CSE 370 Spring 2000
Solutions for Homework 5
05/11/2000

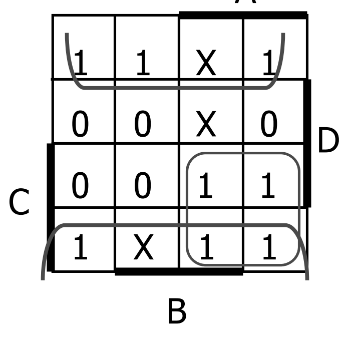
Problem 1

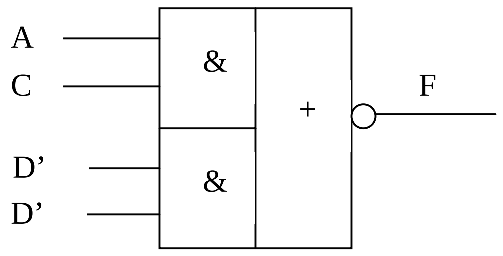
F(A, B, C, D) = Σm(1,3,5,7,9) + Σd(6,12,13)

A	B	C	D	F
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	X
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	0
1	0	1	1	0
1	1	0	0	X
1	1	0	1	X
1	1	1	0	0
1	1	1	1	0

What we actually want to do is get a sum-of-products expression for the complement of F, because of the fact that the AND/OR/Invert block actually has an inverter at its output. Thus, here is the K-map for F':

F' = AC + D'





Problem 2

```
F     = AD + AE + BD + BE + CD + CE + AF
      = A(D + E) + B(D + E) + C(D + E) + AF
      = (A + B + C)(D + E) + AF
```

