Andati Lexy Amwayi A.L.
I39/2218/2013
SPH 336

19th October 2015

2 by 4 Decoder

Objective

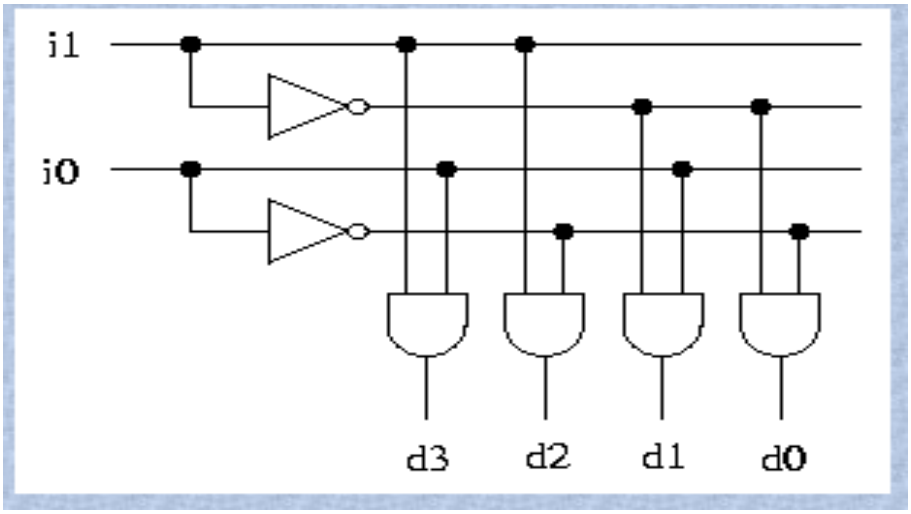
To model and simulate a 2 by 4 decoder using the systemc modelling language.

Introduction

A decoder is a circuit that converts a code into a set of signals. It has n inputs and 2^n outputs . A 2x4 decoder is a circuit just as described above but outputs 1 on the wire corresponding to the binary number represented by the inputs. Its truth table is as follows.

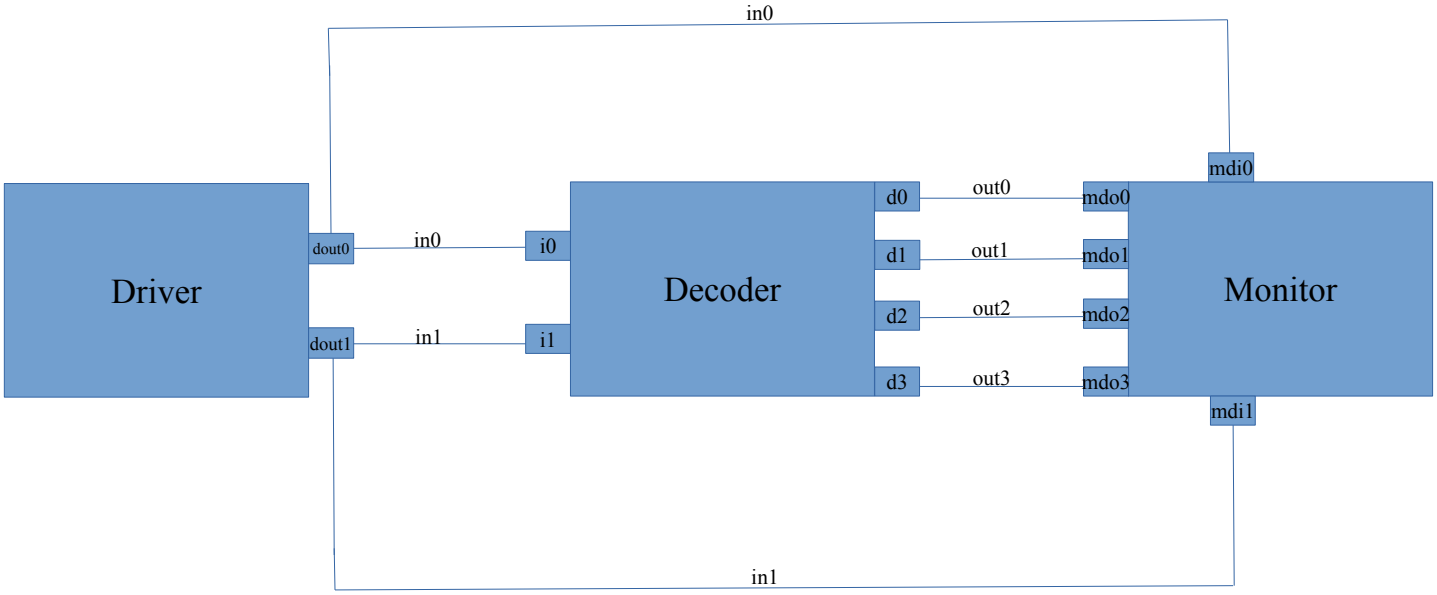
i1	i0	d3	d2	d1	d0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

