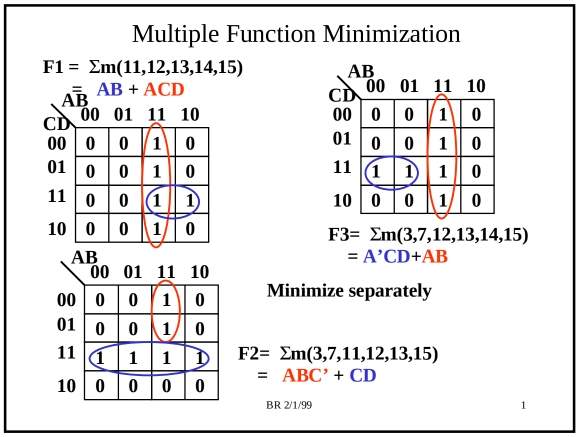


Multiple Function Minimization

F1 = Σm(11,12,13,14,15)
= AB + ACD

AB CD	00	01	11	10
00	0	0	1	0
01	0	0	1	0
11	0	0	1	1
10	0	0	1	0

AB CD	00	01	11	10
00	0	0	1	0
01	0	0	1	0
11	1	1	1	0
10	0	0	1	0

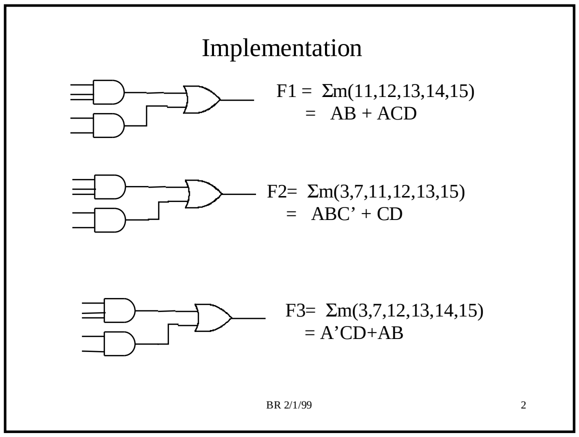
F3= Σm(3,7,12,13,14,15)
= A’CD+AB

AB	00	01	11	10
00	0	0	1	0
01	0	0	1	0
11	1	1	1	1
10	0	0	0	0

Minimize separately

F2= Σm(3,7,11,12,13,15)
= ABC’ + CD

BR 2/1/99 1



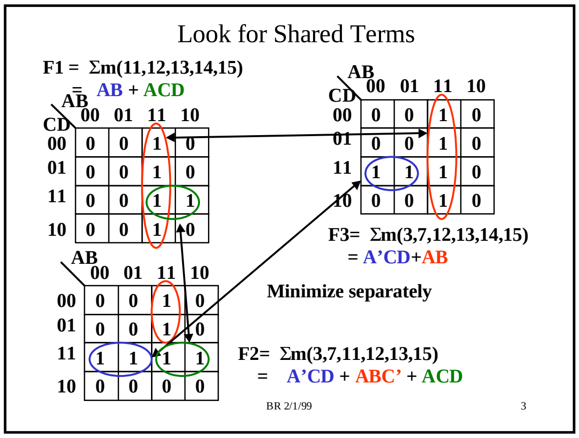
Implementation

F1 = Σm(11,12,13,14,15)
= AB + ACD

F2= Σm(3,7,11,12,13,15)
= ABC’ + CD

F3= Σm(3,7,12,13,14,15)
= A’CD+AB

BR 2/1/99 2



Look for Shared Terms

F1 = Σm(11,12,13,14,15)
= AB + ACD

AB CD	00	01	11	10
00	0	0	1	0
01	0	0	1	0
11	0	0	1	1
10	0	0	1	0

AB CD	00	01	11	10
00	0	0	1	0
01	0	0	1	0
11	1	1	1	0
10	0	0	1	0

F3= Σm(3,7,12,13,14,15)
= A’CD+AB

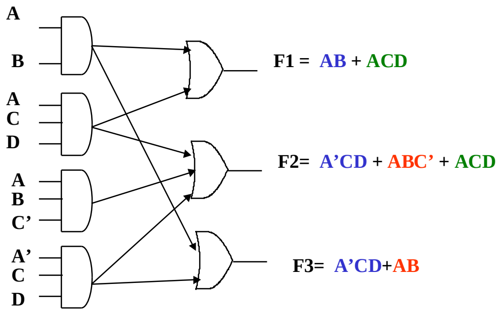
AB	00	01	11	10
00	0	0	1	0
01	0	0	1	0
11	1	1	1	1
10	0	0	0	0

Minimize separately

F2= Σm(3,7,11,12,13,15)
= A’CD + ABC’ + ACD

BR 2/1/99 3

Implementation with Shared Terms



F1 = AB + ACD

F2= A'CD + ABC' + ACD

F3= A'CD+AB

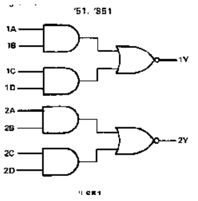
Lab 6: SSN Decoder

- Create a four output combinational block that will recognize fields in your SSN
