Daniel **Fitz**
(43961229)

University Of Queensland

CSSE4010 – Digital System Design

ENGG2800 Lecture Notes

Table of Contents

List of Tables

List of Figures

What this subject covers

- Combinational Circuit Design
  - Minimisation by manipulation of Boolean expressions
  - Two level and multilevel implementation of logic
  - MSI building blocks: multiplexers, decoders, ROM's and PLA's, parity blocks, etc
  - Arithmetic circuits
  - Timing problems in combinational circuits, hazards
- Sequential Circuit Design
  - Revised basic latches and flip-flops (FF), latch / FF timing and triggering
  - Sequential (Finite State Machine) advanced design methods
  - MSI sequential parts: registers, shift registers, counters, memories, special functions
  - Principles of bus design
  - Controller-data path design methodology
  - Design for testability
  - Computer synthesis and optimization tools
- VHDL – VHSIC Hardware Description Language
  - A Language for describing digital systems
  - Concepts of digital circuit simulation and synthesis
  - Skills with VHDL in simulation, synthesis and test → (Register Transfer Level) RTL design in VHDL
- Technology
  - Field Programmable Gate Arrays

Introduction to moden design methodology

`![FPGA](sem2-2017/csse4010/FPGA.png)`$_{50}$

Modern FPGAs

- Millions of usable gates
  - Memory blocks on chip
  - DSP blocks
  - Processors (hard and soft cores)
- Integrated development
  - From 'C' or VHDL
  - From Matlab and Simulink
- Ready to compete with ASICs (Application Specific Integreated Circuits)

How to succeed?

- Structured design methodology
  - Start at high level of abstraction
  - Start simulation as early as possible
  - Make prototypes on FPGAs
- Integrated CAD tools
  - Simulation, synthesis, test, emulation, documentation
- Knowledgeable designers

Logic Minimization with Karnaugh Maps

Circuit Minimization

- Algebraic manipulations can be used to simplify Boolean expressions
  - As we've seen, this process is not always easy

• Karnaugh maps (K-maps) provide an easy and visual technique for finding the minimum cost SOP (or POS) form for a Boolean expression

- This technique has limitations, i.e. works for number inputs less than 7
- Not good for CAD tools, but good for teaching the idea of simplification

Truth Tables, Minterms

Consider majority function - output D=1, if at least 2 inputs are 1

A	B	C	D	Minterm	Minterm number
0	0	0	0	A'B'C'	m_0
0	0	1	0	A'B'C	m_1
0	1	0	0	A'BC'	m_2
0	1	1	1	A'BC	m_3
1	0	0	0	AB'C'	m_4
1	0	1	1	AB'C	m_5
1	1	0	1	ABC'	m_6
1	1	1	1	ABC	m_7

$$D = \Sigma m(3,5,6,7) = A'BCAB'C + ABC' + ABC$$

The Combining Property

• Recall the combining property

$$xyxy' = x(y + y') = x$$

• Example:

$$f = x1'x2'x3'x1'x2x3' + x1x2'x3' + x1x2'x3$$
$$= m0 + m2 + m4 + m5$$

- Minterms m0 and m2 differ in only one variable (x2)
  - m0 and m2 can be combined to get x1'x3'
  - Reduced fan in and reduced number of gates
- Hence, f = x1'x3' + x1x2' (still SOP but not canonical)

Visualizing the Combining Property

$$f(x1, x2) = \sigma m(0,1,3)$$

x1\x2	0	1
0	1	1
1	0	1

Minimum form: f=x1'+x2

3 Variable Map

$$f(A, B, C) = \Sigma m(1,2,6,7)$$

A\BC	00	01	11	10
0	0	1	0	(lightblue) 1
1	0	0	1	(lightblue) 1

Minimum SOP is:

$$f = ABBC' + A'B'C$$

> Gray Code: any two consecutive numbers differ in only a single bit

A\BC	00	01	11	10
0	0	0	1	1
1	1	1	0	0

$$f = A'B + AB'(A \oplus B)$$

A\BC	00	01	11	10
0	0	1	1	0
1	1	0	0	1

The formal Karnaugh Map Method

1) Choose a 1 element
2) Find all maximal groups of 1's adjacent to that element
   - Note: "box" must be a power of 2 in size
1) Repeat steps 1-2 for all 1 elements
2) Select all boxes for which a 1 is "covered" by only that box
   - These boxes are essential!
1) For all 1's not covered by the essential boxes, select the smallest number of other boxes that cover them
   - In case of a choice, select the largest box!

4 Variable Maps

$$f(A, B, C, D) = \Sigma m(0, 1, 2, 3, 6, 8, 9, 11, 13, 14)$$

AB\CD	00	01	11	10
00	1	1	1	1
01	0	0	0	1
11	0	1	0	1
10	1	1	1	0

$$f = AC'D + BCD' + B'C' + B'D + A'B'$$

Terminology

- An **implicant** is a product term in an SOP expression (or a sum term in POS expression)
  - Implicants are always rectangular in shape and the number of 1's covered is a power of 2
- A **prime implicant** is an implicant that is not fully contained in some other larger implicant

`![Prime Implicant](sem2-2017/csse4010/implicant.png)`75

Essential Prime Implicants

- An **essential prime implicant** is a prime implicant that contains a 1 not included in any other prime implicant
  - The minimum Boolean expression *must* use this term
- A **cover** is a collection of implicants that accounts for all valuations in which the function is "on" (e.g. 1)

Dont Care Conditions

- Many times there are incompletely specified conditions
  - Valuations that can never occur, or for which we "don't care what the device does"
- Modeling such a device requires us to specify don't care conditions in those instances
  - Use X as a value to indicate we don't care what happens
- Don't care situations are often called **incompletely specified functions**

$$f(A, B, C, D) = \Sigma m(1, 5, 8, 9, 10)d(3, 7, 11, 15)$$

AB\CD	00	01	11	10
00	0	1	X	0
01	0	1	X	0

11	0	0	X	0
10	1	1	X	1

$$f = AB' + A'D$$

Karnaugh Map Method Restated

1) Choose an element from the "on" set

2) Find all maximal groups (prime implicants) of "on" elements and X elements adjacent to that element

> Note 1: prime implicants are always a power of 2 in size
> Note 2: do not feel compelled to include X's – use them only when they provide a larger implicant

1) Repeat steps 1-2 for all elements in the "on" set

2) Select all *essential* prime implicants

3) For all elements of the "on" set not covered by the essential prime implicants, select the smallest number of prime implicants that cover them