LAB3: VHDL (II) – Opis układów sekwencyjnych

I. CEL ĆWICZENIA

Celem ćwiczenia jest doskonalenie umiejętności projektowych z wykorzystaniem języka opisu sprzętu VHDL, w szczególności w zakresie układów sekwencyjnych. Zapoznanie z instrukcjami języka mającymi na celu detekcję zbocza sygnału oraz przeprowadzaniem symulacji synchronicznych układów sekwencyjnych w środowisku *Active-HDL* firmy *Aldec*.

II. WPROWADZENIE

Układ sekwencyjny – przypomnienie

Układy cyfrowe dzielą się na dwie podstawowe grupy: kombinacyjne oraz sekwencyjne. Stany wyjść układów kombinacyjnych są określane wyłącznie na podstawie aktualnych wartości sygnałów wejściowych, natomiast układy sekwencyjne uwzględniają również poprzednie stany wewnętrzne oraz kolejność ich występowania.

Powszechnym układem sekwencyjnym jest układ synchroniczny, w którym zmiany stanów wewnętrznych synchronizowane są sygnałem zegarowym.

Współbieżna instrukcja *process*

W języku VHDL instrukcja *process* służy do opisu sekwencyjnych fragmentów kodu, czyli takich, które mogą zmieniać swoje zachowanie w czasie – np. w reakcji na zbocze sygnału zegarowego. Proces działa podobnie do funkcji, która „budzi się" przy każdej zmianie sygnałów znajdujących się na liście wrażliwości (ang. sensitivity list).:

```
nazwa_procesu: process(syg_1, syg_2,...)
  -- miejsce deklaracji zmiennych lokalnych variable
begin
  -- ciało procesu
end process;
```

Przykład opisu przerzutnika typu D

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity DFF is
  port (
    clk   : in  std_logic;
    d     : in  std_logic;
    q     : out std_logic
  );
end DFF;

