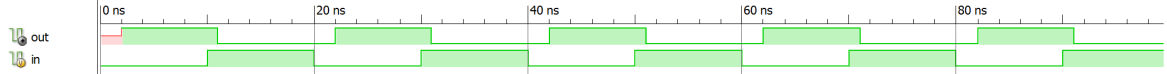
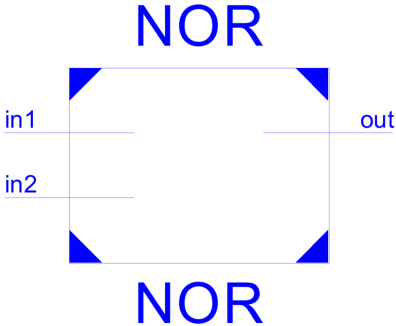
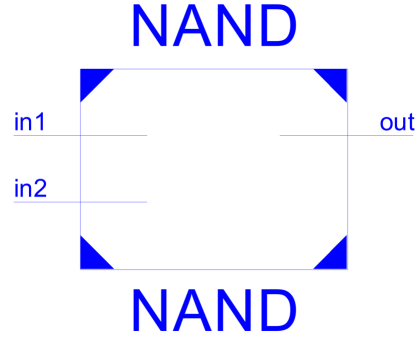
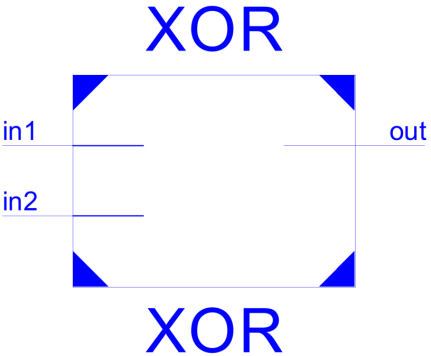
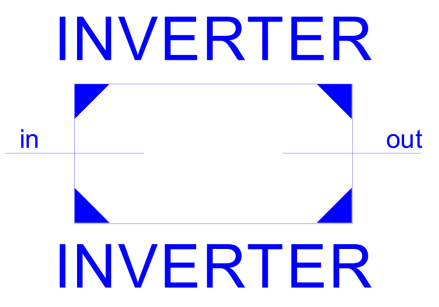
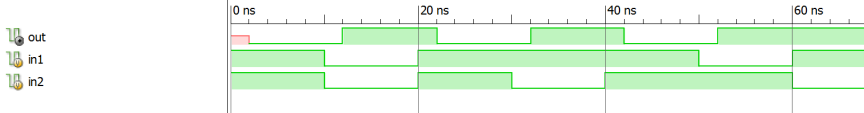
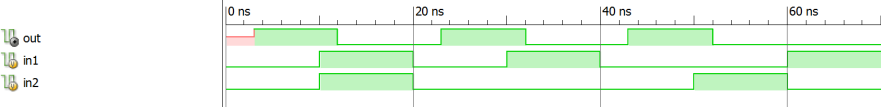
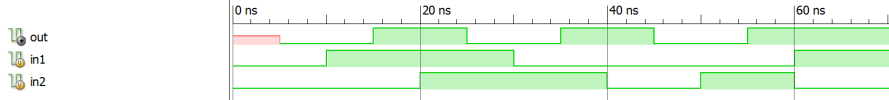
Mehmet Alp Şarkışla

