P.168

```
Module testHam;
    reg     [1:8]   original;
    wire    [1:8]   regenerated;
    wire    [1:8]   encoded,messedUp;
    integer             seed;

    initial begin
        seed = 1;
        forever begin
            original = $random(seed);
            #1
            $display (“original=%h,encoded=%h,messed=%h,regen=%h”,
                        original,encoded,messedUp,regenerated);
        end
    end

    hamEncode   hIn (original,encoded);
    hamDecode   hOut (messedUp,regenerated);

    assign messedUp = encoded ^ 12’b 0000_0010_0000;
endmodule
```

P.185

6.5.3 Specifying Time Units

`timescale <time_unit> / <time_precision>

Table 6.7 Arguments for `timescale compiler directive

Unit of Measurement	Abbreviation
seconds	s
milliseconds	ms
microseconds	us
nanoseconds	ns
picoseconds	ps
femtoseconds	fs

`timescale 10ns/1ns

Table 6.8 Time delay / precision specifications

Unit / precision	Delay specification	Time delayed	Comments
10ns / 1ns	#7	70ns	The delay is 7*time_unit,or 70ns
10ns / 1ns	#7.748	77ns	7.748 is rounded to one decimal place(due to the difference between 10ns and 1ns)and multiplied by the time_unit
10ns / 0.1ns	#7.748	77.5ns	7.748 is rounded to two decimal places and multiplied by the time_unit
10ns / 1ns	#7.5	75ns	7.5 is rounded to one decimal place and multiplied by 10
10ns / 10ns	#7.5	80ns	7.5 is rounded to the nearest integer(no decimal places)and multiplied by 10

P.333

System Tasks and Functions

F.1 Display and Write Tasks

$display(“Some text %d and maybe some more:%h.”,a,b);

h or H	display in hexadeximal
d or D	display in decimal
o or O	display in octal
b or B	display in binary
c or C	display ASCII character
v or V	display net signal strength(see Table 10.4)
m or M	display hierarchical name
s or S	display string

\n	is the new line character
\t	is the tab character
\\	is the \ character
\”	is the “ character
\ddd	is the character specified in up to 3 octal digits

```
$display(“Hello world\n”);
```

F.2 Continuous Monitoring

```
$monitor(parameters as used in the $display task);
$monitor($time,,”regA = ”,regA);
```

F.4 File Output

```
$fdisplay(descriptor,parameters as in the display command);
$fwrite(descriptor,parameters as in the write command);
$fmonitor(descriptor,parameters as in the monitor command);
$fstrobe(descriptor,parameters as in the strobe command);
```

The descriptor is a 32-bit value

```
$fopen(“name of file”);
$fclose(descriptor);
```

F.5 Simulation Time

$time is a function that returns the current time as a 64-bit value.$stime will return a 32-bit value.

```
$monitor($time,,,”regA = ”,regA);
```

F.6 Stop and Finish

The $stop and $finish tasks stop simulation.They differ in that $stop returns control back to the simulator’s command interpreter,while $finish returns back to the host operating system.

```
$stop;
$stop(n);
$finish;
$finish(n);
```

Parameter Value	Diagnostics
0	Prints nothing
1	Gives simulation time and location
2	Same as 1,plus a few lines of run statistics

F.7 Random

```
Parameter SEED = 33;
Reg [31:0] vector;

Always @(posedge clock)
        Vector = $random(SEED);
```

F.8 Reading Data From Disk Files

The $readmemb and $readmemh system tasks are used to load information stored in disk files into Verilog memories.

```
$readmemx(“filename”,<memname>,<<start_addr> <,<finish_addr>>?>?);
```