Math 245: Discrete Mathematics

Logic — Minimization of Expressions

Karnaugh-maps and the Quine-McCluskey Method

Lecture Notes #3$\frac{1}{2}$

Peter Blomgren
Department of Mathematics and Statistics
San Diego State University
San Diego, CA 92182-7720
blomgren@terminus.SDSU.EDU
http://terminus.SDSU.EDU

$Id: lecture.tex,v 1.11 2006/09/19 22:22:01 blomgren Exp $

Minimization of Logic Expressions

One major application area of logic is circuit design.

For many reasons, mostly $$$-related, we want to make our circuits as efficient as possible.

Consider the following logically equivalent statements

$$\begin{aligned}(x \wedge y \wedge z) \vee (x \wedge (\sim y) \wedge z) &\equiv (y \vee (\sim y)) \wedge (x \wedge z) \\ &\equiv t \wedge (x \wedge z) \\ &\equiv (x \wedge z)\end{aligned}$$

Clearly(?), the expression $(x \wedge z)$ is the most *efficient* representation of this particular statement.

In this lecture we will explore two methods for such minimization of logic expressions: — Karnaugh maps (K-maps), and the Quine-McCluskey method.

K-maps

K-maps is a ***graphical method*** for minimizing logic expressions (by hand). K-maps are suited for expressions containing no more than ***six*** statement variables.

We consider an expression with two statement variables x, and y. There are four possible and/not "minterms" of x, and y:

$$x \wedge y, \qquad x \wedge (\sim y), \qquad (\sim x) \wedge y, \qquad (\sim x) \wedge (\sim y)$$

A K-map in these variables consists of four cells

	y	$\sim y$
x		
$\sim x$		

where a 1 is placed in a cell if that "minterm" is part of the expression.

K-maps — First Examples

The K-maps for
(a) $(x \wedge y) \vee ((\sim x) \wedge y)$,
(b) $(x \wedge (\sim y)) \vee ((\sim x) \wedge y)$, and
(c) $(x \wedge (\sim y)) \vee ((\sim x) \wedge y) \vee ((\sim x) \wedge (\sim y)$ are given by:

	y	$\sim y$
x	1	
$\sim x$	1	

(a)

	y	$\sim y$
x		1
$\sim x$	1	

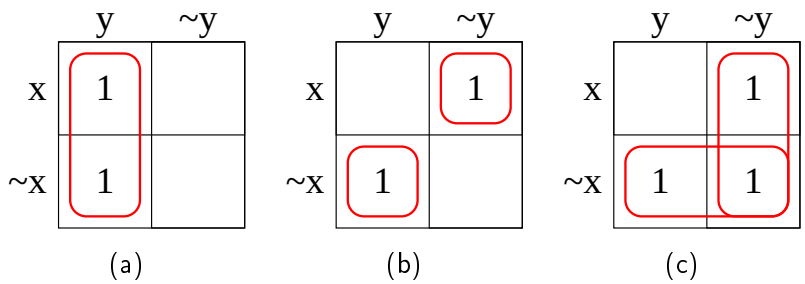
(b)

	y	$\sim y$
x		1
$\sim x$	1	1

(c)

Whenever two adjacent cells in the K-map have 1s; the minterms representing these cells can be combined into an expression involving just ***one*** of the variables.

K-maps, Simplification — First Examples



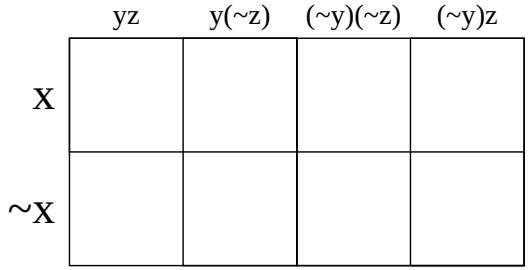
The each block in the K-maps above reduce to one expression, so that the simplified expressions are

(a) y

(b) $(x \wedge (\sim y)) \vee ((\sim x) \wedge y)$

(c) $(\sim x) \vee (\sim y)$

K-maps in Three Variables



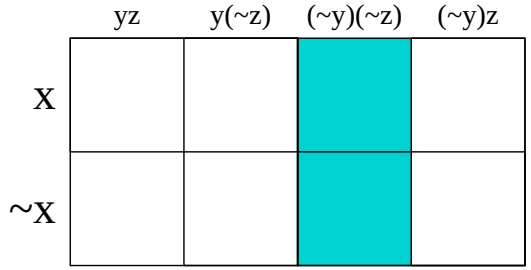
In three variables we set up a 4×2 rectangle of 8 cells.