Also below is the implementation of the 2x4 binary decoder through the use of AND gates and NOT gates.



Methodology

To implement this in systemc, a model of computation was developed as shown below, where the straight lines represent the signals used, while the small boxes represented the ports included. This smoothened the transition into the actual systemc code.



Driver Module
This was used to produce the 2 signals that were to be decoded. This was its only function hence why it has no inputs.

Decoder Module
Here, the signals from the driver module were used as inputs. The input 'codes' were decoded into a set of signals which were then output.

Monitor Module
This module was used to monitor both the inputs from the driver module, and the corresponding otuputs from the decoder module in order to determine whether the decoder model functions correctly.

The following was how the code that was implemented following the above model of computation.

```
//driver module
#include<systemc.h>

SC_MODULE(driver)
{
  sc_out<bool> dout0,dout1;
 // sc_out<bool> dout2;

 SC_CTOR(driver){
```

```
   SC_THREAD(startDrive);
     }

 void startDrive(void)
{
     while(1)
     {
          dout0=0;
          dout1=0;
          wait(5,SC_NS);
          dout1=1;
          wait(5,SC_NS);
          dout0=1;
          dout1=0;
          wait(5,SC_NS);
          dout1=1;
          wait(5,SC_NS);
      }

     }
};
```

//decoder module

```
#include<systemc.h>
SC_MODULE(decoder)
{
//input and output ports
  sc_in<bool> i0,i1;
  sc_out<bool> d0,d1,d2,d3;

  SC_CTOR(decoder)
  {
    SC_METHOD(decode);
    sensitive<<i0<<i1;
   }

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

void decode (void)
 {
```

```
  if (i0==0 && i1==0){
   d0=1;
   d3=0; }
    else if (i0==0 && i1==1){
     d1=1;
     d0=0;
        }
    else  if (i0==1 && i1==0){
       d2=1;
       d1=0;
     }
      else  if (i0==1 && i1==1){
         d3=1;
         d2=0;

         }
      }};
```

//monitor module

```
#include <iostream>
#include<systemc>

using namespace std;

SC_MODULE(monitor)
{
 sc_in<bool> mdi0,mdi1,mdo0,mdo1,mdo2,mdo3;

  SC_CTOR(monitor)
   {
     SC_METHOD(monitoring);
     sensitive<< mdo3<< mdo2<< mdo1<< mdo0;  //sets module
sensitivity
     dont_initialize();

    }

     void monitoring(void)
  {
   cout<<"At "<<sc_time_stamp()<<" inputs are "<<mdi0<<" and
"<<mdi1<<" the outputs are "<<mdo0<<", "<<mdo1<<", "<<mdo2<<",
"<<mdo3<<endl;
```

```
   }

};
```

