

# CSE 140 Homework Two

February 10, 2014

*Only **Problem Set Part B** will be graded. Turn in only **Problem Set Part B** which will be due on February 25, 2014 (Tuesday) at 4:00pm.*

Please note that the first problem of **Part B** is the same as the last problem of your first homework.

## 1 Problem Set Part A

- Roth&Kinney, 6th Ed: 2.1
- Roth&Kinney, 6th Ed: 2.3
- Roth&Kinney, 6th Ed: 2.5
- Roth&Kinney, 6th Ed: 2.6
- Roth&Kinney, 6th Ed: 2.7
- Roth&Kinney, 6th Ed: 2.8
- Roth&Kinney, 6th Ed: 2.12
- Roth&Kinney, 6th Ed: 2.22
- Roth&Kinney, 6th Ed: 2.29
- Roth&Kinney, 6th Ed: 2.30
- Roth&Kinney, 6th Ed: 3.9
- Roth&Kinney, 6th Ed: 3.10
- Roth&Kinney, 6th Ed: 3.12
- Roth&Kinney, 6th Ed: 3.16
- Roth&Kinney, 6th Ed: 3.17
- Roth&Kinney, 6th Ed: 3.19
- Roth&Kinney, 6th Ed: 3.20
- Roth&Kinney, 6th Ed: 3.21
- Roth&Kinney, 6th Ed: 3.29

- Roth&Kinney, 6th Ed: 3.30
- Roth&Kinney, 6th Ed: 3.33
- Roth&Kinney, 6th Ed: 3.34
- Roth&Kinney, 6th Ed: 3.37
- Roth&Kinney, 6th Ed: 4.5
- Roth&Kinney, 6th Ed: 4.6
- Roth&Kinney, 6th Ed: 4.8
- Roth&Kinney, 6th Ed: 4.13
- Roth&Kinney, 6th Ed: 4.17
- Roth&Kinney, 6th Ed: 4.18
- Roth&Kinney, 6th Ed: 4.25
- Roth&Kinney, 6th Ed: 4.32
- Roth&Kinney, 6th Ed: 4.42
- Roth&Kinney, 6th Ed: 5.10
- Roth&Kinney, 6th Ed: 5.11
- Roth&Kinney, 6th Ed: 5.12
- Roth&Kinney, 6th Ed: 5.21
- Roth&Kinney, 6th Ed: 5.28
- Roth&Kinney, 6th Ed: 5.29
- Roth&Kinney, 6th Ed: 5.32
- Roth&Kinney, 6th Ed: 5.33
- Roth&Kinney, 6th Ed: 5.35
- Roth&Kinney, 6th Ed: 5.42
- Roth&Kinney, 6th Ed: 6.4
- Roth&Kinney, 6th Ed: 6.5
- Roth&Kinney, 6th Ed: 6.11
- Roth&Kinney, 6th Ed: 6.17

## 2 Problem Set Part B

### 1 (Boolean Axioms)

The following incomplete truth tables show the 2 input  $*$  and  $+$  operations for a possible 3-value Boolean algebra:

| $*$ | 0 | A | 1 |
|-----|---|---|---|
| 0   |   |   |   |
| A   |   |   |   |
| 1   |   |   |   |

| $+$ | 0 | A | 1 |
|-----|---|---|---|
| 0   |   |   |   |
| A   |   |   |   |
| 1   |   |   |   |

For the following questions, you may find the Boolean Axioms on the back of the exam to be useful.

(Part A) How many 2-input 3-value functions are commutative?

(Part B) Of the commutative 2-input 3-value functions, how many of them have an identity element?

(Part C) Using what you know about the identity and commutativity axioms for Boolean algebra, as well as the fact that 0 and 1 are the identity elements for  $+$  and  $*$  respectively, fill in the above 2 tables.

(Part D) What is the number of valid 3-valued Boolean algebras? Please explain your reasoning (e.g. justify your answer by addressing the Boolean axioms).

## 2 (Tabulation Method)

(Part A) Below, the tables generated by the Tabulation Method for three unknown functions have been partially specified. However, they are not like the normal tables. Instead of showing the specific minterms and implicants included in each group, you are only given the quantity of each type of minterm and/or implicant that is included in the function.

