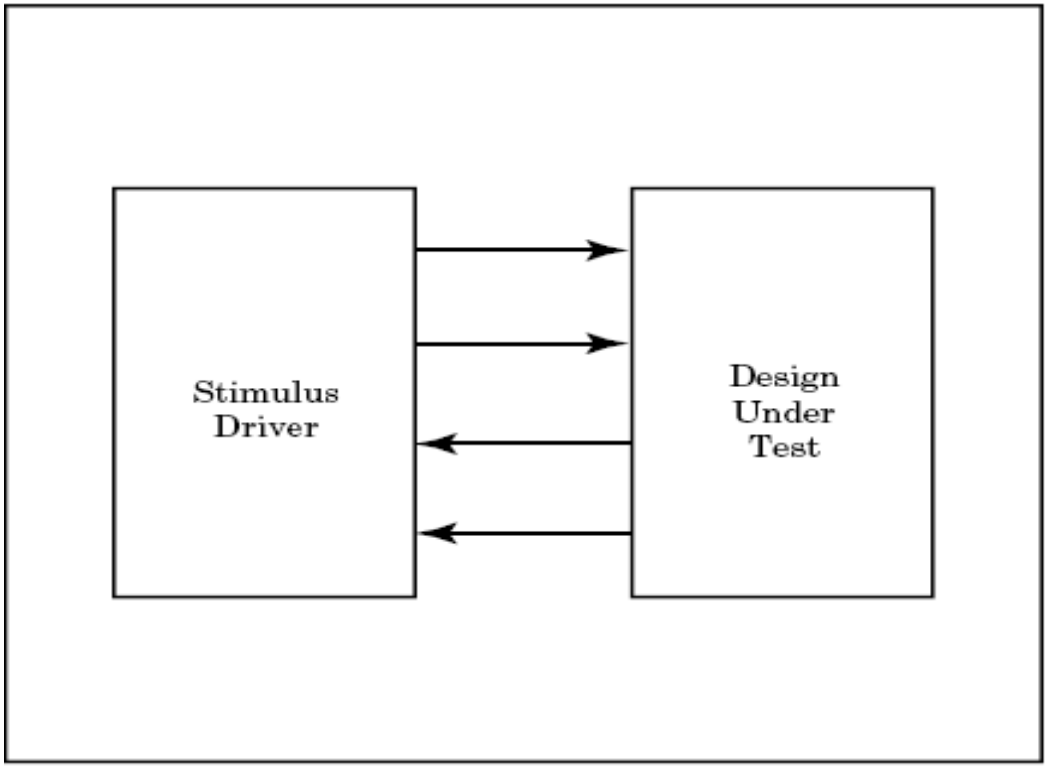
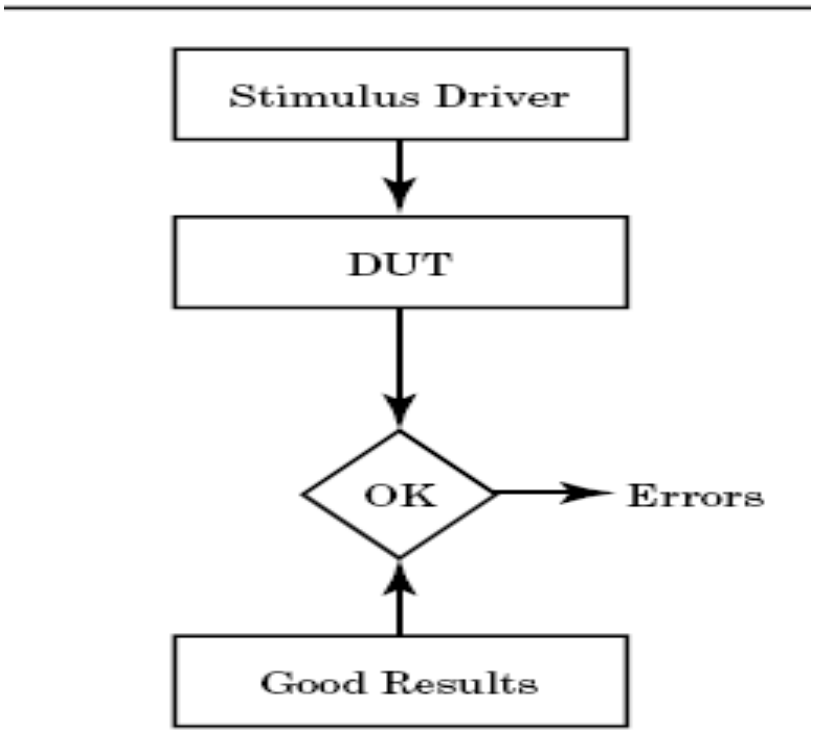
上海交通大學
Shanghai Jiao Tong University

