1.1.1 Verilog 语言中的系统任务和系统函数

Verilog 语言中预先定义了一些任务和函数，用于完成一些特殊的功能，它们被称为系统任务和系统函数，这些函数大多数都是只能在 Testbench 仿真中使用的，使我们更方便的进行验证。

```
`timescale 1ns/1ns  //时间尺度预编译指令 时间单位/时间精度
```

时间单位和时间精度由值 1、10、和 100 以及单位 s、ms、us、ns、ps 和 fs 组成

时间单位：定义仿真过程所有与时间相关量的单位

仿真中使用“#数字”表示延时相应时间单位的时间，例#10 表示延时 10 个单位的时间，即 10ns

时间精度：决定时间相关量的精度及仿真显示的最小刻度

```
`timescale 1ns/10ps  精度 0.01，#10.11 表示延时 10110ps
```

下面这种写法就是错误的，因为时间单位不能比时间精度小

```
`timescale 100ps/1ns
```

主要的函数有如下这些，在支持 Verilog 语法的编辑器中都会显示为高亮关键字

```
//------------------------------------------------
$display   //打印信息，自动换行
$write     //打印信息
$strobe    //打印信息，自动换行，最后执行
$monitor   //监测变量
$stop      //暂停仿真
$finish    //结束仿真
$time      //时间函数
$random    //随机函数
$readmemb  //读文件函数
//------------------------------------------------
```

下面我们单独介绍它们的功能，并在 ModelSim 的 Transcript 界面中打印这些信息。

1、$display 用于输出、打印信息

使用格式为：

```
$display("%b+%b=%d",a, b, c); //格式“%b+%b=%d” 格式控制，未指定时默认十进制
%h 或%H //以十六进制的形式输出
%d 或%D //以十进制的形式输出
%o 或%O //以八进制的形式输出
%b 或%B //以二进制的形式输出
//------------------------------------------------
//a,b,c 输出列表，需要输出信息的变量
//每次打印信息后自动换行
```

```
01 `timescale 1ns/1ns
02
03 module tb_test();
04
05 reg [3:0] a;
06 reg [3:0] b;
07 reg [3:0] c;
08
```

```
09 initial begin
10  $display("Hello");
11      $display("xiangliangzi");
12      a = 4'd5;
13      b = 4'd6;
14      c = a + b;
15      #100;
16      $display("%b+%b=%d", a, b, c);
17 end
18
19 endmodule
//-------------------------------------------------
```

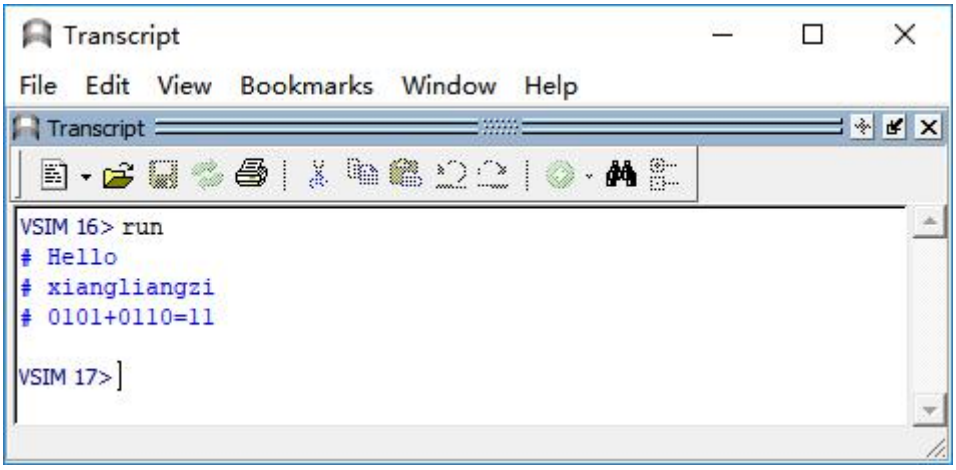

图 5-1

2、$write 用于输出、打印信息

使用格式为：

```
$write("%b+%b=%d\n",a, b, c); //"%b+%b=%d\n" 格式控制，未指定时默认十进制
%h 或%H //以十六进制的形式输出
%d 或%D //以十进制的形式输出
%o 或%O //以八进制的形式输出
%b 或%B //以二进制的形式输出
\n    //换行
//-------------------------------------------------
//a,b,c 为输出列表，需要输出信息的变量
01 `timescale 1ns/1ns
02
03 module tb_test();
04
05 reg [3:0] a;
06 reg [3:0] b;
07 reg [3:0] c;
08
09 initial begin
```

```
10      $write("Hello ");
11      $write("xiangliangzi\n");
12      a = 4'd5;
13      b = 4'd6;
14      c = a + b;
15      #100;
16      $write("%b+%b=%d\n",a, b, c);
17  end
18
19  endmodule
//-------------------------------------------------
```