Problem 3

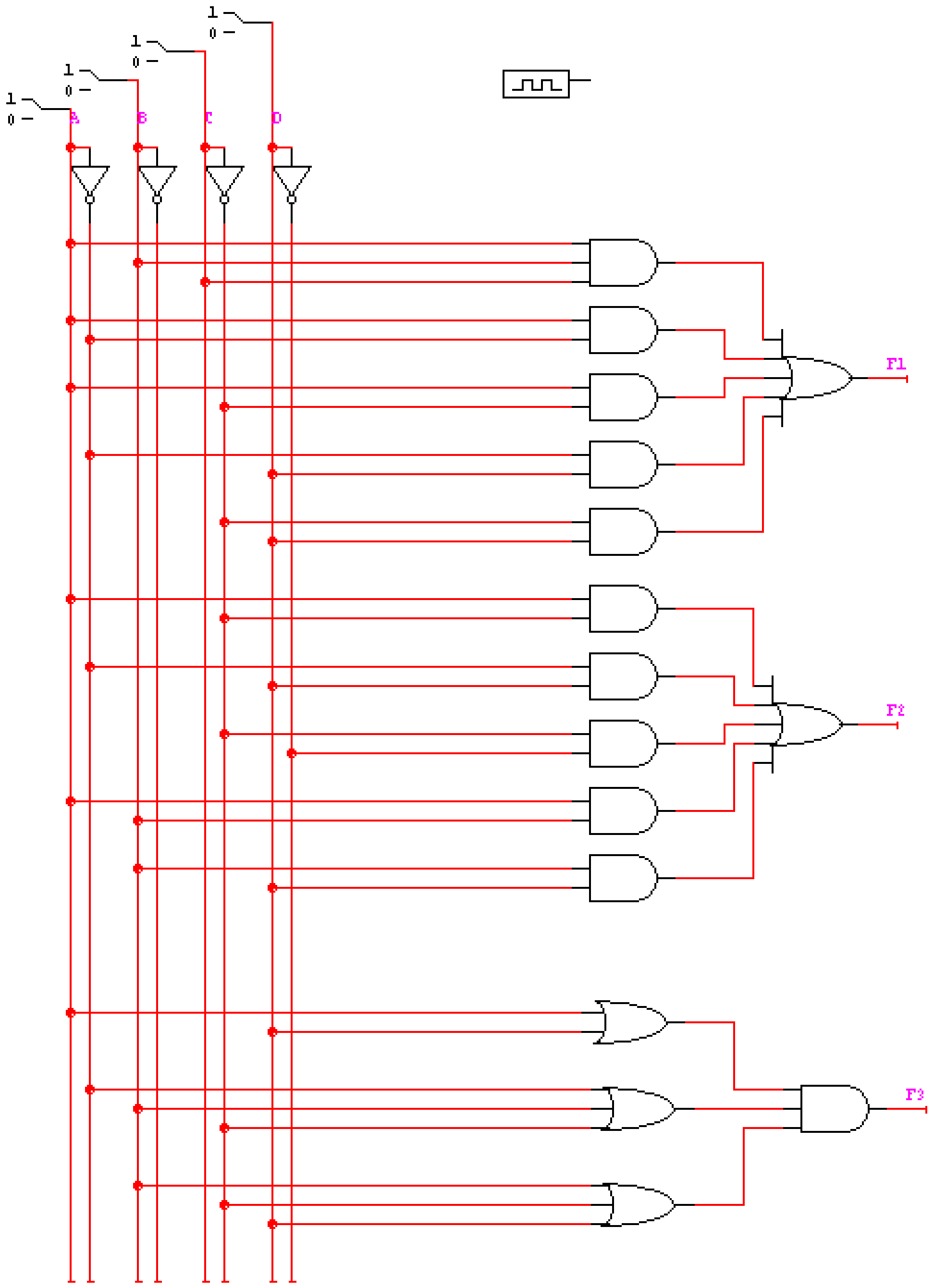


Figure 3.36

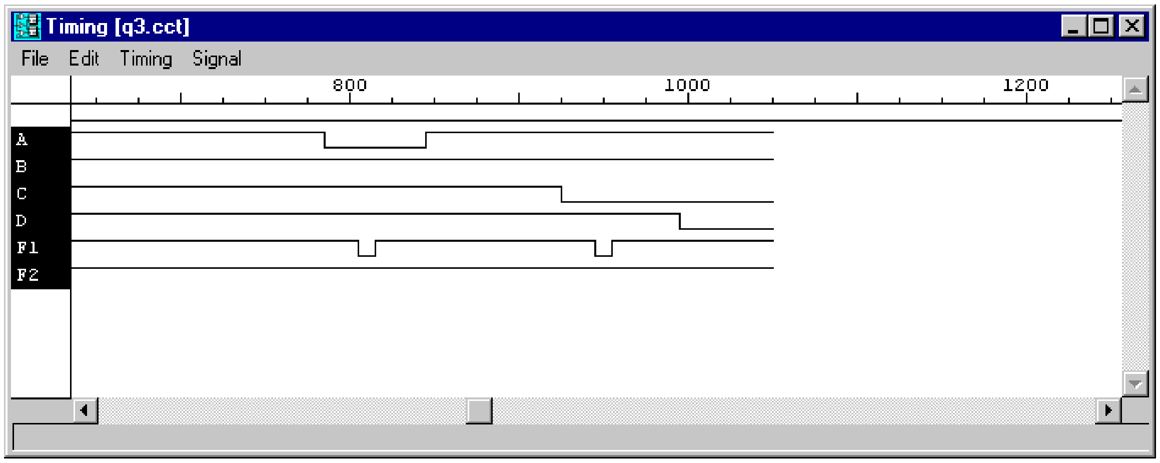
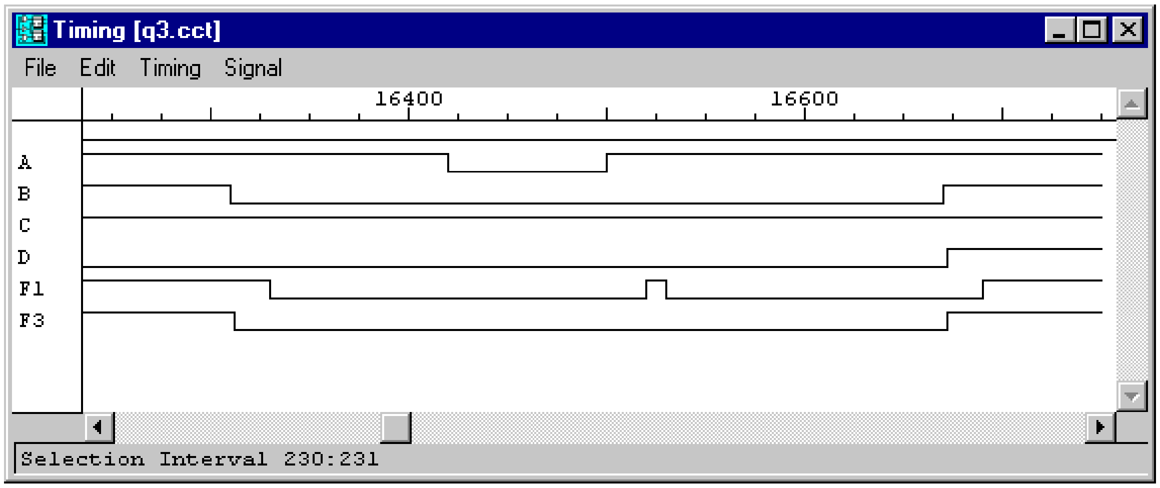