# EE 540 Assignment1

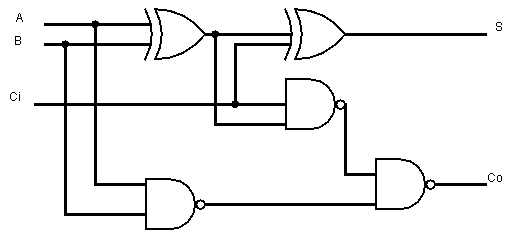
**1-a) Inverter (tplh=2ns, tphl=1ns)**

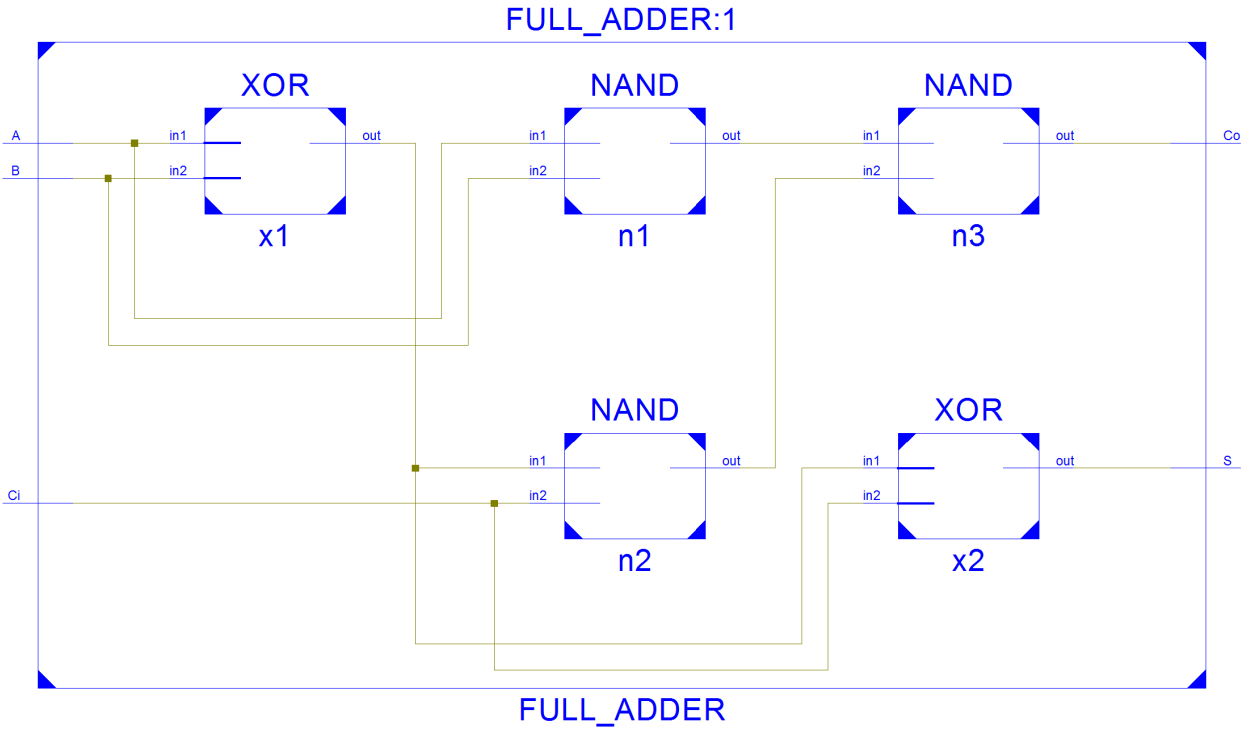
**1-b) NAND (tplh=tphl=2ns)**

**1-c) NOR (tplh=3ns, tphl=2ns)**

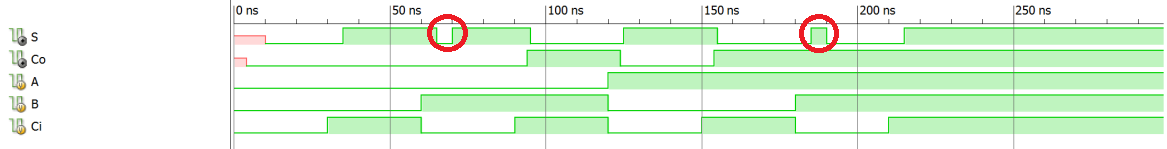
**1-d) XOR (tplh=tphl=5ns)**

**2-a) 1-bit full adder**

****

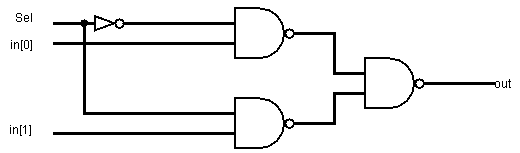