  - SSN is three fields: XXX - YY - ZZZZ
  - F1 = 1 if input is equal to one of the numbers in the first group (XXX)
  - F2 = 1 if input is equal to one of the numbers in the 2nd group (YY)
  - F3 = 1 if input is equal to one of the numbers in the 3rd group (ZZZZ)
  - F4 = 1 if input is equal to numbers in any group

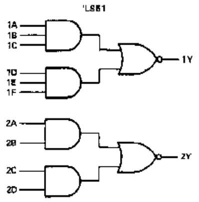
Constraints

- Can only use 1 each of the following devices
  - 7400 (4 two-input NANDs)
  - 7402 (4 two-input NORs)
  - 7404 (6 Inverters)
  - 7408 (4 two-input AND)
  - 7410 (3 three-input NAND)
  - 7432 (4 two-input OR)
  - 7451 (AND-OR-INVERT function)
  - 7486 (4 two-input XOR gates)

7451, 74LS51



7451



74LS51

Lab has both 7451 and 74LS51 -- be careful which one you use.
Design to use the 7451 since it is a subset of the 74LS51

What good is a 7451?

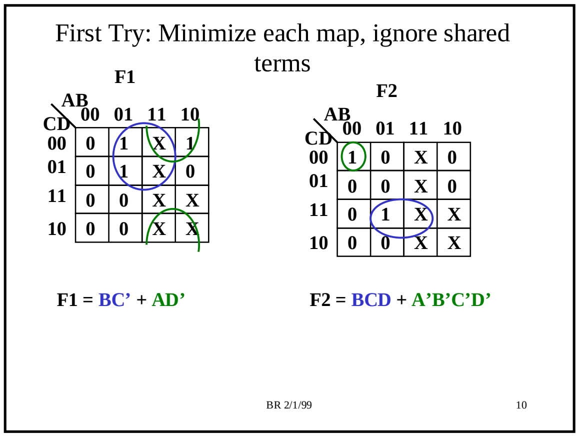
If a SOP equation has two product terms, with only two terms for each product term, then can implement with a 7451.

If minimize ZEROS on K-Map, then get an SOP form for F'. The inverter on the output of the 7451 will convert it to F!!!

OR, can minimize '1's, then put an external inverter on the output of the 7451 to get the high true version.

