Lecture 5 – Quine McCluskey Minimisation

Karl R. Wilcox
Karl@cs.rhul.ac.uk

Administration

- **Lab Session to be arranged**
  - **More information on Monday**
- **Lecture Notes**
  - **Are available at http://www.cs.rhul.ac.uk/~karl**
  - **Formats**
    - **Original Powerpoint 2000 slides**
    - **PDF files, A4 2 slides per page, colour**
    - **Let me know if you would like other formats**

Objectives

- **In this lecture we will discuss**
  - **Review**
  - **Reminder about Karnaugh Maps**
  - **Quine McCluskey minimisation**

Review

- In Computer Engineering I you covered basic logic circuits and discrete hardware
  - Laboratory work to implement logic functions
- We have looked at how to use these basic circuits to implement state machines
  - Laboratory work using ROMs and PLAs
- State machines can be expressed as logic functions
  - But we need to implement these using the minimum amount of logic
- State machines are very simple “CPUs”
- CPUs are very complicated state machines!

Reminder of Karnaugh Maps

- A method of representing boolean function on a plane
- Offers a simple graphical method to minimize boolean functions
- Practical for 4,5,6 variables then too cumbersome to use
- A K-map consists of a grid of squares, and each square representing one canonical minterm combination of the variables or their inverse
- The map is arranged with squares representing minterms which differ by only one variable to be adjacent both vertically and horizontally
- Therefore AB‘C‘ would be adjacent to A ‘ B ‘C ‘ and would also adjacent to AB ‘ C and ABC‘

K-Maps and Minterms

CD \ AB	00	01	11	10
00	0	4	12	8
01	1	5	13	9
11	3	7	15	11
10	2	6	14	10

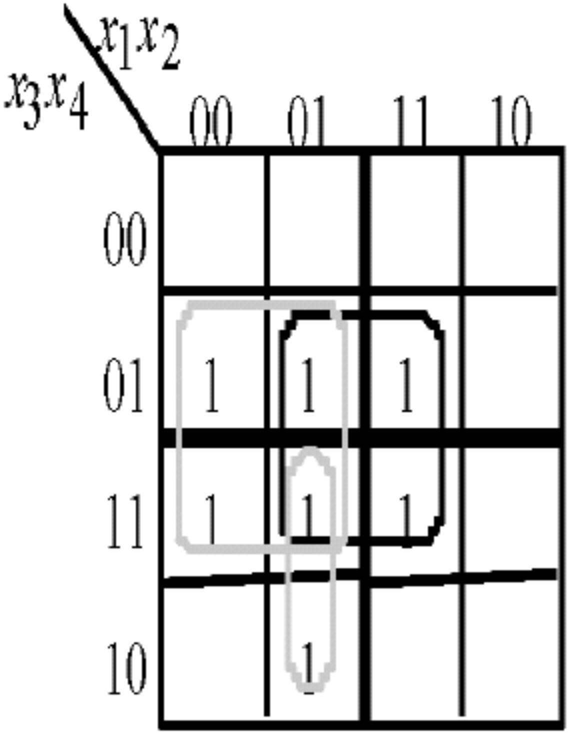
Using K-Maps – 1

- The expression to be minimized should generally be in **sum-of-product** form
  - If necessary, the conversion process are applied to create the sum-of-product form
- The function is ‘mapped‘ onto the K-map by marking a 1 in those squares corresponding to the terms in the expression to be simplified
  - the other squares may be filled with 0‘s
- Pairs of 1‘s on the map which are adjacent are combined using the theorem **P(A+A ‘) = P** where P is any Boolean expression
  - If two pairs are also adjacent, then these can also be combined using the same theorem

