Structural Design With SystemC

Introduction

This tutorial presents a way of using SystemC for structural design. It starts by implementing the NAND gate and then, using only the NAND gate, building the rest of the blocks used in digital design.

Currently we stop at T-flipflop, however, in the future more blocks might be added. This is not a tutorial about learning SystemC, various pointers will be given in time for great tutorials that can be found online.

Directory structure, compiling and running

The directory structure is kept very simple so that the user can follow the design easily. The root directory is called ***structural_sc***. All the digital blocks are implemented only in header files and they belong to the directory named ***common***. Each digital block might have multiple versions, each with its own name. Every digital block has a test directory, named after the digital lock tested (e.g. ***nand_gate*** for testing the nand gate digital block). To build a specific block run the following command in the test directory :

```
#cd tflipflpop
```

```
# g++ -g -O -I. -I /home/USER_NAME/systemc/include/ -L. -L /home/USER_NAME/systemc/lib-linux64/ -o t_flipflop_test.exe t_flipflop_test.cpp -lsystemc -lm
```

From the above command it is supposed that the systemc 2.3 library is installed in the directory ***/home/USER_NAME/systemc*** on a linux 64 bit machine.

The executable *.exe produces a trace file that contains the input and output signals of the block under test. It can be visualized with and VCD waveform viewer (like gtkwave). To run the test:

```
# ./t_flipflop_test.exe
```

The NAND gate

Few versions of the nand gare are implemented in the **nand.h** file. The ideal nand gate with two inputs is named **Nand2Ideal**. Its testbench is in ***nand_gate/nand_test.cpp*** file and consists of a test generator **TestGenerator** that issues all four combinations for the two inputs. The same test generator is used to test all the nand gates, the DUT must be an instance of the nand gate that has to be tested. The inputs and outputs of the nand gate are traced in ***traces.vcd*** file. For detailed explanations about how **Nand2Ideal** and its testbench are implemented please see the excellent tutorial from Doulos (http://www.doulos.com/knowhow/systemc/tutorial/). The **Nand2WithDelay** and **Nand3WithDelay** are used in building the latches and flipflops presented in this tutorial. They have a “hardwired” delay of 100 ps for both propagation times, low-to-high and high-to-low. The fourth gate is **Nand2WithTwoOut** and this served for building and experimental latch that was exposing a port instead of a signal to the outside world. This nand gate has two output ports and uses **sc_port** as output port. More details will come later. Finally the last nand gate is named **Nand** and has a programmable number of inputs as a constructor parameter and same “hardwired” delay like **Nand2WithDelay** gate.

The SetReset latch

The SetReset latch and its TestGenerator are shown in Fig.1.

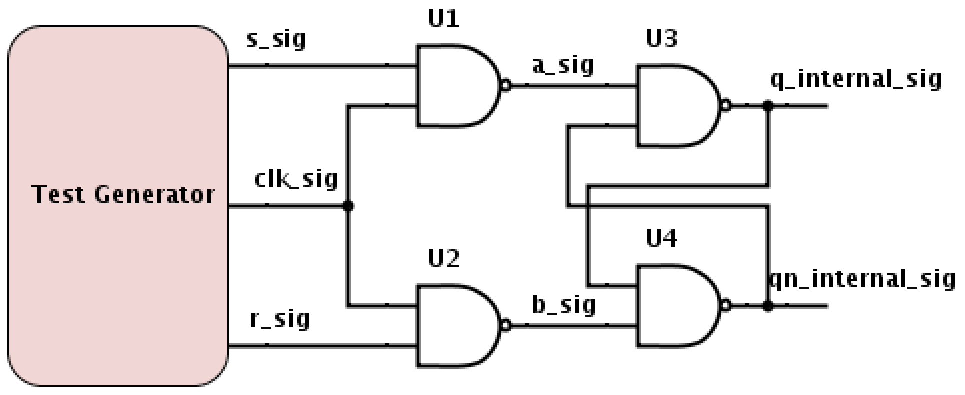

Fig. 1

The SetReset latch is implented with **Nand2WithDelay** gates and because of that it does not have output ports. The testbench traces the q_internal_sig for its output.

The Set latch

The Set latch and its TestGenerator are shown in Fig. 2.

