VHDL Process Constructs

Last updated 1/7/25

VHDL Process Constructs

This is not a programming language
It is a hardware description language

VHDL Process Constructs

- VHDL supports many hardware description constructs
  - 2 process specific constructs:
    - if-else
    - case
  - These are more flexible than the combinatorial constructs with-select and when-else

VHDL Process Constructs

- if-else
  - Choose a value when a certain situation exists

```
if (decision_signal = decision_value_X) then
  result_signal_1 <= result_value_1;
  result_signal_2 <= result_value_2;
  result_signal_3 <= result_value_3;
elsif (decision_signal = decision_value_Y) then
  result_signal_1 <= result_value_1a;
  result_signal_2 <= result_value_2b;
  result_signal_3 <= result_value_3c;
else
  result_signal_1 <= result_value_a;
  result_signal_2 <= result_value_b;
  result_signal_3 <= result_value_c;
end if;
```

Typically use the same decision signal
but not required

Limitation: Must be used in a process

VHDL Process Constructs

- if-else

Exhaustive List

```
if(inA = "00") then
    outW <= "1000";
elsif(inA = "01") then
    outW <= "0100";
elsif(inA = "10") then
    outW <= "0010";
elsif(inA = "11") then
    outW <= "0011";
else
    outW <= "0000";
end if;
```

Partial List

```
if(inA = "00") then
    outX <= "1000";
elsif(inA = "01") then
    outX <= "0100";
else
    outX <= "0000";
end if;
```

Partially Common Result

```
if(inA = "00") then
    outY <= "1000";
elsif(inA = "01") then
    outY <= "0100";
elsif((inA = "10") or (inA = "11")) then
    outY <= "0110";
else
    outX <= "0000";
end if;
```

Complex Selection

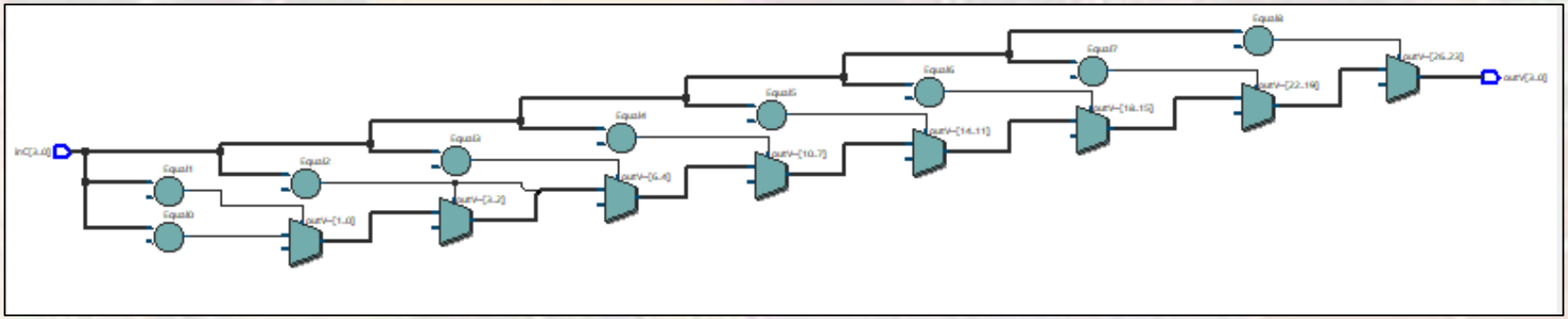
```
if((inA or inB) = "00") then
    outZ <= "1000";
elsif((inA or inB)  = "01") then
    outZ <= "0100";
else
    outZ <= "0000";
end if;
```

VHDL Process Constructs

- if-else

```
--
-- if_else.vhdl
--
-- created 7/5/2018
-- tj
--
-- rev 0
----------------------------------------
--
-- if-else example
--
----------------------------------------
--
-- Inputs: in
-- Outputs: out
--
----------------------------------------
library ieee;
use ieee.std_logic_1164.all;

entity if_else is
   port (
      i_in:    in std_logic_vector(3 downto 0);
      o_out:   out std_logic_vector(3 downto 0)
);
end entity;
```

```
architecture behavioral of if_else is

begin

   process(all)
   begin
      if(i_in = "0000") then
         o_out <= "0001";
      elsif(i_in = "0001") then
         o_out <= "0010";
      elsif(i_in = "0010") then
         o_out <= "0011";
      elsif(i_in = "0011") then
         o_out <= "0100";
      elsif(i_in = "0100") then
         o_out <= "0101";
      elsif(i_in = "0101") then
         o_out <= "0110";
      elsif(i_in = "0110") then
         o_out <= "0111";
      elsif(i_in = "0111") then
         o_out <= "1000";
      elsif(i_in = "1000") then
         o_out <= "1001";
      else
         o_out <= "0000";
      end if;
   end process;

end architecture;
```



VHDL Process Constructs

- case
  - Choose a value when a certain situation exists

```
case decision_signal is
  when decision_value_X => result_signal_1 <= result_value_1;
                           result_signal_2 <= result_value_2;
                           result_signal_3 <= result_value_3;
  when decision_value_Y => result_signal_1 <= result_value_1a;
                           result_signal_2 <= result_value_2b;
                           result_signal_3 <= result_value_3c;
  when others =>           result_signal_1 <= result_value_a;
                           result_signal_2 <= result_value_b;
                           result_signal_3 <= result_value_c;

end case;
```

Limitations: Must be used in a process
Only one decision signal

VHDL Process Constructs

- case

Exhaustive List

```
case inA is
    when "00" => outW <= "1000";
    when "01" => outW <= "0100";
    when "10" => outW <= "0010";
    when "11" => outW <= "0011";
    when others => outW <= "0000";
end case;
```

Partial List

```
case inA is
    when "00" => outX <= "1000";
    when "01" => outX <= "0100";
    when others => outX <= "0000";
end case;
```

Partially Common Result

```
case inA is
    when "00" => outY <= "1000";
    when "01" => outY <= "0100";
    when ("10“ or “11”) => outY <= "0010";
    when others => outY <= "0000";
end case;
```

Complex Selection

```
case (inA or inB) is
    when "00" => outW <= "1000";
    when "01" => outW <= "0100";
    when "10" => outW <= "0010";
    when "11" => outW <= "0011";
    when others => outW <= "0000";
end case;
```

VHDL Process Constructs

- case

```
----------------------------------------
--
-- case_ex.vhdl
--
-- created 7/5/2018
-- tj
--
-- rev 0
----------------------------------------
--
-- case example
--
----------------------------------------
--
-- Inputs: in
-- Outputs: out
--
----------------------------------------
library ieee;
use ieee.std_logic_1164.all;

entity case_ex is
   port (
            i_in:    in std_logic_vector(3 downto 0);
            o_out:   out std_logic_vector(3 downto 0)
        );
end entity;
```

```
architecture behavioral of case_ex is

begin

   process(all)
   begin
      case i_in is
         when "0000" => o_out <= "0001";
         when "0001" => o_out <= "0010";
         when "0010" => o_out <= "0011";
         when "0011" => o_out <= "0100";
         when "0100" => o_out <= "0101";
         when "0101" => o_out <= "0110";
         when "0110" => o_out <= "0111";
         when "0111" => o_out <= "1000";
         when "1000" => o_out <= "1001";
         when others => o_out <= "0000";
      end case;
   end process;

end behavioral;
```