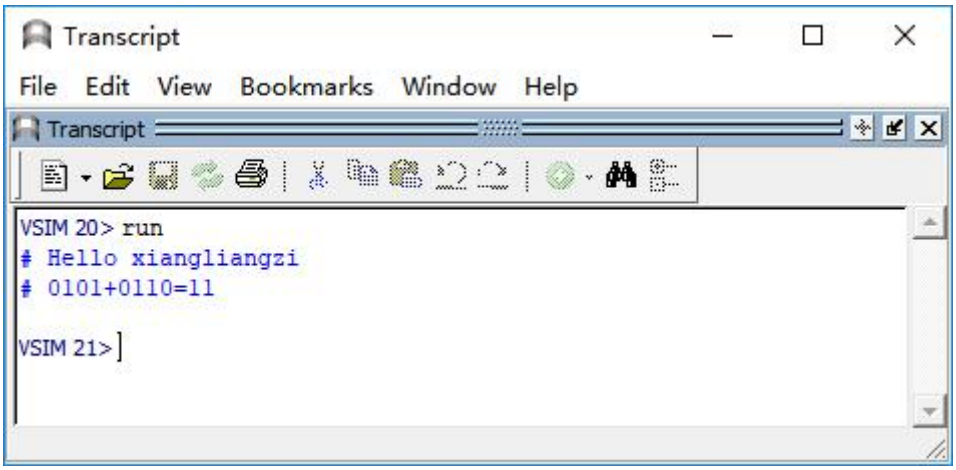

图 5-2

3、$strobe 用于输出、打印信息

使用格式为：

```
$strobe("%b+%b=%d",a,b,c); //"%b+%b=%d" 格式控制，未指定时默认十进制
%h 或%H //以十六进制的形式输出
%d 或%D //以十进制的形式输出
%o 或%O //以八进制的形式输出
%b 或%B //以二进制的形式输出
//-------------------------------------------------
//a,b,c 输出列表，需要输出信息的变量
//打印信息后自动换行，触发操作完成后执行
01  `timescale 1ns/1ns
02
03  module tb_test();
04
05  reg [3:0] a;
06  reg [3:0] b;
07  reg [3:0] c;
08
09  initial begin
10      $strobe("strobe:%b+%b=%d", a, b, c);
```

```
11      a = 4'd5;
12      $display("display:%b+%b=%d", a, b, c);
13      b = 4'd6;
14      c = a + b;
15  end
16
17  endmodule
//-------------------------------------------------
```

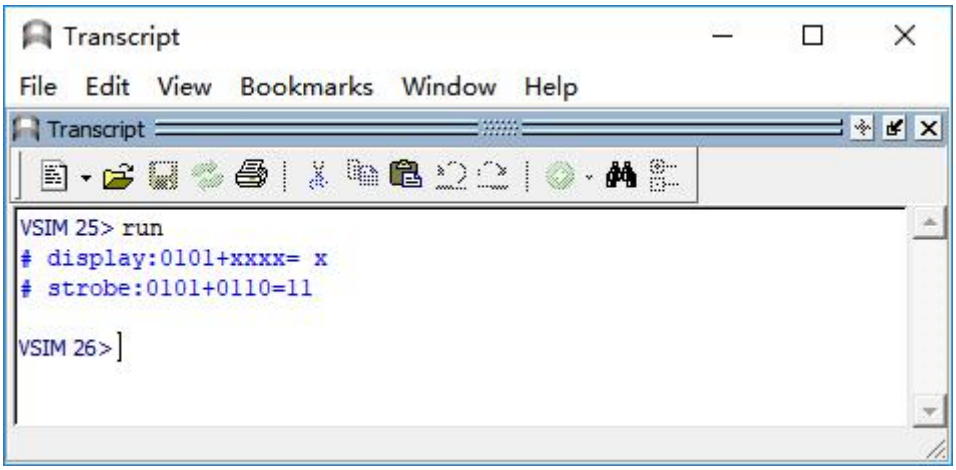

图 5-3

4、$monitor 用于持续监测变量

使用格式为：

```
$monitor("%b+%b=%d",a,b,c); //"%b+%b=%d" 格式控制，未指定时默认十进制
%h 或%H //以十六进制的形式输出
%d 或%D //以十进制的形式输出
%o 或%O //以八进制的形式输出
%b 或%B //以二进制的形式输出
//-------------------------------------------------
//a,b,c 输出列表，需要输出信息的变量
//被测变量变化触发打印操作，自动换行
01  `timescale 1ns/1ns
02
03  module tb_test();
04
05  reg [3:0] a;
06  reg [3:0] b;
07  reg [3:0] c;
08
09  initial begin
10      a = 4'd5;
11      #100;
12      b = 4'd6;
13      #100;
```

```
14      c = a + b;
15  end
16
17  initial $monitor("%b+%b=%d", a, b, c);
18
19  endmodule
//------------------------------------------------
```

