User Defined Primitive

```
primitive UDP1357(Z,W,X,Y);
output Z;
input W,X,Y;
// Truth table for Z=f(W,X,Y)=m(1,3,5,7)
table
//  W     X    Y  :   Z
    0     0    0  :   0;
    0     0    1  :   1;
    0     1    0  :   0;
    0     1    1  :   1;
    1     0    0  :   0;
```

```
    1     0    1  :   1;
    1     1    0  :   0;
    1     1    1  :   1;
endtable
endprimitive
```

//instantiate primitive

```
module UseUDP1357( a,b,c,d,e,f);
input a,b,c,d;
output e,f;
UDP1357(e,a,b,c);
and(f,e,d);
endmodule
```

Defining the constants for the seven-segment decoder.

```
// define segment codes
// seven bit code - one bit per segment, segment is
// bit is high. Bits 6543210 correspond to:
//       6666
//      1   5
//      1   5
//      0000
//      2   4
//      2   4
//       3333
```

```
////-----------------------------------------------
 'define SS_0 7'b1111110
 'define SS_1 7'b0110000
 'define SS_2 7'b1101101
 'define SS_3 7'b1111001
 'define SS_4 7'b0110011
 'define SS_5 7'b1011011
 'define SS_6 7'b1011111
 'define SS_7 7'b1110000
 'define SS_8 7'b1111111
 'define SS_9 7'b1111011
```

```
//-------------------------------------------------
// sseg - converts a 4-bit binary number to
// seven segment code//
// bin - 4-bit binary input
// segs - 7-bit output, defined above
//-------------------------------------------------
```

```
module sseg(bin, segs) ;
input [3:0] bin ;
 // four-bit binary input
 output [6:0] segs ;
  // seven segments
  reg [6:0] segs ;
  always@(bin)
   begin case(bin)
   0: segs = SS_0 ;
   1: segs = SS_1  ;
   2: segs = SS_2 ;
   3: segs = SS_3 ;
   4: segs = SS_4 ;
```

```
5: segs = SS_5 ;
6: segs = SS_6 ;
7: segs = SS_7 ;
8: segs = SS_8 ;
9: segs = SS_9 ;
default: segs = 7b0000000 ;
endcase
end
endmodule
```