Example for SSN= 458 70 2198

Row	A B C D	F1	F2	F3	F4
0	**0 0 0 0**	**0**	**1**	**0**	**1**
1	**0 0 0 1**	**0**	**0**	**1**	**1**
2	**0 0 1 0**	**0**	**0**	**1**	**1**
3	**0 0 1 1**	**0**	**0**	**0**	**0**
4	**0 1 0 0**	**1**	**0**	**0**	**1**
5	**0 1 0 1**	**1**	**0**	**0**	**1**
6	**0 1 1 0**	**0**	**0**	**0**	**0**
7	**0 1 1 1**	**0**	**1**	**0**	**1**
8	**1 0 0 0**	**1**	**0**	**1**	**1**
9	**1 0 0 1**	**0**	**0**	**1**	**1**
10	**1 0 1 0**	**x**	**x**	**x**	**x**
11	**1 0 1 1**	**x**	**x**	**x**	**x**
12	**1 1 0 0**	**x**	**x**	**x**	**x**
13	**1 1 0 1**	**x**	**x**	**x**	**x**
14	**1 1 1 0**	**x**	**x**	**x**	**x**
15	**1 1 1 1**	**x**	**x**	**x**	**x**



First Try: Minimize each map, ignore shared terms

F1

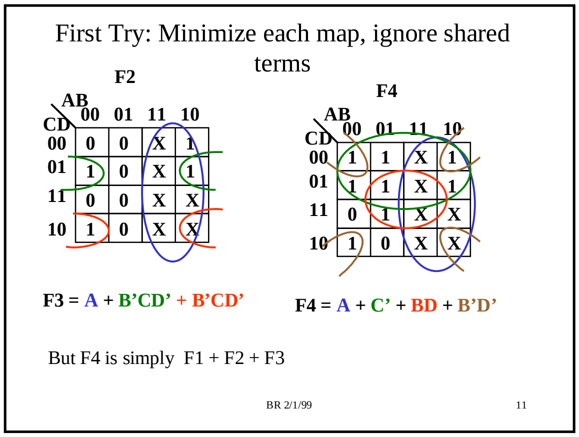
CD\AB	00	01	11	10
00	0	1	X	1
01	0	1	X	0
11	0	0	X	X
10	0	0	X	X

F2

CD\AB	00	01	11	10
00	1	0	X	0
01	0	0	X	0
11	0	1	X	X
10	0	0	X	X

F1 = BC’ + AD’

F2 = BCD + A’B’C’D’



First Try: Minimize each map, ignore shared terms

F2

CD\AB	00	01	11	10
00	0	0	X	1
01	1	0	X	1
11	0	0	X	X
10	1	0	X	X

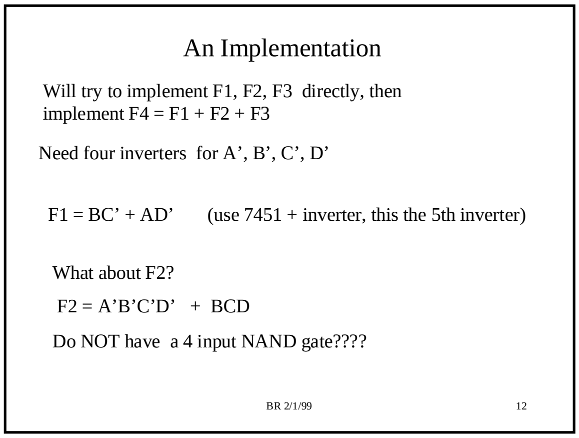
F4

CD\AB	00	01	11	10
00	1	1	X	1
01	1	1	X	1
11	0	1	X	X
10	1	0	X	X

F3 = A + B’CD’ + B’CD’

F4 = A + C’ + BD + B’D’

But F4 is simply F1 + F2 + F3



An Implementation

Will try to implement F1, F2, F3 directly, then
implement F4 = F1 + F2 + F3

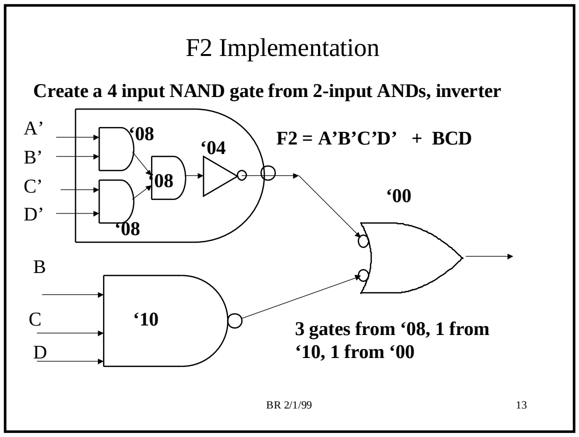
Need four inverters for A’, B’, C’, D’

F1 = BC’ + AD’ (use 7451 + inverter, this the 5th inverter)

What about F2?

