Standard VHDL Architecture Styles

1. Data-Flow Modeling
2. Behavioral Modeling
3. Structural Modeling

Data-Flow Modeling
- Describes paths and operations on signals as a circuit is traversed
- Uses _____________________________
- More closely describes / defines the elements of the circuit to be implemented
- Works well for simple functions
- Limits our ability to describe more complex behaviors
  - Therefore WE have to do more of the “design work” (i.e. translating a desired behavior into logic functions)

Behavioral Modeling
- No details about the circuit implementation are implied by the description
  - Uses a higher level of abstraction
- Describes circuit responses (Behaviors) to input conditions
- Describes Behaviors in terms of “________”
- Relies on the Sequential Nature of a Process
  - Uses “Sequential Statements”

Structural Modeling
- Breaks up the implementation into a hierarchy of subsystems (“components”)
- Describes more complex entities in terms of combinations of simpler components
- Component definitions can be shared by multiple entities
  - Supports design module sharing and reuse.
Behavioral Modeling With Processes

Processes are
• A sequentially executed block of code
  – Order of instructions inside a process matters!
  – All statements in a process are called “______________” (even if look like CSA)

architecture behavioral of my_circuit is
begin
  concurrent statements
  my_proc: process (signals) is
  begin
    sequential statements
  end process my_proc;
end architecture behavioral

Behavioral Modeling With Processes
• The entire process executes
  – Once when a simulation begins
  – Anytime that “an event” happens on any signal in the Process’s ______________

Events:
  – Signal assignments outside the Process
  – Change in input signal from simulation test vectors
• Choose your Sensitivity List signals carefully!
  – Include EVERY signal that will affect the statements and Signal Assignments in the Process; unless you have a deliberate reason not to.

Behavioral Modeling With Processes
• The entire process executes in “0” time
  – Process is executed “concurrently” with other concurrent statements (“1 big CSA”)
  – All signal assignments made inside a process take effect ______________

_______ the _________ process is evaluated
architecture behavioral of my_circuit is
begin
  concurrent statements
  my_proc: process (signals) is
  begin
    sequential statements
  end process my_proc;
end architecture behavioral

Process Syntax

(label : ______________( sensitivity list ) ____)___
  Gives a name to the process
  Defines what signals will initiate the process when they are altered elsewhere in the code

begin
  sequential statement

(ss_label : sequential statement)
  Sequential statements within a process can have their own labels

  end _________________ ;
Behavioral Modeling With Processes

• Since sequential and instantaneous:
  – If multiple assignments made to a signal inside a process, ______________ assignment will have any visible effect

```
architecture behavioral of my_circuit is
  signal A, B : std_logic;
begin
  my_proc: process ( A ) is
    begin
      B <= A;   -- This assignment will have __________
    end process my_proc;
end architecture behavioral
```

Behavioral Modeling With Processes

• Even though a Process is “sequential”
  – A new value assigned to a signal will not take effect until __________ of the Process
  • Can’t rely on signal updates made inside the process when making assignments to other signals appearing later in the process

```
signal B: std_logic;   -- is an “intermediate” signal
...
my_proc: process ( A ) is
  begin
    B <= A;   -- but B does not change until end of process
    C <= not( B );   -- SO this assignment uses __________
                             -- of B that it had when the Process started
  end process my_proc;
end architecture behavioral
```

Behavioral Modeling With Processes

• A new value assigned to a signal will not take effect until THE END of the Process

```
signal B: std_logic;   -- is an “intermediate” signal
...
my_proc: process ( A ) is
  begin
    B <= A;   -- but B does not change until end of process
    C <= not( B );   -- SO this assignment is better
  end process my_proc;
end architecture behavioral
```

Sequential Statements for Modeling Behavior

1. Sequential Signal Assignments
   – Look like Concurrent Signal Assignments
   – But appear _____________________

```
architecture behavioral of my_circuit is
begin
  C <= '0';   -- a Simple “Concurrent” Signal Assignment
  my_proc: process ( A ) is
    begin
      B <= A;   -- A Simple “Sequential” Signal Assignment
      B <= '1';   -- and another
    end process my_proc;
end architecture behavioral
```
Sequential Statements for Modeling Behavior

