VHDL与Verilog语言的语法规则不同，它们的TestBench的具体写法也不同，但是应包含的基本结构大体相似，在VHDL的仿真文件中应包含以下几点：实体和结构体声明、信号声明、顶层设计实例化、提供激励；Verilog的仿真文件应包括：模块声明、信号声明、顶层设计实例化、提供激励。

Testbench 是一种诊断/验证手段，从软件层面对设计的电路进行仿真。一般是在仿真文件里，产生激励信号，作用于设计文件DUT（Design Under Test），产生相应的输出，然后根据输出信号检验设计的电路是否存在问题及存在哪些问题。

这里以一个流水灯的设计来说明编写方法，使用Verilog语言来编写。

led\_run.v

module led\_run(clk,rst,led);

input clk,rst;

output reg [7:0] led;

reg [25:0] clk\_cnt;

reg clk\_tmp;

reg [3:0] temp;

always@(posedge clk or negedge rst)

begin

if(!rst)

begin

clk\_cnt<=26'd0;

clk\_tmp<=1'b1;

end

else

begin

if(clk\_cnt==26'b11111111111111111111111111)

begin

clk\_cnt<=26'd0;

clk\_tmp<=~clk\_tmp;

end

else

clk\_cnt<=clk\_cnt+1'b1;

end

end

always@(posedge clk\_tmp or negedge rst)

begin

if(!rst)

temp<=4'd15;

else

temp<=temp+1'b1;

end

always@(temp)

begin

case(temp)

4'd0 :led<=8'b11111110;

4'd1 :led<=8'b11111100;

4'd2 :led<=8'b11111000;

4'd3 :led<=8'b11110000;

4'd4 :led<=8'b11100000;

4'd5 :led<=8'b11000000;

4'd6 :led<=8'b10000000;

4'd7 :led<=8'b00000000;

4'd8 :led<=8'b00000001;

4'd9 :led<=8'b00000011;

4'd10:led<=8'b00000111;

4'd11:led<=8'b00001111;

4'd12:led<=8'b00011111;

4'd13:led<=8'b00111111;

4'd14:led<=8'b01111111;

4'd15:led<=8'b11111111;

default:;

endcase

end

endmodule

上面是一段流水灯代码，控制8位流水灯依次点亮，再依次熄灭。第一个always语句完成分频功能，第二个always语句用于计数，共16个值，第三个always语句根据计数的值选择LED灯的状态。其中clk、rst分别为时钟和复位信号，led为驱动流水灯的输出信号。接下来针对这一设计编写其TestBench文件。

tb\_led\_run.v

module tb\_led\_run;

reg clk,rst;

wire led;

initial

begin

rst=1;

#30 rst=0;

#40 rst=1;

end

initial

begin

clk=1;

forever #20 clk=~clk;

end

led\_run led1(.clk(clk),.rst(rst),.led(led));

endmodule

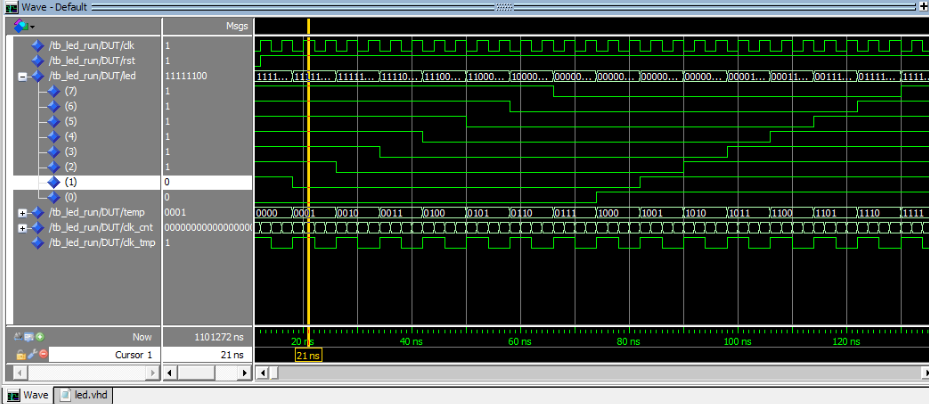
接下来在Modelsim中进行仿真，结果如图1

图1

总结起来，Verilog语言的testbench文件有着固定的格式：  
module test\_bench;

端口声明语句

initial

begin

产生时钟信号

end

initial

begin

提供激励源

end

例化语句

Endmodule

最主要的是在initial语句中进行激励的生成，这要根据具体的设计来分析。

以下是Vhdl语言编写

1 LIBRARY IEEE;

2 USE IEEE.STD\_LOGIC\_1164.ALL;

3 USE IEEE.STD\_LOGIC\_UNSIGNED.ALL;

4

5 ENTITY led\_run IS

6 PORT(clk:in std\_logic;

7 rst:in std\_logic;

8 led:out std\_logic\_vector(7 downto 0):="11111111" );

9 END led\_run;

10

11 ARCHITECTURE arc\_led\_run OF led\_run IS

12 signal temp:std\_logic\_vector(3 downto 0);

13 signal clk\_cnt:std\_logic\_vector(25 downto 0);

14 signal clk\_tmp:std\_logic:='1';