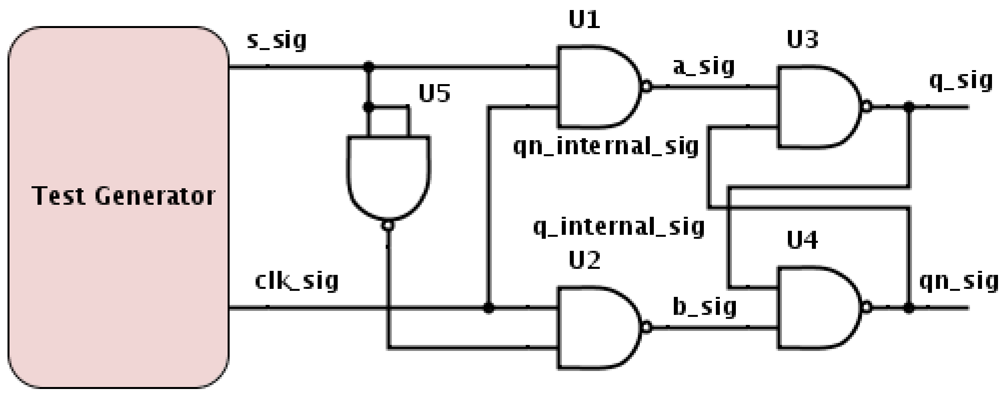

Fig. 2

This latch has three implementations. In **SLatchV0** version the **Nand2WithDelay** gates were used, same like for SetReset latch. In the **SLatchV1** the **Nand2WithDelay** were used with one extra method, **WriteOutPort**(), that helped to introduce the output ports q**_out** and **qn_out**. The third version of the latch is **SlatchWithTwoOut** that used for implementation the **Nand2WithTwoOut** gates. This implementation did not need the extra method as the previous implementation for having **q_out** and **qn_out** ports.

The D flipflop

The Dflipflop has multiple implementations too. Two of them are master and slave and the other two are optimized implementations. The master and slave DflipFlop schematic is shown in Fig.3.

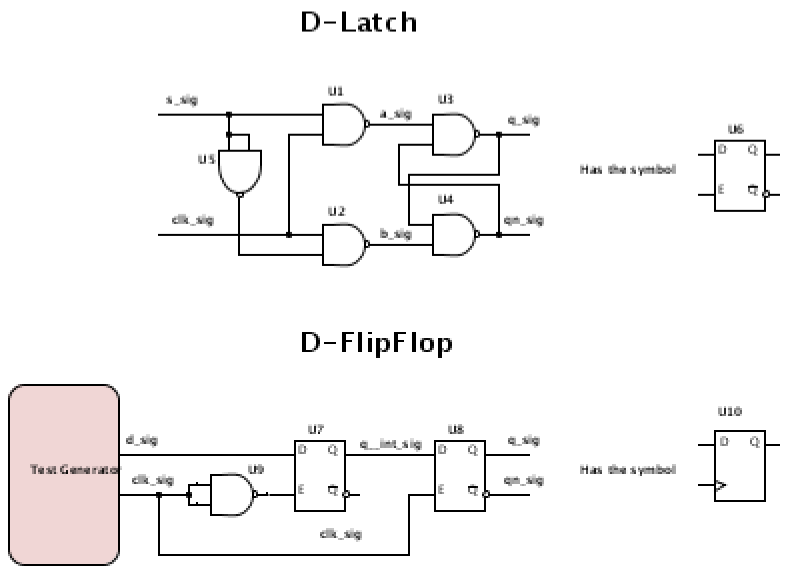

Fig. 3

The first version of the master slave DflipFlop is called **DflipFlopFast** and uses the **SLatchV0.** It runs faster than the **DflipFlopMasterSlave** that uses the **SLatchV1** because does not have the extra methods used by the **SLatchV1** latches.

The versions **DflipFlop** and **DFlipFlop** are the fastest, they are using six nand gates instead of eleven like the master slave flipflop. The **DflipFlopR** version is the flipflop with reset and its the version that is used for all the coming digital blocks.

The T flipflop

The TflipFlop schematic and testbench are shown in Fig.4.

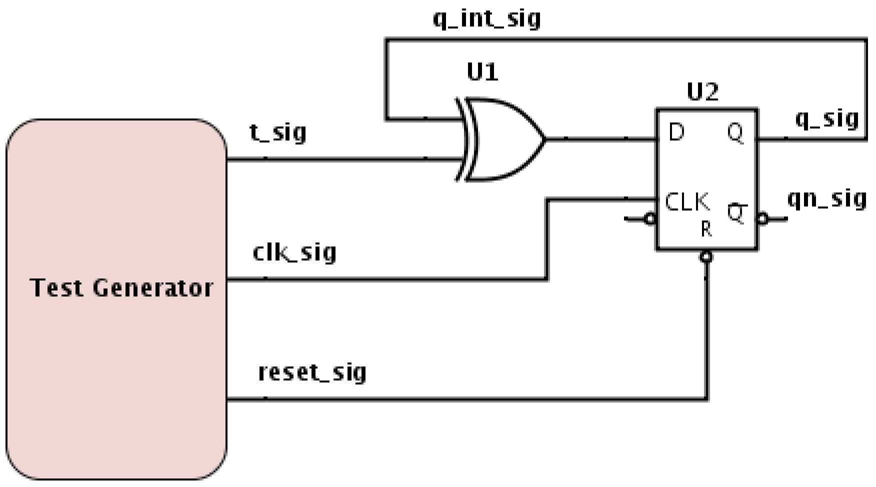
q_int_sig
U1
U2
t_sig
D
Q
q_sig
CLK
Q̄
qn_sig
R
clk_sig
Test Generator
reset_sig

Fig. 4