Figure 3.38

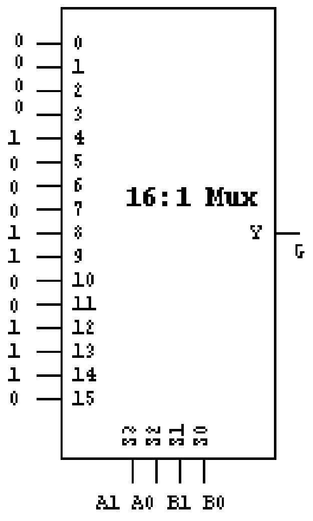


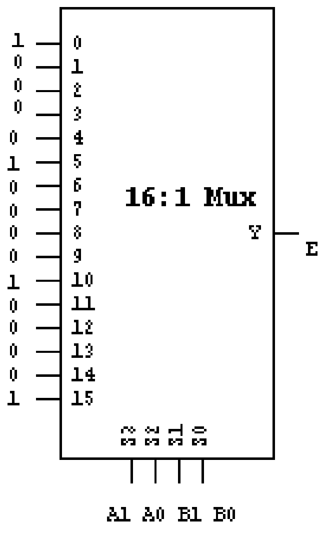
Selection Interval 230:231

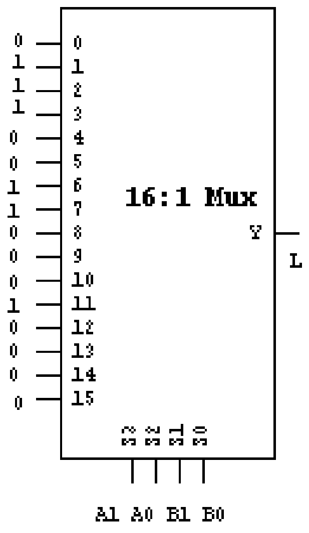
Problem 4

Using 16:1 Muxes

A1	A0	B1	B0	G	E	L
0	0	0	0	0	1	0
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	0	1
0	1	0	0	1	0	0
0	1	0	1	0	1	0
0	1	1	0	0	0	1
0	1	1	1	0	0	1
1	0	0	0	1	0	0
1	0	0	1	1	0	0
1	0	1	0	0	1	0
1	0	1	1	0	0	1
1	1	0	0	1	0	0
1	1	0	1	1	0	0
1	1	1	0	1	0	0
1	1	1	1	0	1	0