****

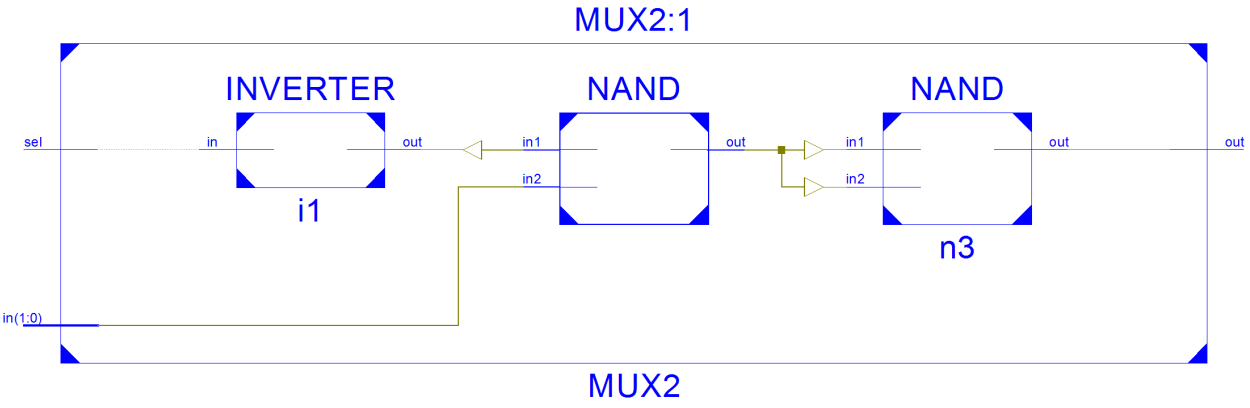
I changed the AND’s and a OR in the classic full adder design to NAND’s .

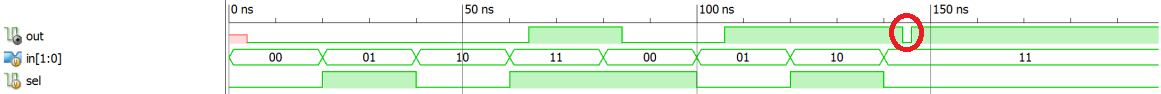
****

As you can see there are two “pulses” on the simulation, thats because S= Ci^(A^B) so there is a 5ns delay in the A^B to change due to XOR.

