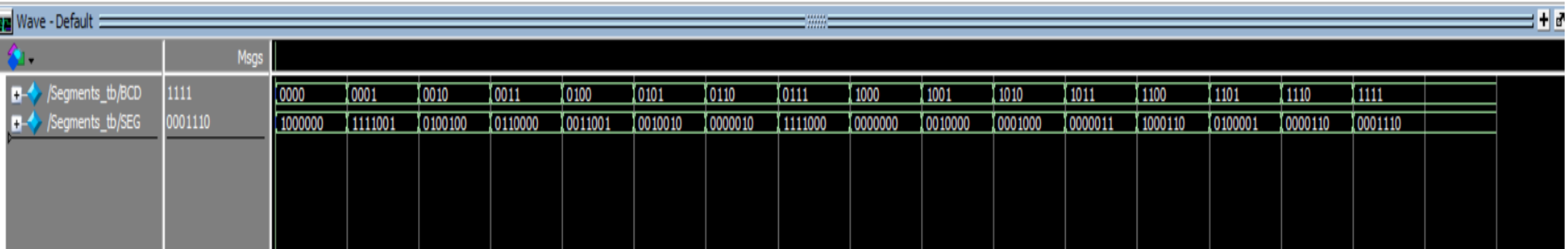
```
# BCD    SEG
# 0000  1000000
# 0001  1111001
# 0010  0100100
# 0011  0110000
# 0100  0011001
# 0101  0010010
# 0110  0000010
# 0111  1111000
# 1000  0000000
# 1001  0010000
# 1010  0001000
# 1011  0000011
# 1100  1000110
# 1101  0100001
# 1110  0000110
# 1111  0001110
```

SSD Testing

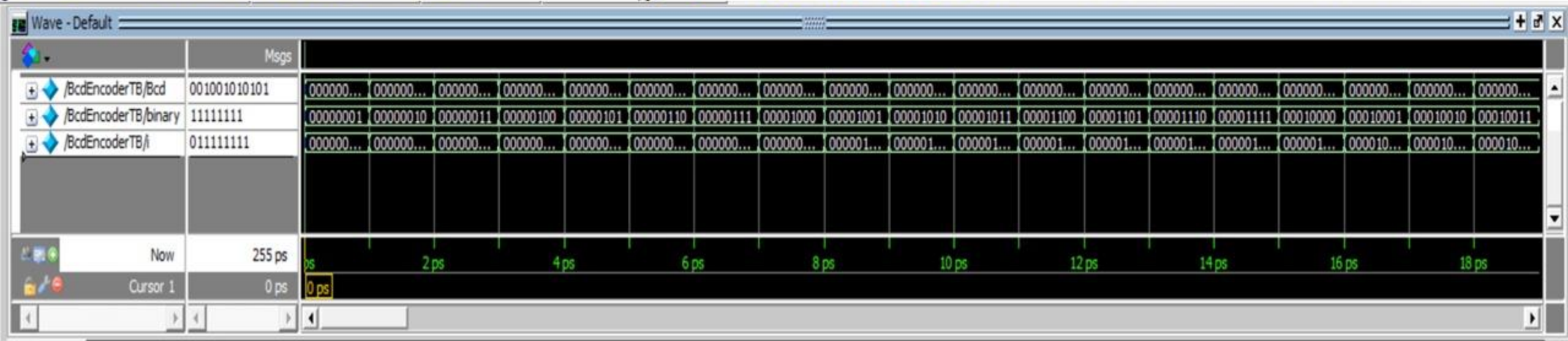
in[3..0]	out[6:0]	Digit
0000	1000000	0
0001	1111001	1
0010	0100100	2
0011	0110000	3
0100	0011001	4
0101	0010010	5
0110	0000010	6
0111	1111000	7

in[3..0]	out[6:0]	Digit
1000	0000000	8
1001	0010000	9
1010	0001000	A
1011	0000011	b
1100	1000110	C
1101	0100001	d
1110	0000110	E
1111	0001110	F