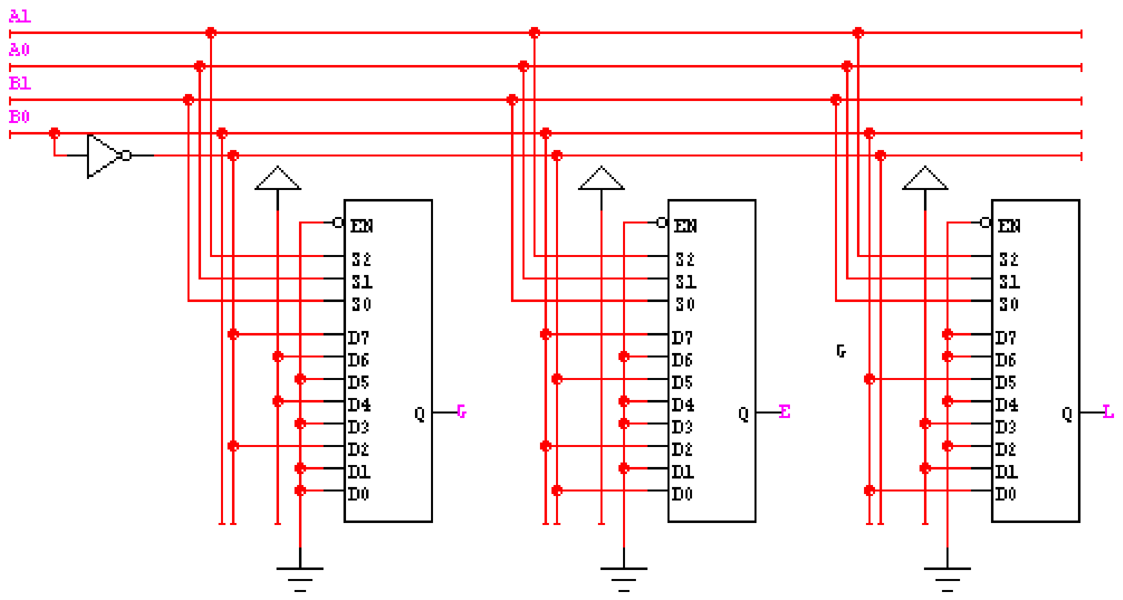






Using 8:1 Muxes

A1	A0	B1	G	E	L
0	0	0	0	B0’	B0
0	0	1	0	0	1
0	1	0	B0’	B0	0
0	1	1	0	0	1
1	0	0	1	0	0
1	0	1	0	B0’	B0
1	1	0	1	0	0
1	1	1	B0’	B0	0

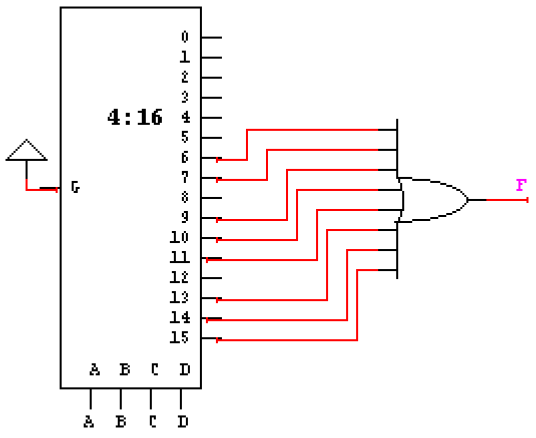


Problem 5

First, we draw a truth table:

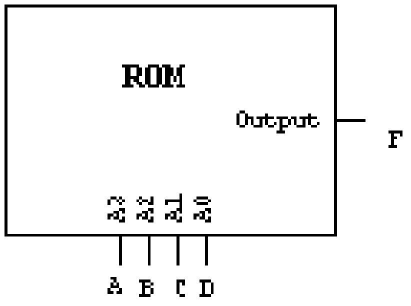
A	B	C	D	F
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

b.



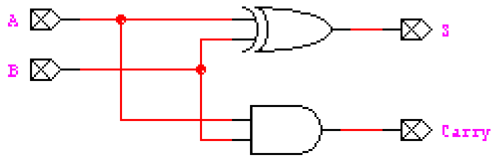
c.
The idea of a ROM is that it maps an n-bit address onto an m-bit word. We call this type of ROM a 2^n words by m bits ROM. In our case, n = 4, and m only needs to be 1, so that we will have a 16 words by 1 bit ROM. This means that our ROM will contains16 words,

and each word will be 1 bit wide. The values inside the ROM are actually given exactly by the truth table of the function F.