1. Sequential Signal Assignments
   NOTE: Selective & Conditional CSA's be used inside of a Process

2. If Statements
3. Case Statements
4. Loop Statements
   - For Loops, While Loops
5. Wait Statements

If-Then-Else Statements

- Generalized version of a Conditional CSA
- Conditionally execute a block of Sequential Statements
  - vs. just making a single, particular signal assignment
- Execution depends on a set of conditions being "True"
  - Uses a Boolean expression for conditions
  - Parenthesis (condition) are optional (help readability)

Syntax:

____(condition)_______
{sequential statements}
___________;

If-Then-Elsif Statements

- Optional "Elsif" clauses
  - provide alternative actions for different conditions
- Follow "Elsif" paths until a "True" condition is found
  - Alternative "Elsif"'s following the "True" condition
- Statements following the final "Else" will be executed if none of the previous conditions are True

Syntax:

IF (condition1) THEN
  {alternative1 sequential statements}
ELSIF (condition2) THEN
  {alternative2 sequential statements}
ELSIF (condition3) THEN
  {alternative3 sequential statements}
ELSE
  {default sequential statements}
END IF;
Case Statements

- Generalized version of a Selected CSA
  - Conditional branching to execute one of several alternative blocks of Sequential Statements
    - vs. just making a single, particular signal assignment
  - Selection depends on the value of a single selection expression

Syntax:

```
select_criterion
    criterionValue1 => {alternative1 sequential statements} ;
    criterionValue2 => {alternative2 sequential statements} ;
    criterionValue3 => {alternative3 sequential statements} ;
    .......... => {default sequential statements} ;
```

Cautions:

- The Criterion Values must be of the same “Type” as the Select Criterion (signal type & size)
- Criterion Values are checked in the order they appear. (Sequential)
- Need a “WHEN” clause for every possible value of `select_criterion` if you omit the "OTHERS"
ENTITY XOR_Gate IS
  PORT ( A, B : IN STD_LOGIC;  F : OUT STD_LOGIC );
END XOR_Gate ;

ARCHITECTURE Behavioral_Example OF XOR_Gate IS

BEGIN

  xor_case : PROCESS ( ) IS
  BEGIN
    CASE ( ) IS
      WHEN WHEN
      END CASE;
    END PROCESS xor_case ;
END Behavioral_Example ;

ENTITY MUX_4_to_1 IS
  PORT ( D0, D1, D2, D3, En : IN STD_LOGIC ;
          Sel: IN STD_LOGIC_VECTOR( 1 downto 0) ;
          F : OUT STD_LOGIC );
END MUX_4_to_1 ;

ARCHITECTURE Behavioral2 OF MUX_4_TO_1 IS

BEGIN
  mux_case: PROCESS (En,Sel,D0,D1,D2,D3) IS

  END PROCESS mux_case ;
END Behavioral2 ;

Example: 4:1 MUX With Strobe Enable (Active High)

<table>
<thead>
<tr>
<th>EN</th>
<th>Sel(1)</th>
<th>Sel(0)</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>D0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>D1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>D2</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>D3</td>
</tr>
<tr>
<td>0</td>
<td>-</td>
<td>-</td>
<td>0</td>
</tr>
</tbody>
</table>

ENTITY MUX_4_4_to_1 IS

<table>
<thead>
<tr>
<th>EN</th>
<th>Sel(1)</th>
<th>Sel(0)</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>D0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>D1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>D2</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>D3</td>
</tr>
<tr>
<td>0</td>
<td>-</td>
<td>-</td>
<td>0</td>
</tr>
</tbody>
</table>

Common Student VHDL Errors

207) You can only use IF or CASE statements if they are__________________.

483) You CAN NOT USE CONCURRENT SELECTED SIGNAL ASSIGNMENTS:

    WITH sel SELECT...

    OR CONDITIONAL SIGNAL ASSIGNMENTS:

    F <= A WHEN (...) ELSE ...

    INSIDE A PROCESS.