

## Study Guide, Patterson & Hennessy, Appendix Sections B.2 & B.3

### Section B.2

Truth tables (tt) for two inputs and one output are a review from earlier computer courses. Here we see that there can be many inputs and many outputs.

This section introduces axioms for Boolean algebra. You do not need to memorize them in the sense that you would ever be called on to list all 12. However, given the names, you should be able to write the equations. For example, “Write the law that says that OR distributes over AND.”

The big idea related to Boolean algebra is this: Everything equation in boolean variables can be proven in two ways. In the first way, we create the tt of the left hand side of the equation, and the tt of the right hand side. If the results are the same, then the equation is **true**. This is called “proof by modeling,” because the set  $\{0, 1\}$  together with the operations of AND, OR and NOT are a model for the axioms of Boolean algebra.

In the second way, we use the same style of proving equations (getting theorems) that we may have learned in high school geometry: A sequence of equations is written down, each statement following from the axioms and from previously developed theorems. The last equation of that sequence is then a theorem, and we say that it is **valid** (according to the axioms).

It is one of the beautiful results of Boolean algebra (which we state but do not prove) that every equation that is true is valid, and every equation that is valid is true. Notice that in this result, the word “true” is not being used as a synonym for 1 in the model, but means “true in the world,” even though the “world”  $\{0, 1\}$  is a very small world indeed.

There is an error on page B-7. Three lines above the title **Gates** is an equation  $E = \dots$ . Replace it with  $E = (A \wedge B \wedge C) * (A + B + C) * (A * B * C)$ . [I’m sad that Verilog uses  $\wedge$  to mean EXCLUSIVE OR because mathematicians use  $\wedge$  to mean AND; reserving something like  $\oplus$  to mean EXCLUSIVE OR.

And speaking of terminology, we use the term “adequate” for a set of operations to mean that you can get all boolean functions from just those. Your text introduces the term “universal” to mean that a single gate is adequate.

### Section B.3

We use the term “disjunctive normal form” (dnf) in class. Your text calls it more simply “sum of products.” Here we learn that we could do everything in terms of products of sums instead (so-called “conjunctive normal form,” cnf), and we learn of the benefit of DeMorgan’s laws for converting any boolean expression into a normal form.

I prefer to use dnf over cnf because there is a hardware element that corresponds directly to it, which is pictured on page B-13, called a **Programmable Logic Array**, or PLA.

This section explains how “**don’t care**” values in a truth table can simplify the logic of a circuit. You will become adept at handling “don’t cares” both in inputs and in outputs.