CENG Final Exam– Due 12/8/2020

1. Introduction

You have just landed a position at Old Fashion Computing and they have decided to put you on the B18 project. The B18 is a simple FPGA (Field-Programmable Gate Array). An FPGA is an integrated circuit designed to be configured by a customer or a designer after manufacturing, they usually contain an array of programmable logic blocks, and a hierarchy of reconfigurable interconnects that allow the blocks to be "wired together". The logic blocks can be configured to perform complex combinational functions, or merely be simple logic gates like `AND` and `XOR`. Gate array chips like this are widely used for putting custom circuits on a chip because most of the work (depositing the gate array on the chip) is independent of the circuit to be implemented. Only the "wiring" is specific to each design. The B18 will be consisting of $m \times n$ array of two-input `NAND` gates. Your supervisor has decided that your first task will be to develop a simulation for the B18. The plan is to let the B18 be part of a development board where they plan to let the B17 be an embedded processor.

2. Grid Array Architecture

The B18 is a grid array of $m \times n$ two-input `NAND` gates. The circuit is contained on a chip, that has *j* input pins and *k* output pins (see figure 1).

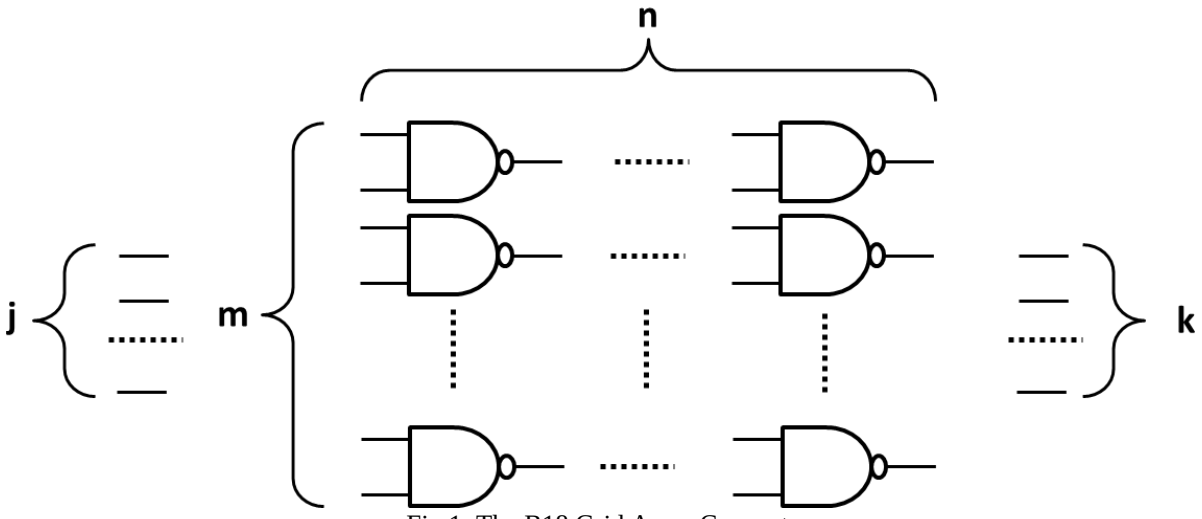


Fig 1: The B18 Grid Array Concept.

The values of *j, k, m* and *n* are compile time parameters of the simulation, where the default values depend on what grade you aim for (see later text).

2.1 Problem description

You will be tasked with creating the simulation of the B18. Another team will work on the hardware description language (HDL), the tool that will be used by the consumer to configure the B18. The output from the HDL, will a "wiring" list of each wire which specifies an input and an output. These "wiring" lists are currently just simple text files, which are read at program start. The other team is behind schedule and can currently not produce these files, but the format is specified so you will have to create them for your own testing. The simulation should display the output from every possible input.

