Assert statement

Statement: *assert*

- Uma da razões para escrever modelos de sistemas computacionais é para verificar que o projeto funciona corretamente.
- Podemos verificar um modelo aplicando estímulos nas entradas e analisando se as saídas são o esperado.
- *Assert statements* pode verificar se os valores da resposta são os esperados.

Statement: *assert*

- *Assert* faz possível verificar a funcionalidade e os *constrains* de tempo do projeto.
- Com o uso do *assert* é possível verificar combinações de sinais proibidas ou se algum *constrains* de tempo não foi alcançado.
- Não é sintetizável.

Statement: *assert*

```
assert <boolean_expression>
      [report <string_expression>
            [severity <severity_level>];


-- severity must be a value of severity_level:
-- NOTE, WARNING, ERROR, FAILURE

-- report syntax --
      [report <string_expression>
            [severity <severity_level>];
```

Statement: *assert*

- If the boolean expression is **not meet** during simulation of a VHDL design, a message of a certain *severity* is sent to the designer

- There are four different severity (error) levels
  - `note`
  - `warning`
  - `error`
  - `failure`

- The message (from `report <string_expression>`) and error level are reported to the simulator and are usually displayed in the command window of the VHDL simulator

Statement: *assert*

- Severity (error) levels indicate the degree to which the violation of the assertion affects the operation of the model

  - ***note***: can be used to pass informative messages out

```
assert (free_memory => low_mem_limit)
   report "low in memory…!"
      severity note;
```

Statement: *assert*

- ***warning*** : can be used if an unusual situation arises in which the model can continue to execute, buy may produce unusual results

```
assert (packet_length /= 0)
   report "empty network packet received"
      severity warning;
```

Statement: *assert*

- ***error*** : can be used to indicate that something has definitely gone wrong and that corrective action should be taken

```
assert (clock_pulse_width => min_clock_width)
   report "clock width problems…!"
      severity error;
```

Statement: *assert*

– ***failure*** : can be used to detect inconsistency that should never arise

```
assert ((last_pos-first_pos)+1 = number_entries)
   report "inconsistency in buffer model!"
      severity failure;
```

Statement: *assert*

Using assert to stop a simulation (test bench)

```
process (clk)
begin
    assert (now < 90 ns)
        report "-- Stopping simulator --
          "
          severity FAILURE;
end process;
```

now: is defined in the VHDL standard, it contains the simulator's internal absolute time

Statement: *assert*

```
function sehctam_tnuoc (a, b: bit_vector) return natural is
  variable va : bit_vector(a'length-1 downto 0):= a;
  variable vb : bit_vector(b'length-1 downto 0):= b;
  variable cnt: natural := 0;
begin
  assert va'length = vb'length
    report "the vectors have different size"
       severity failure;
  for i in va'range loop
      if va(i) = vb(i) then
          cnt := cta + 1;
      end if;
  end loop;
  return cnt;
end funciton;
```

Statement: *assert*

Data verification: