

2 by 4 Binary Decoder.

Objectives:

1. To understand the functioning of a 2 by 4 binary decoder.
2. To create a model of computation and simulate the decoder using SystemC language.
3. To analyze the simulation output and compare it with real values and results.

Tools:

1. A computer with SystemC, Eclipse and Gtkwave installed.

Theory:

SystemC is a set of C++ classes and macros which provide an event-driven simulation interface. These facilities enable a designer to simulate concurrent processes, each described using plain C++ syntax.

To decode is to translate data from a code into another form, especially its original form. A binary decoder is a combinational logic circuit that converts a binary integer value to an associated pattern of output bits. A 2 by 4 binary decoder consists of 4 AND gates. It has 2 binary inputs which are decoded into one of 4 outputs. It uses the concept of minterms where each output represents one of the minterms of the 2 inputs. With this concept, the binary inputs determine which output is logic level 1 while the rest are held at logic level 0. Figure 1 shows the 2 by 4 decoder circuit with inputs A₀ and A₁ and outputs D₀, D₁, D₂ and D₃ while figure 2 shows the associated truth table.

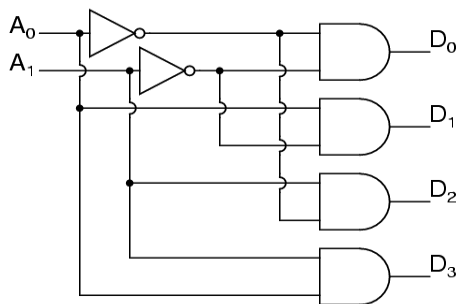


Figure 1. 2 by 4 binary decoder circuit

A ₁	A ₀	D ₃	D ₂	D ₁	D ₀
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

Figure 2. 2 by 4 binary decoder truth table

For any given time, only one output can be at logic level 1 and this helps determine the binary code at the input.

Methodology.

The model of computation in Figure 3 was used. The model above was implemented using SystemC and has 3 modules as shown in the figure above.

Driver module

The driver produces the decoder's input signals, A₀ and A₁. It has no inputs itself.

Monitor module

This module consists of 6 ports, 2 of which monitor the input signals, A₀ and A₁, while the other 4 monitor the output signals, D₀, D₁, D₂ and D₃. At the input, it checks whether the signals produced by the driver are correct and at the output, it checks whether we are receiving the intended output.

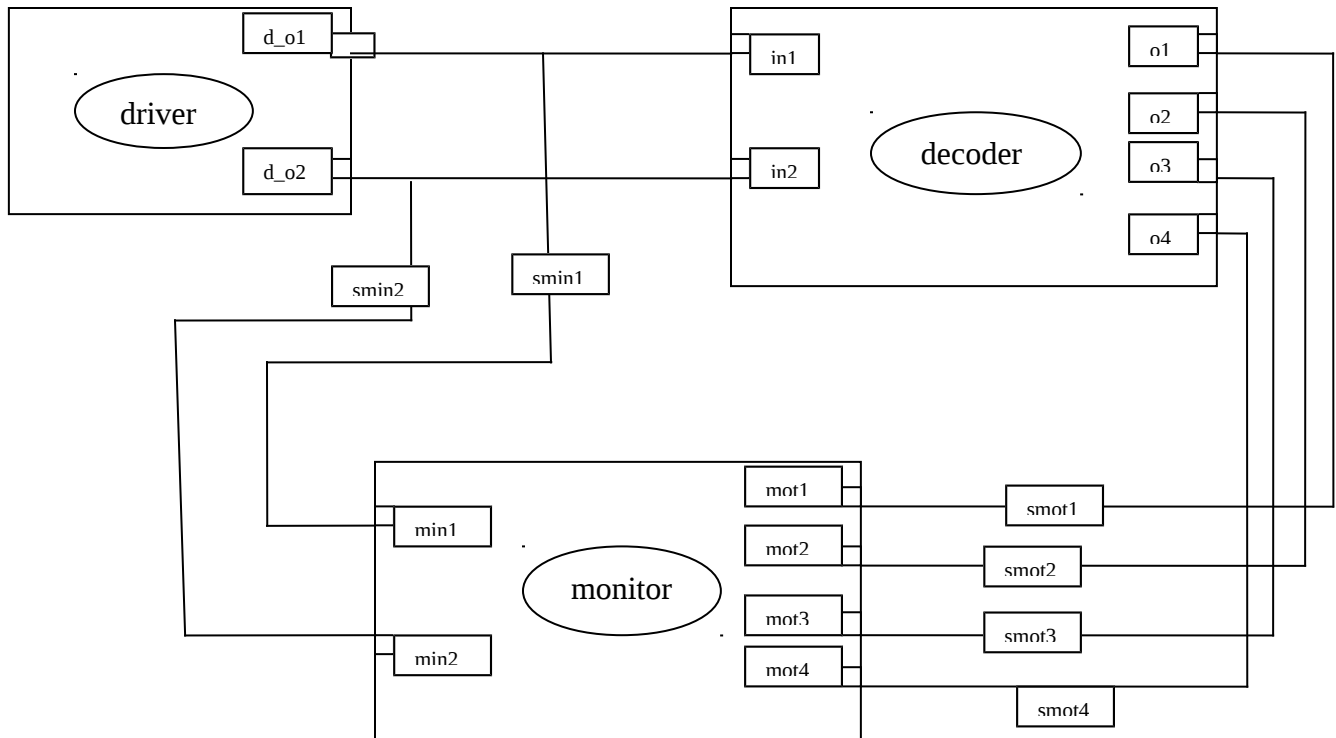


Figure 3. The model of computation

Decoder

This module has 2 input ports which take in signals from the driver and 4 outputs that produce the decoded signals of the inputs.

The code used to implement the 3 modules is as below:

//The driver module

#ifndef DRIVER_H_

#define DRIVER_H_

#include<systemc>

SC_MODULE(driver){
sc_out<bool> d_o1, d_o2;

SC_CTOR(driver){
SC_THREAD(drive);
}

void drive(void){
while(1){
d_o1=0;
wait(5,SC_NS);
d_o1=1;
wait(5,SC_NS);
d_o2=0;
wait(5,SC_NS);
}
}

```

        d_o2=1;
        wait(5,SC_NS);
    }
}

};
#endif /* DRIVER_H_ */

```

//The monitor module

```

#ifndef MONITOR_H_
#define MONITOR_H_

#include<iostream>
#include<systemc>

using namespace std;

SC_MODULE(monitor){
    sc_in<bool> m_in1, m_in2, m_out1, m_out2, m_out3, m_out4;

    SC_CTOR(monitor){
        SC_METHOD(monita);
        sensitive<<m_out1<<m_out2<<m_out3<<m_out4;
        dont_initialize();
    }

    void monita(void){
        cout<<"at "<<sc_time_stamp()<<" input is: "<<m_in1<<" outputs are: "<<m_out1<<" and "<<m_out2<<endl;
        cout<<"at "<<sc_time_stamp()<<" input is: "<<m_in2<<" outputs are: "<<m_out3<<" and "<<m_out4<<endl;
    }

};
#endif /* MONITOR_H_ */

```

//The decoder module

```

#ifndef DECODER_H_
#define DECODER_H_
#include "systemc.h"

SC_MODULE (decoder) {
    sc_in<bool> in1, in2;
    sc_out<bool> out1, out2, out3, out4;

    //constructor: where the processes are bound to simulation kernel
    SC_CTOR(decoder){
        SC_METHOD(decode);
        sensitive<<in1<<in2;
        //dont_initialize();
    }

    ~decoder(){
        //delete stuff :P
    }
}

```

```

        void decode(void){
            out1=in1&&in2;
            out2=in1&&!in2;
            out3=!in1&&in2;
            out4=!in1&&!in2;
        }
};

#endif /* DECODER_H_ */

#include"decoder.h"
#include"driver.h"
#include"monitor.h"
#include<systemc>

int sc_main(int argc, char *argv[]){
//some signals for interconnections
sc_signal<bool> s_in1, s_in2, s_out1, s_out2, s_out3, s_out4;
//module instances
decoder dec("decoder_instance");
driver dr("driver");
monitor mn("monitor");
//interconnections b2in modules
dr.d_o1(s_in1);
dec.in1(s_in1);
mn.m_in1(s_in1);

dr.d_o2(s_in2);
dec.in2(s_in2);
mn.m_in2(s_in2);

dec.out1(s_out1);
mn.m_out1(s_out1);

dec.out2(s_out2);
mn.m_out2(s_out2);

dec.out3(s_out3);
mn.m_out3(s_out3);

dec.out4(s_out4);
mn.m_out4(s_out4);

//create a trace file with nanosecond resolution
sc_trace_file *tf;
tf = sc_create_vcd_trace_file("2by4timing_diagram");
tf->set_time_unit(1, SC_NS);

//trace the signals interconnecting modules
sc_trace(tf, s_in1, "in1"); // signals to be traced
sc_trace(tf, s_out1, "out1");
sc_trace(tf, s_out2, "out2");
sc_trace(tf, s_in2, "in2"); // signals to be traced
sc_trace(tf, s_out3, "out3");

```

```

sc_trace(tf, s_out4, "out4");

//run a simulation for 45 systemc nano-seconds
if( !sc_pending_activity() )
sc_start(50,SC_NS);
//close the trace file
sc_close_vcd_trace_file(tf);
return 0;
}

```

Results.

Upon implementing the model of computation in Figure 3 using the above code, I obtained the following command line output:

```

SystemC 2.3.1-Accellera --- Sep 22 2015 12:00:37
Copyright (c) 1996-2014 by all Contributors,
ALL RIGHTS RESERVED

```

```

Info: (I703) tracing timescale unit set: 1 ns (2by4timing_diagram.vcd)
at 0 s with inputs: 0 and 0 outputs are: 0 and 0 and 0 and 1
at 5 ns with inputs: 1 and 0 outputs are: 0 and 1 and 0 and 0
at 15 ns with inputs: 1 and 1 outputs are: 1 and 0 and 0 and 0
at 20 ns with inputs: 0 and 1 outputs are: 0 and 0 and 1 and 0
at 25 ns with inputs: 1 and 1 outputs are: 1 and 0 and 0 and 0
at 30 ns with inputs: 1 and 0 outputs are: 0 and 1 and 0 and 0
at 35 ns with inputs: 1 and 1 outputs are: 1 and 0 and 0 and 0
at 40 ns with inputs: 0 and 1 outputs are: 0 and 0 and 1 and 0
at 45 ns with inputs: 1 and 1 outputs are: 1 and 0 and 0 and 0
gtkwave 2by4timing_diagram.vcd

```

GTKWave Analyzer v3.3.58 (w)1999-2014 BSI

On the Gtksave, I obtained timing diagrams as shown in Figure 4, the output VCD trace file. The truth table I obtained from both the command line output and and wave form was as follows.

in1	in2	out1	out2	out3	out4
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

