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

Case

- Variations

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;
```

Case

- Example

```
----------------------------------------
--
-- 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;
```