For each function, complete the table by filling in each blank entry with the number of implicants that would populate it in the normal Tabulation Method table. Furthermore, in each case below, it is possible to identify the function (or a very small set of functions) which can produce the original table. In each case, please give a Sum-of-Products implementation and a behavioral description of the function. (Hint: Can you find any spatial relationships between minterms in the different groups?)

Please keep in mind that in the question below “i-subcubes” corresponds to “column-(i+1)” in the exposition of your book.

i.

| Group | 0-Subcubes | 1-Subcubes | 2-Subcubes | 3-Subcubes |
|-------|------------|------------|------------|------------|
| $G_0$ | 0          |            |            |            |
| $G_1$ | 3          |            |            |            |
| $G_2$ | 0          |            |            |            |
| $G_3$ | 1          |            |            |            |

Sum-of-Products:

Behavioral Description:

ii.

| Group | 0-Subcubes | 1-Subcubes | 2-Subcubes | 3-Subcubes |
|-------|------------|------------|------------|------------|
| $G_0$ | 0          |            |            |            |
| $G_1$ | 0          |            |            |            |
| $G_2$ | 3          |            |            |            |
| $G_3$ | 1          |            |            |            |

Sum-of-Products:

Behavioral Description:

iii.

| Group | 0-Subcubes | 1-Subcubes | 2-Subcubes | 3-Subcubes |
|-------|------------|------------|------------|------------|
| $G_0$ | 0          |            |            |            |
| $G_1$ | 2          | 2          |            |            |
| $G_2$ | 1          |            |            |            |
| $G_3$ | 1          |            |            |            |

Sum-of-Products:

Behavioral Description:

### Exposition: The 5-Summand Adder

In this and subsequent parts of the question, you will be analyzing a 5-Summand Adder. To be clear, this is not a circuit that adds two 5-bit numbers to produce another 5-bit number. Rather, it takes in 5 numbers (summands),  $v, w, x, y$  and  $z$ , and computes their sum. Normally, one would need four 2-Summand Adders to compute this sum, but the 5-Summand adder does this addition in a single scoop. At bit position  $i$ , the 5 related summand bits,  $v_i, w_i, x_i, y_i$ , and  $z_i$ , are summed. The result can obviously be as large as 5, which means that the single carry-out bit  $C$  is not sufficient to capture the nature of the ripple effect the sum at bit  $i$  has on its neighbors. Thus, a second carry-out,  $DC$  (Double Carry), is needed, which will be propagated to position  $i + 2$ . We can think of the three outputs,  $DC$ ,  $C$ , and  $S$  (the Sum), as the different bits in what is essentially a 3-bit count of ones, where the sum is the concatenation  $(DC)(C)(S)$ .

Thus we see that in fact, not only must the 5-Summand Single-bit adder at position  $i$  take in the 5 summand bits from position  $i$  of the summands, but it must also take in two carries –  $C_{in}$ , from the  $C$  output of the adder at position  $i - 1$ , and  $DC_{in}$  from the adder at position  $i - 2$ . The circuit must then compute  $v_i + w_i + x_i + y_i + z_i + C_{(i-1)} + DC_{(i-2)}$ . Since the new maximum digital sum of 7 can still fit in 3 bits, no additional carries are necessary.

(Part B) Now that we have explained the 5-Bit Adder, we want to think about how we would generate the boolean functions for each of its three outputs. We will start with the *Sum*. As you may see, generating the set of Prime Implicants for a 7-variable function can be quite tedious, but because of the nature of the *Sum* function, a recognizable pattern is present in the set of prime implicants that constitute it. In the space below, please describe these implicants. State clearly **which rows and columns in the Tabulation method table the Prime Implicants will end up in**, and also **how many Prime Implicants will occupy each entry**. (Refer to Part A to refresh your idea of this table.) Give an example of a Prime Implicant that would occur in each of the entries you specified. (You may leave your answer in a combinatorial form – no need to compute lengthy factorials.)

(Part C) Next, we will consider the  $DC$  output of the 5-Bit Adder. Under what conditions does the 5-Summand Adder produce a  $DC$  value of 1? How does this information determine which Prime Implicants will be included in the expression for  $DC$ ? Again, **identify the row(s) and column(s) in the Tabulation method table the Prime Implicants will end up in, state how many Implicants will occupy that entry, and give an example of a Prime Implicant that would occur in each of the entries you specified.** (Remember: You need only concern yourself with table entries in which Prime Implicants appear.)