Figure 5. Truth table

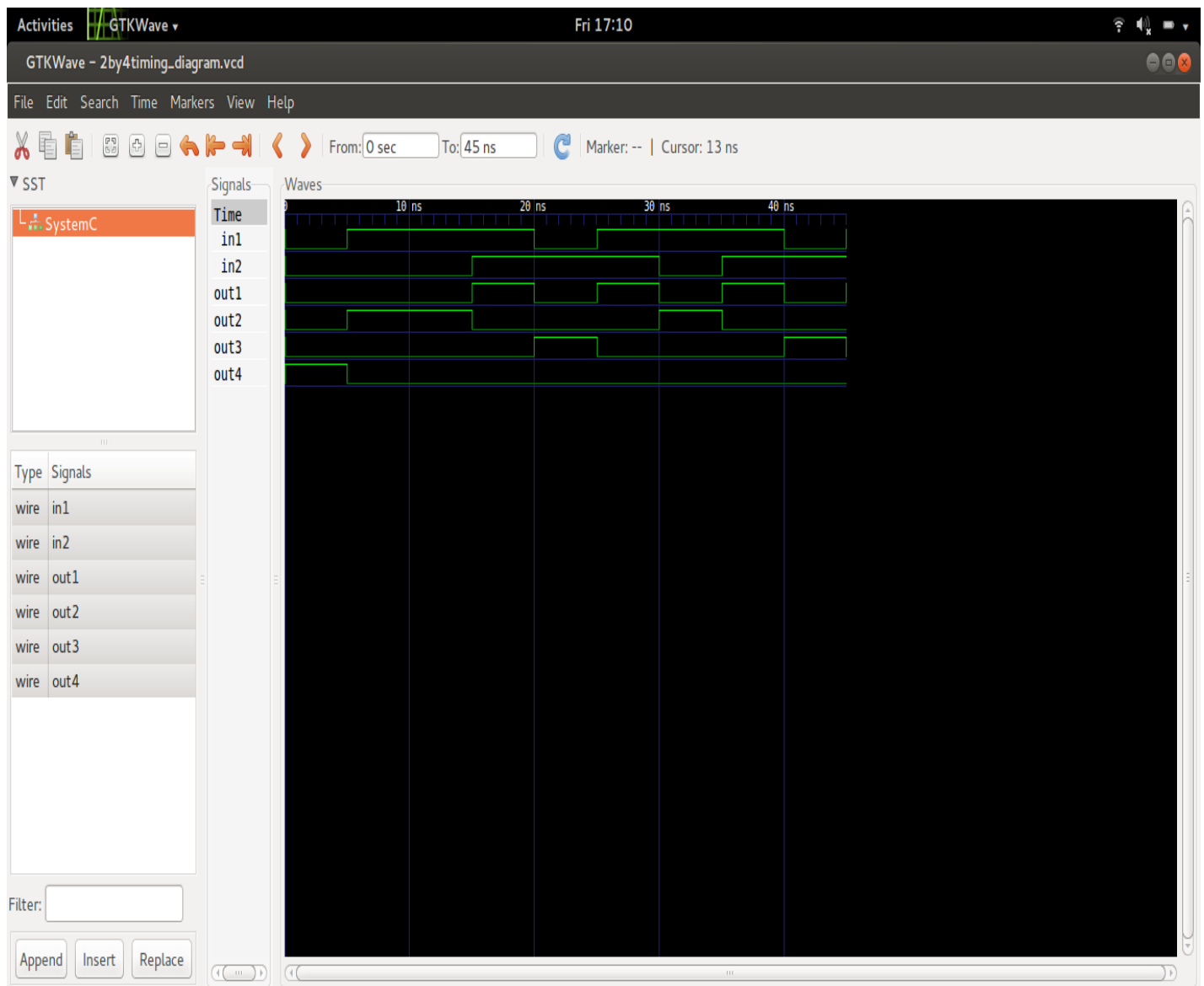


Figure 4. VCD trace file.

Discussion

From the results, it can be deduced that the 2 by 4 decoder gives an output of logic level 1 at the port corresponding to the binary number at the inputs.

Conclusion.

From the timing diagrams, truth table and command line outputs, the objectives were met since I was able to understand the functioning of a 2 by 4 binary decoder, to create a model of computation and simulate the decoder using SystemC language and to analyze the simulation output and compare it with real values and results.

References

1. http://www.electronics-tutorials.ws/combinational/comb_5.html
2. <https://en.wikipedia.org/wiki/SystemC>
3. A SystemC Primer by J. BHASKER . Published by Star Galaxy Publishing 1015 Treeline Drive, Allentown, PA 18103
4. https://github.com/Muriukidavid/cplusplus_examples/tree/mastersystemc