Adjacent cells should differ in only ***one*** statement variable.

The left-most and right-most cells should be considered adjacent (think of wrapping the strip around a cylinder).

Notation: Here yz represents $(y \wedge z)$, and a 1 in the upper left cell corresponds to the three-variable minterm $(x \wedge y \wedge z)$.

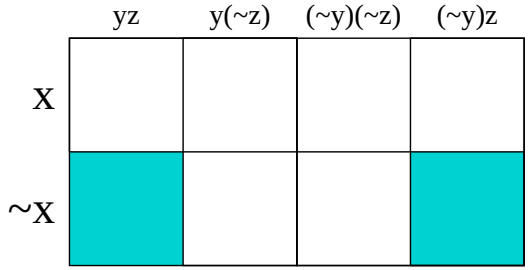
K-maps in Three Variables — Rules 1/5



$$(x \wedge (\sim y) \wedge (\sim z)) \vee ((\sim x) \wedge (\sim y) \wedge (\sim z)) \equiv (\sim y) \wedge (\sim z)$$

Two adjacent cells represent two three-variable minterms, which can be reduced to a two-variable term.

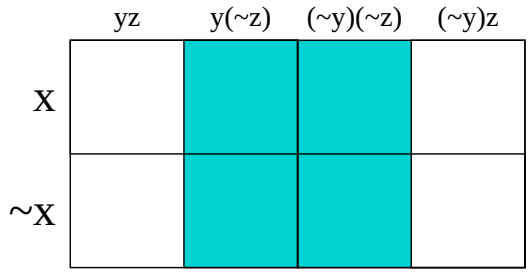
K-maps in Three Variables — Rules 2/5



$$((\sim x) \wedge (\sim y) \wedge z) \vee ((\sim x) \wedge y \wedge z) \quad \equiv \quad (\sim x) \wedge z$$

Two adjacent cells represent two three-variable minterms, which can be reduced to a two-variable term. (Note that due to wrap-around the two high-lighted cells ***are*** considered to be adjacent.)

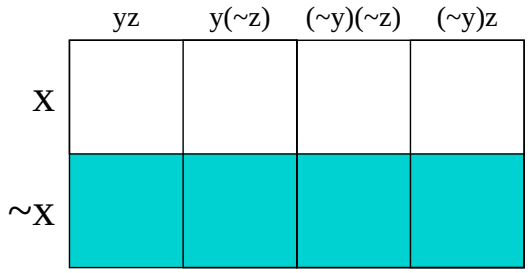
K-maps in Three Variables — Rules 3/5



$$(x \wedge y \wedge (\sim z)) \vee (x \wedge (\sim y) \wedge (\sim z)) \vee$$
$$((\sim x) \wedge y \wedge (\sim z)) \vee ((\sim x) \wedge (\sim y) \wedge (\sim z)) \equiv \sim z$$

A 2×2-block of cells represents 4 minterms which can be reduced to a single variable.

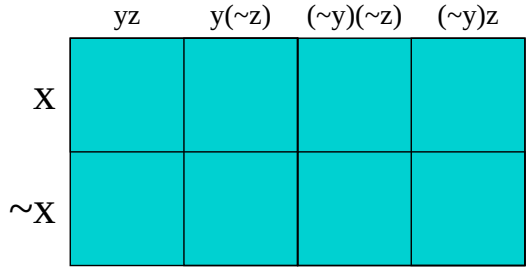
K-maps in Three Variables — Rules 4/5



$$((\sim x) \wedge y \wedge z) \vee ((\sim x) \wedge y \wedge (\sim z)) \vee$$
$$((\sim x) \wedge (\sim y) \wedge (\sim z)) \vee ((\sim x) \wedge (\sim y) \wedge z) \equiv \sim x$$

A 4×1-block of cells represents 4 minterms which can be reduced to a single variable.

K-maps in Three Variables — Rules 5/5



$$(x \wedge y \wedge z) \vee (x \wedge y \wedge (\sim z)) \vee$$
$$(x \wedge (\sim y) \wedge (\sim z)) \vee (x \wedge (\sim y) \wedge z) \vee$$
$$((\sim x) \wedge y \wedge z) \vee ((\sim x) \wedge y \wedge (\sim z)) \vee$$
$$((\sim x) \wedge (\sim y) \wedge (\sim z)) \vee ((\sim x) \wedge (\sim y) \wedge z) \equiv t$$

A 4×2-block of cells represents 8 minterms which can be reduced to a ***tautology***.

K-maps: Language, and Goal

A block of 1s is called an ***implicant*** of the expression being minimized. It is a ***prime implicant*** if it is not contained in a larger block.

The goal is to identify the largest possible blocks in the map, and cover all the 1s in the map by the fewest number of blocks.

In general, the largest blocks are chosen; since they give the most simplification.

A block, which is the only one to cover a particular 1, is called a ***essential prime implicant***, and must be chosen.

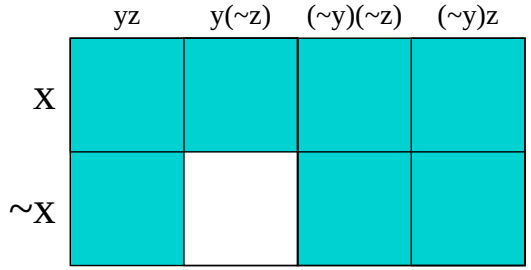
By covering all blocks with prime implicants, we can massage the expression into an $\vee$-connected set of prime implicants.

Example #1 — 1 of 2

Minimize

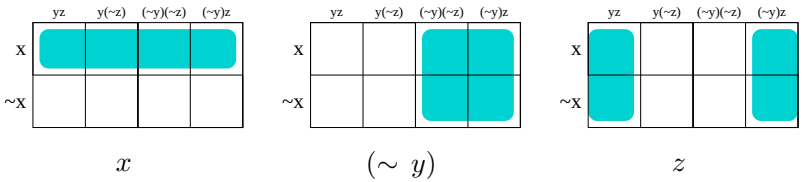
$$\begin{array}{lcll} (x \wedge y \wedge z) & \vee & (x \wedge y \wedge (\sim z)) & \vee \\ (x \wedge (\sim y) \wedge (\sim z)) & \vee & (x \wedge (\sim y) \wedge z) & \vee \\ ((\sim x) \wedge y \wedge z) & \vee & ((\sim x) \wedge (\sim y) \wedge (\sim z)) & \vee \\ ((\sim x) \wedge (\sim y) \wedge z) & \equiv & ??? & \end{array}$$

First, we fill in the K-map:



Example #1 — 2 of 2

We can break the K-map into three blocks



x $\quad$ $(\sim y)$ $\quad$ z

which shows that the expression is equivalent to

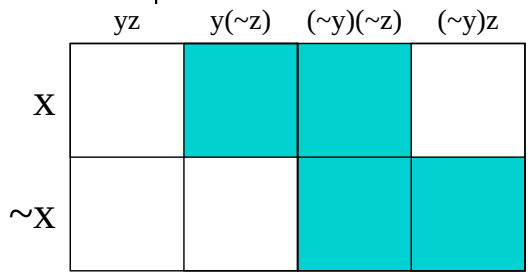
$$x \vee (\sim y) \vee z$$

Example #2 — 1 of 2

Minimize

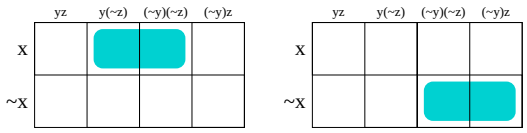
$$\begin{array}{lcll} (x \wedge y \wedge (\sim z)) & \vee & (x \wedge (\sim y) \wedge (\sim z)) & \vee \\ ((\sim x) \wedge (\sim y) \wedge (\sim z)) & \vee & ((\sim x) \wedge (\sim y) \wedge z) & \equiv \quad ??? \end{array}$$