(Part D) Finally, we will consider the  $C$  function. Its computation is more complex than one might initially think. For the standard 2-Summand Adder, the carry-out is 1 whenever the sum is greater than 1, but for a 5-Summand Adder the generalization of this intuition requires careful thought. Again, one must think of  $C$  as the middle bit in the 3-Bit sum. With this in mind, under what conditions will  $C$  be 1? How does this information determine which Prime Implicants will be included in the expression for  $C$ ? **Describe again the locations in the Tabulation method table that the Prime Implicants will end up in, and give an example of a Prime Implicant that would occur in each of the entries you specified.**

### 3 (Karnaugh Maps)

One well-known issue with Karnaugh Maps that was discussed in class is that they cannot be easily extended to more than 4-variables. However, as you will learn in the next two weeks, state encoding heuristics are useful in generating larger cubes of 2, and the benefits of the heuristics may not be immediately obvious unless a function has variables well in excess of 4.

With this in mind, one CSE 140 professor has decided to give his students a problem set in which they need to derive several minimal sum-of-product implementations of several 6-variable functions, with variables  $a, b, c, d, e$ , and  $f$ . Several of these poor students have come to you begging for help, and because you have nothing better to do on a Tuesday night, you elect to help them. However, being a student of academic integrity, you elect not to actually give them the answers, but to instead give them hints as to how to potentially simplify the problem.

(Part A) One thing that you notice is that, if certain minterm patterns never occur in the function, then the corresponding row labels or column labels could be eliminated, resulting in the 6-variable K-map being reduced from an 8-by-8 table. For example, if you know that the minterms  $a'b'c'xxx$ ,  $a'b'cxxx$ ,  $xxde'f$ , and  $xxd'e'f$  (where  $x$  denotes an unspecified bit pattern) never occur in the function, those with the row labels 000 and 001 do not need to be drawn; neither do the column labels 101 and 001 need to be drawn, thus resulting in a 6-by-6 table. The simplified Karnaugh Map with the aforementioned row and column eliminations is shown below.

| abc\def | 010 | 000 | 100 | 110 | 111 | 011 |
|---------|-----|-----|-----|-----|-----|-----|
| 010     |     |     |     |     |     |     |
| 011     |     |     |     |     |     |     |
| 111     |     |     |     |     |     |     |
| 101     |     |     |     |     |     |     |
| 100     |     |     |     |     |     |     |
| 110     |     |     |     |     |     |     |

One of the students notices that the Karnaugh map shown above will not facilitate the circling of all cubes of 2 since some of the Hamming adjacent minterms that may be useful in constructing implicants fail to appear next to each other on the same table (for example  $abcxxx$  and  $abc'xx$ ). What is another set of possibly useful adjacent minterms that fail to lie next to each other on the table shown above?

(Part B) Despite the student's observation, you realize that certain minterm restrictions can be applied such that all Hamming adjacent minterms will be next to each other on the same table. Please list one set of possible minterm restrictions, and write down the remaining column labels on the blank spaces in the Karnaugh Map below. Also, please provide your reasoning as to why the problem in (Part A) will not manifest itself with your new minterm restrictions.

| abc\def | _____ | _____ | _____ | _____ | _____ | _____ |
|---------|-------|-------|-------|-------|-------|-------|
| _____   |       |       |       |       |       |       |
| _____   |       |       |       |       |       |       |
| _____   |       |       |       |       |       |       |
| _____   |       |       |       |       |       |       |
| _____   |       |       |       |       |       |       |
| _____   |       |       |       |       |       |       |

(Part C) Upon presenting the students your modified Karnaugh Map, they are now in a furious but furtive debate about whether it can effectively enable one to circle cubes of size 4. One of the students believes that four adjacent minterms can be circled only if they are on the same row (an example of which is shown in the bottom row of the table below), while another student believes that four adjacent minterms can be circled only if they are on the same column (an example of which is shown in the leftmost column of the table below). A third student believes that adjacent minterms can only be circled if they span 2 rows and 2 columns (an example of which is shown in the top right corner of the table below), while a fourth student believes that the Karnaugh Map above is essentially useless, since no cubes of 4 can be circled. Finally, a fifth student believes that all cubes of 4 can be circled. Which of the students is right? Please provide your reasoning below.