硬件描述语言-VHDL

Testbench

Testbench


Stimulus
Driver
Design
Under
Test


Stimulus Driver
DUT
OK
Errors
Good Results

计数器设计

位宽为 8 位的计数器，此计数器可实现向上计数和向下计数(由 up_dwn 信号 控制)，可设置计数初值(ld 信号为高时，设置计数初值)，并且带时钟使能信号 clk_en,

```
ENTITY count IS
PORT (clk : IN std_logic;
        ld : IN std_logic;
        up_dwn : IN std_logic;
        clk_en : IN std_logic;
        din : IN std_logic_vector(7 downto 0);
        qout : buffer std_logic_vector(7 downto 0);
END count;
```

Testbench要点

上海交通大学 Shanghai Jiao Tong University

- 时钟信号的产生
- 激励信号的产生

  利用语句和仿真软件命令产生

  通过读取文件产生
- 输出信号的验证

  观察输出信号波形

  通过和文件中存储的真值比对

时钟的产生

上海交通大学 Shanghai Jiao Tong University

```
-- Declare a clock period constant.
TIME := 10 ns;

-Clock 产生方法1:
Clock <= not Clock after ClockPeriod / 2;

-Clock 产生方法2:
GENERATE_CLOCK: process
begin
   wait for (ClockPeriod / 2) Clock <= '1';
   wait for (ClockPeriod / 2) Clock <= '0';
end process;
```

激励信号的产生

上海交通大学 Shanghai Jiao Tong University

利用VHDL的语句产生：

```
process begin
     Ld <= '0';
    UpDwn <= '0';
    wait for 100 ns;
    wait for 20 ns;
     Ld <= '1';
     din <= 20;
     wait for 20 ns;
     UpDwn <= '1';
 end process;
```

利用仿真软件的命令产生:

