Synchronous VHDL

Last updated 2/16/21

Synchronous VHDL

- Synchronous Logic in HDL
  - Synchronous activities only happen in certain situations
    - Rising edge of clock
  - Synchronous activities are identified by placing them in a “process” block
  - Process blocks are only executed when a signal in the blocks “sensitivity list” changes
  - The code in a process block is evaluated sequentially but signals are ONLY updated at the end of the process block
  - Process blocks themselves are concurrent activities

Synchronous VHDL

- Process block format

```
process (sensitivity list)
   begin
      actions
end process;
```

```
label process (sensitivity list)
   begin
      actions
end process;
```

Synchronous VHDL

- Processes allow two kinds of conditional statements
  - if-else
  - case

  - In VHDL 2008, with-select and when-else are also allowed but seldom used due to the added flexibility if-else and case provide

Synchronous VHDL

- VHDL – Selection
  - 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
Exception – creating registers (Flip-Flops)

Limitation: Must be used in a process

Synchronous VHDL

- VHDL – Selection
  - 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;
```

Synchronous VHDL

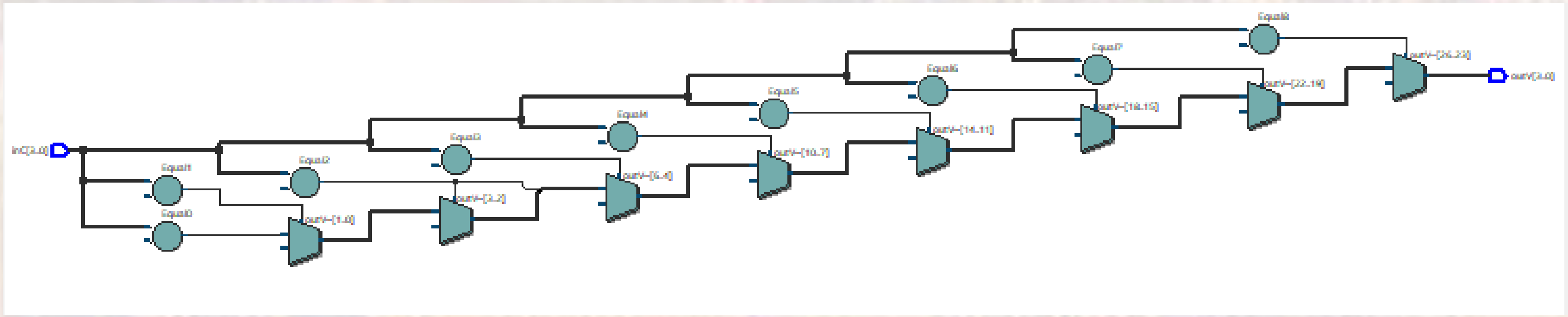
- VHDL – Selection
  - if-else

```
---------------------------------------
--
-- if_else.vhdl
--
-- created 7/5/2018
-- tj
--
-- rev 0
-----------------------------------------
--
-- if-else example
--
-----------------------------------------
--
-- Inputs: inA, inB, inC
-- Outputs: outV, outW, outX, outY, outZ
--
-----------------------------------------
library ieee;
use ieee.std_logic_1164.all;

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

```
architecture behavioral of if_else is

begin
    process(i_C)
    begin
        if(i_ C = "0000") then
            o_V <= "0001";
        elsif(i_ C = "0001") then
            o_V <= "0010";
        elsif(i_ C = "0010") then
            o_V <= "0011";
        elsif(i_ C = "0011") then
            o_V <= "0100";
        elsif(i_ C = "0100") then
            o_V <= "0101";
        elsif(i_ C = "0101") then
            o_V <= "0110";
        elsif(i_ C = "0110") then
            o_V <= "0111";
        elsif(i_ C = "0111") then
            o_V <= "1000";
        elsif(i_ C = "1000") then
            o_V <= "1001";
        else
            o_V <= "0000";
        end if;
    end process;
end behavioral;
```