First, we fill in the K-map:



Example #2 — 2 of 2

We can break the K-map into two blocks

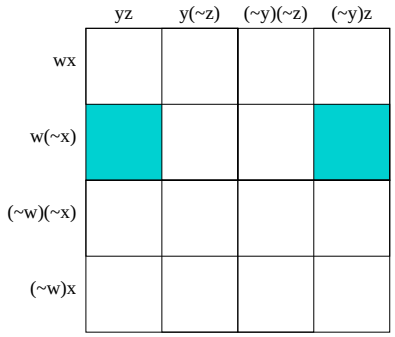


$x \wedge (\sim z)$ $\quad$ $(\sim x) \wedge (\sim y)$

which shows that the expression is equivalent to

$$(x \wedge (\sim z)) \vee ((\sim x) \wedge (\sim y))$$

K-maps in Four Variables — Rules 1/5



$$(w \wedge (\sim x) \wedge y \wedge z) \quad \vee \quad (w \wedge (\sim x) \wedge (\sim y) \wedge z)$$
$$\equiv w \wedge (\sim x) \wedge z$$

A 2×1-block of cells represents 2 minterms which can be reduced to a 3-variable expression.

K-maps in Four Variables — Rules 2/5

Note: The top and bottom of the square are considered adjacent, and so are the left and right; topologically it is a torus (donut)

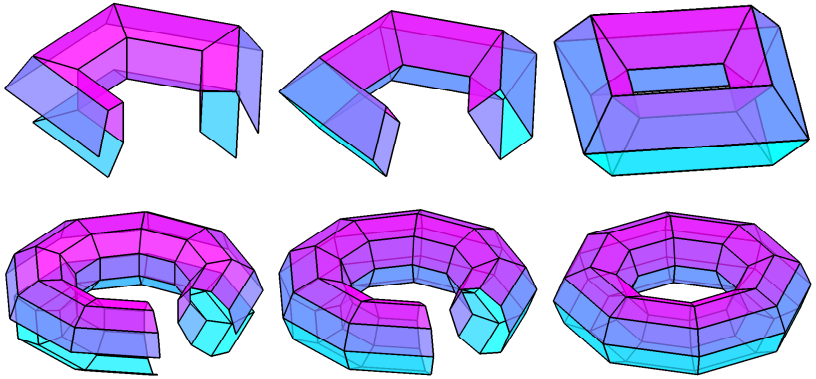
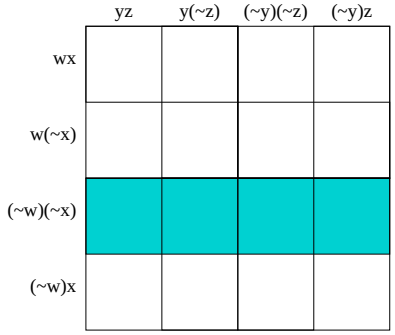


Figure: The folding of a 4×4, and a 8×8 square into a torus.

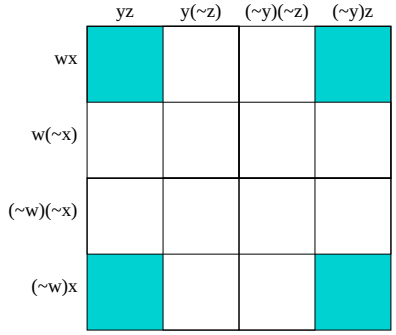
K-maps in Four Variables — Rules 3/5



$$((\sim w) \wedge (\sim x) \wedge y \wedge z) \quad \vee \quad ((\sim w) \wedge (\sim x) \wedge y \wedge (\sim z)) \quad \vee$$
$$((\sim w) \wedge (\sim x) \wedge (\sim y) \wedge (\sim z)) \quad \vee \quad ((\sim w) \wedge (\sim x) \wedge (\sim y) \wedge z)$$
$$\equiv (\sim w) \wedge (\sim x)$$

A 4×1-block of cells represents 4 minterms which can be reduced to a 2-variable expression.