**2-b) 1-bit 2-input multiplexer**

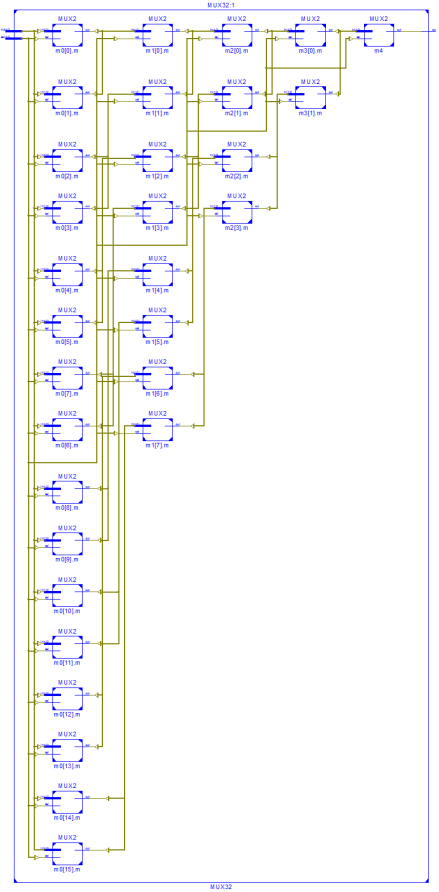
****

****

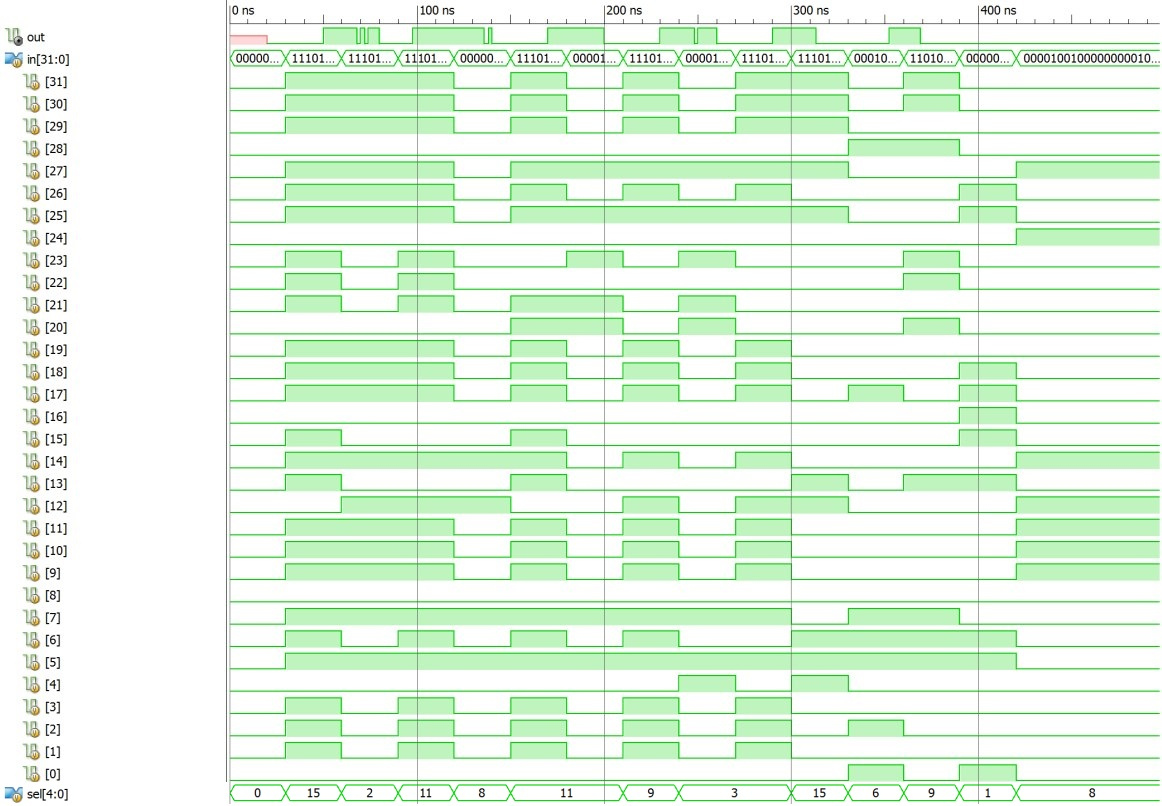
****My origial design consisnt of 3 NAND’s and one INVERTER , synthesize changed that.

There is one 2ns pulse due to the NAND delay.

**3) 32-bit multiplexer**

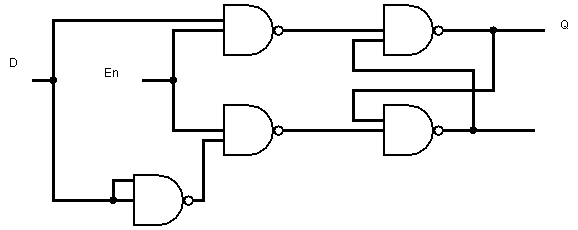


Ther is 2 ^32 \* 2 ^5 = 137438953472 possibilities . It is impossible to simulate. So I select few to show.