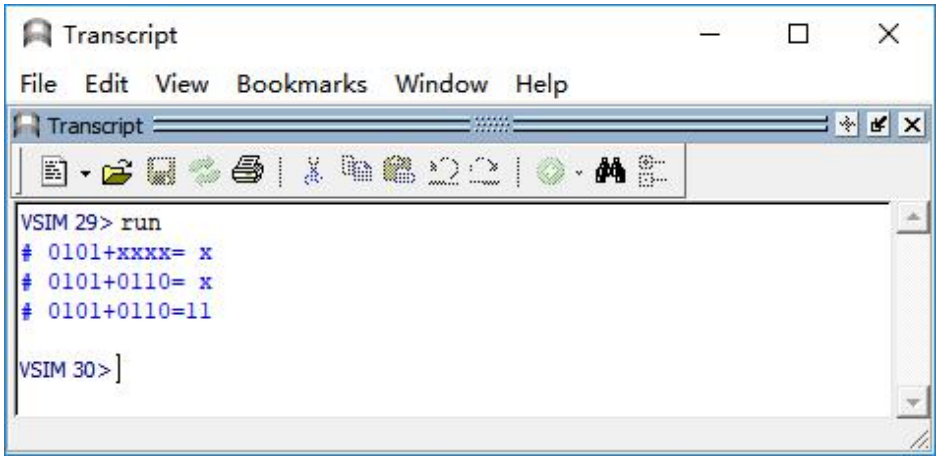

图 5-4

5、$stop 用于暂停仿真，$finish 用于结束仿真

```
//------------------------------------------------
01  `timescale 1ns/1ns
02
03  module tb_test();
04
05  initial begin
06      $display("Hello");
07      $display("xiangliangzi");
08      #100;
09      $display("Stop Simulation");
10      $stop; //暂停仿真
11      $display("Continue Simulation");
12      #100;
13      $display("Finish Simulation");
14      $finish;   //结束仿真
15  end
16
17  endmodule
//------------------------------------------------
```

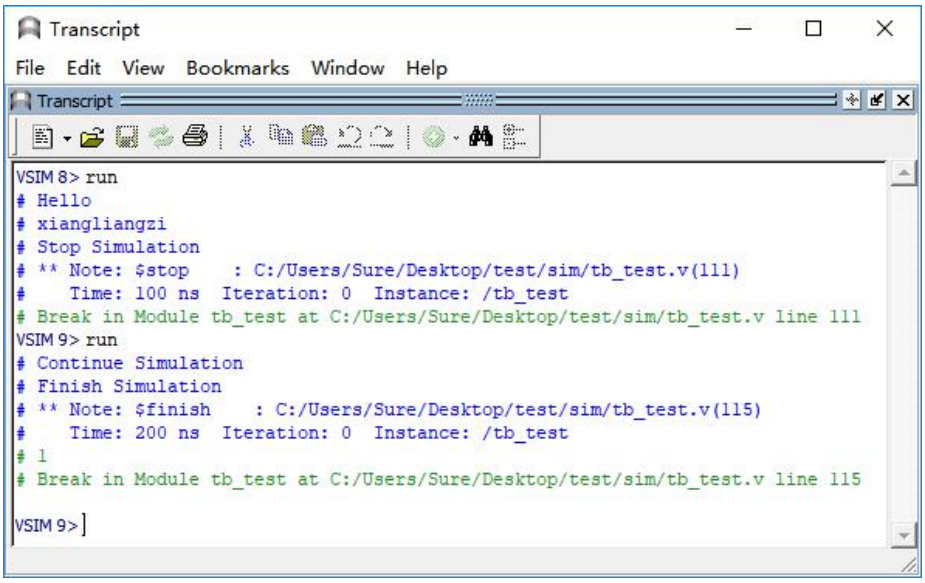

图 5-5

6、$time 为时间函数，返回 64 位当前仿真时间；$random 用于产生随机函数，返回随机数

```
//------------------------------------------------------
01 `timescale 1ns/1ns
02
03 module tb_test();
04
05 reg [3:0] a;
06
07 always #10 a = $random;
08
09 initial $monitor("a=%d @time %d", a, $time);
10
11 endmodule
//------------------------------------------------------
```

