CS 211 Homework 7

Circuit simulator

Fall 2020

Introduction

You have to write a circuit simulator. One of the inputs to your program will be a circuit description file that will describe a circuit using various directives. Your program will print the output of the circuit for all possible input values.

Circuit Description Directives

The input variables used in the circuit are provided using the `INPUTVAR` directive. The `INPUTVAR` directive is followed by the number of input variables and the names of the input variables. All the input variables will be named with capitalized identifiers. An identifier consists of at least one character (A-Z) followed by a series of zero or many characters (A-Z) or digits (0-9). For example, some identifiers are `IN1`, `IN2`, and `IN3`. An example specification of the inputs for a circuit with three input variables: `IN1`, `IN2`, `IN3` is as follows:

```
INPUTVAR 3 IN1 IN2 IN3
```

The outputs produced by the circuit is specified using the `OUTPUTVAR` directive. The `OUTPUTVAR` directive is followed by the number of outputs and the names of the outputs.

An example specification of the circuit with output `OUT1` is as follows:

```
OUTPUTVAR 1 OUT1
```

The circuits used in this assignment will be built using the following building blocks: `NOT`, `AND`, `OR`, `NAND`, `NOR`, and `XOR`.

The building blocks can produce temporary variables as outputs, and can use either the input variables or temporary variables as input. Output variables will never be used as inputs in a building block.

All the temporary variables will also be named with lower case identifiers (i.e., `temp1`, `temp2`, `temp3`, ...).

The specification of each building block is as follows, where $\cdot$ denotes logical AND, $+$ logical OR, and $\oplus$ XOR:

Block	Arguments	Semantics
`NOT`	`IN1 OUT1`	$\mathrm{OUT}_1 = \overline{\mathrm{IN}_1}$
`AND`	`IN1 IN2 OUT1`	$\mathrm{OUT}_1 = \mathrm{IN}_1 \cdot \mathrm{IN}_2$
`OR`	`IN1 IN2 OUT1`	$\mathrm{OUT}_1 = \mathrm{IN}_1 + \mathrm{IN}_2$
`NAND`	`IN1 IN2 OUT1`	$\mathrm{OUT}_1 = \overline{\mathrm{IN}_1 \cdot \mathrm{IN}_2}$
`NOR`	`IN1 IN2 OUT1`	$\mathrm{OUT}_1 = \overline{\mathrm{IN}_1 + \mathrm{IN}_2}$
`XOR`	`IN1 IN2 OUT1`	$\mathrm{OUT}_1 = \mathrm{IN}_1 \oplus \mathrm{IN}_2$

Describing Circuits using the Directives

It is possible to describe any combinational circuit using the above set of directives. For example, the circuit $\mathrm{OUT}_1 = \mathrm{IN}_1 \cdot \mathrm{IN}_2 + \mathrm{IN}_1 \cdot \mathrm{IN}_3$ can be described as follows:

```
INPUTVAR 3 IN1 IN2 IN3
OUTPUTVAR 1 OUT1
AND IN1 IN2 temp1
AND IN1 IN3 temp2
OR temp1 temp2 OUT1
```

Note that `OUT1` is the output variable. `IN1`, `IN2`, and `IN3` are input variables. `temp1` and `temp2` are temporary variables.

You can assume:

- The circuit descriptions given will be sorted.
  - The first line gives the input variable(s).
  - The second line gives the output variable(s).
  - Every temporary variable occurs as an output variable before occurring as an input variable.
- A temporary variable can be an output variable in at most one directive.

Example Execution

Suppose our input has this description for the circuit $\text{OUT}_1 = \text{IN}_1 \cdot \text{IN}_2 + \text{IN}_1 \cdot \text{IN}_3$:

```
INPUTVAR 3 IN1 IN2 IN3
OUTPUTVAR 1 OUT1
AND IN1 IN2 temp1
AND IN1 IN3 temp2
OR temp1 temp2 OUT1
```

Then your program should take the circuit description file as an argument and produce the following output:

```
$ ./first circuit.txt
0 0 0 0
0 0 1 0
0 1 0 0
0 1 1 0
1 0 0 0
1 0 1 1
1 1 0 1
1 1 1 1
```

The output of the first three columns are the input variables `IN1`, `IN2`, and `IN3` respectively (as they iterate through all possible values). And the last column denotes the output variable `OUT1`.

The printed values of the input and output variables should be space separated and be in the same order (left to right) as listed in the `INPUTVAR` and `OUTPUTVAR` directive.

The order of the lines of input (top to bottom) should match the normal numeric order when counting in binary.

Submission

Please submit a tar file named `hw7.tar` on Canvas that contains a `hw7` directory. The `hw7` directory in your tar file should contain the Makefile and your source code:

```
hw7
├── Makefile
└── first.c
```