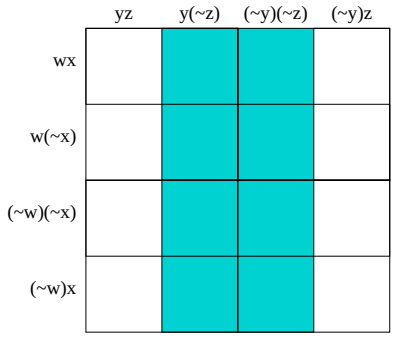
K-maps in Four Variables — Rules 4/5



$$(w \wedge x \wedge y \wedge z) \quad \vee \quad (w \wedge x \wedge (\sim y) \wedge z) \quad \vee$$
$$((\sim w) \wedge x \wedge y \wedge z) \quad \vee \quad ((\sim w) \wedge x \wedge (\sim y) \wedge z)$$
$$\equiv x \wedge z$$

A 2×2-block of cells represents 4 minterms which can be reduced to a 2-variable expression.

K-maps in Four Variables — Rules 5/5



$$
\begin{array}{lcl}
(w \wedge x \wedge y \wedge (\sim z)) & \vee & (w \wedge x \wedge (\sim y) \wedge (\sim z)) \quad \vee \\
(w \wedge (\sim x) \wedge y \wedge (\sim z)) & \vee & (w \wedge (\sim x) \wedge (\sim y) \wedge (\sim z)) \quad \vee \\
((\sim w) \wedge (\sim x) \wedge y \wedge (\sim z)) & \vee & ((\sim w) \wedge (\sim x) \wedge (\sim y) \wedge (\sim z)) \quad \vee \\
((\sim w) \wedge x \wedge y \wedge (\sim z)) & \vee & ((\sim w) \wedge x \wedge (\sim y) \wedge (\sim z)) \\
\equiv (\sim z) & &
\end{array}
$$

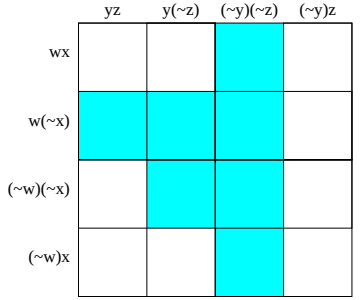
A 2×4-block of cells represents 8 minterms which can be reduced to a 1-variable expression.

Example #3 — 1 of 2

Minimize

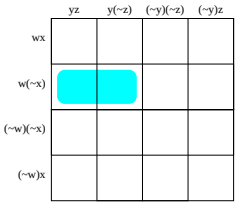
$$
\begin{array}{lcl}
(w \wedge x \wedge (\sim y) \wedge (\sim z)) & \vee & (w \wedge (\sim x) \wedge y \wedge z) \quad \vee \\
(w \wedge (\sim x) \wedge y \wedge (\sim z)) & \vee & (w \wedge (\sim x) \wedge (\sim y) \wedge (\sim z)) \quad \vee \\
((\sim w) \wedge x \wedge (\sim y) \wedge (\sim z)) & \vee & ((\sim w) \wedge (\sim x) \wedge y \wedge (\sim z)) \quad \vee \\
((\sim w) \wedge (\sim x) \wedge (\sim y) \wedge (\sim z)) & \equiv & ???
\end{array}
$$

We fill in the K-map:

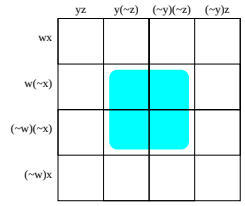


Example #3 — 2 of 2

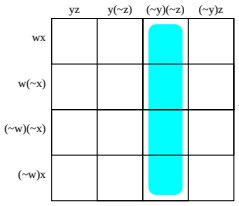
We can break the K-map into three blocks



$w \wedge (\sim x) \wedge y$



$(\sim x) \wedge (\sim z)$



$(\sim y) \wedge (\sim z)$

which shows that the expression is equivalent to

$$(w \wedge (\sim x) \wedge y) \vee ((\sim x) \wedge (\sim z)) \vee ((\sim y) \wedge (\sim z))$$

K-maps with Even More Variables

Each additional variable doubles either the number of columns, or the number of rows (alternating) in the associated K-map.

