Ref: David Pellerin VHDL Made Easy

Background: What is a Test Bench?

After a VHDL model is written, it must be tested to verify correct operation. To do this, the designer provides a VHDL simulator with a set of input signal transitions (stimulus vectors) to exercise the circuit. These stimulus vectors are usually grouped together in an entity called a **test bench.** Grouping the stimulus vectors together in an isolated entity provides a means to reuse the stimulus vectors as the VHDL code for the circuit changes (e.g. as the simulation models are changed from behavioral models to structural models).

In VHDL there are two ways to write stimulus vectors: using **wait** statements or using **transport** statements. Transport-based test benches are smaller and easier to read than wait-based test benches, but wait-based test benches are easier to understand when single-stepping through a simulation to debug it. Figure D-1 shows examples of both wait-based and transport-based test benches.

Writing a VHDL test bench is one of the most tedious and error prone parts of the simulation process. In VHDL you are required to specify the time for each signal transition. It is easy to make mistakes when writing an HDL test bench, because it is difficult to visualize the temporal relationships between transitions on different waveforms. For instance, take a look at the **transport** statements in the listings of Figure D-1 and see if you can figure out if the first rising edge of **SIG0** comes before the rising edge of the third cycle of clock **clk**. This is where a graphical timing diagram editor like The WaveFormer excels. Figure D-2 shows a timing diagram that generated the code in Figure D-1. Looking at the timing diagram, it is easy to see that the **SIG0** transition does occur before the rising edge of the clock.

sted to verify
ovides a
ansitions
se stimulus
ntity called a
gether in an
stimulus
nges (e.g. as
iavioral mod-

us vectors:
itements.
d easier to read
d test benches
ng through a
amples of both
s.

st tedious and
In VHDL you
nal transition. It
IDL test bench,
ral relation-
forms. For
ents in the
ure out if the
sing edge of
graphical
r excels. Figure
the code in
it is easy to see
he rising edge

Transport VHDL Test Bench

```
library ieee, std; use std.textio.all;
use ieee.std_logic_1164.all;
entity testbench is
 port( CLK0: out std_logic;
    SIG0 : out std_logic;
    SIG1 : out integer;
    SIG2 : out std_logic);
end testbench;
architecture test of testbench is
begin
 process
 begin
  CLK0 <= '1'; wait for 0 ps;
  while true loop
   CLK0 <= '0'; wait for 25000 ps;
   CLK0 <= '1'; wait for 25000 ps;
  end loop;
 end process;
 process
 begin
  SIG0 <= transport '1' after 0 ps,
          '0' after 50000 ps,
          '1' after 90000 ps,
          '0' after 175000 ps;
  SIG1 <= transport 4 after 0 ps,
          16 after 55000 ps,
          9 after 120000 ps,
          12 after 180000 ps,
          3 after 235000 ps;
  SIG2 <= transport '0' after 0 ps,
          '1' after 20000 ps,
          '0' after 40000 ps,
          '1' after 60000 ps,
          '0' after 80000 ps,
          '1' after 100000 ps,
          '0' after 120000 ps,
          '1' after 140000 ps,
          '0' after 160000 ps,
          '1' after 170000 ps,
          '0' after 180000 ps,
          '1' after 190000 ps,
          '0' after 200000 ps,
          '1' after 210000 ps,
          '0' after 220000 ps,
          '1' after 230000 ps,
          '0' after 240000 ps;
  wait;
 end process;
end test;
```

Wait VHDL Test Bench

```
library ieee, std;
use std.textio.all;
use ieee.std_logic_1164.all;
entity testbench is
 port( CLK0: out std_logic;
    SIG0 : out std_logic;
    SIG1 : out integer;
    SIG2 : out std_logic);
end testbench;
architecture test of testbench is
begin
 process
 begin
  CLK0 <= '1'; wait for 0 ps;
  while true loop
   CLK0 <= '0'; wait for 25000 ps;
   CLK0 <= '1'; wait for 25000 ps;
  end loop;
 end process;
 process
 begin
   SIG0 <= '1';
   SIG1 <= 4;
   SIG2 <= '0';
   wait for 20000 ps;
   SIG2 <= '1';
   wait for 20000 ps;
   SIG2 <= '0';
   wait for 10000 ps;
   SIG0 <= '0';
   wait for 5000 ps;
   SIG1 <= 16;
   wait for 5000 ps;
   SIG2 <= '1';
   wait for 20000 ps;
   SIG2 <= '0';
   wait for 10000 ps;
   SIG0 <= '1';
   wait for 10000 ps;
   -- More wait statement code
   -- removed because of space limitations
   wait for 848 ps;
   wait;
 end process;
end test;
```

Figure D-1: *VHDL test benches using* ***transport*** *and* ***wait*** *statements.*