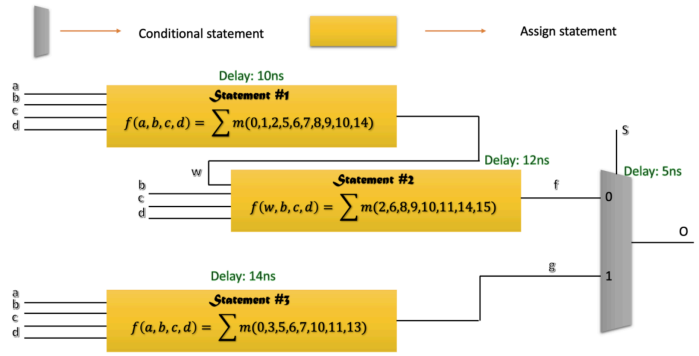
Computer Assignment 3 Report

Logic minimization and SystemC simulation

Erfan Iravani
810197462
Professor Z.Navabi
May 13, 2021

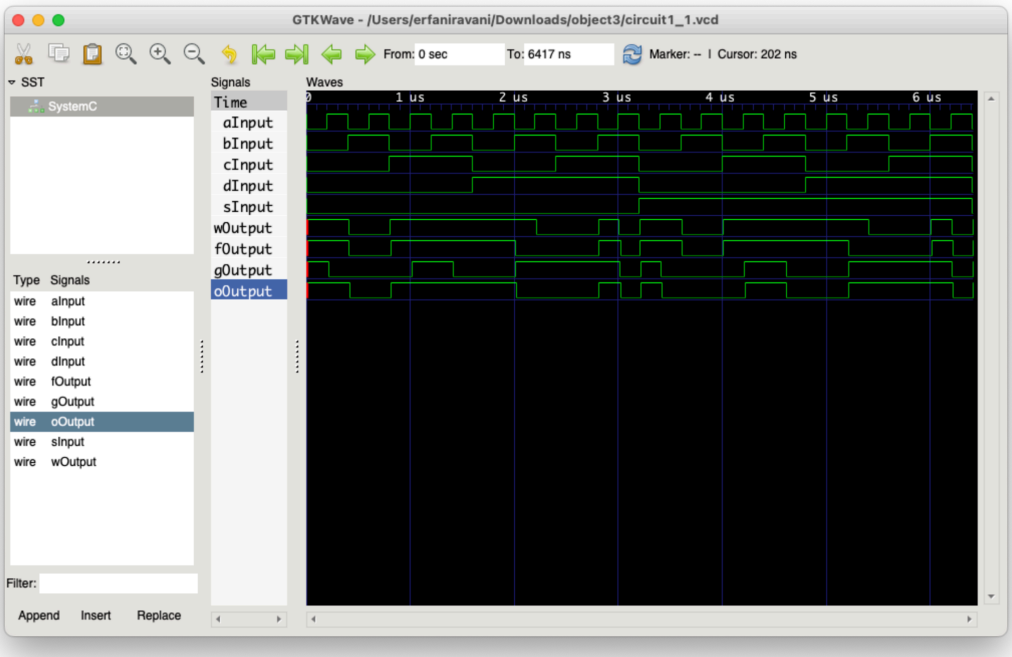
SystemC simulation

The circuit shown below is asked to be manually translated to SystemC code




```
module main(input a, b, c, d, s, output o);

assign #10 w = ~a&~b&~c&~d | ~a&~b&~c&d | ~a&~b&c&~d | ~a&b&~c&~d | ~a&b&c&d | a&~b&~c&~d | a&~b&~c&d | a&~b&c&~d|
             a&b&c&~d;

assign #12 f = ~w&~b&c&~d | ~w&b&c&~d | w&~b&~c&~d | w&~b&~c&d | w&~b&c&~d | w&b&c&d | w&b&c&~d | w&b&c&d;

assign #14 g = ~a&~b&~c&~d | ~a&~b&c&d | ~a&b&~c&d | ~a&b&c&~d | ~a&b&c&d | a&~b&c&~d | a&~b&c&d | a&b&~c&d;

assign #5 o = s ? g : f;

endmodule;
```

After implementing the circuit and developing a test bench we get the results below where we can see the results are as accurate as compiling verilog codes



SOP minimization using QM method in C++

The SOP that we should minimize in shown below

ooTest1.txt

```
module main (a, b, c, d,s,o);

input a;
input b;
input s;
input d;
input c;
output o;
wire w;
wire f;
wire g;


assign #10 w = ~b&~a&~c&~d | ~b&~a&~c&d | ~a&~b&c&~d | ~a&b&~c&d | ~a&b&c&~d | ~a&b&c&d | a&~b&~c&~d
| a&~b&~c&d | ~b&a&c&~d | a&b&c&~d;

assign #12 f = ~w&~b&c&~d | ~w&b&c&~d | w&~b&~c&~d | w&~b&~c&d | w&~b&c&~d | w&~b&c&d | w&b&c&~d |
w&b&c&d;

assign #14 g = ~a&~b&~c&~d | ~a&~b&c&d | ~a&b&~c&d | ~a&b&c&~d | ~a&b&c&d | a&~b&c&~d | a&~b&c&d |
a&b&~c&d;

assign #5 o = s ? g : f;

endmodule
```

And the result after minimization is prepared as a .txt file and we can see that it is working properly

ca3verilog.txt

```
module main(a,b,c,d,s,o);
input a;
input b;
input c;
input d;
input s;
output o;
wire w;
wire f;
wire g;
assign #10 w = ~b&~c | c&~d | b&~a&d;
assign #12 f = c&~d | w&~b | w&c;
assign #14 g = ~a&~b&~c&~d | ~a&b&c | a&~b&c | b&~c&d | ~a&c&d;
assign #5 o = s ? g : f;
endmodule
```

Simulate assign statements with C++

We have classes for wires, not gates, And gates and Or gates and a brief part of them is shown below

```
class wire{
public:
  string value;
  string name;
  int time;
public:
  wire(string value_ , string n) :  value(value_) , time(0) , name(n) {};
  wire(string n) : value("X") , time(0) , name(n) {};
  wire() : value("X") , time(0) {};
};
```

```
class Or{
private:
  wire *o1;
  vector<wire*> i;
public:
  int delay;
  Or(vector<wire*> ins , wire* w);
  void eval();
};
Or::Or(vector<wire*> ins , wire* w){
  o1 = w;
  i = ins;
  delay = 0;
}
```

Two input And and Or gates have 5ns delay. For having more realistic delays we consider the gates having more than 2 inputs as a set of 2 input gates ; in other words for example a 3 input gate is considered as 2 gates with 2 inputs .

Test bench and simulation results for this part is shown below , on top of next page.

testbench.txt

```
#10 01011
#40 10011
#40 11100
#40 01101
```

results.txt

```
<<<<<<<< at time 10
input values: 0  input values: 1  input values: 0  input values: 1  input values: 1
input names: a  input names: b  input names: c  input names: d  input names: s
output w becomes 1  after 30
output f becomes 0  after 45
output g becomes 1  after 45
output o becomes 1  after 58
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
<<<<<<<< at time 50
input values: 1  input values: 0  input values: 0  input values: 1  input values: 1
input names: a  input names: b  input names: c  input names: d  input names: s
output w becomes 1  after 30
output f becomes 1  after 65
output g becomes 0  after 85
output o becomes 0  after 98
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
<<<<<<<< at time 90
input values: 1  input values: 1  input values: 1  input values: 0  input values: 0
input names: a  input names: b  input names: c  input names: d  input names: s
output w becomes 1  after 30
output f becomes 1  after 65
output g becomes 0  after 85
output o becomes 1  after 78
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
<<<<<<<< at time 130
input values: 0  input values: 1  input values: 1  input values: 0  input values: 1
input names: a  input names: b  input names: c  input names: d  input names: s
output w becomes 1  after 30
output f becomes 1  after 65
output g becomes 1  after 165
output o becomes 1  after 178
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
```

We can see that the circuit is working correctly

Comparing to what we had in SystemC we can see that there is not much difference and the fact that SystemC is a C++ library can be seen throw these computer assignment

The delays might be slightly different because in C++ part we implement a gate level logic which is more accurate but the wire values are the same.