Clearly, as the number of variables grows this procedure becomes more complicated, and it gets more difficult to see what the best splitting of the K-map is.

Note: In most cases, there are more than one "minimal" expression, *i.e.* the "blocking step" is non-unique.

Note: The use of K-maps relies manual visual inspection. This approach is difficult to automate.

As the use of K-maps gets cumbersome for large (> 4) number of variables, we now look at a method which can be automated — the ***Quine-McCluskey Method***.

The Quine-McCluskey Method

The Quine-McCluskey method is an automatable two-stage scheme for minimizing an expression.

⇒ In the first stage we figure out what terms can be combined to form terms with fewer variables.

⇒ In the second stage we figure out which of the fewer-variable-terms are needed to "cover" the expression.

A couple of examples will shed some light on the issue...

Quine-McCluskey Example #1 1 of 4

Minimize

$$(x \wedge y \wedge z) \vee (x \wedge (\sim y) \wedge z) \vee ((\sim x) \wedge y \wedge z) \vee ((\sim x) \wedge (\sim y) \wedge z) \vee ((\sim x) \wedge (\sim y) \wedge (\sim z)) \equiv ???$$

For notational convenience, we "code" the expressions using binary notation (1s for the variables, and 0s for not'ed variables,) here:

Expression	Binary Code
$(x \wedge y \wedge z)$	111
$(x \wedge (\sim y) \wedge z)$	101
$((\sim x) \wedge y \wedge z)$	011
$((\sim x) \wedge (\sim y) \wedge z)$	001
$((\sim x) \wedge (\sim y) \wedge (\sim z))$	000

Quine-McCluskey Example #2 2 of 4

The ***Hamming distance*** between two binary expressions, is the number of differing bits. In our case, codes with Hamming distance 1 correspond to logic expressions which can be simplified.

We build a table in the following way:

#	Code	Combo	Code#2	Combo	Code#3
$(1)^1$	111	$(1,2)^2$	1*1	(1,2,3,4)	**1
$(2)^1$	101	$(1,3)^2$	*11		
$(3)^1$	011	$(2,4)^2$	*01		
$(4)^1$	001	$(3,4)^2$	0*1		
$(5)^1$	000	(4,5)	00*		

(1,2,3,4) can be formed in two ways: (1,2)+(3,4), and (1,3)+(2,4). The result is the same, therefore we consider (1,2), (1,3), (2,4), and (3,4) used in the second step.

Quine-McCluskey Example #2 3 of 4

At this point, not more combinations can be made; we set up a table with columns for all the initial (n) combinations, and one row for each terminal ("unused") combination, and mark the common elements:

	(1)	(2)	(3)	(4)	(5)
(1,2,3,4)	√	√	√	√	
(4,5)				√	√

Next, we select the smallest number of combinations which cover all the initial singlets; in this case we need both (1,2,3,4) and (4,5).

We have

Tag	Binary	Logical
(1,2,3,4)	**1	z
(4,5)	00*	$(\sim x) \wedge (\sim y)$

Quine-McCluskey Example #1 4 of 4

Therefore, we have

$$\begin{array}{lllll} (x \wedge y \wedge z) & \vee & (x \wedge (\sim y) \wedge z) & \vee & ((\sim x) \wedge y \wedge z) \quad \vee \\ ((\sim x) \wedge (\sim y) \wedge z) & \vee & ((\sim x) \wedge (\sim y) \wedge (\sim z)) & \equiv & \mathbf{z} \vee ((\sim \mathbf{x}) \wedge (\sim \mathbf{y})) \end{array}$$

Comment: In this case, the minimal expression is unique; that is not true in general — there may be several (equivalent) minimal expressions.