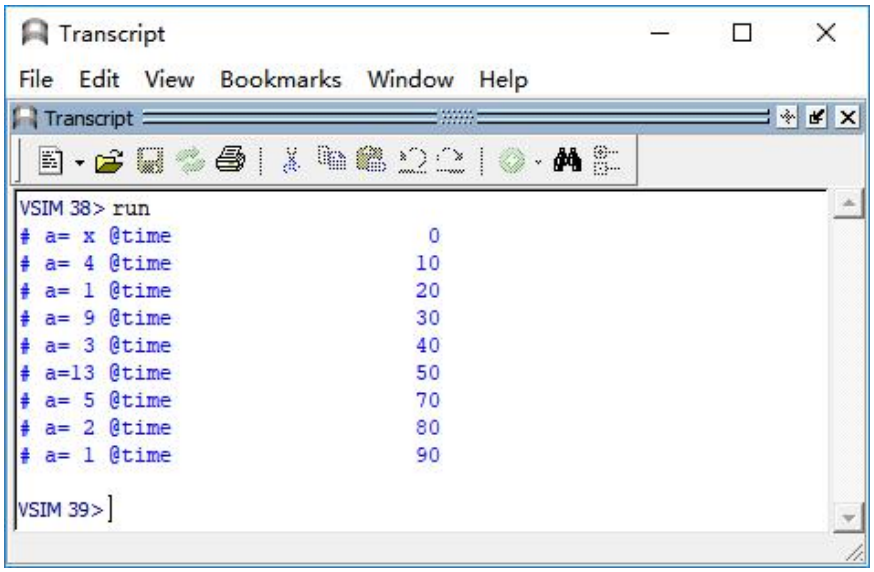

图 5-6

7、$readmemb 用于读二进制文件函数 ， $readmemh 用于读十六进制文件函数

使用格式为：

```
$readmemb("<数据文件名>", <寄存器变量名>);
$readmemh("<数据文件名>", <寄存器变量名>);
```

```
//------------------------------------------------
01 `timescale 1ns/1ns
02
03 module tb_test();
04
05 integer i;
06
07 reg [7:0] a [23:0];
08
09 initial begin
10    $readmemb("xiangliangzi.txt", a);
11    for(i=0; i<=23; i=i+1) begin
12       #10;
13       $write("%s", a[i]);
14    end
15 end
16
17 endmodule
//------------------------------------------------
```

读取的 txt 文件为

```
//------------------------------------------------
01010111 // W
01100101 // e
01101100 // l
01100011 // c
01101111 // o
01101101 // m
01100101 // e
00100000 //空格
01110100 // t
01101111 // o
00100000 //空格
01111000 // x
01101001 // i
01100001 // a
01101110 // n
01100111 // g
01101100 // l
01101001 // i
01100001 // a
```

```
01101110 // n
01100111 // g
01111010 // z
01101001 // i
00100001 // !
//------------------------------------------------
```

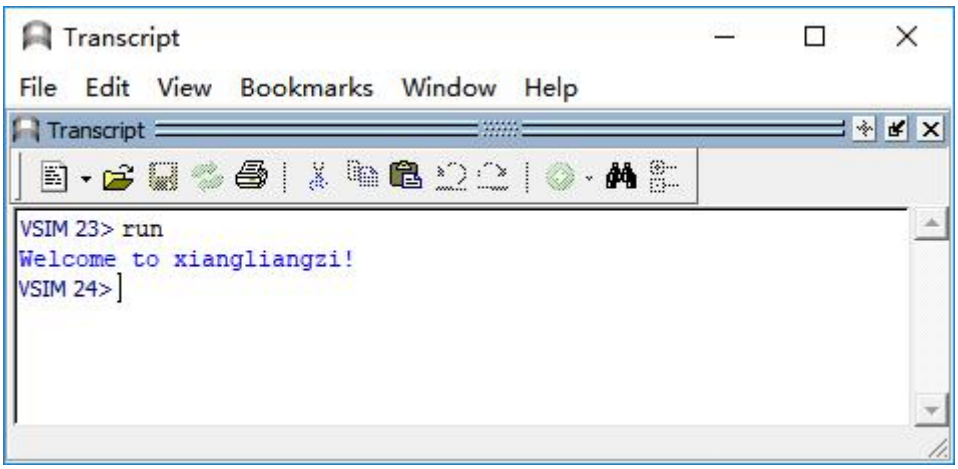

图 5-7