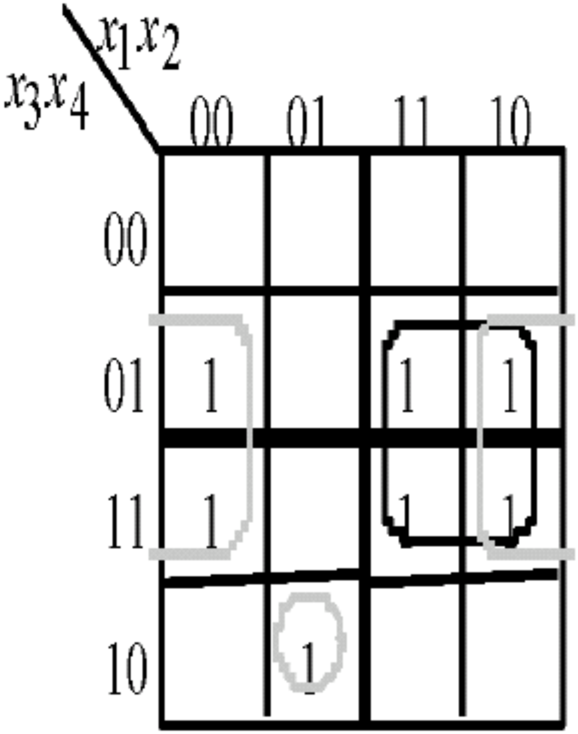
Using K-maps – 2

- The minimization procedure consists of **recognizing those pairs and multiple pairs**
  - these are circled indicating reduced terms
  - groups which can be circled are those which have two (20) 1‘s, four (21) 1‘s, eight (22) 1‘s, and so on
  - note that because squares on one edge of the map are considered adjacent to those on the opposite edge, group can be formed with these squares
  - groups are allowed to overlap
- the objective is to cover all the 1‘s on the map in the fewest number of groups and to create the largest group to do this

K-Map Example



x_3x_4 \ x_1x_2	00	01	11	10
00				
01	1	1	1	
11	1	1	1	
10		1		



x_3x_4 \ x_1x_2	00	01	11	10
00				
01	1		1	1
11	1		1	1
10		1		

Important Terms

- **Prime Implicant**
  - It is the name given to the groups formed on the map
  - may include groups not actually necessary in the final solutio
- **Essential Prime Implicants**
  - It is the name given to the groups which include at least one 1 not covered by any other group
- **Non-essential Prime Implicants**
  - it is the name given to the prime implicants which are not essential but nevertheless cover 1‘s on the map
- **Redundant Prime Implicants**
  - Non-essential prime implicants which only cover 1‘s already covered by essential prime implicants
- From these definitions it can be seen that the minimal solution is given by **all the essential prime implicants plus** a careful selection of **the non-essential prime implicants**

Quine McCluskey – 1

- Quine-McCluskey minimization method uses the same theorem to produce the solution as the K-map method, namely P(A+A')=P
- Necessary Steps:
  - the expression is represented in the canonical SOP form if not already in that form
  - the function is converted into numeric notation
  - the numbers are converted into binary form
  - the minterms are arranged in a column divided into groups
  - begin with the minimization procedure
    - each minterm of one group is compared with each minterm in the group immediately below
    - each time a number is found in one group which is the same as a number in the group below except for one digit, the pair numbers is ticked and a new composite is created

Quine McCluskey – 2

- This composite number has the same number of digits as the numbers in the pair except the digit different which is replaced by an X
  - The above procedure is repeated on second column to generate a third column
  - The next step is to identify the essential prime implicants which can be done using a **prime implicant chart**
- Where a prime implicant covers a minterm, the intersection of the corresponding row and column is marked with a cross
  - Those columns with only one cross identify the essential prime implicants → these prime implicants must be in the final answer
- The single crosses on a column are circled and all the crosses on the same row are also circled, indicating that these crosses are covered by prime implicants selected

Quine McCluskey – 3

- Once one cross on a column is circled, all the crosses on that column can be circled since the minterm is now covered
  - If any non-essential prime implicant has all its crosses circled, the prime implicant is redundant and need not be considered further
- Next, a selection must be made from the remaining nonessential prime implicants, by considering how the crosses not circled can best be covered
  - one generally would take those prime implicants which cover the greates number of crosses on their row
  - if all the crosses in one row also occur on another row which includes further crosses, then the latter is said to dominate the former and can be selected
  - the dominated prime implicant can then be deleted

Q-M Example A – 1

- Consider the function
  - f(A,B,C) = !A!B!C + !A!BC + A!B!C + A!BC
- Change this to the binary notation
  - f(A,B,C) = SUM(000, 001, 100, 101)
- This comes from:
  - !A!B!C : Binary notation 000 : Index 0 : Decimal value 0
  - !A!BC : Binary notation 001 : Index 1 : Decimal value 1
  - A!B!C : Binary notation 100 : Index 1 : Decimal value 4
  - A!BC : Binary notation 101 : Index 2 : Decimal value 5

Q-M Example A – 2

```
 -----------------------------------------------------------------------------
|         1st List        |         2nd List        |         3rd List        |
 -----------------------------------------------------------------------------
|                         |                         |                         |
|             A B C       |             A B C       |             A B C       |
|  ---------- -----       |  ---------- -----       |  ---------- -----       |
|      (0)    0 0 0 [x]   |     (0,1)   0 0 - [x]   |   (0,1,4,5) - 0 -       |
|  ---------- -----       |     (0,4)   - 0 0 [x]   |   (0,4,1,5) - 0 -       |
|      (1)    0 0 1 [x]   |  ---------- -----       |                         |
|      (4)    1 0 0 [x]   |     (1,5)   - 0 1 [x]   |                         |
|  ---------- -----       |     (4,5)   1 0 - [x]   |                         |
|      (5)    1 0 1 [x]   |                         |                         |
|                         |                         |                         |
 -----------------------------------------------------------------------------
```