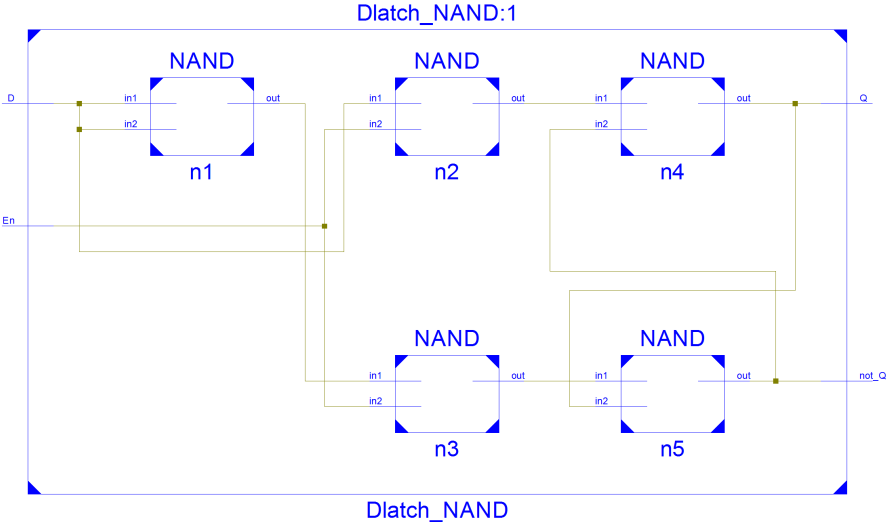
After in changed out changes ~30ns after. Set number shoved as decimal. MUX picks in[n] looking the sets decimal number and send that bit as out.

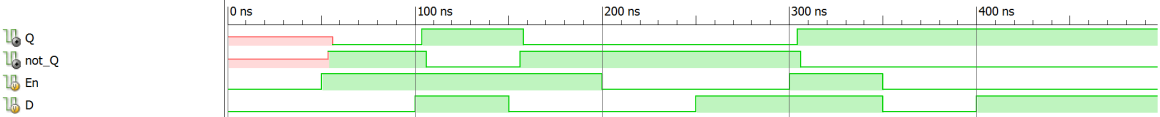
**4-a) positive and negative D latch with NAND gates**

