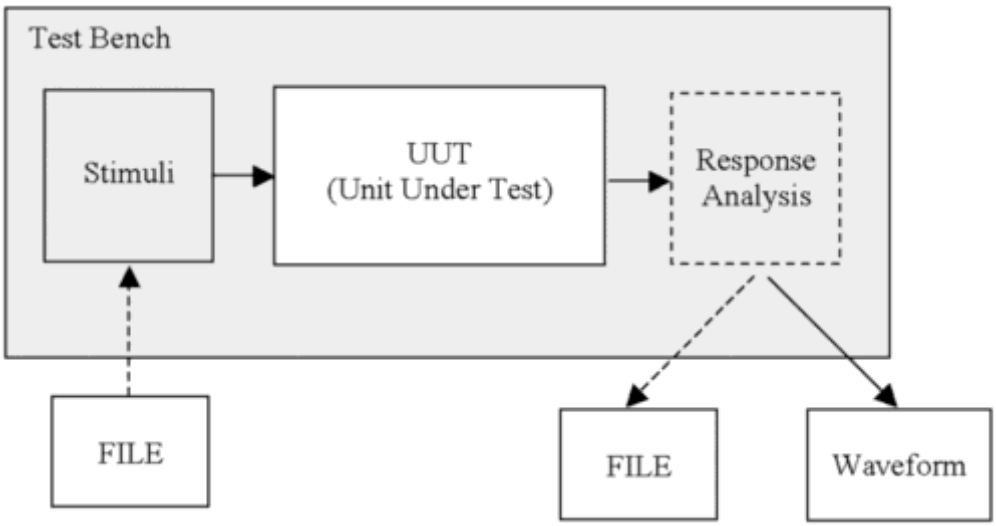
KATEDRA
INŻYNIERII KOMPUTEROWEJ

ModelSim® - VHDL

1. **Zakres tematyczny ćwiczenia:**
   - symulacja modeli z wykorzystaniem testbench'a VHDL,
   - opis behawioralny układów sekwencyjnych w języku VHDL,

   użycie poleceń `compare` i `dataset`.

2. **Symulacja modeli w środowisku testowym VHDL**



Testbench jest specyficzną jednostką projektową opisaną w języku VHDL. Z reguły nie posiada on żadnych portów i jest przeznaczony do symulacji projektowanego urządzenia (UUT na rys. powyżej). W architekturze testbench'a zdefiniowane są wymuszenia (*stimuli*) oraz sposób interpretacji wyników symulacji (szczególnie przydatne przy dużych projektach). Testbench może być pisany „ręcznie" przez projektanta lub generowany automatycznie przez graficzne edytory wymuszeń na podstawie opisu portów UUT.

[!] Polecenie: utwórz nowy projekt o nazwie <nr_indeksu_2> 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 'shift_reg.vhd' (model rejestru 8-bitowego) oraz 'shift_reg_tb.vhd' (testbench dla powyższego modelu) i zapisz je w dowolnym katalogu tymczasowym; dodaj pliki do projektu z opcją 'Copy to project directory' i domyślnym typem źródła;

[!] Polecenie: uruchom symulację modelu rejestru przesuwnego 8-bitowego korzystając z testbench'a VHDL; wytłumacz działanie i przeznaczenie procesu `sim_end_process`;

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
-----------------------------------------------
entity shift_reg_tb is
end entity shift_reg_tb;
-----------------------------------------------
architecture behav of shift_reg_tb is
signal clk_i, d_i : std_logic := '0';
signal q_i : std_logic_vector(7 downto 0);

begin
  -- instantiation of unit under test
  UUT: entity work.shift_reg(double_reg)
       port map (
       clk => clk_i,
       q => q_i,
       d => d_i);

  -- stimuli for clk
  clk_process: process
  begin
    wait for 20 ns;
    clk_i <= not clk_i;
  end process clk_process;

  -- stimuli for d_i
  d_i <= '0', '1' after 45 ns, '0' after 123 ns, '1' after 146 ns;

  -- end of simulation
  sim_end_process: process
  begin
    wait for 450 ns;
    assert false
      report "End of simulation at time " & time'image(now)
      severity Failure;
  end process sim_end_process;

end architecture behav;
-----------------------------------------------
```

[!] Polecenie: dokonaj interpretacji wyników symulacji; wyszukaj różnice pomiędzy symulowanym urządzeniem a rzeczywistym rejestrem SRR 8b (Shift Right Register);

[!Z] Polecenie: zdefiniuj **nową architekturę behawioralną** dla istniejącej jednostki projektowej `shift_reg`; działanie nowej architektury powinno odpowiadać rzeczywistemu układowi SRR; dokonaj symulacji nowozdefiniowanej architektury korzystając z testbench'a VHDL; wykonane modele prześlij na serwer kursu;

3. Weryfikacja wyników symulacji

Automatyczna weryfikacja wyników symulacji może być zdefiniowana wewnątrz testbench'a HDL lub realizowana przy pomocy funkcji wbudowanych symulatora. Do automatycznego porównania zestawów danych symulacyjnych służy funkcja `compare`. Funkcja ta operuje na plikach .wlf przechowujących spakowane przebiegi czasowe. Przed rozpoczęciem porównań należy najpierw zgromadzić kilka zestawów danych (funkcja `dataset`) lub dokonać konwersji danych ze standardowego formatu przebiegów czasowych (funkcja `vcd`) [1]. Porównywanie może być dokonywane asynchronicznie (wywołanie domyślne) lub synchronicznie względem

zadeklarowanego wcześniej zegara. Wydając polecenie `compare` można również zdefiniować limity różnic w porównywanych sygnałach i tolerancję przesunięć czasowych [1].

Podstawowe działania na zestawach danych:
- archiwizacja zestawu danych *datasetname* do pliku *filename*
```
dataset save <datasetname> <filename>
```
- załadowanie zestawu danych z pliku *filename*
```
dataset open <filename> [<logicalname>]
```
- zamknięcie zestawu danych
```
dataset close <logicalname> | [-all]
```

Podstawowa składnia funkcji porównania:
- stworzenie zestawu danych porównania
```
compare start <reference_dataset> [<test_dataset>]
```
- dodanie sygnałów do porównania /wersja asynchroniczna i synchroniczna/
```
compare add <name>
compare add -clock <name>
```
- uruchomienie wcześniej skonfigurowanego porównania /czasy opcjonalne/
```
compare run [<startTime>] [<endTime>]
```
- zapis wyników porównania do pliku tekstowego
```
compare info [-write <filename>]
```

Na zestawach danych porównania można operować tak samo jak na innych zestawach danych symulacyjnych.

[!Z] Polecenie:
a) zapisz wyniki symulacji modelu `shift_reg(double_reg)` do pliku ‘double_reg.wlf’;
b) zapisz wyniki symulacji nowej architektury modelu `shift_reg` do pliku .wlf o innej nazwie;
c) dokonaj porównania wszystkich sygnałów w obu zestawach danych korzystając z funkcji `compare` symulatora;
d) składnię wydawanych poleceń archiwizacji danych i automatycznego porównania zapisz w pliku makra; **uruchom makro w celu kontroli poprawności działania!**

[!Z] Zadanie:
a) Zbuduj testbench VHDL do symulacji licznika 2-dekadowego z wyświetlaniem na wskaźnikach 7-segmentowych (model wykonany w czasie lab.1);
b) W testbench’u zadeklaruj stałą (lub stałe) służącą do określania czasu trwania symulacji (powinny być powiązane z zadeklarowanym okresem zegara);
c) Wyniki symulacji, model i kod testbench’a przedstaw prowadzącemu.

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