Q-M Example B – 1

- Consider the function
  - f(A,B,C,D) = SUM(0,1,2,3,5,7,8,10,12,13,15)
- in decimal form
  - SUM(0000,0001,0010,0011,0101,0111,1000,1010,1100,1101,1111)
- in binary form
  - SUM(0,1,1,2,2,3,1,2,2,3,4)

Q-M Example B - 2

```
 -----------------------------------------------------------------------------------
|         1st List          |         2nd List          |         3rd List          |
 -----------------------------------------------------------------------------------
|                           |                           |                           |
|               A B C D     |               A B C D     |               A B C D     |
| ------------- -------     | ------------- -------     | ------------- -------     |
|     (00)      0 0 0 0 [x] |    (00,01)    0 0 0 - [x] | (00,01,02,03) 0 0 - - (3) |
| ------------- -------     |    (00,02)    0 0 - 0 [x] | (00,02,01,03) 0 0 - -     |
|     (01)      0 0 0 1 [x] |    (00,08)    - 0 0 0 [x] | (00,02,08,10) - 0 - 0 (4) |
|     (02)      0 0 1 0 [x] | ------------- -------     | (00,08,02,10) - 0 - 0     |
|     (08)      1 0 0 0 [x] |    (01,03)    0 0 - 1 [x] | ------------- -------     |
| ------------- -------     |    (01,05)    0 - 0 1 [x] | (01,03,05,07) 0 - - 1 (5) |
|     (03)      0 0 1 1 [x] |    (02,03)    0 0 1 - [x] | (01,05,03,07) 0 - - 1     |
|     (05)      0 1 0 1 [x] |    (02,10)    - 0 1 0 [x] | ------------- -------     |
|     (10)      1 0 1 0 [x] |    (08,10)    1 0 - 0 [x] | (05,07,13,15) - 1 - 1 (6) |
|     (12)      1 1 0 0 [x] |    (08,12)    1 - 0 0 (1) | (05,13,07,15) - 1 - 1     |
| ------------- -------     | ------------- -------     |                           |
|     (07)      0 1 1 1 [x] |    (03,07)    0 - 1 1 [x] |                           |
|     (13)      1 1 1 1 [x] |    (05,07)    0 1 - 1 [x] |                           |
| ------------- -------     |    (05,13)    - 1 0 1 [x] |                           |
|     (15)      1 1 1 1 [x] |    (12,13)    1 1 0 - (2) |                           |
|                           | ------------- -------     |                           |
|                           |    (07,15)    - 1 1 1 [x] |                           |
|                           |    (13,15)    1 1 - 1 [x] |                           |
 -----------------------------------------------------------------------------------
```

The prime implicants are:

```
!A!B + !B!D + !AD + BD + A!C!D + AB!C
```

Q-M Example B - 3

IMPLICANTS	00	01	02	03	05	07	08	10	12	13	15
!A!B	x	x	x	x							
!B!D	x		x				x	(x)			
!AD		x		x	x	x					
BD					x	x				x	(x)
A!C!D							x		x		
AB!C									x	x	
Essential	x		x		x	x	x	(x)		x	(x)

Q-M Example B – 3

- From the above chart, BD is an essential prime implicant. It is the only prime implicant that covers the minterm decimal 15 and it also includes 5, 7 and 13. !B!D is also an essential prime implicant. It is the only prime implicant that covers the minterm denoted by decimal 10 and it also includes the terms 0, 2 and 8. The other minterms of the function are 1, 3 and 12. Minterm 1 is present in !A!B and !AD. Similarly for minterm 3. We can therefore use either of these prime implicants for these minterms. Minterm 12 is present in A!C!D and AB!C, so again either can be used.
- Thus, one minimal solution is:
  - Z = !B!D + BD + !A!B + A!C!D

Q-M Features

- Like Karnaugh maps, Q-M relies on “Theorem T10,” that is
  - AB + AB’ = A.
- Computerized
- Theoretically no limit to number of input variables
- Effective, always finds all the prime implicants.
- However, the complexity (worst case) is exponential in the number of variables!

Summary

- State machines are just “digital logic”
  - But more complicated
- They can be represented as “truth tables”
  - Or logic functions
- We can use automated techniques (like Q-M) to minimise the amount of logic we need to implement the function
- Used in CAD tools to create programmable logic designs
- All “real” state machines (e.g. CPUs) require this automated support

Next Week

- Introduction to Central Processing Units

- Next Laboratory session, To be arranged
- Next Lecture, Monday C336