Input	Output in BCD	BCD digit to decimal

```
#   1    0000 0000 0001   0 0 1
#   2    0000 0000 0010   0 0 2
#   3    0000 0000 0011   0 0 3
#   4    0000 0000 0100   0 0 4
#   5    0000 0000 0101   0 0 5
#   6    0000 0000 0110   0 0 6
#   7    0000 0000 0111   0 0 7
#   8    0000 0000 1000   0 0 8
#   9    0000 0000 1001   0 0 9
#  10    0000 0001 0000   0 1 0
#  11    0000 0001 0001   0 1 1
#  12    0000 0001 0010   0 1 2
#  13    0000 0001 0011   0 1 3
#  14    0000 0001 0100   0 1 4
#  15    0000 0001 0101   0 1 5
#  16    0000 0001 0110   0 1 6
#  17    0000 0001 0111   0 1 7
#  18    0000 0001 1000   0 1 8
#  19    0000 0001 1001   0 1 9
#  20    0000 0010 0000   0 2 0
#  21    0000 0010 0001   0 2 1
#  22    0000 0010 0010   0 2 2
#  23    0000 0010 0011   0 2 3
#  24    0000 0010 0100   0 2 4
#  25    0000 0010 0101   0 2 5
#  26    0000 0010 0110   0 2 6
#  27    0000 0010 0111   0 2 7
#  28    0000 0010 1000   0 2 8
#  29    0000 0010 1001   0 2 9
#  30    0000 0011 0000   0 3 0
#  31    0000 0011 0001   0 3 1
#  32    0000 0011 0010   0 3 2
#  33    0000 0011 0011   0 3 3
#  34    0000 0011 0100   0 3 4
#  35    0000 0011 0101   0 3 5
#  36    0000 0011 0110   0 3 6
#  37    0000 0011 0111   0 3 7
# 230    0010 0011 0000   2 3 0
# 231    0010 0011 0001   2 3 1
# 232    0010 0011 0010   2 3 2
# 233    0010 0011 0011   2 3 3
# 234    0010 0011 0100   2 3 4
# 235    0010 0011 0101   2 3 5
# 236    0010 0011 0110   2 3 6
# 237    0010 0011 0111   2 3 7
# 238    0010 0011 1000   2 3 8
# 239    0010 0011 1001   2 3 9
# 240    0010 0100 0000   2 4 0
# 241    0010 0100 0001   2 4 1
# 242    0010 0100 0010   2 4 2
# 243    0010 0100 0011   2 4 3
# 244    0010 0100 0100   2 4 4
# 245    0010 0100 0101   2 4 5
# 246    0010 0100 0110   2 4 6
# 247    0010 0100 0111   2 4 7
# 248    0010 0100 1000   2 4 8
# 249    0010 0100 1001   2 4 9
# 250    0010 0101 0000   2 5 0
# 251    0010 0101 0001   2 5 1
# 252    0010 0101 0010   2 5 2
# 253    0010 0101 0011   2 5 3
# 254    0010 0101 0100   2 5 4
# 255    0010 0101 0101   2 5 5
```