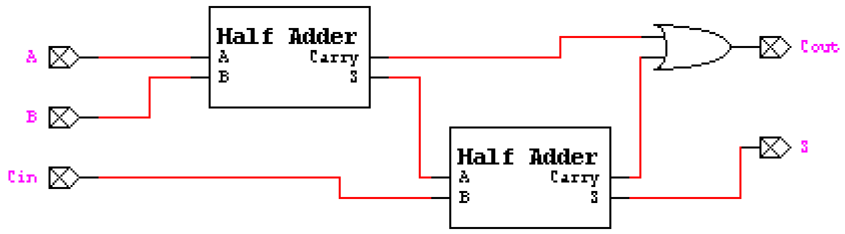


Problem 7

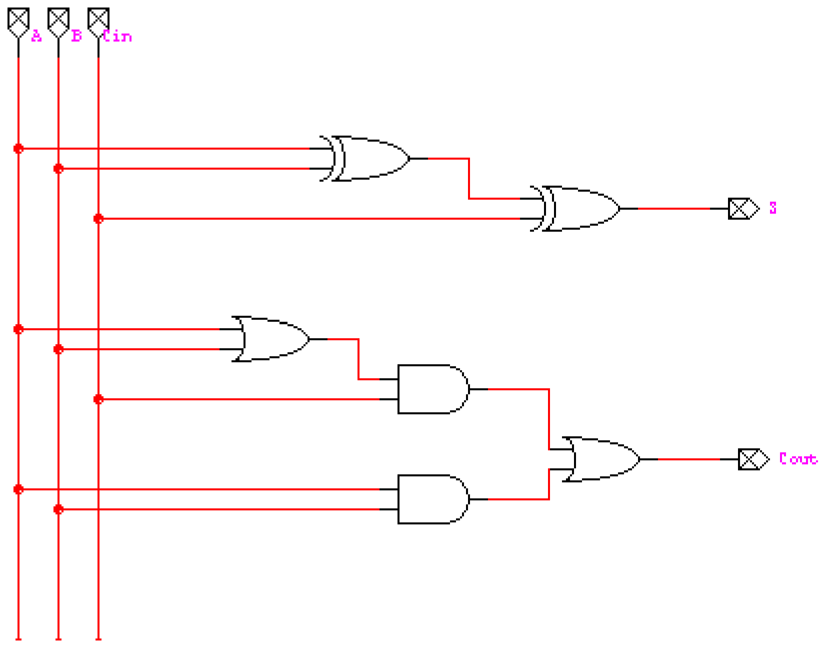
Half adder



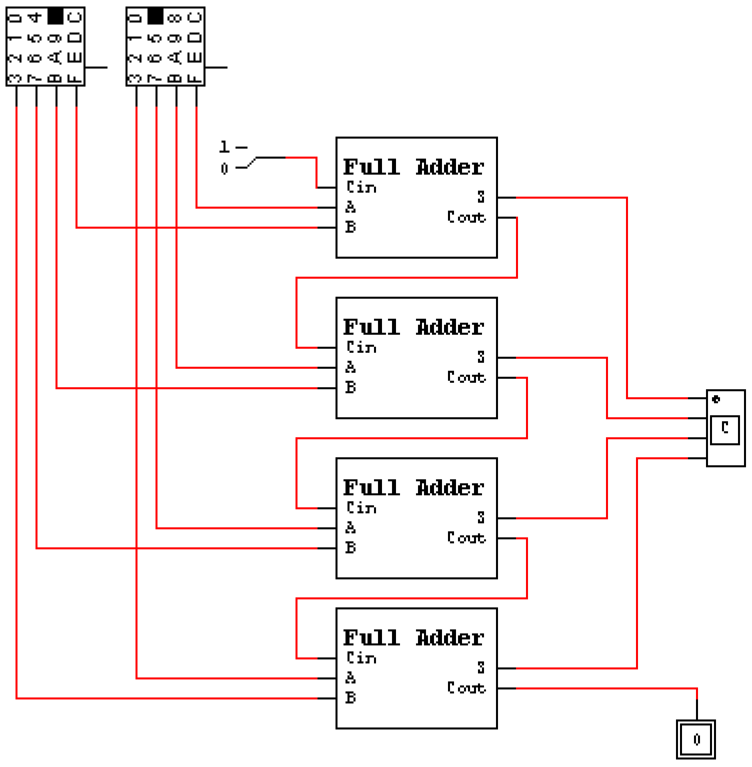
Full Adder Using sub-blocks



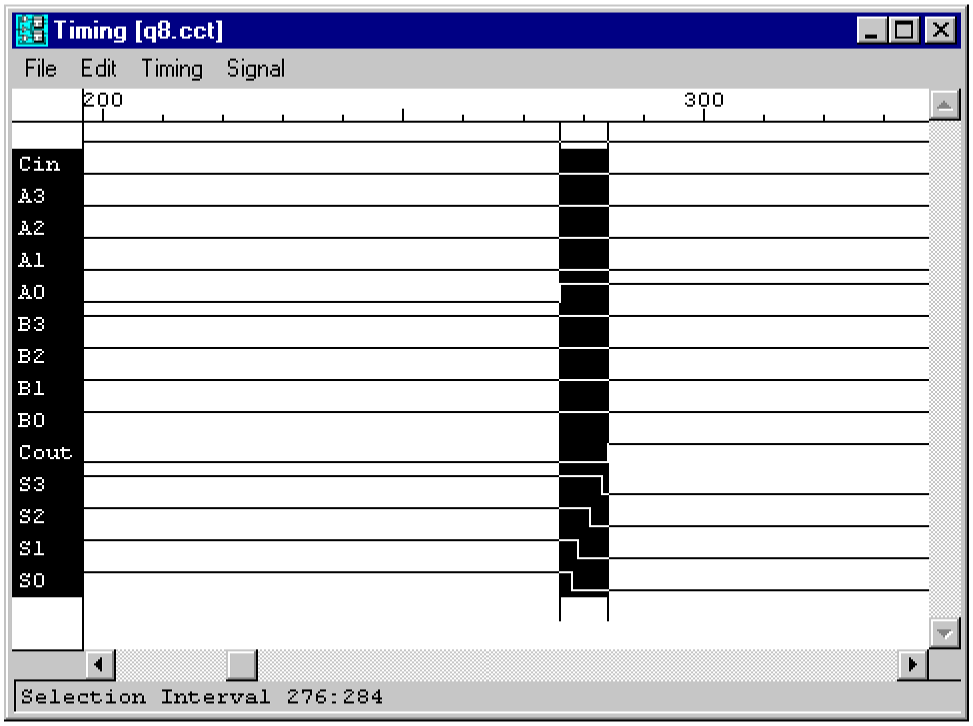
Full Adder without sub-blocks



Problem 8



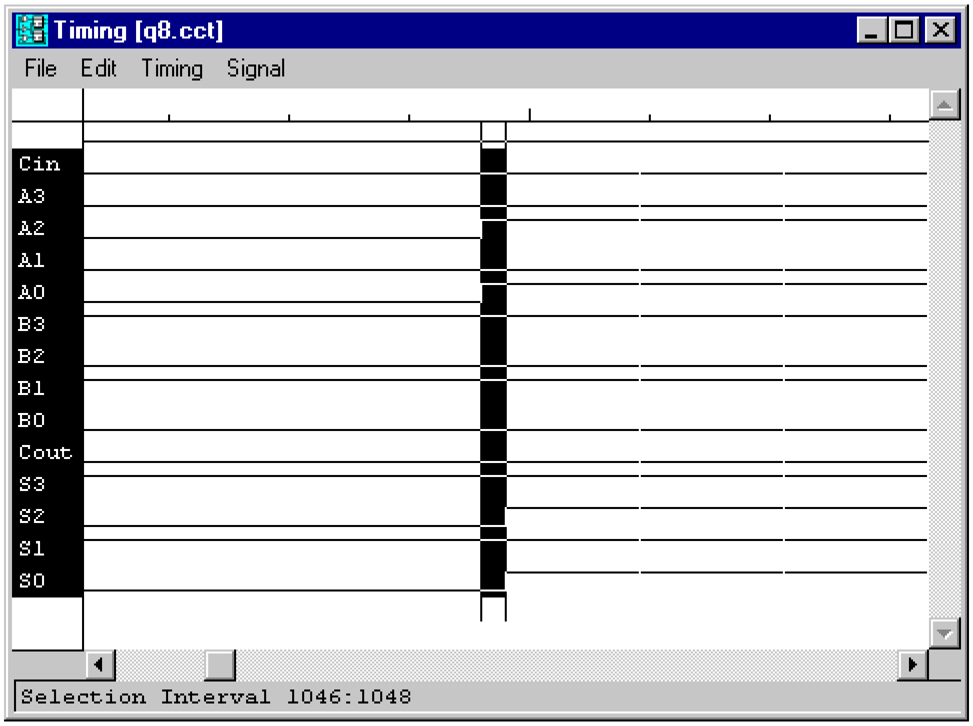
First Test case: 1111 + 0000 changed to 1111 + 0001



Selection Interval 276:284

From the selection interval, we can see that the propagation delay is 8. This is consistent with the theoretical expectation. When we change the 0000 to 0001, the result will be 0000 with a carry out of 1. The carry out of 1 will have to propagate all the way from the least significant bit that was changed through each full adder. Since the full adders are implemented using two-level logic, each full adder will have a propagation delay of 2. Since there are 4 full adders to propagate through, the total delay should be 8.

Test case 2: 1010 + 0000 changed to 1010 + 0101



```
Selection Interval 1046:1048
```

From the selection interval above, we can see that the delay is only 2, which is much smaller than the previous the case. This again corresponds to the theoretical expectation. When 1010 is added with 0101 there are no carries that need to be propagated: only the sum bits need to propagate to the output. This only takes the propagation delay of a full adder, which is 2.