LAB4: Język VHDL – Przykłady opisu bloków cyfrowych

I. CEL ĆWICZENIA

Celem ćwiczenia jest doskonalenie umiejętności projektowania z wykorzystaniem języka opisu sprzętu VHDL oraz zapoznanie z parametryzacją jednostek projektowych i opisem strukturalnym w środowisku *Active-HDL* firmy *Aldec*.

II. WPROWADZENIE

Parametr typu generic w języku VHDL

W sekcji *entity* opisywany jest interfejs projektowanej jednostki: **nazwa jednostki**, porty **wejściowe** i **wyjściowe**. Dodatkową opcją są parametry **generic**, które służą do parametryzacji projektowanej jednostki. Dzięki parametryzacji możemy w łatwy i szybki sposób zmienić np. liczbę bitów w całym projekcie. Bez parametrów **generic**, należy prześledzić cały kod i dokonać zmiany w wielu miejscach.

Przykład opisu licznika N-bitowego z resetem synchronicznym

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity Licznik is
  generic (N : integer := 4);
  port (
    CLK : in  std_logic;
    RST : in  std_logic;
    Q   : out std_logic_vector(N-1 downto 0)
  );
end Licznik;

architecture L1 of Licznik is
  signal q_s : unsigned(N-1 downto 0) := (others => '0');
begin
  process(CLK)
  begin
    if rising_edge(CLK) then
      if RST = '1' then
        q_s <= (others => '0');
      else
        q_s <= q_s + 1;
      end if;
    end if;
  end process;

  Q <= std_logic_vector(q_s);
end L1;
```

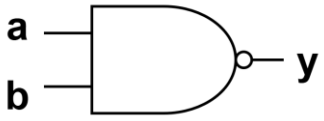
Opis strukturalny (hierarchiczny) w języku VHDL

W języku VHDL układ cyfrowy można opisać na różne sposoby, m.in. behawioralnie (opis działania, algorytmu) oraz strukturalnie (opis połączeń między blokami). **Opis strukturalny** traktuje układ jako sieć połączonych ze sobą mniejszych modułów (komponentów), podobnie jak schemat zbudowany z bramek, liczników, multiplekserów itd. Aby wykorzystać wcześniej zdefiniowaną jednostkę projektową (*entity*) wewnątrz innego bloku, stosuje się **komponenty (*component*)** oraz instrukcję **port map**.

Przykład opisu bramki XOR w oparciu o komponenty bramki NAND

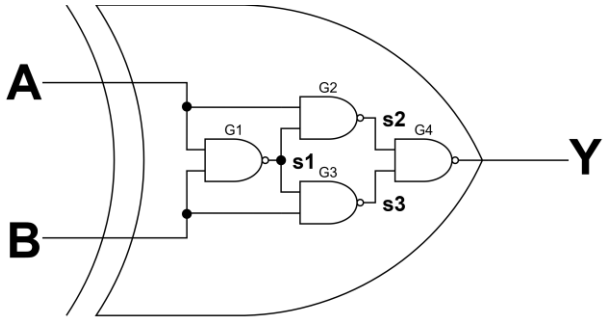
1. **Opisanie dwuwejściowej bramki NAND**

```
library ieee;
use ieee.std_logic_1164.all;

entity nand2 is
    port (
        a : in  std_logic;
        b : in  std_logic;
        y : out std_logic
    );
end nand2;

architecture a of nand2 is
begin
    y <= not (a and b);
end a;
```



2. **Opisanie bramki XOR w oparciu o komponent bramki NAND**

```
library ieee;
use ieee.std_logic_1164.all;

entity XOR_z_NAND is
    port (
        A : in  std_logic;
        B : in  std_logic;
        Y : out std_logic
    );
end XOR_z_NAND;

architecture struct of XOR_z_NAND is

    -- Deklaracja komponentu NAND (musi odpowiadac entity nand2 utworzonej wyzej)

    component nand2 is
        port (
            a : in  std_logic;
            b : in  std_logic;
            y : out std_logic
        );
    end component;

    -- Sygnaly wewnetrzne
    signal s1, s2, s3 : std_logic;

begin
    -- Przyporzadkowanie jawne
    G1 : nand2 port map (a => A, b => B, y => s1);
    G2 : nand2 port map (a => A, b => s1, y => s2);
    G3 : nand2 port map (a => s1, b => B, y => s3);
    G4 : nand2 port map (a => s2, b => s3, y => Y);

