VHDL reference

General VHDL file structure

```
[library and use statements]

ENTITY [entity name] IS
  PORT(
    [port signals]
  );
END [entity name];

ARCHITECTURE [architecture name] of [entity name] IS

  [type and signal declarations]

  BEGIN

    [hardware description]

  END [architecture name];
```

WITH-SELECT statement

```
with b select m <=
  '0' when "00",
  '1' when "01",
  '1' when "10",
  '0' when others;
```

IF-ELSIF-ELSE statement

```
if a = '0' then
  m <= '0';
elsif b = '0' then
  m <= '0';
else
  m <= '1';
end if;
```

CASE statement

```
case b is
  when "00" =>
    m <= '0';
  when "01" =>
    m <= '1';
  when "10" =>
    m <= '1';
  when others =>
    m <= '0';
end case;
```

```vhdl
-- Lab 5 example state machine

LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.All;

ENTITY SM_VHDL IS  -- Do not modify this entity statement!
  PORT(X       : IN STD_LOGIC_VECTOR(1 DOWNTO 0);
       RESETN,
       CLOCK   : IN  STD_LOGIC;
       Z       : OUT STD_LOGIC;
       Q       : OUT STD_LOGIC_VECTOR(1 DOWNTO 0)  );
END SM_VHDL;       -- Do not modify this entity statement!

ARCHITECTURE behavior of SM_VHDL IS
  TYPE STATE_TYPE IS (A, B, C);
  SIGNAL state : STATE_TYPE;

  BEGIN
    PROCESS(CLOCK, RESETN)
      BEGIN
        IF RESETN = '0' THEN
          state <= A;
        ELSIF CLOCK'EVENT AND CLOCK = '1' THEN
          CASE state IS
            WHEN A =>
              CASE X IS
                WHEN "00" =>
                  state <= B;
                WHEN "01" =>
                  state <= C;
                WHEN OTHERS =>
                  state <= A;
              END CASE;
            WHEN B =>
              CASE X IS
                WHEN "01" =>
                  state <= A;
                WHEN "11" =>
                  state <= B;
                WHEN OTHERS =>
                  state <= C;
              END CASE;
            WHEN C =>
              CASE X IS
                WHEN "10" =>
                  state <= B;
                WHEN "11" =>
                  state <= A;
                WHEN OTHERS =>
                  state <= C;
              END CASE;
          END CASE;
        END IF;
      END PROCESS;
    Z  <= '1' WHEN state = C ELSE '0';
    Q  <= "00" WHEN state = A ELSE "01" WHEN state = B ELSE "10";

  END behavior;
```

```
-- Lab 6 example state machine

LIBRARY IEEE;
USE  IEEE.STD_LOGIC_1164.all;
USE  IEEE.STD_LOGIC_ARITH.all;
USE  IEEE.STD_LOGIC_UNSIGNED.all;


ENTITY Tcontrol IS
   PORT(
      reset, clock, sensor1, sensor2      : IN std_logic;
      sensor3, sensor4, sensor5, sensor6  : IN std_logic;
      switch1, switch2, switch3, switch4  : OUT std_logic;
      dirA, dirB                          : OUT std_logic_vector(1 DOWNTO 0)
   );
END Tcontrol;


ARCHITECTURE a OF Tcontrol IS
   TYPE STATE_TYPE IS (
      ABout,
      Ain,
      Bin,
      Astop,
      Bstop
   );
   SIGNAL state                                : STATE_TYPE;
   SIGNAL sensor12, sensor13, sensor24         : std_logic_vector(1 DOWNTO 0);

BEGIN
   PROCESS (clock, reset)
   BEGIN
      IF reset = '1' THEN
         -- Reset to this state
         state <= ABout;
      ELSIF clock'EVENT AND clock = '1' THEN
         CASE state IS
            WHEN ABout =>
               CASE Sensor12 IS
                  WHEN "00" => state <= ABout;
                  WHEN "01" => state <= Bin;
                  WHEN "10" => state <= Ain;
                  WHEN "11" => state <= Ain;
                  WHEN OTHERS => state <= ABout;
               END CASE;

            WHEN Ain =>
                  CASE Sensor24 IS
                  WHEN "00" => state <= Ain;
                  WHEN "01" => state <= ABout;
                  WHEN "10" => state <= Bstop;
                  WHEN "11" => state <= Bin;
                  WHEN OTHERS => state <= Ain;
               END CASE;

            WHEN Bin =>
               CASE Sensor13 IS
                  WHEN "00" => state <= Bin;
                  WHEN "01" => state <= ABout;
                  WHEN "10" => state <= Astop;
                  WHEN "11" => state <= Ain;
                  WHEN OTHERS => state <= Bin;
               END CASE;

            WHEN Astop =>
               -- Although a case statement could be used with
               -- one sensor, a simple IF statement is sufficient.
               IF Sensor3 = '1' THEN
                  state <= Ain;
```

```
                ELSE
                   state <= Astop;
                END IF;

             WHEN Bstop =>
                IF Sensor4 = '1' THEN
                   state <= Bin;
                ELSE
                   state <= Bstop;
                END IF;

          END CASE;
       END IF;
    END PROCESS;

    -- Combine bits for the internal signals declared above.
    -- ("&" operator concatenates bits)
    sensor12 <= sensor1 & sensor2;
    sensor13 <= sensor1 & sensor3;
    sensor24 <= sensor2 & sensor4;

    -- The following outputs depend on the state.
    WITH state SELECT Switch1 <=
       '0' WHEN ABout,
       '0' WHEN Ain,
       '1' WHEN Bin,
       '1' WHEN Astop,
       '0' WHEN Bstop;
    WITH state SELECT Switch2 <=
       '0' WHEN ABout,
       '0' WHEN Ain,
       '1' WHEN Bin,
       '1' WHEN Astop,
       '0' WHEN Bstop;
    WITH state SELECT DirA <=
       "01" WHEN ABout,
       "01" WHEN Ain,
       "01" WHEN Bin,
       "00" WHEN Astop,
       "01" WHEN Bstop;
    WITH state SELECT DirB <=
       "01" WHEN ABout,
       "01" WHEN Ain,
       "01" WHEN Bin,
       "01" WHEN Astop,
       "00" WHEN Bstop;

    -- These outputs happen to be constant values for this solution;
    -- they do not depend on the state.
    Switch3 <= '0';
    Switch4 <= '0';

 END a;
```