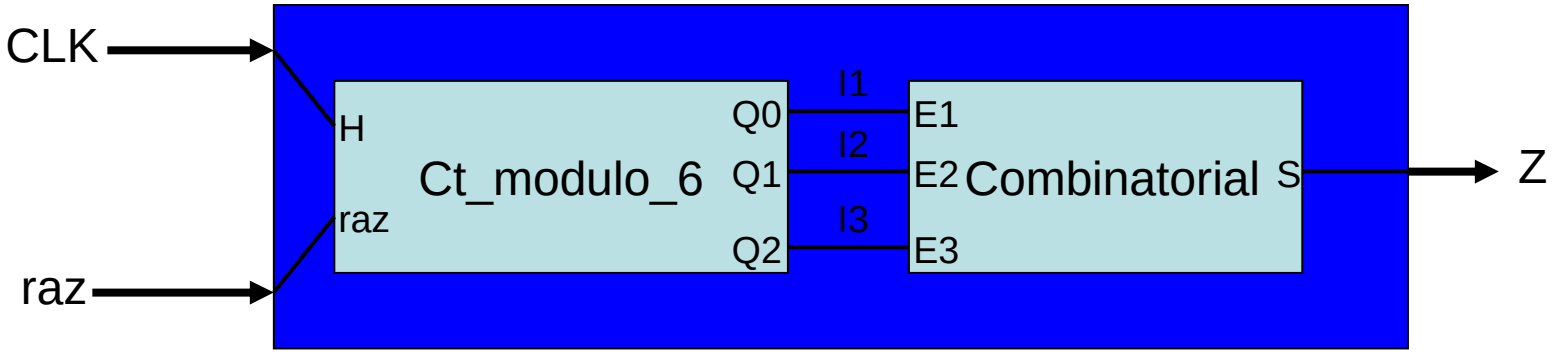


NB: Xor and inv are supposed as being already compiled and declared in one package

VHDL Language

- Other examples of a structural description: synthesis of a signal Z from a clock H





VHDL Language

```
library ieee;
Use ieee.std_logic_1164.all;
entity signal_Z is
      port (CLK,raz: in std_logic;
                  Z: out std_logic);
end signal_Z;
Architecture ARCH of signal_Z is
Signal I1,I2,I3:bit;
begin
       IC0: ct_modulo_6 port map (CLK=>H,raz=>raz,Q0=>I1,Q1=>I2,Q2=>I3);
       IC1: combinatorial port map (I1=>E1,I2=>E2,I3=>E3,S=>Z);
End ARCH;
```

NB: With XILINX, we associate symbols with every file VHDL representing a function. Then We connect them then in a graphic way in a schematic window.

VHDL Language

V - Process

The process allows the description of synchronous sequential systems

→ The process is the main construction allowing this description

NB: The synchronous character of the description authorizes a parallelism of the operations. It results from in a gain of speed.

