Chapter 2

Basic Definitions and Concepts

Most of the terminology used in this book is standard and in common use in the synthesis community [54, 13, 14]. This chapter is devoted to the definition of the nontrivial terminology and an elucidation of some of the basic concepts.

2.1 Two-Valued Logic

A binary variable is a symbol representing a single coordinate of the Boolean space (e.g. \(a\)). A literal is a variable or its negation (e.g. \(a\) or \(\bar{a}\)). A cube is a set \(C\) of literals such that \(x \in C\) implies \(\bar{x} \notin C\) (e.g., \(\{a, b, c\}\) is a cube, and \(\{a, \bar{a}\}\) is not a cube). A cube (sometimes called a product term) represents the conjunction, i.e. the Boolean product of its literals. The trivial cubes, written 0 and 1, represent the Boolean functions 0 and 1 respectively. A sum-of-products expression is the disjunction, i.e. a Boolean sum, \(f\), of cubes. For example, \(\{\{a\}, \{b, \bar{c}\}\}\) is an expression consisting of the two cubes \(\{a\}\) and \(\{b, \bar{c}\}\).

A cube may also be written as a bit vector on a set of variables with each bit position representing a distinct variable. The values taken by each bit can be 1, 0 or 2 (don’t-care), signifying the true form, negated form and non-existence respectively of the variable corresponding to that position. A minterm is a cube with only 0 and 1 entries. Cubes can be classified based on the number of 2 entries in the cube. A cube with
2.1. TWO-VALUED LOGIC

\[ \begin{array}{cccccc}
1-00 & (\text{inp1, inp3}) & & & & 1-00 & 101 & 001 & 100 \\
111- & (\text{inp1, inp3}) & & & & 111- & 110 & 010 & 110 \\
101- & (\text{inp1, inp2}) & & & & 101- & 110 & 100 & 101 \\
1-01 & (\text{inp1, inp3}) & & & & 1-01 & 101 & 100 & 101 \\
0--- & (\text{inp1}) & & & & 0--- & 110 & 100 & 111 \\
\end{array} \]

(a)

\[ \begin{array}{cccccc}
1-00 & 100 & & & & 1-00 & 101 & 001 & 100 \\
111- & 011 & & & & 111- & 110 & 010 & 111 \\
101- & 110 & & & & 101- & 110 & 100 & 110 \\
1-01 & 101 & & & & 1-01 & 101 & 100 & 101 \\
0--- & 111 & & & & 0--- & 110 & 100 & 111 \\
\end{array} \]

(b)

Figure 2.1: An example of a symbolic cover and its multiple-valued representation

\[ k \] entries or bits which take the value 2 is called a \( k \)-cube. A minterm thus is a 0-cube. A cube \( c_1 \) is said to cover (contain) another cube \( c_2 \), if \( c_1 \) evaluates to 1 for every minterm for which \( c_2 \) evaluates to 1. A super-cube of a set of cubes is defined as the smallest cube containing all the minterms contained in the set of cubes.

The on-set of a function \( j \) is the set of minterms for which the function evaluates to 1, the off-set of \( j \) is the set of minterms for which \( f \) evaluates to 0, and the don't-care set or the DC-set is the set of minterms for which the value of the function is unspecified. A function which does not have a DC-set is a completely-specified function. A function with a non-empty DC-set is termed incompletely-specified.

An implicant of \( j \) is a cube that does not contain any minterm in the off-set of \( j \). A prime implicant of \( j \) is an implicant which is not contained by any other implicant of \( f \), and which is not entirely contained in the DC-set.

A two-level cover for a Boolean function is a set of implicants which cover all the minterms in the on-set, and none of the minterms in the off-set. A cover is prime if it composed entirely of prime implicants. A cover is irredundant if removing any single implicant results in a set of implicants that is not a cover. A minimum cover is a cover of minimum cardinality over all the possible covers for the function.

A minterm \( m_1 \) is said to dominate another minterm \( m_2 \) if for each bit position in which \( m_2 \) has a 1, \( m_1 \) also has a 1. If \( m_1 \) dominates \( m_2 \), we write \( m_1 \succ m_2 \) or \( m_2 \prec m_1 \).

The distance between two minterms is defined as the number of bit positions they differ in.