

Name \_\_\_\_\_

CSE 241 Digital Systems

Hourly Exam #2 Solutions

Nov 13, 2009

*Instructions: Write your name on the top of each sheet. Show all work in the space provided. No calculators or other electronic devices allowed. 50 min closed book.*

1. For the Boolean function  $f(w,x,y,z) = y'z' + xz' + wx'z' + w'x'z$

(b) There are 5 prime implicants, 3 of size 4 (left column  $y'z'$ , middle two in first and last columns  $xz'$ , bottom two in first and last columns  $wz'$ ) and two of size 2 (left two cells in first row  $w'x'y'$ , middle two in first row  $w'x'z$ ).

(c) Three of the prime implicants contain essential 1-cells and are thus irredundant product terms:  $xz'$ ,  $wz'$ ,  $w'x'z$ . Either of the two others can be omitted without leaving a 1-cell uncovered, but not both. So there are two irredundant sums, each containing all 3 of the irredundant product terms and one of the others.

(a) Fill in the Karnaugh Map, showing the prime implicant subcubes

		yz			
		00	01	11	10
wx	00	1	1	1	0
	01	1	0	0	1
	11	1	0	0	1
	10	1	0	0	1

(b) List all prime implicants

$y'z'$ ,  $xz'$ ,  $wz'$ ,  $w'x'y'$ ,  $w'x'z$

(c) Write all irredundant sums

There are two:  $f_1(w,x,y,z) = xz' + wz' + w'x'y' + w'x'z$   
 $f_2(w,x,y,z) = y'z' + xz' + wz' + w'x'z$

2.

$$f(v,w,x,y,z) = (v+w)(x+y)(z)$$

Use the Quine-McCluskey Method to express the Boolean function  $f(v,w,x,y,z)$  as a sum of all its prime implicants. Show all your work. Note: do not worry about redundant prime implicants. Not necessary to construct a prime implicant table or use Petrick Method. Just find the sum of all prime implicants.

The first list in QM contains all minterms of  $f$ . To be a minterm, one of the first two columns ( $v$  or  $w$ ) must be a 1, one of the next two columns ( $x$  or  $y$ ) must be a 1, and the fifth column must be a 1. Using this rule we can easily write the first list in natural order:

```
v w x y z
0 1 0 1 1
0 1 1 0 1
0 1 1 1 1
1 0 0 1 1
1 0 1 0 1
1 0 1 1 1
1 1 0 1 1
1 1 1 0 1
1 1 1 1 1
```

Now lets sort this list in index-order to make a starting list for QM and then proceed:

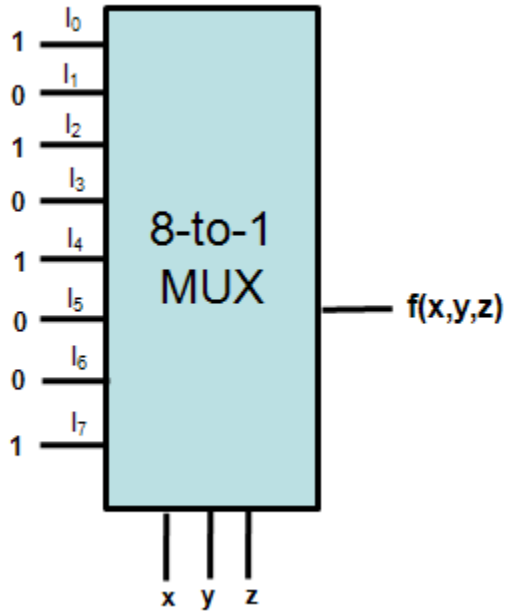
0 1 0 1 1 ✓	0 1 - 1 1 ✓	- 1 - 1 1 (wyz)
0 1 1 0 1 ✓	- 1 0 1 1 ✓	- 1 1 - 1 (wxz)
1 0 0 1 1 ✓	- 1 1 0 1 ✓	1 - - 1 1 (vyz)
1 0 1 0 1 ✓	0 1 1 - 1 ✓	1 - 1 - 1 (vxz)
-----	1 - 0 1 1 ✓	By QM, these are the four prime implicants of $f(v,w,x,y,z)$
0 1 1 1 1 ✓	1 0 - 1 1 ✓	
1 0 1 1 1 ✓	1 - 1 0 1 ✓	
1 1 0 1 1 ✓	1 0 1 - 1 ✓	
1 1 1 0 1 ✓	-----	
-----	- 1 1 1 1 ✓	
1 1 1 1 1 ✓	1 - 1 1 1 ✓	
	1 1 - 1 1 ✓	
	1 1 1 - 1 ✓	

Express  $f(v,w,x,y,z)$  as the sum of all its prime implicants:

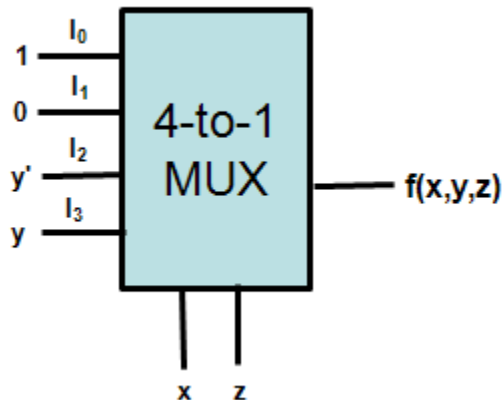
$$f(v,w,x,y,z) = wyz + wxz + vyz + vxz$$

Name \_\_\_\_\_

3. A circuit with output  $f(x,y,z)$  which uses a 8-to-1 MUX is shown. Design a circuit creating exactly the same output but which uses a 4-to-1 MUX. Use  $x$  and  $z$  as the select input lines as shown. Label the data inputs.



From the 8-to-1-MUX we see that  $f(x,y,z) = \sum m(0,2,4,7) = x'y'z' + x'yz' + xy'z' + xyz = x'z' + xy'z' + xyz$ . So in the 4-to-1 MUX for  $I_0$  ( $x'z'$ ) we need input 1,  $I_1$  ( $x'z$ ) we need 0,  $I_2$  ( $xz'$ ) we need  $y'$  and  $I_3$  ( $xz$ ) we need  $y$ .



4. As discussed in class and the textbook, a 1-bit comparator has outputs

$$G_{i+1} = A_i B_i' + A_i G_i + B_i' G_i$$

$$E_{i+1} = (A_i' B_i' + A_i B_i) E_i$$

$$L_{i+1} = A_i' B_i + B_i L_i + A_i' L_i$$

where  $G_i=1$  if " $A_i A_{i-1} \dots A_0$ " > " $B_i B_{i-1} \dots B_0$ ",  $E_i=1$  if they are equal, and  $L_i=1$  otherwise.

(a) If we use a 5-32 decoder to build a 1-bit comparator, we will also need one OR gate for each of the three outputs  $G_{i+1}$ ,  $E_{i+1}$ ,  $L_{i+1}$ . Specify how many inputs each of these OR gates will require, ie. how many lines will connect each OR gate to the decoder.

(b) A 2-bit comparator compares two bits " $A_{i1} A_{i0}$ " and " $B_{i1} B_{i0}$ " at a time instead of just one. Let  $G$ ,  $E$  and  $L$  be defined as above except  $G_{i+1}=1$  if

$$"A_{i1} A_{i0} A_{(i-1)1} A_{(i-1)0} \dots A_{01} A_{00}" > "B_{i1} B_{i0} B_{(i-1)1} B_{(i-1)0} \dots B_{01} B_{00}"$$

$E_{i+1}=1$  if they are equal, otherwise  $L_{i+1}=1$ . Write formulas similar to those on the top of this page for the three outputs of a 2-bit comparator. Note that your formulas should involve  $A_{i1}$ ,  $A_{i0}$ ,  $B_{i1}$  and  $B_{i0}$  instead of  $A_i$  and  $B_i$ .

(a) DEC outputs are minterms, so we need to count the minterms for each output. Lets count them for E, then by symmetry G and L will split the other minterms equally. There are 4 minterms involving  $A_i' B_i' E_i$ , and 4 for  $A_i B_i E_i$ . That leaves 12 for each of the other two outputs. Note: this solution sets all the "don't-care" outputs, which correspond to impossible inputs such as  $G_i$  and  $E_i$  both 1, all equal to 1. If you set them equal to 0, fewer inputs to the OR gates are required: 5 each for  $G_i$  and  $L_i$ , 2 for  $E_i$ .

(b)  $G_{i+1}=1$  if " $A_{i1} A_{i0}$ " > " $B_{i1} B_{i0}$ " or if they are equal and  $G_i=1$ .  $E_{i+1}=1$  if " $A_{i1} A_{i0}$ " = " $B_{i1} B_{i0}$ " and  $E_i=1$ . Otherwise,  $L_{i+1}=1$ , which we can get just by switching A and B, G and L in the formula for  $G_{i+1}$ .

(a) The  $G_{i+1}$  OR gate will have 12 inputs

The  $E_{i+1}$  OR gate will have 8 inputs

The  $L_{i+1}$  OR gate will have 12 inputs

$$(b) \quad G_{i+1} = \underline{A_{i1} B_{i1}' + (A_{i1} B_{i1} + A_{i1}' B_{i1}')((A_{i0} B_{i0}') + (A_{i0} B_{i0} + A_{i0}' B_{i0}'))(G_i)}$$

$$E_{i+1} = \underline{(A_{i1} B_{i1} + A_{i1}' B_{i1}') (A_{i0} B_{i0} + A_{i0}' B_{i0}')(E_i)}$$

$$L_{i+1} = \underline{A_{i1}' B_{i1} + (A_{i1} B_{i1} + A_{i1}' B_{i1}')((A_{i0}' B_{i0}') + (A_{i0} B_{i0} + A_{i0}' B_{i0}'))(L_i)}$$