CS 251 — Lecture 2

Bartosz Antczak *Instructor: Stephen Mann* *January 5, 2017*

2.1 Computer performance

When measuring computer performance, we focus on two factors:

- *Response time:* time between the start and completion of a task. Some factors that affect this include the speed of the computer's clock, complexity of the instruction set, and the efficiency of compilers
- *Throughput:* the total amount of work done in a given time

Despite this, it isn't easy to measure computer performance accurately.

Clock speed

Clock speed is measured in Hertz (Hz). Faster clock speed means more instructions executed per second, and one instruction is executed per clock cycle.

Comparing speeds of computers

We can use benchmarks (which are a standard set of programs and data chosen to measure performance) as a basis to compare the speeds of different computers.

2.1.1 Logic blocks

We have two types:

- Combinational (without memory): takes in n inputs and runs them through a combinational circuit and produces m outputs
- Sequential (with memory): takes in inputs, runs through a combinational circuit and outputs some output and stores the rest in storage

2.1.2 Specifying input/output behaviour

We use truth tables to specify outputs for each possible input combination.

Boolean algebra

In our truth tables, we'll be working with OR ($+$), AND ($\cdot$), and NOT ($\neg$):

- OR: has result 1 if and only if either operand has value 1
- AND: has result 1 if and only if both operands have value 1 ($A \cdot B$ is often written as AB)
- NOT: negates the operand (0 becomes 1, and 1 becomes 0)

Example 2.1.1. *The truth table for AND with two variables*

A	B	AB
0	0	0
0	1	0
1	0	0
1	1	1

These variables (i.e., A, $\bar{A}$) are called *literals*.
A conjunction of literals (i.e., $A\bar{B}C$) is called a *minterm*, and a disjunction of literals (i.e., $A + B$) is called a *maxterm*.
In these truth tables, any Boolean function can be represented as a sum of products (OR of ANDs) of literals.

Don't cares in truth tables

For outputs that we don't care what value it has, we represent them with an 'X'. For instance, if we apply function F on literals A, B, C

A	**B**	**C**	**F**
0	0	X	0
0	1	X	1
1	X	X	X

This method is used to simplify truth tables. But be cautious, because we may break the table. For instance, if we swap 1 and X in the second row of the previous table, we have a paradox:

A	**B**	**C**	**F**
0	0	X	0
0	X	1	1
1	X	X	X

Notice that the first row could be 001 with a result of 0, but 001 could also be in the second row with a result of 1 — not allowed!