architecture Beh1 of DFF is
begin
  p1: process(clk)            -- na liscie wrazliwosci jest sygnal zegarowy clk
  begin
    if rising_edge(clk) then  -- jezeli wystapi zbocze narastajace clk,
      q <= d;                 -- to przypisz d do q
    end if;
  end process;
end Beh1;
```

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 podstawowych układów sekwencyjnych (np. przerzutników, liczników) w języku VHDL. Opracowane projekty są następnie poddawane syntezie oraz analizie działania poprzez symulacje funkcjonalne. Przed pierwszym wykonaniem symulacji należy zapoznać się ze wskazówkami dotyczącymi przygotowania poprawnej symulacji – **ACTIVE-HDL – WSKAZÓWKI.**

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.

ACTIVE-HDL – WSKAZÓWKI

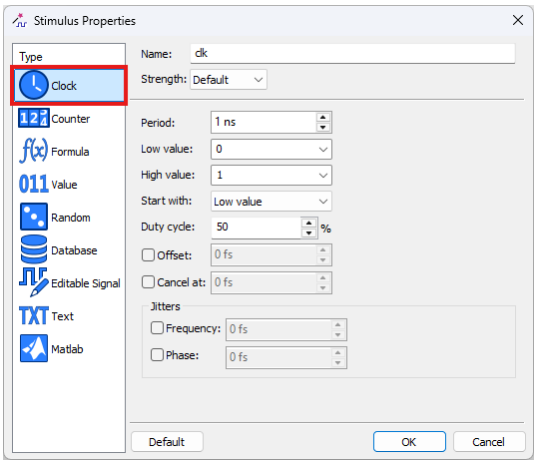
Dla poprawnej symulacji synchronicznych układów sekwencyjnych, niezmiernie ważny jest dobór zależności czasowych pomiędzy wymuszeniami sygnałów wejściowych. Wynika to z możliwości sterowania synchronicznego względem sygnału zegarowego oraz asynchronicznego sygnałami bezpośredniego zerowania i ustawiania stanu wyjściowego przerzutników.

Zatem zerowanie układu powinno być wykonane na początku symulacji oraz incydentalnie w trakcie dla wprowadzenia układu w stan początkowy. Natomiast sygnał zegarowy należy zastosować o możliwie dużej częstotliwości tak, aby można było wykonać symulację pełnego cyklu pracy układu sekwencyjnego.

Program *Active-HDL* posiada kilka sposobów ustawiania wartości sygnałów wejściowych:

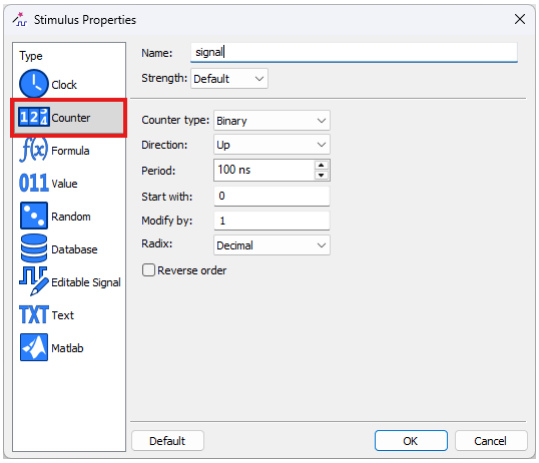
- sygnał prostokątny, używany do sterowania sygnałami zegarowymi:

  Stimulus Properties → Type: → Clock



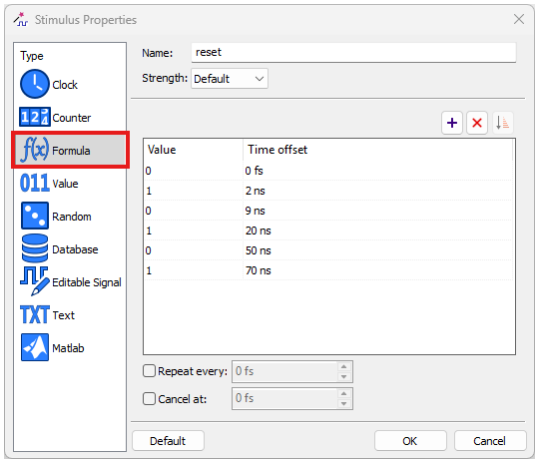
Rys. 1. Widok okna Stimulus Properties – Type: Clock

- sekwencja wartości odpowiadających kolejnym stanom licznika:
  Stimulus Properties → Type: → Counter



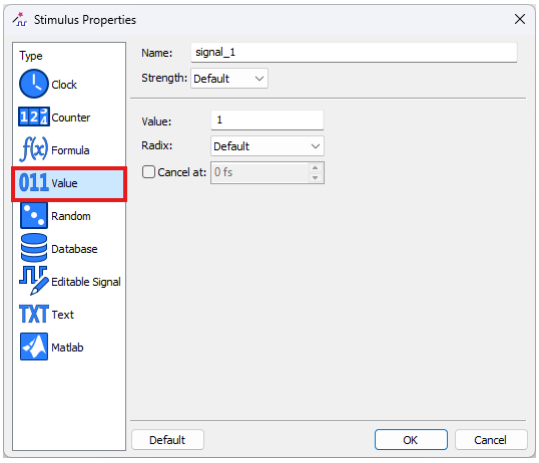
Rys. 2. Widok okna Stimulus Properties – Type: Counter

- przebieg zdefiniowany przez wyrażenie oparte na sekwencji par wartość – czas (Value – Time offset):
  Stimulus Properties → Type: → Formula



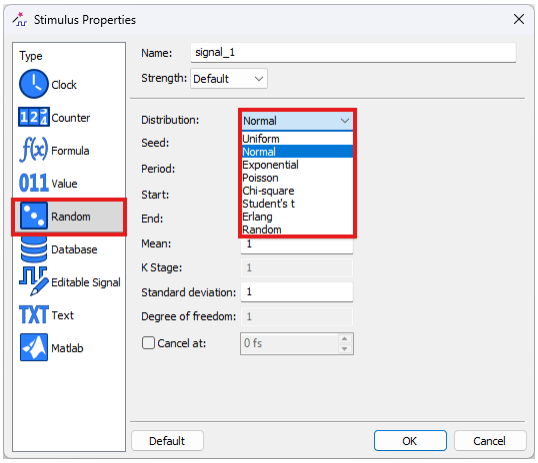
Rys. 3. Widok okna Stimulus Properties – Type: Formula

- wymuszenie stałej wartości:
  Stimulators → Type: → Value



Rys. 4. Widok okna Stimulus Properties – Type: Value

- generator liczb losowych zwracający wartości całkowite rozłożone zgodnie z wybranym rozkładem:
  Stimulus Properties → Type: → Random



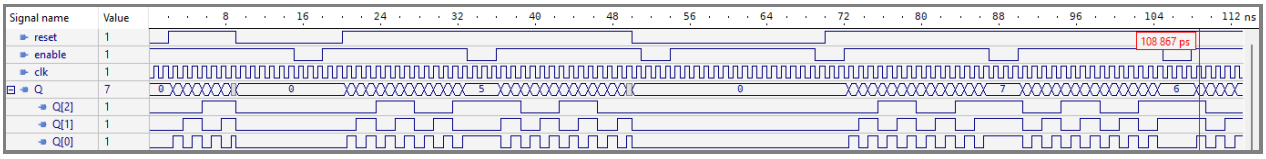
Rys. 5. Widok okna Stimulus Properties – Type: Random

Pozostałe typy (**Database, Editable Signal, Text oraz Matlab**) generują wymuszenia na podstawie danych odczytywanych z plików zewnętrznych.

Przykład symulacji