| abc\def | _____ | _____ | _____ | _____ | _____ | _____ |
|---------|-------|-------|-------|-------|-------|-------|
| _____   | 1     |       |       | 1     | 1     |       |
| _____   | 1     |       |       | 1     | 1     |       |
| _____   | 1     |       |       |       |       |       |
| _____   | 1     |       |       |       |       |       |
| _____   |       |       |       |       |       |       |
| _____   |       | 1     | 1     | 1     | 1     |       |

#### 4 (Setting up Quine-McCluskey Prime Implicant Tables)

This question concerns the validity of various initial Quine-McCluskey tables as they presumably stand at *setup time*. As you are aware, the minterms contained by various implicants are restricted by certain properties of Karnaugh Maps, and you are tasked with using them to determine the validity of the following Quine-McCluskey tables.

(Part A) Given that the following Quine-McCluskey tables were all generated upon completion of the Tabulation Method on Karnaugh Maps with **no** *don't-cares* for their respective functions, for each table, tell us if these tables can represent a valid initial state (i.e. prior to any essentiality, row/column dominance applications). If you believe that no Karnaugh Maps can give rise to such initial prime implicant tables, please provide a brief reasoning. For the ones that you believe are valid, please draw a Karnaugh Map that can generate such a table.

|    | $m_i$ | $m_j$ | $m_k$ | $m_l$ | $m_n$ | $m_o$ |
|----|-------|-------|-------|-------|-------|-------|
| P0 | ✓     | ✓     | ✓     |       |       |       |
| P1 |       |       |       | ✓     | ✓     |       |
| P2 |       |       |       |       | ✓     | ✓     |
| P3 | ✓     |       |       | ✓     |       |       |

|    | $m_i$ | $m_j$ | $m_k$ | $m_l$ | $m_n$ | $m_o$ | $m_p$ |
|----|-------|-------|-------|-------|-------|-------|-------|
| P0 | ✓     | ✓     | ✓     | ✓     |       |       |       |
| P1 |       | ✓     |       | ✓     |       |       |       |
| P2 |       | ✓     |       | ✓     | ✓     |       | ✓     |
| P3 |       |       | ✓     | ✓     |       | ✓     | ✓     |

|    | $m_i$ | $m_j$ | $m_k$ | $m_l$ |
|----|-------|-------|-------|-------|
| P0 | ✓     | ✓     |       |       |
| P1 |       | ✓     | ✓     |       |
| P2 |       |       | ✓     | ✓     |
| P3 | ✓     |       |       | ✓     |

(Part B) Now we will extend our analysis to Quine-McCluskey tables that presumably emanate from Karnaugh Maps with *don't cares* embedded. Please let us know which of the following Quine-McCluskey tables are valid after the *don't-care* minterms have been eliminated. For the ones that are not, please provide a brief reasoning. For the ones that are, please draw a Karnaugh Map that can generate such a table.

|    | $m_i$ | $m_j$ | $m_k$ | $m_l$ | $m_n$ | $m_o$ | $m_p$ |
|----|-------|-------|-------|-------|-------|-------|-------|
| P0 | ✓     | ✓     | ✓     |       |       |       |       |
| P1 | ✓     | ✓     |       | ✓     |       |       |       |
| P2 |       |       |       | ✓     |       | ✓     | ✓     |
| P3 |       |       |       |       | ✓     |       |       |

|    | $m_i$ | $m_j$ | $m_k$ | $m_l$ | $m_n$ | $m_o$ |
|----|-------|-------|-------|-------|-------|-------|
| P0 | ✓     | ✓     |       |       |       |       |
| P1 | ✓     | ✓     |       |       |       |       |
| P2 |       |       | ✓     | ✓     |       |       |
| P3 |       |       |       |       | ✓     | ✓     |

|    | $m_i$ | $m_j$ | $m_k$ | $m_l$ | $m_n$ | $m_o$ |
|----|-------|-------|-------|-------|-------|-------|
| P0 | ✓     | ✓     | ✓     |       |       |       |
| P1 | ✓     | ✓     | ✓     |       |       |       |
| P2 |       |       |       |       | ✓     | ✓     |
| P3 |       |       |       | ✓     | ✓     |       |