F2 = A’B’C’D’ + BCD

Do NOT have a 4 input NAND gate????

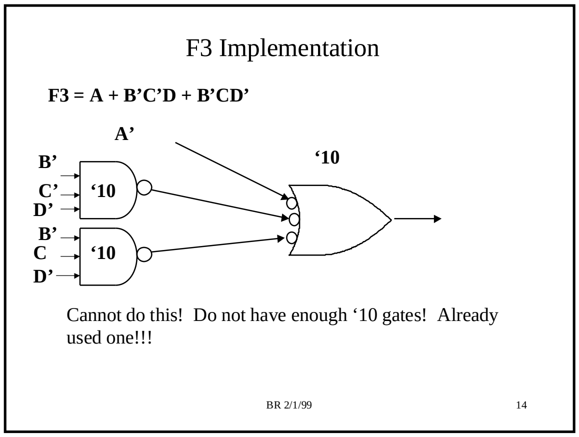


F2 Implementation

Create a 4 input NAND gate from 2-input ANDs, inverter

F2 = A’B’C’D’ + BCD

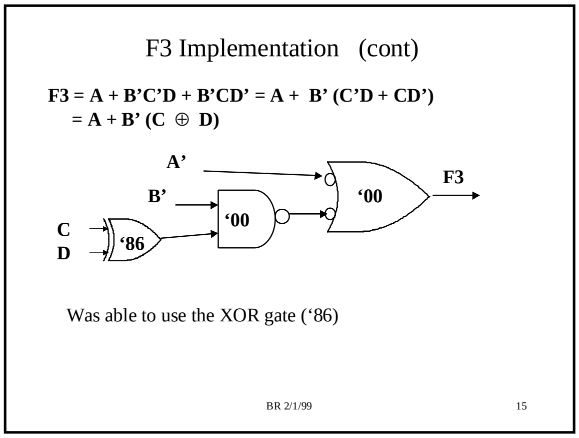
3 gates from ‘08, 1 from ‘10, 1 from ‘00



F3 Implementation

F3 = A + B’C’D + B’CD’

Cannot do this! Do not have enough ‘10 gates! Already used one!!!



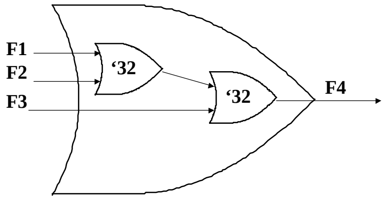
F3 Implementation (cont)

F3 = A + B’C’D + B’CD’ = A + B’ (C’D + CD’)
= A + B’ (C ⊕ D)

Was able to use the XOR gate (‘86)

F4 Implementation

F3 = F1 + F2 + F3



I used Nand-Nand implementation for SOP forms so that I could save the OR-gates in the '32 for this.

Final Gate Count

All gates from the '04 (6 inverters)

Part of the 7451

Three gates from the '08

One gate from the '10

Three gates from the '10

One gate from the '86

Two gates from the '32

There are MOST certainly other solutions……… Did not consider shared terms, did not minimize zeros. Many other avenues to try if these failed.

Debugging

- Wire up ONE function at a time
  - Start with whatever your simplest function is, then work towards hardest function.
  - No need to proceed to next function until current one works
- If F1, F2, F3 all work, then getting F4 to work will be trivial.
- The TAs do NOT know what the correct solution is for your SSN! They will only be able to offer general debugging help.