3. Simulator Requirements

3.1. Current simulator

The current concept-drawing for the B18 simulation looks like this:

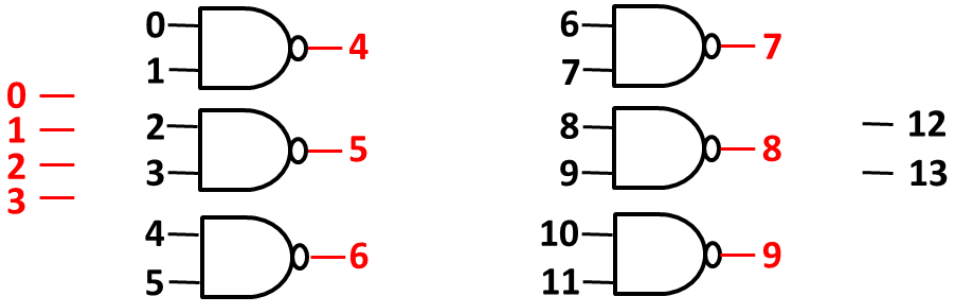


Fig 2: Schematic of a B18 with *j*=4, *k*=2, *m*=3 and *n*=2

The numbers shows the numbering of the different input and output wires. No more work has currently been done on the simulation but the hardware and HDL team are currently working on their implementations.

3.3. Input and Output formats

The input "wiring" lists should be simple text files, which shows the input (either one of the *j* input pins or the output of some `NAND` gate) and the output (either one of the *k* output pins or an input to some `NAND` gate) of a wire, for example:

```
0   0
1   1
2   2
2   3
3   4
3   5
4   6
4   7
5   10
6   11
7   12
9   13
```

Is the “wiring” list for the following circuit:

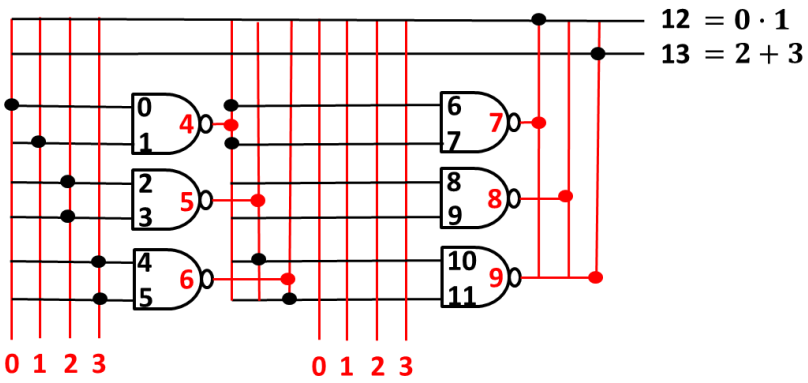


Fig 3: Circuit with j=4, k=2, m=3 and n=2 and `circuit1.txt` as input file.

And it will provide the following output information:

```
-----------------------------------
  0  |  1  |  2  |  3  | 12  | 13  |
-----------------------------------
  0  |  0  |  0  |  0  |  0  |  0  |
  0  |  0  |  0  |  1  |  0  |  1  |
  0  |  0  |  1  |  0  |  0  |  1  |
  0  |  0  |  1  |  1  |  0  |  1  |
  0  |  1  |  0  |  0  |  0  |  0  |
  0  |  1  |  0  |  1  |  0  |  1  |
  0  |  1  |  1  |  0  |  0  |  1  |
  0  |  1  |  1  |  1  |  0  |  1  |
  1  |  0  |  0  |  0  |  0  |  0  |
  1  |  0  |  0  |  1  |  0  |  1  |
  1  |  0  |  1  |  0  |  0  |  1  |
  1  |  0  |  1  |  1  |  0  |  1  |
  1  |  1  |  0  |  0  |  1  |  0  |
  1  |  1  |  0  |  1  |  1  |  1  |
  1  |  1  |  1  |  0  |  1  |  1  |
  1  |  1  |  1  |  1  |  1  |  1  |
```

3.2. Task

Create a simulation program for the B18. The values of *j*, *k*, *m* and *n* are compile time parameters, but should be set to default values until the hardware team provide more accurate parameters (see required elements). The program should start by reading in the “wiring” list (just a simple text file looking like the example above). An input to a wire is either one of the *j* input pins or the output of some `NAND` gate (marked red in the schematics above). An output is either one of the *k* output pins or an input to some `NAND` gate (marked black in the schematics above). Unused inputs are logical 1 (but you don’t need to care about that). After reading in the list the B18 simulation will evaluate and display the output for each of the 2^j possible inputs (for example of an output see previous subsection).