* **Positive**

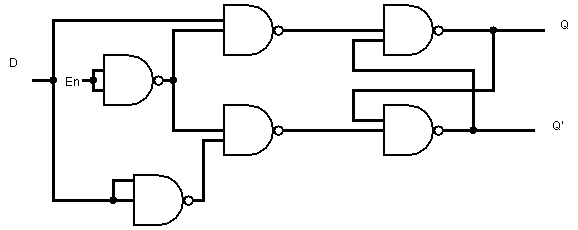
**Simple D latch Q=D whenever En =1**

**And Q retains its old value whenever En = 0**

****

****

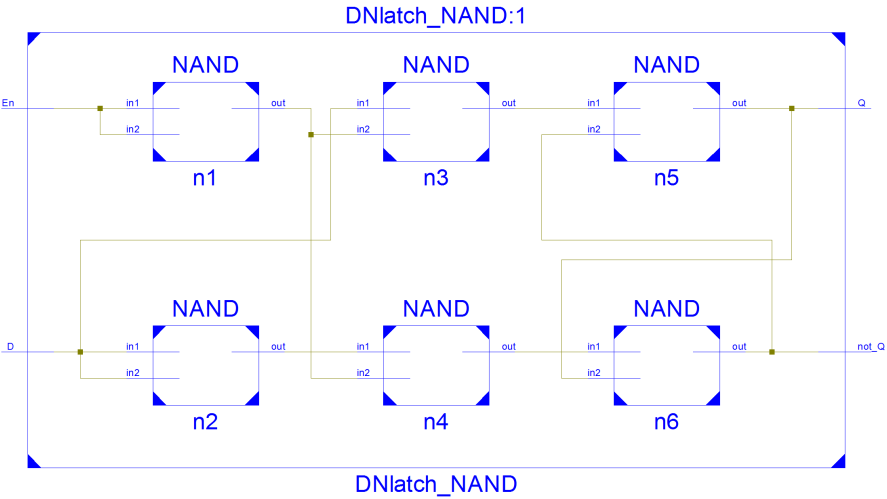
* **Negative**

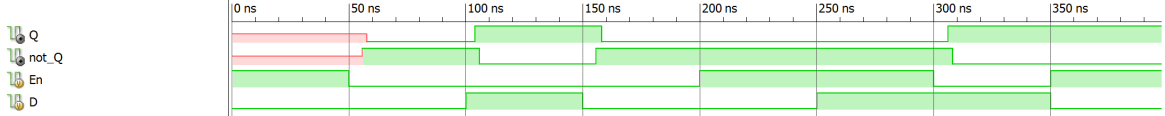
****

**Still same Dlatch but this time Q=D if En =0**

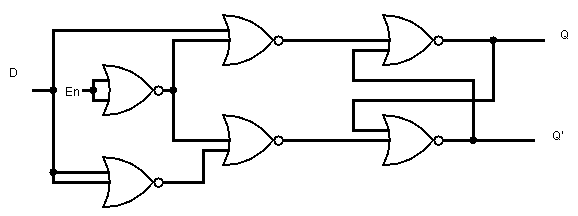
**And Q = old\_Q when En = 1**

**I had to use one more NAND gate as an inverter**

****

**4-b) positive and negative D latch with NOR gates**

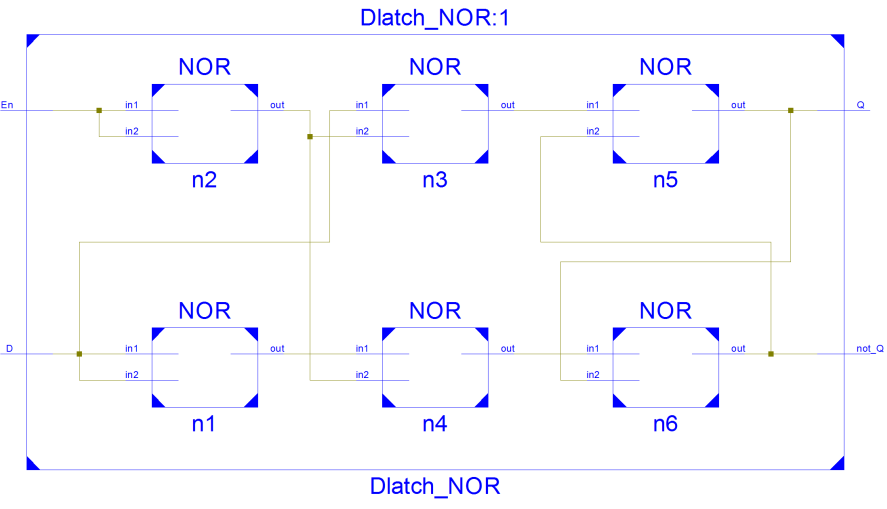
* **Positive**

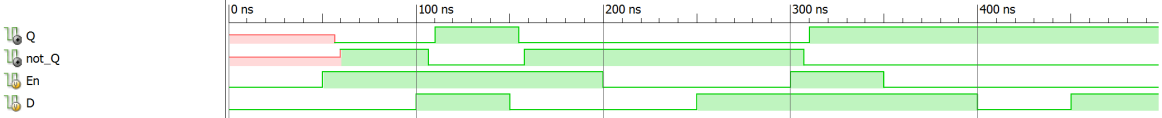
****

**Simple D latch Q=D whenever En =1**

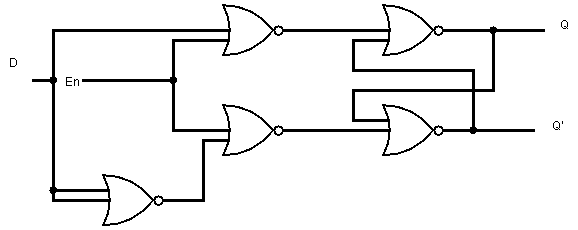
**And Q retains its old value whenever En = 0**

