Computer Organization and Architecture

Program 2: The NAND Array

B18 Simulator program: This program simulates the B18 wired grid array so that wiring designs can be tested before they are built in hardware. The program implements an oop -like structure in pure C wherein the NAND-grid is simulated by an array of objects designed to function like ideal NAND gates. The program reads an input file consisting of pairs of numbers assigning an output to a connection and which input it will connect to. The program then performs the simulation and outputs a formatted truth table using all the inputs and outputs. It is important to remember that the values for the NAND array are unique to a wiring diagram. Whether a wiring input will function on its non-native presets is unknown. The default values are J = 8, k = 4, M= 10, N = 10.

Compilation instructions: This program can be compiled using the included Makefile on linux or any machine that has a basic C compiler and math libraries using the command:

cc b18.c -lm

Running: The program can be run using the statement

./b18 <inputfile>

Changing Default Simulation values: After the library #includes the user will find 4 statements :

```
#define INNUM 8
#define OUTNUM 4
#define WIDTHNUM 10
#define HEIGHTNUM 10
```

These are the default simulation values referred to above. They may be defined at compile time and the program must be recompiled in order to change them.

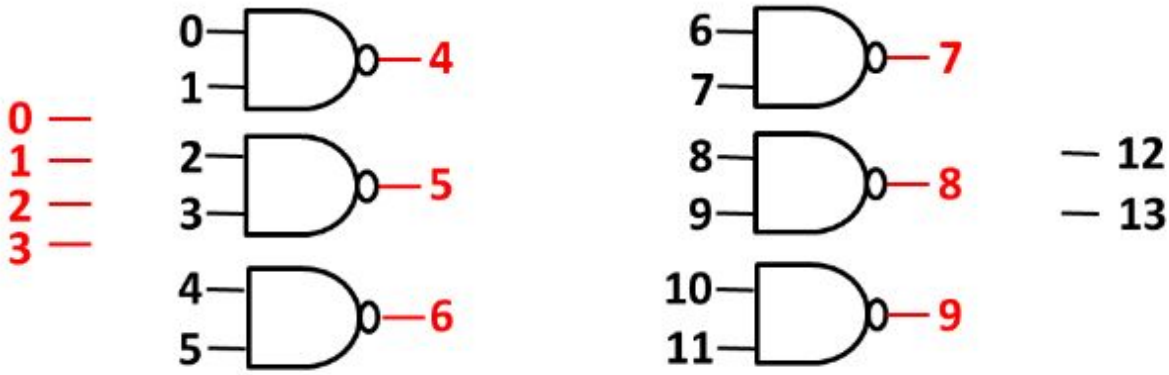
Testing: The programs output was tested against known truth tables for the three wiring diagrams included with this program.

B18 User manual

Personal usage of the b18 simulator program is predicated upon 2 things, the object file and the simulation parameters:

Compile time parameters:

The program begins at compilation where the values of INNUM, OUTNUM, WIDTHNUM and HEIGHTNUM are set in the #defines at the top of the program. These values represent INNUM= number of inputs to the program, OUTNUM = number of outputs from the program, WIDTHNUM = Width in gates of the NAND array, and HEIGHTNUM = Height of the NAND array. These values must be changed before the program is compiled for them to take effect



This is a drawing with INNUM set to 4 OUTNUM set to 2, WIDTHNUM set to 2 and HEIGHTNUM set to 3. Mind how the numbers are set upon the NAND gates as it will impact how the wiring is done. **ALL VALUES MUST BE GREATER THAN ZERO.**

Final note: no strict bar is placed on the upper limits of the these parameters and as #define's they are allowed to be any text macro string. What this means is greater freedom at greater risk. As long as the computer the program is running upon has the RAM to spare and OS to manage it the values can in theory be made as large as you would like them to be. The program will run properly up to the point when you begin overflowing integers but then again that size depends on the host machine.

Wiring:

The wiring of the circuit is what makes the program run, Any circuit can be simulated as long as it contains nothing but NAND gates and contains no Latches. A latch is when an output on the

right connects to its left into a previously used NAND gate This will not be allowed and will cause the program to exit if it is encountered. Correct functionality for that type of circuit is beyond the scope of this program.

A wiring diagram is as follows.
<output wire> <input wire>
The numbers are colored the same way they are in the diagram with outputs in red and inputs in black.
This is an example wiring for a Full 2-bit Adder circuit.

0 0
1 1
0 20
8 21
8 22
1 23
18 40
19 41
28 60
3 61
28 80
38 81
38 82
3 83
38 84
8 85
48 100
49 101
50 201
58 200

Notice the large jump in input values in the last two entries These are connected to the output wires and will be printed to the screen, in the default setting these wires will be in the 200 <-> 204 range if the user changes these values they will need to be recalculated using this equation (M*N) <-> (M*N + k)

The only other thing to note is that non-used inputs to NAND gates whose outputs are used are assumed to be in a low state.

TRUTH TABLE

A	B	A NAND B
0	0	1
0	1	1
1	0	1
1	1	0