Poniżej przedstawiono przykładowy wynik symulacji 3-bitowego licznika binarnego z zerowaniem asynchronicznym oraz sygnałem zezwolenia. Sygnał **reset** to sekwencja stanów 0 i 1 określona przy zastosowaniu wymuszenia typu **Formula**. Sygnał zezwolenia **enable** to cyklicznie powtarzana sekwencja stanów 0 i 1 co 18 ns (wymuszenie typu **Formula**). Jako sygnał zegarowy **clk** zastosowano wymuszenie typu **Clock** o okresie 1 ns.

Symulację rozpoczęto dla sygnału **reset = 0**, czyli od wyzerowania układu. Widoczne są chwile wstrzymania pracy dla **enable = 0**. Ponadto można zaobserwować wpływ sygnału **reset**, który powoduje zerowanie wyjść licznika niezależnie od pozostałych sygnałów wejściowych.



Rys. 3. Przebiegi symulacyjne dla 3-bitowego licznika binarnego

PRZYKŁADOWE ZADANIA PROJEKTOWE

1. Zaprojektować synchroniczny przerzutnik typu D/T/JK, a następnie sprawdzić działanie z użyciem symulatora.
2. Zaprojektować synchroniczny przerzutnik typu D z zerowaniem asynchronicznym / synchronicznym, a następnie sprawdzić działanie z użyciem symulatora.
3. Zaprojektować synchroniczny przerzutnik typu D z ustawianiem asynchronicznym / synchronicznym, a następnie sprawdzić działanie z użyciem symulatora.
4. Zaprojektować synchroniczny przerzutnik typu D z ustawianiem i zerowaniem asynchronicznym / synchronicznym, a następnie sprawdzić działanie z użyciem symulatora.
5. Zaprojektować *n*-bitowy licznik binarny z zerowaniem asynchronicznym / synchronicznym – liczba bitów wskazana przez prowadzącego ćwiczenie, a następnie sprawdzić działanie z użyciem symulatora.
6. Do poprzedniego zadanie dodać wejście zezwolenia na zliczanie.
7. Zaprojektować licznik modulo N z zerowaniem asynchronicznym – liczba N wskazana przez prowadzącego ćwiczenie, a następnie sprawdzić działanie z użyciem symulatora.
8. Do poprzedniego zadanie dodać wejście zezwolenia na zliczanie.
9. Zaprojektować 4-bitowy rejestr przesuwający, szeregowo-równoległy, a następnie sprawdzić działanie rejestru z użyciem symulatora.
10. Zaprojektować *n*-bitowy licznik pierścieniowy - liczba bitów wskazana przez prowadzącego ćwiczenie, a następnie sprawdzić działanie z użyciem symulatora.

IV. ZAGADNIENIA DO OPRACOWANIA PRZED PRZYSTĄPIENIEM DO ĆWICZENIA

1. Wyjaśnić różnicę między układem kombinacyjnym a sekwencyjnym.
2. Podać definicję układu synchronicznego.
3. Jaka jest różnica między zerowaniem synchronicznym a asynchronicznym?
4. Opisać w języku VHDL przerzutnik typu D/T/JK.
5. Opisać w języku VHDL przerzutnik typu D z zerowaniem asynchronicznym/synchronicznym.
6. Opisać w języku VHDL przerzutnik typu D z ustawianiem asynchronicznym/synchronicznym.
7. Opisać w języku VHDL przerzutnik typu D z zerowaniem i ustawianiem asynchronicznym/synchronicznym.
8. Omówić instrukcję *process*.
9. Jakie instrukcje wykonywane są wewnątrz *processu*, a jakie poza? Podać przykłady takich instrukcji.

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.