**Uses one more gate compaed to the NAND positive latch**

****

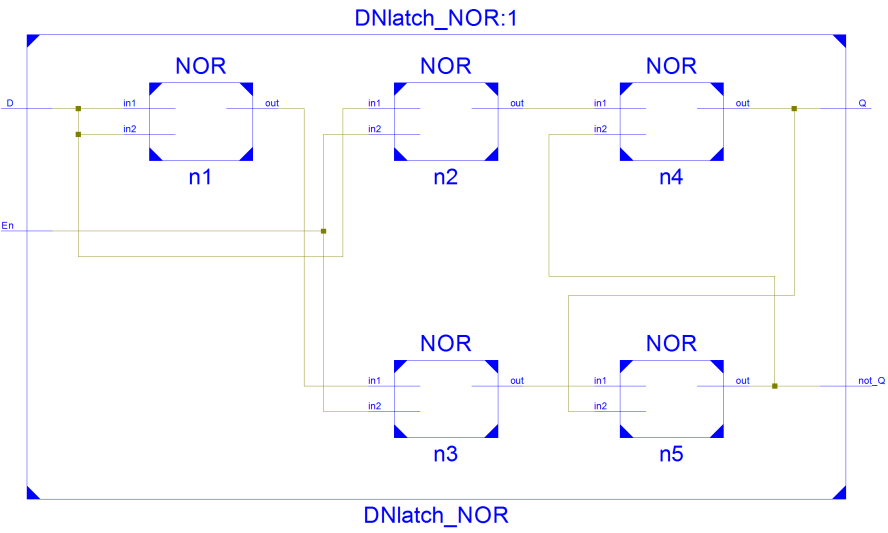
****

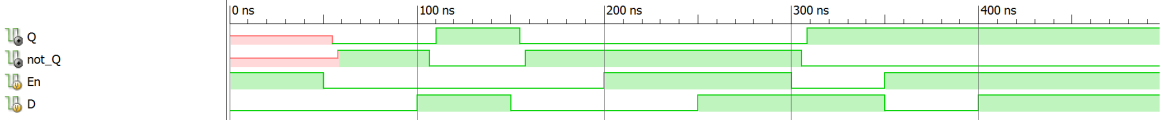
* **Negative**



**Still same Dlatch but this time Q=D if En =0**

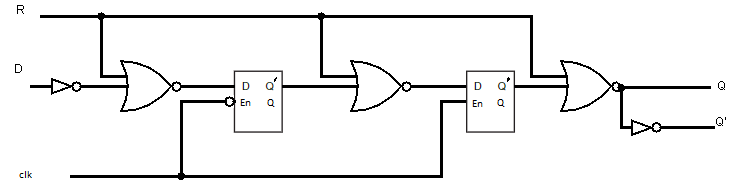
**And Q = old\_Q when En = 1**

**Uses one less gate compaed to the NAND negative latch**

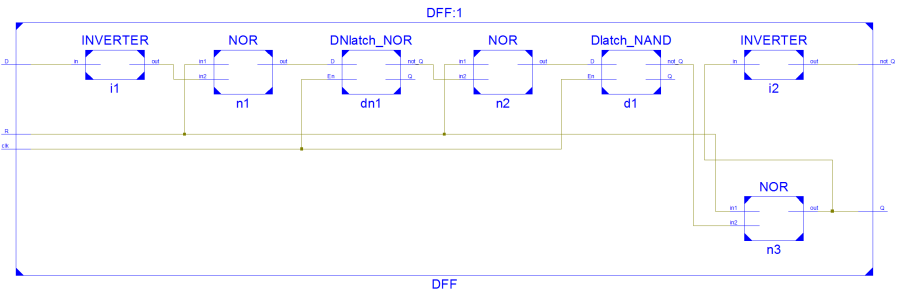


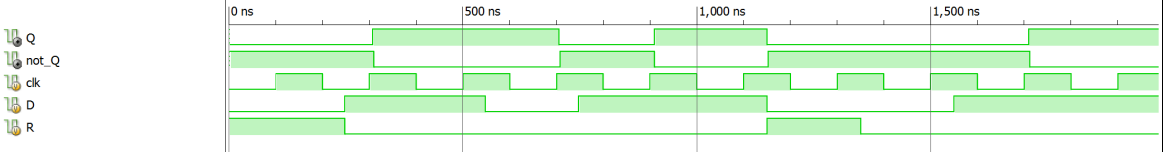
If gate numbers are equal NAND takes less time to adjust new values ((NORtplh > NANDtplh) but if game number is less lesser one tkaes less time to adjust.