Quine-McCluskey Example #2 1 of 3

Minimize

$$\begin{array}{lllll} (w \wedge x \wedge y \wedge (\sim z)) & \vee & (w \wedge (\sim x) \wedge y \wedge z) & \vee \\ ((\sim w) \wedge x \wedge y \wedge z) & \vee & (w \wedge (\sim x) \wedge y \wedge (\sim z)) & \vee \\ ((\sim w) \wedge x \wedge (\sim y) \wedge z) & \vee & ((\sim w) \wedge (\sim x) \wedge y \wedge z) & \vee \\ ((\sim w) \wedge (\sim x) \wedge (\sim y) \wedge z) & \equiv & ??? \end{array}$$

Expression	Binary Code
$(w \wedge x \wedge y \wedge (\sim z))$	1110
$(w \wedge (\sim x) \wedge y \wedge z)$	1011
$((\sim w) \wedge x \wedge y \wedge z)$	0111
$(w \wedge (\sim x) \wedge y \wedge (\sim z))$	1010
$((\sim w) \wedge x \wedge (\sim y) \wedge z)$	0101
$((\sim w) \wedge (\sim x) \wedge y \wedge z)$	0011
$((\sim w) \wedge (\sim x) \wedge (\sim y) \wedge z)$	0001

Quine-McCluskey Example #2 2 of 3

Singlet	Code	Pair	Code	Quad	Code
(1)$^{\times}$	1110	(1,4)	1*10	(3,5,6,7)	0**1
(2)$^{\times}$	1011	(2,4)	101*		
(3)$^{\times}$	0111	(2,6)	*011		
(4)$^{\times}$	1010	(3,5)$^{\times}$	01*1		
(5)$^{\times}$	0101	(3,6)$^{\times}$	0*11		
(6)$^{\times}$	0011	(5,7)$^{\times}$	0*01		
(7)$^{\times}$	0001	(6,7)$^{\times}$	00*1		

	(1)	(2)	(3)	(4)	(5)	(6)	(7)
(3,5,6,7)			x		x	x	x
(1,4)	x			x			
(2,4)		x		x			
(2,6)		x				x	

Quine-McCluskey Example #2 3 of 3

	(1)	(2)	(3)	(4)	(5)	(6)	(7)
0**1			x		x	x	x
1*10	x			x			
101*		x		x			
*011		x				x	

The top two expressions are needed — to cover (3) and (1). Additionally one of the two bottom ones are needed. Therefore,

1. $((\sim w) \wedge z) \quad \vee \quad (w \wedge y \wedge (\sim z)) \quad \vee \quad (w \wedge (\sim x) \wedge y)$
2. $((\sim w) \wedge z) \quad \vee \quad (w \wedge y \wedge (\sim z)) \quad \vee \quad ((\sim x) \wedge y \wedge z)$

are two possible minimal expressions equivalent to the initial one.

Homework Homework #3 — Due Friday 9/29/2006 Final Version

Use ***both*** K-maps and the Quine-McCluskey method to minimize the following two expressions:

1.

$$
\begin{array}{lcl}
(x \wedge (\sim y) \wedge z) & \vee & (x \wedge (\sim y) \wedge (\sim z)) \quad \vee \\
((\sim x) \wedge y \wedge z) & \vee & ((\sim x) \wedge (\sim y) \wedge z) \quad \vee \\
((\sim x) \wedge (\sim y) \wedge (\sim z)) & \equiv & ???
\end{array}
$$

2.

$$
\begin{array}{lcl}
(w \wedge x \wedge y \wedge (\sim z)) & \vee & (w \wedge x \wedge (\sim y) \wedge (\sim z)) \quad \vee \\
(w \wedge (\sim x) \wedge y \wedge z)) & \vee & (w \wedge (\sim x) \wedge y \wedge (\sim z))) \quad \vee \\
(w \wedge (\sim x) \wedge (\sim y) \wedge (\sim z))) & \vee & ((\sim w) \wedge x \wedge y \wedge z)) \quad \vee \\
((\sim w) \wedge x \wedge y \wedge (\sim z))) & \vee & ((\sim w) \wedge x \wedge (\sim y) \wedge (\sim z))) \quad \vee \\
((\sim w) \wedge x \wedge (\sim y) \wedge z)) & \vee & ((\sim w) \wedge (\sim x) \wedge y \wedge (\sim z))) \quad \vee \\
((\sim w) \wedge (\sim x) \wedge (\sim y) \wedge (\sim z))) & \equiv & ???
\end{array}
$$