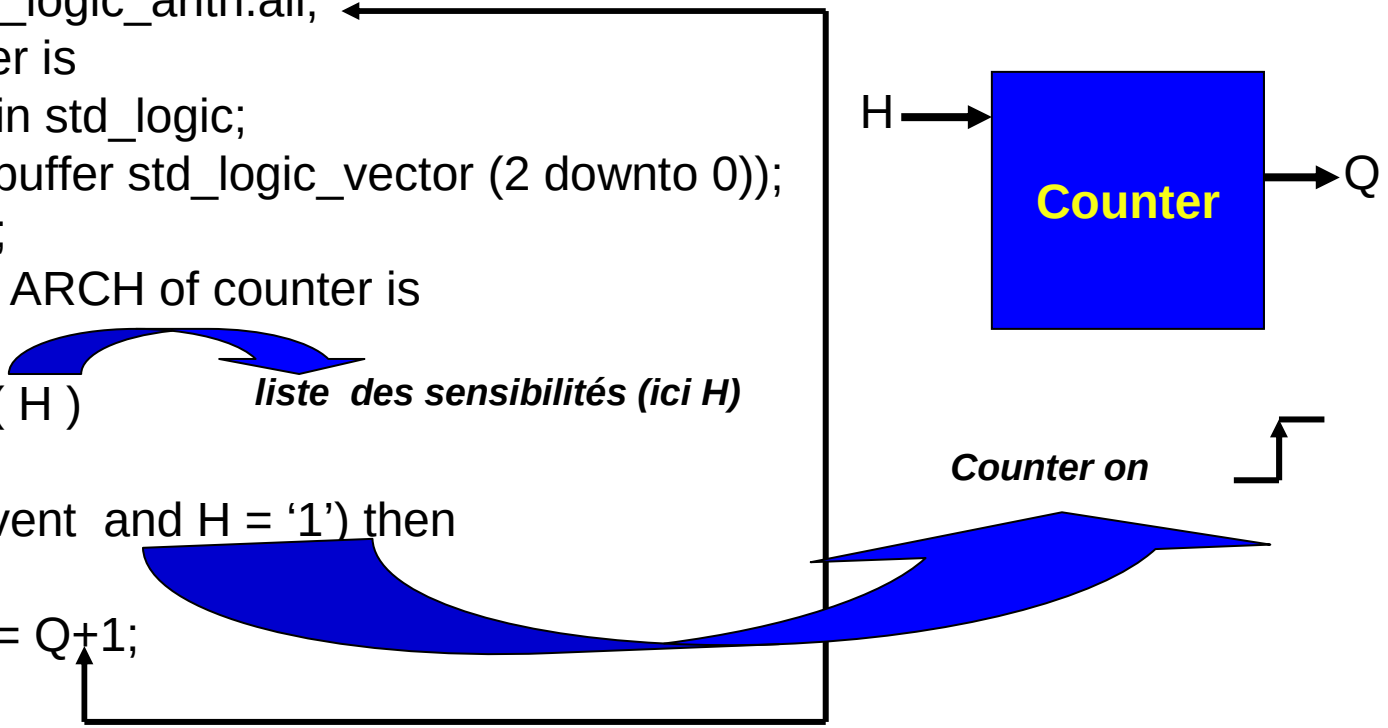
- The process is located in the architecture
- Inside a process, the instructions run in a sequential way
- Outside of a process the operations run simultaneously
- A process runs in every change of state of the input signals to which it is sensitive. We create an inventory of these signals.
  It is entitled " list of sensibilities "

VHDL Language

Example of description of a sequential system: counter

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
entity counter is
      port (H: in std_logic;
            Q: buffer std_logic_vector (2 downto 0));
end counter;
Architecture ARCH of counter is
begin
    process ( H )
      begin
      if (H' event  and H = '1') then

          Q <= Q+1;
      end if;
    end process;
end ARCH;
```



VHDL Language

VI - Syntactical tools

- Logical operators:

```
AND    { Y <= A and B;  }
OR     { Y <= A or B; }
NOT  { Y <= NOT (A); Y <= NOT (A AND B); }
NAND  { Y <= A NAND B; }
NOR  { Y <= A NOR B; }
XOR { Y <= A XOR B; }
```

- Instructions in mode contributing (Except process)
  - Assignment <=
  - Conditional assignment <= … WHEN
  - Selective assignment WITH ….SELECT

VHDL Language

- Instructions in sequential mode (IN A PROCESS)
  assignment <=
  Conditional assignment <= …. If…THEN…
  end if;
  selective assignment case ….when
  LOOP(NEXT,EXIT,FOR…LOOP, WHILE…LOOP

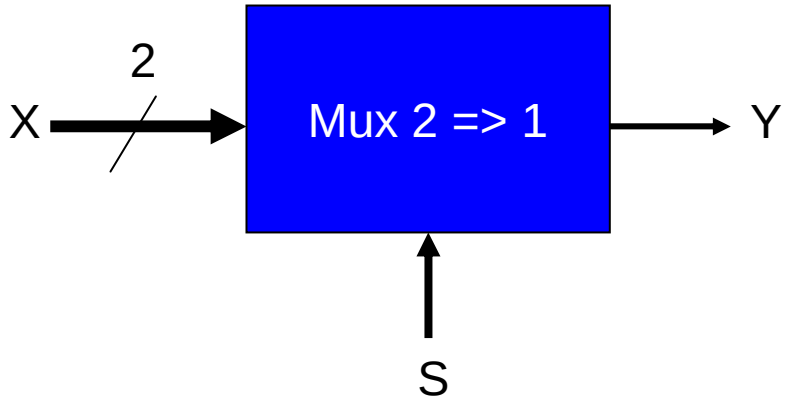
- Arithmetical operators:
  sum: +
  substract: -
  multiplication: *
  division: /
  not equal: /=
  equal =
  ……..

VHDL Language

VI – Examples of VHDL descriptions

Example n°1: Multiplexer 2=>1

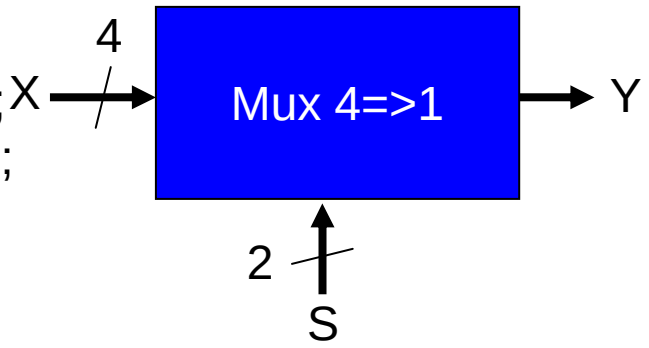
```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
entity mux2_1 is
      port (X:  in std_logic_vector(1 downto 0);
            S: in std_logic;
            Y: out std_logic);
end mux2_1;
Architecture ARCH of mux2_1 is
begin
    with S select
         Y <= X(0) when '0',
                 X(1) when others;
end ARCH;
```



VHDL Language

Exemple n°2: Multiplexer 4=>1

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
entity mux4_1 is
      port (X: IN std_logic_vector(3 downto 0);
            S:  IN std_logic_vector(1 downto 0);
            Y: out std_logic);
end mux4_1;
```



VHDL Language

```
architecture ARCH of mux4_1 is

begin
   seq:process(X,S)
               begin
                  case S is
                    when "00" => Y <= X(0);
                    when "01" => Y <= X(1);
                    when "10" => Y <= X(2);
                    when others => Y <= X(3);
                  end case;
   end process seq;
end ARCH;
```

VHDL Language

Exemple: counter modulo 4

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
entity cpt_4 is
   port (H,raz: in std_logic;
         Q: out std_logic_vector(1 downto 0));
end cpt_4;

Architecture ARCH of cpt_4 is
signal etat: std_logic_vector(1 downto 0);
   begin
      Q<=etat;
      process (H,raz)
       begin
        if (H' event and H = '1') then
            if raz = '1' then etat <= "00";
               else etat <= etat+1;
            end if;
        end if;
       end process;
End ARCH;
```

Definition of a signal: variable interns allowing to define releases from sequential releases(cf example synthesis of the signal Z taken out: Z out sequential: Q of the counter)

VHDL Language

Additive: instruction ELSIF

Used in the architecture when several conditions take place

NB: placed after an if, this instruction is not finished by an end if

Architecture ARCH of system is

```
   begin
      process (H)
        begin
            if (H' event and H = '1') then
                if …… then…..  ;
                   elsif……then….   ;
                end if;
            end if;
       end process;
End ARCH;
```