Theory Of Computer Arithmetic

Disclaimer

This follows the lessons from Mr. Oudjida given in course, major additions, changes or rearrangements can be applied on the course for the sake of clarity, better explainations or just subjective opinions, written by HADIOUCHE Azouaou.

> Some extra information will be added. To seperate the contents of the course to actual additions or out of context information, a black band will be added by its side like the one englobing this comment.

USE THIS AT YOUR OWN RISK.

Contents

Chapter 1

Classical logic

The most basic part of executing a computation on a machine is to describe the most basic information, which is true/false, and then compose them into a statement or a proposition.

informally, given a sentence, it is said to be a *statement* if
- its declaritive, either affirmative or negative.
- its possible truth values are true or false.
- its verifiable in reality.

on those statements, we have some rules to give them truth values
- law of identity: $A = A$ is a true statement.
- law of non-contradiction: $\neg(A \land \neg A)$ is false statement.
- law of excluded middle: either $\neg A$ or A is true statement.

Now we will formalize calculations on boolean variables which is known as Boolean algebra, it will help us analyse and create circuits later on and also simplify them to have less components.

1.1. Boolean Algebra

Let $\mathbb{B} := \{0, 1\}$ denote the set of boolean values, which can be represented too with true/false. Any variable ta

Definition 1.1.1 (Boolean Variable): *Let x be a variable, x is said to be a* boolean variable *if it can assume values in $\mathbb{B}$. Let $f : \mathbb{B}^n \to \mathbb{B}$ a map, f is called a* boolean function.

Boolean functions will be the main study of Boolean algebra, how they can be written, expressed and modified without altering its values, the following operations will be useful for operating on boolean variables and construct functions.

Definition 1.1.2 (Boolean Operations): *Let x, y be two boolean variables, we define the operations $+, \cdot, \overline{}$ to be the logical or, and, not respectively, which have the following truth tables.*

y	x	$\overline{x}$	$x + y$	$x \cdot y$
0	0	1	0	0
0	1	1	1	0
1	0	1	1	0
1	1	1	1	1

there are some other operations that are as follows
- $x \mid y = \overline{x \cdot y}$.
- $x \otimes y = x \cdot \overline{y} + \overline{x} \cdot y$.
- $x \Rightarrow y = \overline{x} + y$.

Proposition 1.1.3 (Boolean Identites):

$\overline{\overline{x}} = x$	
$x + x = x$	$x \cdot x = x$
$x + 0 = x$	$x \cdot 1 = x$
$x + 1 = 1$	$x \cdot 0 = 0$
$x + y = y + x$	$x \cdot y = y \cdot x$
$x + (y + z) = (x + y) + z$	$x \cdot (y \cdot z) = (x \cdot y) \cdot z$
$x(y + z) = xz + yz$	$x + yz = (x + y) \cdot (x + z)$
$\overline{x + y} = \overline{x} \cdot \overline{y}$	$\overline{x \cdot y} = \overline{x} + \overline{y}$
$x + \overline{x} = 1$	$x \cdot \overline{x} = 0$

Definition 1.1.4 (Duality): *Let $f : \mathbb{B}^n \to \mathbb{B}$ be a boolean function, we define the dual of f as the map $(x_1, ..., x_n) \mapsto \overline{f(\overline{x}_1, ..., \overline{x}_n)}$. We can obtain the dual of a function f by swapping $+$ with $\cdot$, 0 with 1 and keep the variables unchanged.*

Definition 1.1.5 (Literal/Minterm/Maxterm): *Let $x_1, \ldots, x_n$ be boolean variables and $y_1, \ldots, y_n$ such that $\forall i \in [\![1, n]\!], y_i = x_i \vee y_i = \overline{x_i}$.*

- *A* literal *is a proposition in the form of x or $\overline{x}$ with x a boolean variable.*
- *A* minterm *of $x_1, \ldots, x_n$ is the product $y_1 \cdot y_2 \cdots y_n$.*
- *A* maxterm *of $x_1, \ldots, x_n$ is the sum $y_1 + y_2 \ldots + y_n$.*

Definition 1.1.6 (Conjunctive/Disjunctive Normal Form): *Let $f : \mathbb{B}^n \to \mathbb{B}$ be a boolean function, $x_1, \ldots, x_n$ boolean variables.*

- *DNF: $f(x_1, \ldots, x_n) = \sum_{i=1}^{k} X_i$ where X_i are minterms.*
- *CNF: $f(x_1, \ldots, x_n) = \prod_{i=1}^{k} X_i$ where X_i are maxterms.*

Proposition 1.1.7:

1. *the dual of a DNF is a CNF and vice versa.*
2. *Every boolean function can be written with only the defined connectives.*
3. *Every boolean function can be expressed only using one of those sets $\{+, \overline{}\}$, $\{\cdot, \overline{}\}$, $\{|\}$, we call them a complete set of connectives.*
4. *Any boolean function can be written in the CNF or DNF.*

Proof.

1. A minterm is of the form $y_1 \ldots y_n$ then its dual is $y_1 + \ldots + y_n$ which is a maxterm, now if f is in a DNF then it is the sum of minterms, the dual will become a product of maxterms which is a CNF, its easy to verify the rest.

2. Let $f : \mathbb{B}^n \to \mathbb{B}$ a boolean function, we define $g : \mathbb{B}^n \to \mathbb{B}$ as follows $(x_1, \ldots, x_n) \mapsto \sum_{i=1}^{2^n} \sigma_i(x_1, \ldots, x_n) \cdot f(x_1, \ldots, x_n)$ where $\sigma_i(x_1, \ldots, x_n)$ is defined as if we take the i in base 2, $i = \overline{i_n i_{n-1} \ldots i_1 i_0}^2$ then $\sigma_i(x_1, \ldots, x_n) = y_1 \ldots y_n$ and if $i_j = 1$ then $y_j = x_j$ otherwise $y_j = \overline{x_j}$. Notice that $\sigma_i(x_1, \ldots, x_n) = 1$ if and only if $\overline{x_n \ldots x_1}^2 = i$. So we have for $(x_1, \ldots, x_n) \in \mathbb{B}^n$, $g(x_1, \ldots, x_n) = f(x_1, \ldots, x_n)$ thus f can written only with the connectives.

3. To prove this statement, we just need to use the fact that any function can be written using only $+, \cdot, \overline{}$.
   - We have by De Morgans laws that $x \cdot y = \overline{\overline{x} + \overline{y}}$ thus $+, \overline{}$ is enough to express every function.
   - Same can be used to express $x + y = \overline{\overline{x} \cdot \overline{y}}$.
   - Now we can use the NAND to write everything, notice that $x|x = \overline{x}$ and $(x|y)|(x|y) = x \cdot y$ thus we use the previous statement and we get that $\{|\}$ is a complete set of connectives.

4. Notice that in the first statement we proved that any boolean funtion can be written in the DNF, using the same function σ_i we can construct a DNF, it will be of the form $g : (x_1, \ldots, x_n) \mapsto \prod_{i=1}^{2^n} \overline{f(x_1, \ldots, x_n)} \cdot \overline{\sigma_i(x_1, \ldots, x_n)}$.

□

1.2. Transistors

One of the biggest advancements in our modern world is the creation of a transistor, in principle the idea is simple, a transistor is simply an electrically controlled valve.

> Multiple families of transistors exist, the ones used in course are FET transistors, but due to their unfamiliarity, I would prefer to use BJT transistors that have a slightly different workflow but are almost the same, one regulates current and the other votage.