KATEDRA INŻYNIERII KOMPUTEROWEJ

ModelSim® - VHDL

1. **Zakres tematyczny ćwiczenia:**
   - symulacja modeli z wymuszeniami Tcl,
   - opis układów kombinacyjnych w języku VHDL,
   - budowa architektury strukturalnej, konkretyzacja komponentów.

2. **Symulacja modeli z wymuszeniami Tcl**
   Proste modele urządzeń można testować z użyciem poleceń **force** wydawanych z okna transkrypcji lub zapisanych w pliku makra (.do)

   [!] Polecenie: utwórz nowy projekt o nazwie <nr_indeksu_1> w katalogu CADHDL na dysku wskazanym przez prowadzącego (jeśli katalog nie istnieje utwórz go); pozostaw domyślną nazwę dla biblioteki roboczej;

   [!] Polecenie: pobierz z serwera kursu pliki ‘mux_bit.vhd’ i ‘mux_slv.vhd’ oraz plik ‘symuluj.do’ i zapisz je w dowolnym katalogu tymczasowym; dodaj powyższe pliki do projektu z opcją ‘Copy to project directory’ i domyślnym typem źródła;

   [!] Polecenie: uruchom symulację obu modeli wykorzystując do tego celu makro ‘symuluj.do’, zlokalizuj i usuń błędy w kompilacji pliku ‘mux_slv.vhd’;

   [!] Polecenie: dokonaj porównania wyników symulacji dla obu modeli; z czego wynikają różnice w wynikach?

3. **Konkretyzacja komponentu**
   Instrukcja podstawienia (łączenia) komponentu to współbieżna instrukcja określająca wzajemne połączenie sygnałów i podukładów zwanych komponentami. Istnieją dwa sposoby przyporządkowania sygnałów:

   a) asocjacja pozycyjna,
   b) asocjacja nazw.

   przykład:

```
port map (d, clk, rst, q);    -- asocjacja pozycyjna
port map (clk=>clk_s, rst=>rst_s, d=>d_s, q=>q_s); -- asocjacja nazw
```

Wg standardu z roku 1987 komponent powinien być najpierw zadeklarowany w bloku architektury lub w pakiecie. Następnie można dokonać jego konkretyzacji w ciele architektury:

składnia:

```
component <nazwa_jednostki>
      port (...);
end component;
...
<etykieta>: <nazwa_jednostki>
            port map (...);
```

przykład:

```
component dff
      port (...);
end component;

przerzutnik: dff
            port map (...);
```

Konkretyzacja wg standardu z 1993 nie wymaga wcześniejszej deklaracji komponentu w zakresie deklaracyjnym architektury.

składnia:

```
<etykieta>: entity <nazwa_biblioteki>.<nazwa_jednostki>(<architektura>)
            port map (...);
```

przykład:

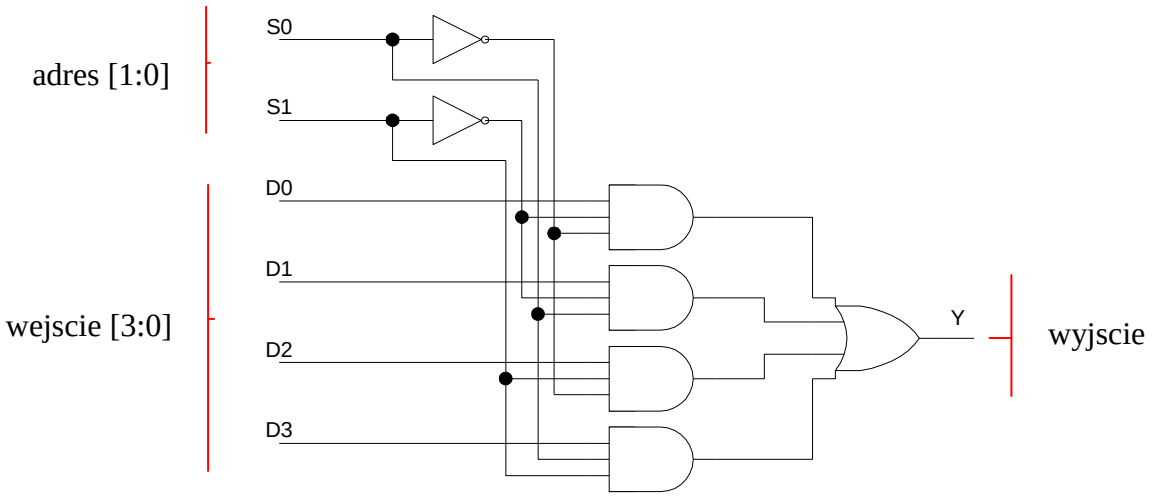
```
przerzutnik: entity work.dff(flip_flop)
            port map (...);
```

4. Budowa opisu strukturalnego

Aby zrealizować architekturę strukturalną wymagana jest znajomość budowy wewnętrznej modelowanego urządzenia. Opis strukturalny składa się z instrukcji przypisań komponentów i definicji połączeń między osadzonymi modułami. Połączenia muszą być realizowane przy pomocy **sygnałów** wewnętrznych, których typ odpowiada typowi portów w łączonych modułach.

[!] Polecenie: pobierz z serwera kursu plik ‘gates.vhd’ i zapisz go w dowolnym katalogu tymczasowym; dodaj powyższy plik do projektu z opcją ‘Copy to project directory’ i typem źródła ‘VHDL’;

[!Z] Polecenie: zbuduj model strukturalny multipleksera 4x1 (1-bitowa ścieżka danych) i dołącz go do projektu (w opisie strukturalnym należy wykorzystać elementy z pakietu ‘gates’);



[!Z] Polecenie: utwórz makro do kompilacji i symulacji powyższego modelu multipleksera (można wykorzystać lub zmodyfikować makro ‘symuluj.do’ z pkt.2); dokonaj symulacji urządzenia – **wyniki przedstaw prowadzącemu**.

[!Z] Zadanie:

a) Zbuduj model strukturalny licznika 2-dekadowego z wyświetlaniem na wskaźnikach 7-segmentowych, wykorzystaj gotowe komponenty z pakietu ‘devices.vhd’ [dostępny na serwerze kursu];

   licznik_u – licznik dekadowy z resetem asynchronicznym i wejściem clock_enable;
   seven_seg – dekoder kodu NKB na kod wskaźnika 7-segmentowego, układ kombinacyjny, zakres działania 0-9;

b) Dokonaj symulacji urządzenia (wymuszenia podawane przy pomocy makra!);
c) Wyniki symulacji i opis strukturalny przedstaw prowadzącemu.

[1] ModelSim®SE Reference Manual, Software Version 6.4a, Mentor Graphics 2008.