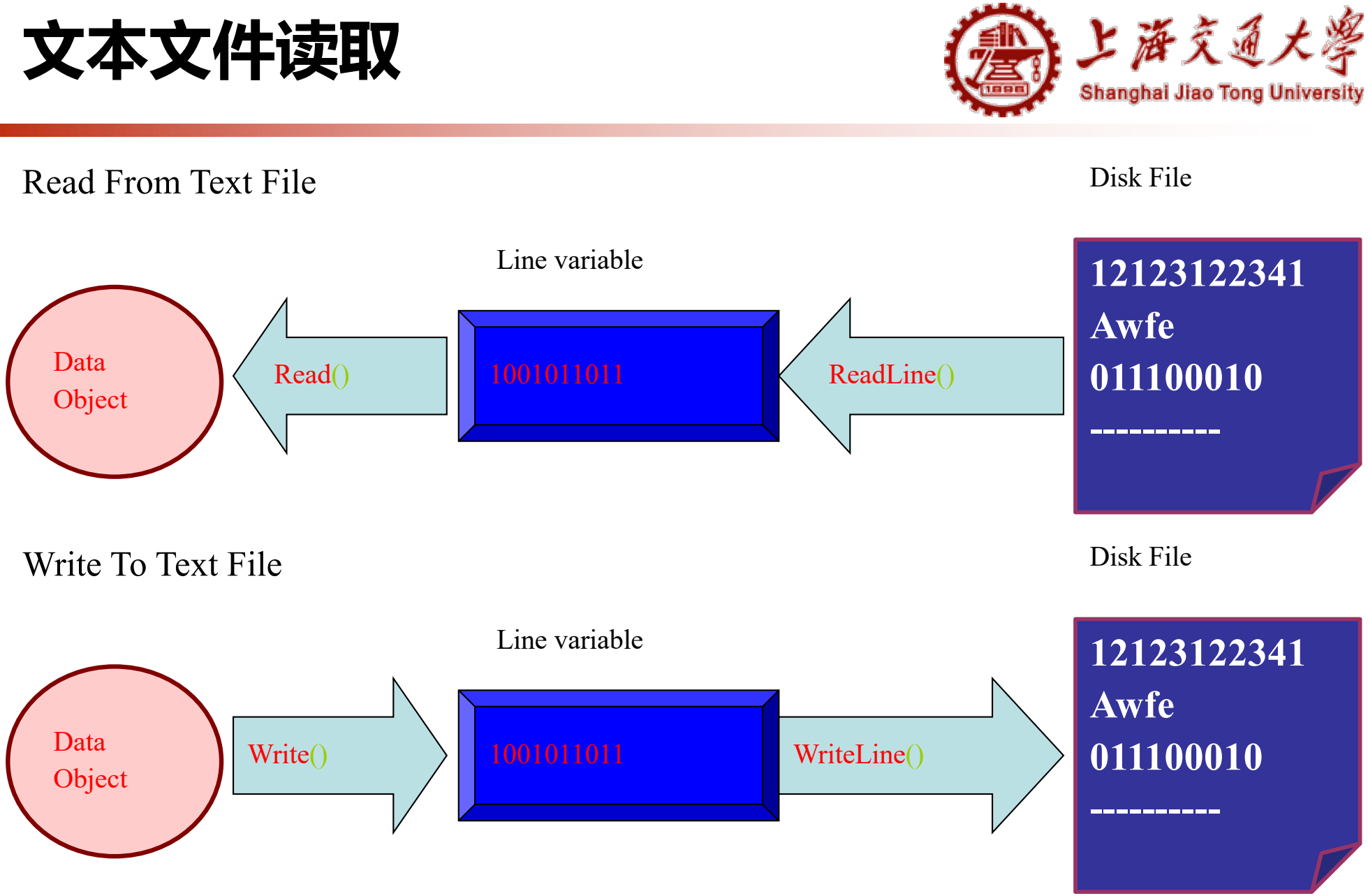
```
force ld 0
force updwn 0
run 120
Force ld 1
Force din 16#14
Run 20
Force UpDwn  1
```

激励信号的产生-通过文件读取

上海交通大學
Shanghai Jiao Tong University

```
time clk ld up_dwn clk_en din
10 0001 0
20 1101 50
30 0001 0
40 1001 0
50 0001 0
60 1001 0
70 0001 0
80 1001 0
90 0001 0
100 1101 10
110 0001 0
120 1001 0
130 0001 0
140 1001 0
150 0001 0
160 1001 0
```

文本文件读取


Read From Text File
Disk File
Line variable
Data Object
Read()
1001011011
ReadLine()
12123122341
Awfe
011100010

Write To Text File
Disk File
Line variable
Data Object
Write()
1001011011
WriteLine()
12123122341
Awfe
011100010

文本文件读取主要步骤

上海交通大学 Shanghai Jiao Tong University

- 定义待读取的数据类型DataObj

  VARIABLE tmpclk, tmpld

- 定义Line对象

  variable LineObj: line;

- 定义文件对象

  file FileObject: text is in "FileName";
  file FileObject: text is out "FileName";

- 读取一行数据到Line对象中

  readLine(FileObj, LineObj);

- 读取Line对象中的数据到相应的数据类型中

  read(LineObj, DataObj);

文本文件写入主要步骤

上海交通大学 Shanghai Jiao Tong University

- 定义待读取的数据类型DataObj

  VARIABLE tmpclk, tmpld

- 定义Line对象

  variable LineObj: line;

- 定义文件对象

  file FileObject: text is in "FileName";
  file FileObject: text is out "FileName";