//decoder2x4.cc

```
#include "driver2x4.h"
#include "decoder2x4.h"
#include "monitor2x4.h"
#include <systemc.h>

int sc_main(int argc, char *argv[])
{
 sc_signal<bool> in_a, in_b, out_a, out_b, out_c, out_d;
//signals to connect different modules

 driver myDriver("driver instance");
 monitor myMonitor("monitor instance");
 decoder myDecode("decoder instance");

 //inputs format: module instance.port(assign signal 2 port)
 myDriver.dout0(in_a);
 myDriver.dout1(in_b);
 myMonitor.mdi0(in_a);  //monitor for inputs
 myMonitor.mdi1(in_b);
 myDecode.i0(in_a);
 myDecode.i1(in_b);

 // outputs format: module instance.portDeclaredInMethod(assign
signal 2 port)
 myDecode.d0(out_a);
 myDecode.d1(out_b);
 myDecode.d2(out_c);
 myDecode.d3(out_d);
 myMonitor.mdo0(out_a);  //monitor for outputs
 myMonitor.mdo1(out_b);
 myMonitor.mdo2(out_c);
 myMonitor.mdo3(out_d);

//creating trace file
sc_trace_file *tf;
tf= sc_create_vcd_trace_file("timing_diagram"); //sets the name
of the file
```

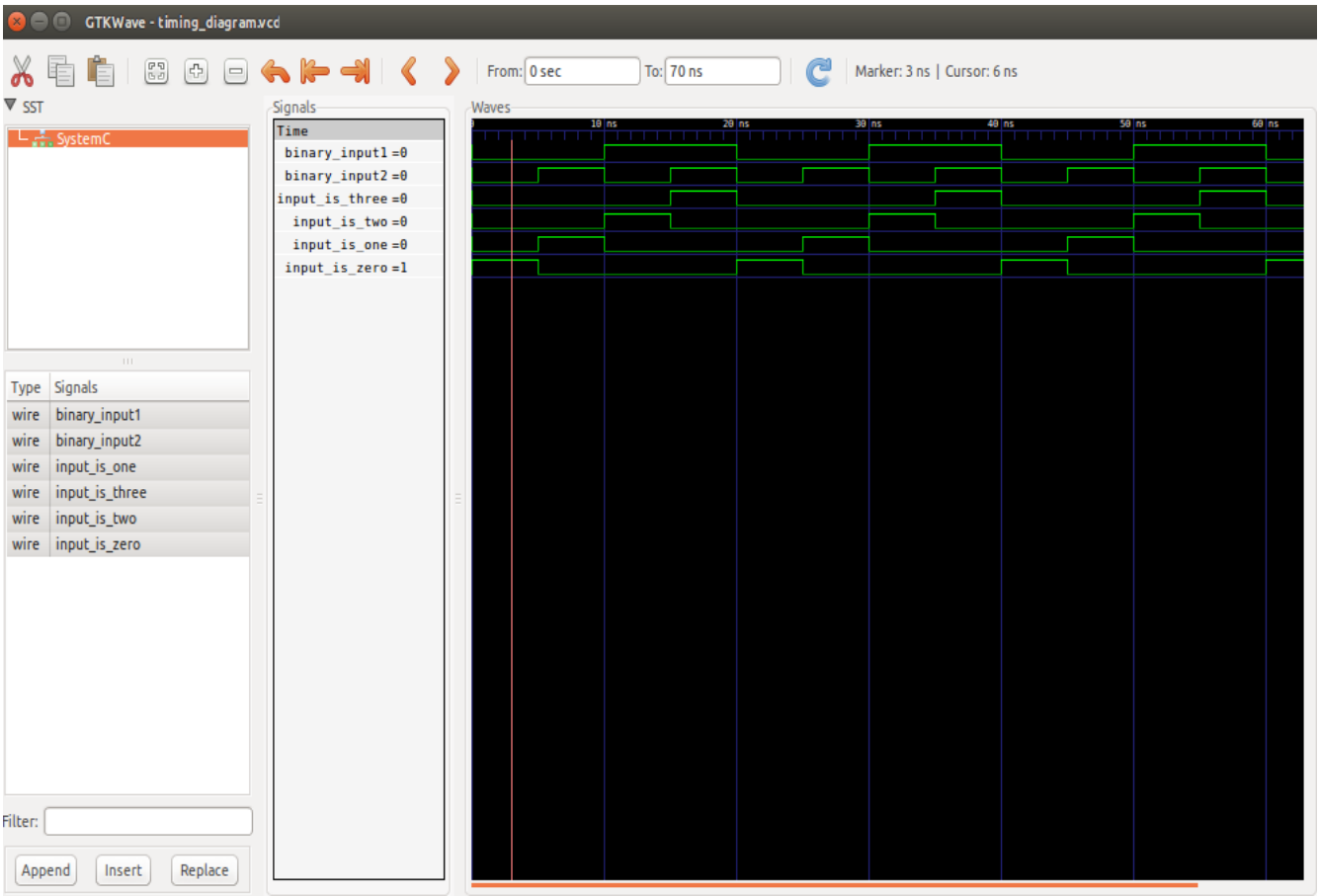
```
tf->set_time_unit(1, SC_NS);

//trace the signals interconnecting modules
sc_trace(tf, in_a, "binary_input1"); // signals to be traced
sc_trace(tf, in_b, "binary_input2");  //those inquotations show
the label of the 'axes'
sc_trace(tf, out_d, "input_is_three");
sc_trace(tf, out_c, "input_is_two");
sc_trace(tf, out_b, "input_is_one");
sc_trace(tf, out_a, "input_is_zero");
//point to note frm above ^^: while specifying the names of
these outputs SPACES are INVALID xters and thus should not be
included rather be replaced with an underscore '_' . failure to
do this will result in an error that causes the gtkwave to
refuse to open.

//run a simulation for 70 systemc nano-seconds
if( !sc_pending_activity() )
sc_start(75,SC_NS);
//close the trace file
sc_close_vcd_trace_file(tf);

 return 0;
 }
```