Synchronous VHDL

- VHDL – Selection
  - if-else



Synchronous VHDL

- VHDL – Selection
  - 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

Synchronous VHDL

- VHDL – Selection
  - 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;
```

Synchronous VHDL

- VHDL – Selection
  - case

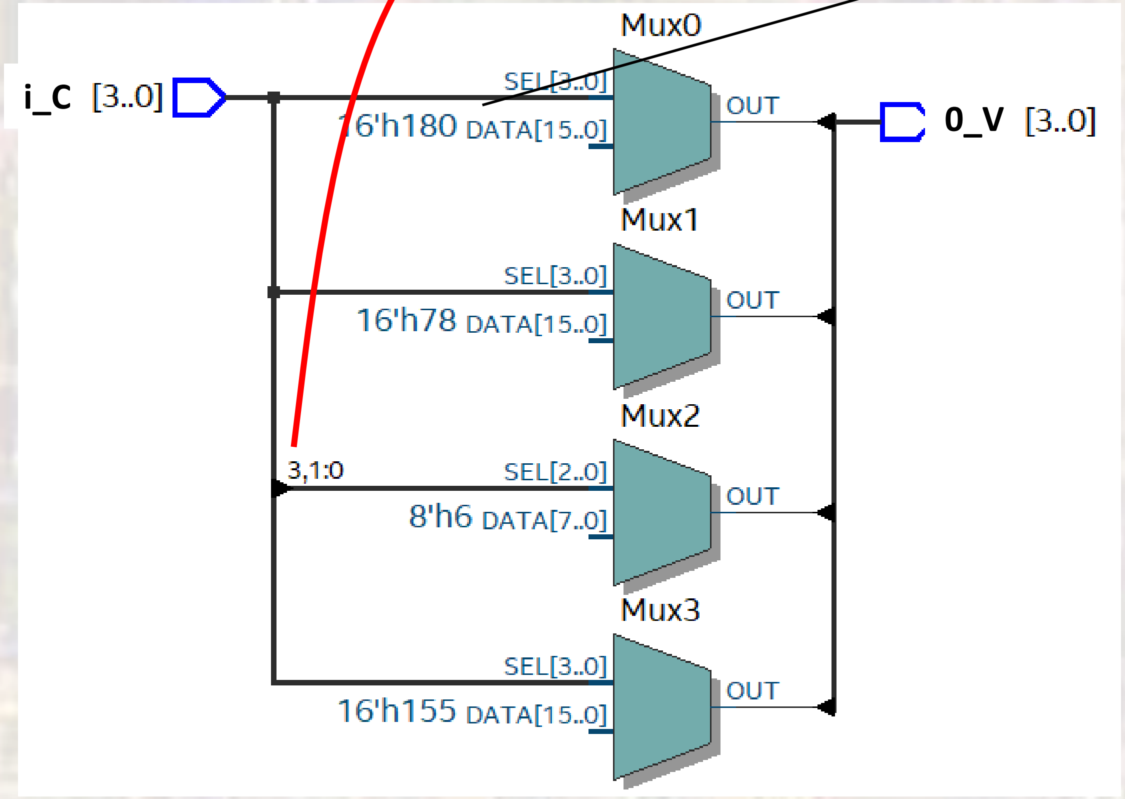
```
---------------------------------------
--
-- case_ex.vhdl
--
-- created 7/5/2018
-- tj
--
-- rev 0
-----------------------------------------
--
-- case example
--
-----------------------------------------
--
-- Inputs: inA, inB, inC
-- Outputs: outV, outW, outX, outY, outZ
--
-----------------------------------------
library ieee;
use ieee.std_logic_1164.all;

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

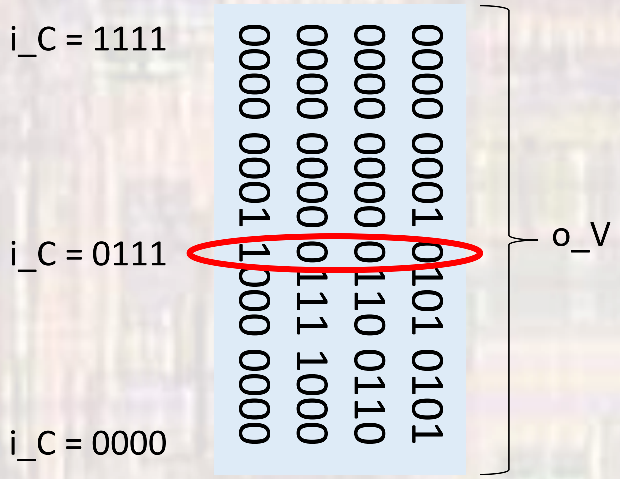
```
architecture behavioral of case_ex is

begin
    process(i_C)
    begin
        case i_C is
            when "0000" => o_V <= "0001";
            when "0001" => o_V <= "0010";
            when "0010" => o_V <= "0011";
            when "0011" => o_V <= "0100";
            when "0100" => o_V <= "0101";
            when "0101" => o_V <= "0110";
            when "0110" => o_V <= "0111";
            when "0111" => o_V <= "1000";
            when "1000" => o_V <= "1001";
            when others => o_V <= "0000";
        end case;
    end process;
end behavioral;
```

Synchronous VHDL

- VHDL – Selection
  - case

Mux3	0000 0001 0101 0101
	0000 0000 0110 0110
	0000 0000 0111 1000
Mux0	0000 0001 1000 0000





```
when "0110" => o_V <= "0111";
when "0111" => o_V <= "1000";
```