    -- Przyporzadkowanie niejawne
    -- G1 : nand2 port map (A, B, s1);
    -- G2 : nand2 port map (A, s1, s2);
    -- G3 : nand2 port map (s1, B, s3);
    -- G4 : nand2 port map (s2, s3, Y);

end struct;
```



III. REALIZACJA ĆWICZENIA LABORATORYJNEGO

STANOWISKO KOMPUTEROWE

Zadania projektowe realizowane są z użyciem stanowiska komputerowego z zainstalowanym systemem operacyjnym Windows i środowiskiem projektowo-symulacyjnym *Active-HDL* firmy *Aldec*.

WYKONANIE ĆWICZENIA

Podczas ćwiczenia realizowane są indywidualne zadania projektowe zdane przez prowadzącego ćwiczenia. Zadania obejmują implementację układów projektowanych na poprzednich zajęciach, ale stosując parametryzację komponentów (***generic***). Opracowane projekty są następnie poddawane syntezie oraz analizie działania poprzez symulacje funkcjonalne.

OPRACOWANIE WYNIKÓW

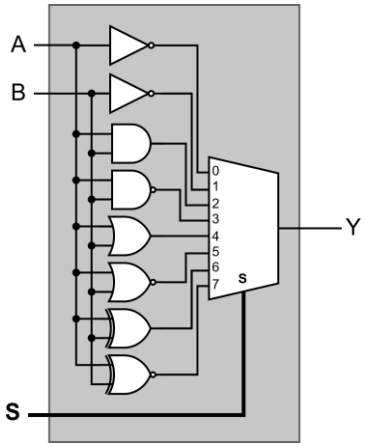
1. Sprawozdanie z ćwiczenia laboratoryjnego powinno zawierać zwięzłe opisy postawionych zadań projektowych wraz z rozwiązaniami (kod VHDL i zrzuty ekranów z wynikami symulacji).
2. W sprawozdaniu należy również zawrzeć rozwiązania dodatkowych problemów projektowych postawionych przez prowadzącego ćwiczenie.

ZALICZENIE ĆWICZENIA

1. Zaliczenie kolokwium wstępnego oraz poprawne wykonanie zadań laboratoryjnych postawionych przez osobę prowadzącą ćwiczenie.
2. Złożenie sprawozdania, zawierającego zwięzły opis wykonanych zadań, wnioski i poprawne odpowiedzi na postawione pytania.

PRZYKŁADOWE ZADANIA PROJEKTOWE

1. Zaprojektować dzielnik częstotliwości przez N stosując parametryzację N – wartość N wskazana przez prowadzącego ćwiczenie, a następnie sprawdzić działanie z użyciem symulatora.
2. Zaprojektować *n*-bitowy licznik binarny z zerowaniem asynchronicznym / synchronicznym stosując parametryzację *n* – liczba bitów wskazana przez prowadzącego ćwiczenie, a następnie sprawdzić działanie z użyciem symulatora.
3. Do poprzedniego zadanie dodać wejście zezwolenia na zliczanie.
4. Zaprojektować licznik modulo N z zerowaniem asynchronicznym (zliczanie w górę / w dół / dwukierunkowy) stosując parametryzację N – liczba N wskazana przez prowadzącego ćwiczenie, a następnie sprawdzić działanie z użyciem symulatora.
5. Do poprzedniego zadanie dodać wejście zezwolenia na zliczanie.
6. Zaprojektować *n*-bitowy licznik Johnsona, stosując parametryzację *n* - liczba bitów wskazana przez prowadzącego ćwiczenie, a następnie sprawdzić działanie z użyciem symulatora.
7. Zaprojektować w **stylu strukturalnym** jednostkę logiczną (rys. 1) realizującą operacje logiczne NOT, AND, NAND, OR, NOR, XOR, XNOR, która na wyjście przypisuje wynik operacji logicznej w zależności od sygnału sterującego.



Rys. 1. Schemat blokowy jednostki logicznej

IV. ZAGADNIENIA DO OPRACOWANIA PRZED PRZYSTĄPIENIEM DO ĆWICZENIA

1. Wyjaśnić na czym polega parametryzacja jednostki projektowej.
2. W którym miejscu w kodzie VHDL występuje deklaracja *generic*?
3. Opisać w języku VHDL n-bitowy licznik z zezwoleniem na zliczanie i parametryzacją.
4. Omówić strukturalny styl opisu układu cyfrowego.

V. LITERATURA I MATERIAŁY POMOCNICZE

1. Materiały wykładowe.
2. J. Kalisz: Podstawy elektroniki cyfrowej, WKŁ, Warszawa 2007.
3. J. Kalisz: Język VHDL w praktyce, WKŁ, Warszawa 2002.
4. J.F. Wakerly: Digital Design, Principles and Practices, 4th Edition, Pearson/Prentice Hall, 2005.
5. R.E. Haskell, D.M. Hanna: Introduction to Digital Design: Using Digilent FPGA Boards. LBE Books, 2009.
6. B.J. LaMeres, Quick Start Guide to VHDL, Springer Cham, 2019.
7. B.J. LaMeres, Introduction to Logic Circuits & Logic Design with VHDL, Springer Cham, 2019.