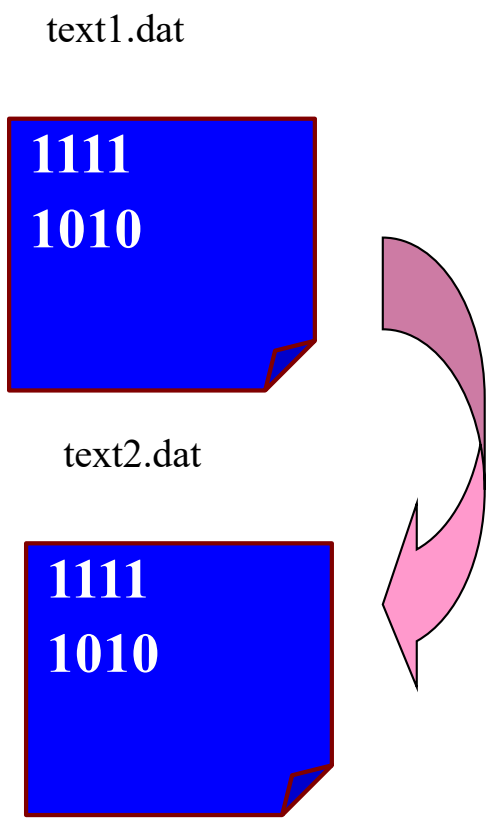
- 写数据到Line对象中

  Write(LineObj, DataObj);

- 写Line对象中的数据到相应的文件中

  WriteLine(FileObj, LineObj);

Read/Write Text File (Example)




```
library ieee;
use ieee.std_logic_1164.all;
use std.textio.all;
entity text_file_read is
end text_file_read;
architecture text_file_read_a of text_file_read is
begin
    process
        variable bv: bit_vector(3 downto 0);
        variable ln_in: line;
        variable ln_out: line;
        file file_in: text is in "text1.dat";
        file file_out: text is out "text2.dat";
    begin
        loop
            exit when endfile(file_in);
            readline(file_in, ln_in);
            read(ln_in, bv);
            write(ln_out, bv);
            writeline(file_out, ln_out);
        end loop;
        wait;
    end process;
end text_file_read_a;
```

激励信号的产生-通过文件读取

上海交通大学
Shanghai Jiao Tong University

```
time clk ld up_dwn clk_en din
10 0001 0
20 1101 50
30 0001 0
40 1001 0
50 0001 0
60 1001 0
70 0001 0
80 1001 0
90 0001 0
100 1101 10
110 0001 0
120 1001 0
130 0001 0
140 1001 0
150 0001 0
160 1001 0
```

计数器的Testbench文件


```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE std.textio.ALL;
USE ieee.std_logic_textio.all;
USE WORK.count_types.all;

ENTITY testbench IS END;

ARCHITECTURE full OF testbench IS
COMPONENT count
PORT (clk : IN std_logic;
          ld : IN std_logic;
          up_dwn : IN std_logic;
          clk_en : IN std_logic;
          din : IN bit8;
         qout : INOUT bit8);
END COMPONENT;
SIGNAL clk, ld, up_dwn, clk_en : std_logic;
SIGNAL qout, din : std_logic_vector(7 downt
    o 0);
BEGIN
uut: count PORT MAP(clk => clk,
ld => ld, up_dwn => up_dwn,clk_en => c
      lk_en,din => din,qout => qout);
test: PROCESS
VARIABLE tmpclk, tmpld, tmpup_dwn, tm
      pclk_en :std_logic;
VARIABLE tmpdin :integer;
FILE vector_file : text IS IN  "counter.txt";
VARIABLE l : line;
VARIABLE vector_time : time;
VARIABLE r : integer;
VARIABLE good_number, good_val : bool
      ean;
VARIABLE space : character;
BEGIN
WHILE NOT endfile(vector_file) LOOP
readline(vector_file, l);
```

计数器的Testbench文件


```
read(l, r);
vector_time := r * 1 ns;
IF (now < vector_time) THEN
WAIT FOR vector_time - now;
END IF;
read(l, space); --- skip a space
-- read clk value
read(l, tmpclk, good_val);
assert good_val REPORT  "bad clk value" ;
-- read ld value
read(l, tmpld, good_val);
assert good_val REPORT  "bad ld value" ;
-- read up_dwn value
read(l, tmpup_dwn, good_val);
assert good_val REPORT  "bad up_dwn value" ;
-- read clk_en value
read(l, tmpclk_en, good_val);
assert good_val REPORT  "bad clk_en value" ;
read(l, space); --- skip a space
read(l, tmpdin, good_val);
assert good_val REPORT  "bad din value"
clk <= tmpclk;
ld <= tmpld;
up_dwn <= tmpup_dwn;
clk_en <= tmpclk_en;
din <= conv_std_logic_vector(tmpdin,din'
      length);
END LOOP;
ASSERT false REPORT  "Test complete" ;
WAIT;
END PROCESS;
END full;
```

