Project GF2: Software
First Interim Report
Software Design Team 1

George Ayris	James Glanville	Andrew Holt
gdwa2	jg597	ah635
Emmanuel	Emmanuel	Emmanuel

21 May 2013

1 Introduction

This project requires the development of a logic simulation program, implemented in C++. When completed, the application will read a definition file containing a list of devices and the connections between them. It will then graphically display the values of specified monitor points in the circuit as the simulation is run. Some legacy code has been provided, but lacks a scanner, parser and GUI which will be developed in this project.

1.1 Teamwork Planning

The GUI development requires learning and reference to the wxWidgets and OpenGL packages, so makes sense for one team member to focus efforts here. The scanner and parser will require robust coding and testing, so it was decided to combine the two as a peer programming project to allow implementation and testing to be carried out by different team members.

The work is to be distributed as follows: Andrew will code the GUI, and James and George will write and test the scanner and parser. The time frame for the development is that the software should be designed by the end of Tuesday 21st May; implemented and unit tested by Tuesday 28th; allowing time for system integration and testing by 11.00am on Friday 31st May.

2 Syntax Specification

The syntax for the circuit definition files was defined and described using the following EBNF grammar:

```
DEFINITION = '{' DEVICES CONNECTIONS MONITORS INIT '}'

DEVICES = 'DEVICES' '{' device {device} '}'
device = devicename '=' devicetype [ '(' digit {digit} ')' ] ';'
devicename = letter {letter | digit}
devicetype = 'AND' | 'NAND' | 'OR' | 'NOR' | 'XOR' | 'DTYPE' | 'CLK' | 'SW'

CONNECTIONS = 'CONNECTIONS' '{' connection {connection} '}'
connection = input '<=' output ';'
input = letter {letter|digit} ['.'letter|digit{letter|digit}]
output = letter {letter|digit} ['.'letter|digit{letter|digit}]
```

```
MONITORS = 'MONITORS' '{' monitor {monitor} '}'
monitor = monitorname '<=' output ';'
monitorname = letter{letter|digit}

INIT = 'INIT' '{' {init} '}'
init = devicename '=' digit{digit} ';'
```

3 Semantics Specification

The following set of semantic descriptions was created to describe the semantics.

- A logic circuit is defined in a file that opens with "{" and closes with "}".
- Each file has four sections: "`DEVICES`", "`CONNECTIONS`", "`MONITORS`" and "`INIT`". Each section starts with the section name followed by a "{" and terminates with a "}".
- Within "`DEVICES`" there must be at least one device.
- All names and keywords are case-insensitive, i.e. `GATE1.O` and `gate1.o` are the same, and must be unique.
- A device is defined by "`devicename = devicetype(N);`", where the parentheses and parameter are only required for certain device types.
- A device name is alphanumeric and must begin with a letter.
- If the device has a variable number of inputs then the number required for the device must be specified in parentheses after the device type. The number of inputs specified must lie within the allowable range for that device type.
- Within "`CONNECTIONS`" every input (of defined devices) must be connected to a single device output.
- A connection is defined as "`input <= output;`". Where input must correspond to a defined device input and output to a defined device output.
- An input is specified by "`devicename.inputidentifier`".
- An output is specified by "`devicename.outputidentifier`".
- The clock, switch and gate outputs are accessed with "`.o`".
- The gate inputs are accessed with "`.n`" where `n` is 1 or 2 for XOR gates and 1-16 for all the other gates.
- The `DTYPE` inputs are accessed with "`.d`", "`.c`", "`.s`" and "`.r`", which represent `DATA`, `CLK`, `SET` and `RESET` (rather than `CLEAR` to avoid confusion with `CLK`).
- The `DTYPE` outputs are accessed with "`.o`" and "`.no`" for normal output (`Q`) and inverting output (`QBAR`).
- Within "`MONITORS`" at least one monitor must be defined.
- A monitor is defined as "`monitorname <= output`", where outputs are specified as above.

- A monitorname is alphanumeric and must begin with a letter.
- Within "`INIT`" all defined switches and clocks must have an initialisation.
- Clocks are initialised with a period of `n` simulation cycles, where `n` is a positive integer.
- Switches are initialised with a state that is either 1 or 0, where 1 is a logic high state on the output and 0 is a logic low state.
- Initialisation is done by "`devicename = value;`", where devicename is an already defined device and value is legal for the device type.
- Comments in the definition file are defined as any data between an opening "`/*`" and a closing "`*/`". Comments may be nested.

4 Error Handling

5 Example Circuits and Definition Files

5.1 XOR Circuit Composed of NAND Gates

5.2 3-bit Gray Code Counter