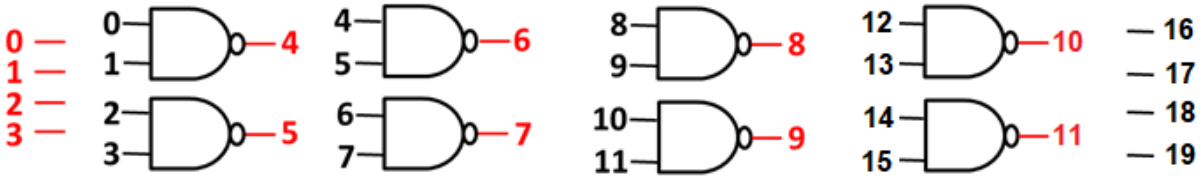
3.3 Required elements:

For a D (63 – 73%)

- Implement what is in the text, i.e. the default values for the `NAND` grid should be:
  j = 4
  k = 2
  m = 3
  n = 2
  with the `0 AND 1` on output 12, and `2 OR 3` on output 13.

For a C (73 – 83%)

- All the things needed for a D
- Implement a grid with the following default values:
  j = 4
  k = 4
  m = 2
  n = 4



- Provide the input file for wiring two sample circuits (simple txt-files):
  - An XOR gate
  - 2 to 4 decoder

For a B (83 – 93%)

- All the things for a C
- The values of j, k, m and n are compile time parameters, but should be set to default values until the hardware team provide more accurate parameters.:
  j = 8
  k = 4
  m = 10
  n = 10
- Proved the input file for wiring five sample circuits:
  - An `XOR` gate
  - 2 to 4 decoder
  - 4 to 1 multiplexor
  - Full Adder
  - Any other example circuit

For an A (93 -100%)

- Everything needed for a B
- All the input files for the four mandatory sample circuits for a B
    - An `XOR` gate
    - 2 to 4 decoder
    - 4 to 1 multiplexor
    - Full Adder
- The input file for an S/R latch (Note: you have to backpropagate the signal and you have to omit the 'forbidden' inputs).
- Any other circuit, you feel would be beneficiary to show your skill and the ability of the code.

4. Turning In Your Solution

4.1. General Information

The program must be submitted by the end of the final slot for the class (7 p.m. on 12/8). NO EXCEPTION! The files need to be tarred and gzipped prior to submission. Please empty the directory prior to tar/zip. [You will tar up the directory containing the files: `tar -czf final.tgz  final`]

Submission contents:

- Documentation
    - What level you want to be graded at.
    - Complete description of the implementation.
    - Objective files with wiring information for the appropriate grade:
        - For a D
            - The wire file provided in the text
        - For a C
            - `XOR` gate
            - 2 to 4 decoder
        - For a B
            - An XOR gate
            - 2 to 4 decoder
            - 4 to 1 multiplexor
            - Full Adder
            - Any other example circuit
        - For an A
            - An XOR gate
            - 2 to 4 decoder
            - 4 to 1 multiplexor
            - Full Adder
            - S/R latch
            - Any other example circuit

    - Complete description of the testing.
    - Complete description of what the system requirements are, how to build, how to run.
    - An instruction manual for the user. Including information on how to run the program with the example objective files, a description on how the user creates their own objective files for wiring their own logical circuits, and how to interpret the output information.
- Main programs.
- Any required external functions (ex. `*.c, *.h`).
- Any include files you use (other than the standard ones).
- You will need to produce a Makefile to build the executable.

The file which contains your `main()` function *must* be named `b18.cpp` (or `b18.py`). Your program will be invoked with the command line:

`./b18 <input-file>` or `python(3) b18.py <input-file>`

where `input-file` is the name of the objective file containing the wiring information.

5.2. Submitting Your Solution

You need to submit your folder in the D2L Dropbox.