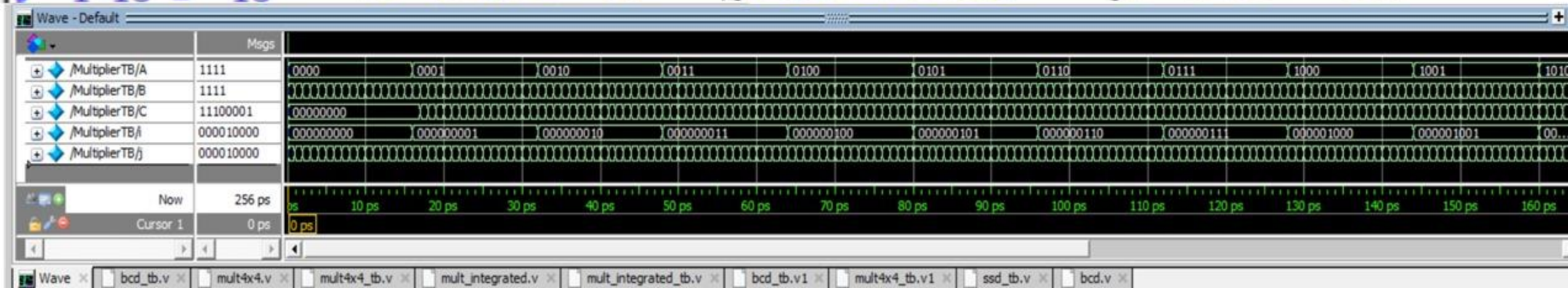
Input1 * input2 = output

```
#  0* 0 =  0
#  0* 1 =  0
#  0* 2 =  0
#  0* 3 =  0
#  0* 4 =  0
#  0* 5 =  0
#  0* 6 =  0
#  0* 7 =  0
#  0* 8 =  0
#  0* 9 =  0
#  0*10 =  0
#  0*11 =  0
#  0*12 =  0
#  0*13 =  0
#  0*14 =  0
#  0*15 =  0
#  1* 0 =  0
#  1* 1 =  1
#  1* 2 =  2
#  1* 3 =  3
#  1* 4 =  4
#  1* 5 =  5
#  1* 6 =  6
#  1* 7 =  7
#  1* 8 =  8
#  1* 9 =  9
#  1*10 = 10
#  1*11 = 11
#  1*12 = 12
#  1*13 = 13
#  1*14 = 14
#  1*15 = 15
```

```
#  5* 0 =  0
#  5* 1 =  5
#  5* 2 = 10
#  5* 3 = 15
#  5* 4 = 20
#  5* 5 = 25
#  5* 6 = 30
#  5* 7 = 35
#  5* 8 = 40
#  5* 9 = 45
#  5*10 = 50
#  5*11 = 55
#  5*12 = 60
#  5*13 = 65
#  5*14 = 70
#  5*15 = 75
#  6* 0 =  0
#  6* 1 =  6
#  6* 2 = 12
#  6* 3 = 18
#  6* 4 = 24
#  6* 5 = 30
#  6* 6 = 36
#  6* 7 = 42
#  6* 8 = 48
#  6* 9 = 54
#  6*10 = 60
#  6*11 = 66
#  6*12 = 72
#  6*13 = 78
#  6*14 = 84
#  6*15 = 90
```

```
# 14* 0 =  0
# 14* 1 = 14
# 14* 2 = 28
# 14* 3 = 42
# 14* 4 = 56
# 14* 5 = 70
# 14* 6 = 84
# 14* 7 = 98
# 14* 8 = 112
# 14* 9 = 126
# 14*10 = 140
# 14*11 = 154
# 14*12 = 168
# 14*13 = 182
# 14*14 = 196
# 14*15 = 210
# 15* 0 =  0
# 15* 1 = 15
# 15* 2 = 30
# 15* 3 = 45
# 15* 4 = 60
# 15* 5 = 75
# 15* 6 = 90
# 15* 7 = 105
# 15* 8 = 120
# 15* 9 = 135
# 15*10 = 150
# 15*11 = 165
# 15*12 = 180
# 15*13 = 195
# 15*14 = 210
# 15*15 = 225
```

```
#  7* 0 =  0
#  7* 1 =  7
#  7* 2 = 14
#  7* 3 = 21
#  7* 4 = 28
#  7* 5 = 35
#  7* 6 = 42
#  7* 7 = 49
#  7* 8 = 56
#  7* 9 = 63
#  7*10 = 70
#  7*11 = 77
#  7*12 = 84
#  7*13 = 91
#  7*14 = 98
#  7*15 = 105
#  8* 0 =  0
#  8* 1 =  8
#  8* 2 = 16
#  8* 3 = 24
#  8* 4 = 32
#  8* 5 = 40
#  8* 6 = 48
#  8* 7 = 56
#  8* 8 = 64
#  8* 9 = 72
#  8*10 = 80
#  8*11 = 88
#  8*12 = 96
#  8*13 = 104
#  8*14 = 112
#  8*15 = 120
```



Testing The Whole Integrated Circuit

```
#  A1    A2       Seg1     Seg2     Seg3
#  0000  0000  1000000  1000000  1000000
#  1111  0001  1000000  1111001  0010010
#  1111  1111  0100100  0100100  0010010
#  1001  1000  1000000  1111000  0100100
#  0110  1001  1000000  0010010  0011001
#  0100  0111  1000000  0100100  0000000
#  0111  0111  1000000  0011001  0010000
```

0 * 0 = 0

15 * 1 = 15

15 * 15 = 225

9 * 8 = 72

6 * 9 = 54

4 * 7 = 28

7 * 7 = 49

in[3..0]	out[6:0]	Digit
0000	1000000	0
0001	1111001	1
0010	0100100	2
0011	0110000	3
0100	0011001	4
0101	0010010	5
0110	0000010	6
0111	1111000	7

in[3..0]	out[6:0]	Digit
1000	0000000	8
1001	0010000	9
1010	0001000	A
1011	0000011	b
1100	1000110	C
1101	0100001	d
1110	0000110	E
1111	0001110	F