Daniel Campos

1. Simplify the following Boolean equation using only the Boolean logic laws and identities:

(X + bar(Y) + X\*bar(Y))(X\*Y + bar(X)\*Z + Y\*Z)

I used the table of identities from the discrete mathematics textbook so may be slightly different than the handout

1. (x+bar(y)+x\*bar(y))(x\*y+z\*bar(x)+z\*y) communicative
2. (x+bar(y) +x\*bar(y))(x\*y+z(bar(x)+y))distributive
3. (x+bar(y))(x\*y+z(bar(x)+y))absorption
4. (x+bar(y))(x\*y)(z(bar(x)+y) (bar(y)+x))distributive
5. (x\*x\*y)(x\*bar(y)\*y)+(z(bar(x)+y)(bar(y)+x))multiplication
6. (x\*y)(x\*0)+(z(bar(x)+y) (bar(y)+x))identity of table (y+bar(y))=0
7. (x\*y) +(z(bar(x)+y) (bar(y)+x)) value of x\*0
8. (x\*y)+(z((x\*bar(x))+(bar(x)\*bar(y))+(x\*y)+(y\*bar(y)) Foil(first outside inside last)
9. (x\*y)+(z(0+(bar(x)\*bar(y))+(x\*y)+0) identity of table (y+bar(y))=0
10. (x\*y)+z(bar(x)\*bar(y))+z(x\*y) distributive and communicative(I removed some non essential steps.
11. (x\*y)+z(bar(x)\*bar(y)) absorption

2. Simplify the Boolean equation in Problem 1 using a K-Map:

step one truth table

|  |  |  |  |
| --- | --- | --- | --- |
| X | Y | Z | F |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |

Step two K-Map Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| XY-> to right | Bar(X)Bar(Y) | Bar(X)Y | XY | Bar(Y) X |
| Bar(Z) | 0 | 0 | 1 | 0 |
| Z | 1 | 0 | 1 | 0 |

Step Three Simply the equation

Z(Bar(X)Bar(Y))+(Bar(Z)+Z)(XY)= Z(Bar(X)Bar(Y))+(XY)

3. Simplify the following Boolean equation using a K-Map:

F(A,B,C,D) = bar(A)\*bar(B)\*bar(C)\*D +

ABC +

A\*bar(B)\*bar(C)\*bar(D) +

A\*bar(B)\*bar(C)\*D +

bar(A)\*bar(B)\*bar(C)\*bar(D) +

bar(A)\*B\*C\*D +

A\*bar(B)\*C\*bar(D) +

bar(B)\*C\*D

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | D | F |
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

Step two K-Map

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Bar(A)Bar(B) | Bar(A)B | Bar(B)A | AB |
| CD | 1 | 1 | 1 | 1 |
| Bar(C )D | 1 | 0 | 1 | 0 |
| Bar(D)C | 0 | 0 | 1 | 1 |
| Bar(C )Bar(D) | 1 | 0 | 1 | 0 |

Step Three Collect the values

Bar(A)\*Bar(B)\*C\*D+Bar(A)\*B\*C\*D+A\*Bar(B)\*C\*D+A\*B\*C\*D+Bar(A)\*Bar(B)\*Bar(C)\*D+A\*Bar(B)\*Bar(C)\*D+A\*B\*C\*Bar(D)+A\*Bar(B)\*C\*Bar(D)+

Bar(A)\*Bar(B)\*Bar(C)\*Bar(D)+A\*Bar(B)\*Bar(C)\*BAR(D)+

Step Four Remove adjacent values and leave only those that matter

1. Since C\*D always produces a 1 it is important and thus added
2. Since A\*Bar(B) always produces a 1 it is essential and thus added
3. Since Bar(A)\*Bar(B)\*Bar(C) no matter what the D is always produces a 1 and thus is added.
4. Also A\*C always produce a 1 and thus is added

These are the major items that aren’t canceled out and thus create the equation present bellow

A\*Bar(B)+C\*D+A\*C+Bar(A)\*Bar(B)\*Bar(C)

4. If your friend tells you that a 4 bit block CLA adder for 128 bit inputs is the same speed in terms of number of gate delays for 256 bits inputs, would you believe your friend? Justify your belief or disbelief in your friend's statement.

I would disagree. I disagree since the more bits in a group the more complex the carry operation become and the more time is spend on slow road in each group rather than faster roads( provided by the look ahead carry logic) and since more times have to be moved since the input is bigger there will be more delay gates.