15 BEGIN

16 divider:PROCESS(clk,rst)

17 BEGIN

18 if(rst='0') then

19 clk\_cnt<="00000000000000000000000000";

20 elsif(clk'event and clk='1') then

21 clk\_cnt<=clk\_cnt+1;

22 if(clk\_cnt="11111111111111111111111111") then

23 clk\_cnt<="00000000000000000000000000";

24 clk\_tmp<=NOT clk\_tmp;

25 end if;

26 end if;

27 END PROCESS;

28

29 PROCESS(clk\_tmp,rst)

30 BEGIN

31 if(rst='0') then

32 temp<="1111"; --all the led off

33 elsif(clk\_tmp'event and clk\_tmp='1') then

34 temp<=temp+1;

35 end if;

36 END PROCESS;

37

38 PROCESS(temp)

39 BEGIN

40 case temp is

41 when"0000"=>led<="11111110";

42 when"0001"=>led<="11111100";

43 when"0010"=>led<="11111000";

44 when"0011"=>led<="11110000";

45 when"0100"=>led<="11100000";

46 when"0101"=>led<="11000000";

47 when"0110"=>led<="10000000";

48 when"0111"=>led<="00000000";

49 when"1000"=>led<="00000001";

50 when"1001"=>led<="00000011";

51 when"1010"=>led<="00000111";

52 when"1011"=>led<="00001111";

53 when"1100"=>led<="00011111";

54 when"1101"=>led<="00111111";

55 when"1110"=>led<="01111111";

56 when"1111"=>led<="11111111";

57 when others=>NULL;

58 end case;

59 END PROCESS;

60

61 END arc\_led\_run;

根据语法要求，首先声明库，接着定义实体和结构体。在结构体中用三个进程(PROCESS)分别实现分频、计数、流水灯状态分配的功能，功能相当于上面Verilog程序中的三个always语句。接下来写TestBench文件：

2 LIBRARY IEEE;

3 USE IEEE.STD\_LOGIC\_1164.ALL;

4

5

6 ENTITY tb\_led\_run IS --空实体

7 END tb\_led\_run;

8

9

10 ARCHITECTURE arc\_tb\_led\_run OF tb\_led\_run IS --结构体

11

12 COMPONENT led\_run IS --元件声明

13 PORT(clk:in std\_logic;

14 rst:in std\_logic;

15 led:out std\_logic\_vector(7 downto 0));

16 END COMPONENT;

17

18 signal clk,rst:std\_logic;

19 signal led:std\_logic\_vector(7 downto 0);

20 constant clk\_period:time:=5 ns;

21

22 BEGIN

23

24 DUT:led\_run PORT MAP(clk=>clk,rst=>rst,led=>led); --元件例化

25

26 clk\_gen:PROCESS

27 BEGIN

28 clk<='1';

29 wait for clk\_period/2;

30 clk<='0';

31 wait for clk\_period/2;

32 END PROCESS;

33

34 tb:PROCESS

35 BEGIN

36 rst<='0';

37 wait for 12 ns;

38 rst<='1';

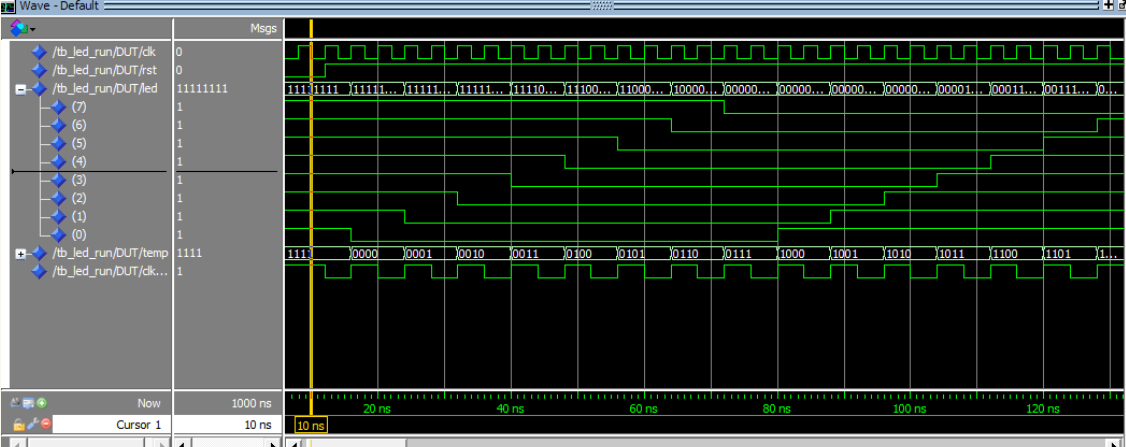
39 wait;

40 END PROCESS;

41

42 END arc\_tb\_led\_run;

在这个TestBench中同样只需要提供clk和rst信号，分别在两个进程实现，Modelsim中的仿真结果如图2(同样在仿真的时候将分频系数改为2)：

图2

相对与Verilog语言来说，VHDL的TestBench除了自身的库声明以及Entity和Architecture之外，还需要进行元件的声明，即将被测试的设计声明为一个元件，然后对其例化。