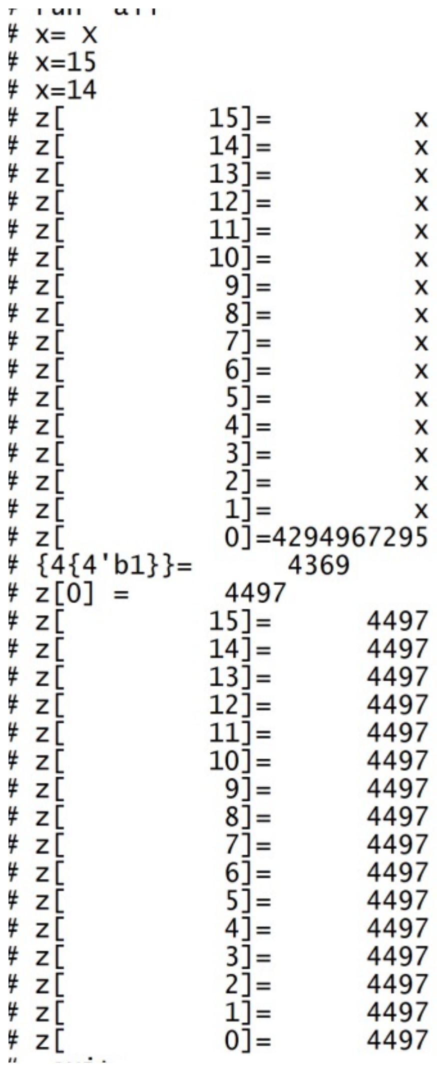
System Verilog

Exercise 1: What are the packed and unpacked dimensions of each declaration?

	packed	unpacked
`logic [3:0] x ;`	4	— (1)
`logic signed [15:0] y ;`	16	— (1)
`logic [3:0] [7:0] z [15:0] ;`	4, 8	16

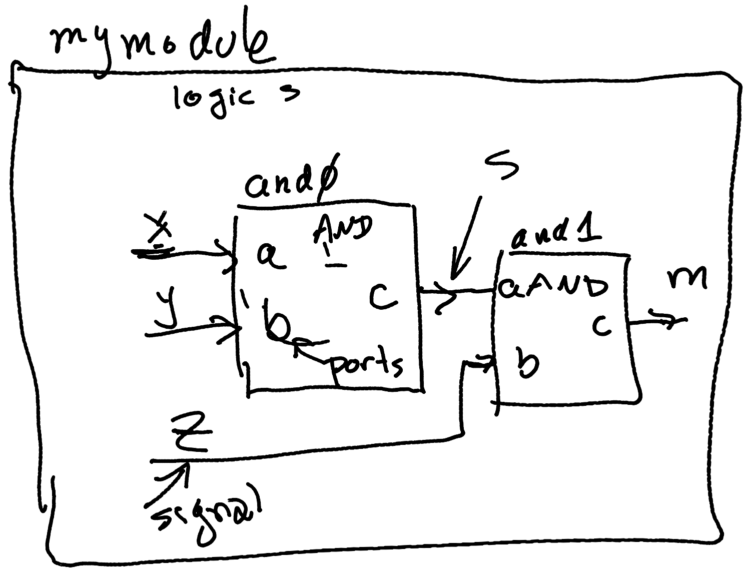
Exercise 2: What are the signedness, size and value of each constant and each expression above?

```
// what are the values of x?
x = 4'b01xz ;
x = -1 + 0 ;
y = -1 + 4'shf ;
x = y ;

// what are the values of z?
z[0] = '1 ;
z[0] = {4{4'b1}} ;
z[0][0][7] = 1 ;
z[15:0] = '{16{z[0]}} ;
```



```
# x= X
# x=15
# x=14
# z[          15]=          x
# z[          14]=          x
# z[          13]=          x
# z[          12]=          x
# z[          11]=          x
# z[          10]=          x
# z[           9]=          x
# z[           8]=          x
# z[           7]=          x
# z[           6]=          x
# z[           5]=          x
# z[           4]=          x
# z[           3]=          x
# z[           2]=          x
# z[           1]=          x
# z[           0]=4294967295
# {4{4'b1}}=       4369
# z[0] =         4497
# z[          15]=       4497
# z[          14]=       4497
# z[          13]=       4497
# z[          12]=       4497
# z[          11]=       4497
# z[          10]=       4497
# z[           9]=       4497
# z[           8]=       4497
# z[           7]=       4497
# z[           6]=       4497
# z[           5]=       4497
# z[           4]=       4497
# z[           3]=       4497
# z[           2]=       4497
# z[           1]=       4497
# z[           0]=       4497
```

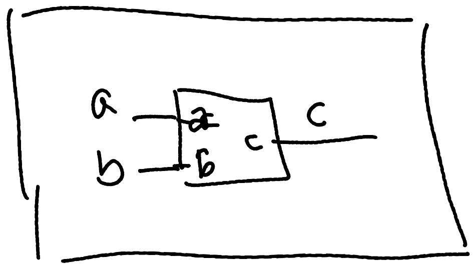
```
logic [3:0] x ;
logic signed [15:0] y ;
logic [3:0] [7:0] z [15:0] ;

// what are the values of x?

// 4-bits, some undefined some 'z':
x = 4'b01xz ;
$display("x= %b %d",x,x) ; // prints: # x= 01xz  X

// signed 32-bit (-1) assigned (truncated) to unsigned 4-bit:
x = -1 + 0 ;
$display("x=",x) ; // prints: # x=15

// signed 4-bit (4'shf) is sign-extended (not zero-padded) to 16
// bits before assignment: 1111_1111_1111_1111 +
// [1111_1111_1111_]1111 = 1111_1111_1111_1110
y = -1 + 4'shf ;
$display("y=",y) ; // prints: # y=    -2

// signed 16 bit is assigned (truncated) to 4 bits
x = y ;
$display("x=",x) ; // prints: # x=14
```



and and0 (x, y, s);

module and (int a, b;
ot. c)

and and0 (.a(x), .b(y), .c(s));



and and0 (.*)

Exercise 3: Should each of the following nets (or variables) be declared `wire` or `reg`?

```
module test (a,b,c,d,q) ;
dff d0 (clk,d,q) ; // assume only q is an output
assign d = a & b ;
always@* clk = a & c ;
endmodule
```

```
module dff ( input wire d, clk, output reg q ) ;
   always@(posedge clk)
     q = d ;
endmodule

module test ;

   wire a, b, c, d ;
   reg clk ;

   dff d0 (clk,d,q) ;

   assign d = a & b ;

   always@* begin
     clk = a & c ;
   end

endmodule
```