Results

The following were the command line outputs

```
At 0 s inputs are 0 and 0 the outputs are 0, 0, 0, 1
At 5 ns inputs are 0 and 1 the outputs are 0, 0, 1, 0
At 10 ns inputs are 1 and 0 the outputs are 0, 1, 0, 0
At 15 ns inputs are 1 and 1 the outputs are 1, 0, 0, 0
At 20 ns inputs are 0 and 0 the outputs are 0, 0, 0, 1
At 25 ns inputs are 0 and 1 the outputs are 0, 0, 1, 0
At 30 ns inputs are 1 and 0 the outputs are 0, 1, 0, 0
At 35 ns inputs are 1 and 1 the outputs are 1, 0, 0, 0
At 40 ns inputs are 0 and 0 the outputs are 0, 0, 0, 1
```

And the output VCD tracefile



Discussion

The above timing diagram and the command line output is as was expected theoretically. It is observed that from the above timing diagram, when the binary inputs are 0 and 0,which is binary for 0, the wire of input is 0 goes high, the same applies for 0 and 1, binary for 1, the wire of input is 1 goes high while all the other outputs are low. This trend is the same for all inputs thereby meaning that for every binary input, the output of the decoder is its decimal equivalent.

Conclusion

It is therefore concluded that the modelling and simulation of the 2 by 4 decoder was successful and therefore the objective was met.

References

1.A SystemC Primer
J. BHASKER
Published by
Star Galaxy Publishing
015 Treeline Drive, Allentown,PA 18103

2.SystemC: From the Ground Up
D.C. Black, J. Donovan, B. Bunton, A. Keist

3. https://github.com/Muriukidavid/cplusplus_examples/tree/master/systemc