加入输出信号验证的测试文件


```
-- time clk ld up_dwn clk_en din dout
0 0001 0 0
10 1001 0 255
20 0101 10 255
30 1001 0 10
40 0001 0 10
50 1001 0 8
60 0001 0 8
70 1001 0 7
80 0001 0 7
90 1001 0 6
100 0101 100 100
110 1001 0 100
120 0001 0 100
130 1001 0 98
140 0001 0 98
150 1001 0 97
160 0001 0 97
```

计数器的Testbench文件

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE std.textio.ALL;
USE ieee.std_logic_textio.all;
USE WORK.count_types.all;

ENTITY testbench IS END;

ARCHITECTURE full OF testbench IS
COMPONENT count
PORT (clk : IN std_logic;
        ld : IN std_logic;
        up_dwn : IN std_logic;
        clk_en : IN std_logic;
        din : IN bit8;
       qout : INOUT bit8);
END COMPONENT;
SIGNAL clk, ld, up_dwn, clk_en : std_logic;
SIGNAL qout, din : std_logic_vector(7 downt
    o 0);
BEGIN
uut: count PORT MAP(clk => clk,
ld => ld, up_dwn => up_dwn,clk_en => c
      lk_en,din => din,qout => qout);
test: PROCESS
VARIABLE tmpclk, tmpld, tmpup_dwn, tm
      pclk_en :std_logic;
VARIABLE tmpqout ,tmpdin :integer;
VARIABLE tmpqout_s: std_logic_vector(7
      down to 0);
FILE vector_file : text IS IN  "counter.txt" ;
VARIABLE l : line;
VARIABLE vector_time : time;
VARIABLE r : integer;
VARIABLE good_number, good_val : bool
      ean;
VARIABLE space : character;
BEGIN
WHILE NOT endfile(vector_file) LOOP
readline(vector_file, l);
```

计数器的Testbench文件


```
read(l, r);
vector_time := r * 1 ns;
IF (now < vector_time) THEN
WAIT FOR vector_time - now;
END IF;
read(l, space); --- skip a space
-- read clk value
read(l, tmpclk, good_val);
assert good_val REPORT  "bad clk value" ;
-- read ld value
read(l, tmpld, good_val);
assert good_val REPORT  "bad ld value" ;
-- read up_dwn value
read(l, tmpup_dwn, good_val);
assert good_val REPORT  "bad up_dwn value" ;
-- read clk_en value
read(l, tmpclk_en, good_val);
assert good_val REPORT  "bad clk_en value" ;
read(l, space); --- skip a space
read(l, tmpdin, good_val);
assert good_val REPORT  "bad din valu
      e" ;
read(l, space); --- skip a space
---- the difference in the file is below
read(l, tmpqout, good_val);
assert good_val REPORT  "bad qout valu
      e" ;
Tmpqout_c = conv_std_logic_vector(tmp
      qout,qout' length);
assert tmpqout_c = qout REPORT  "vect
      or mismatch" ;
clk <= tmpclk;
ld <= tmpld;
up_dwn <= tmpup_dwn;
clk_en <= tmpclk_en;
din <= conv_std_logic_vector(tmpdin,din'
      length);
END LOOP;
ASSERT false REPORT  "Test complete" ;
WAIT;
END PROCESS;
END full;
```

Testbench-仿真软件命令产生


```
-- setup the clock
force -repeat 20 clk 0 0, 1 10
-- log the results to a file
list *
-- setup initial signal conditions
force ld 0
force up_dwn 0
force clk_en 1
force din 16#00
-- run the simulation
run 100
--- set next signal conditions
force ld 1
force up_dwn 0
force clk_en 1
force din 16#AA
--- run the simulation
run 200
--- set next signal conditions
```

```
force ld 1
force up_dwn 0
force clk_en 1
force din 16#55
--- run the simulation
run 200
write list data.out
quit -f
```