**5) DFF with asynchronous reset**

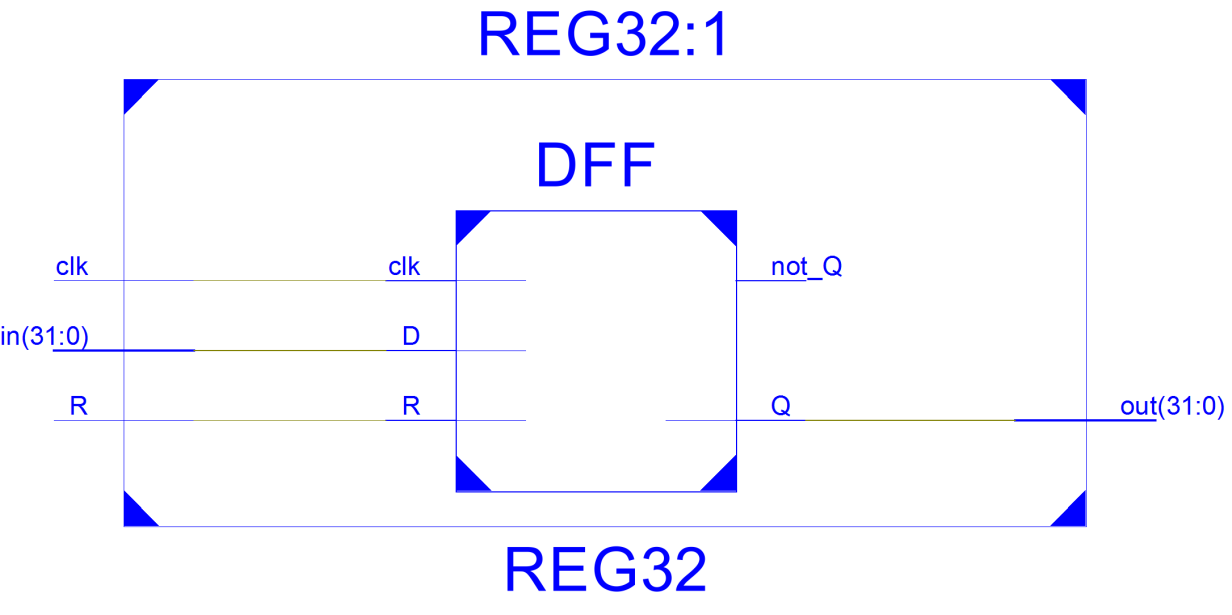


İf R = 1 Q = 0 independent from all , else in every rising edge of clk Q becomes D

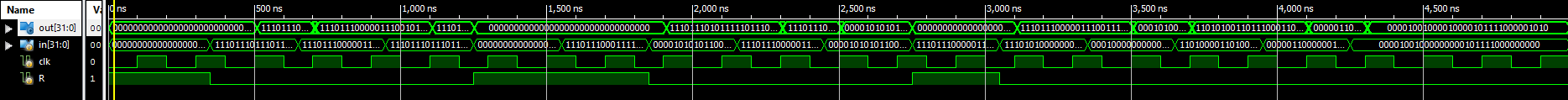




**6) 32 bit Register**



Consist of 32 DFF’s stacked together.



**7) UDP 4bit MUX**

Again too many (3(1,0,X)^4 \* 2^2 ) inputs to simulate. Tried couple of them.

ISim P.20131013 (signature 0x7708f090)

WARNING: A WEBPACK license was found.

WARNING: Please use Xilinx License Configuration Manager to check out a full ISim license.

WARNING: ISim will run in Lite mode. Please refer to the ISim documentation for more information on the differences between the Lite and the Full version.

This is a Lite version of ISim.

Time resolution is 1 ps

Simulator is doing circuit initialization process.

Finished circuit initialization process.

in = 0000 sel = 00 out = 0

in = 0001 sel = 00 out = 1

in = 0010 sel = 01 out = 1

in = 0100 sel = 01 out = 0

in = xxx0 sel = 00 out = 0

in = 1x10 sel = 11 out = 1

in = 1101 sel = 10 out = 1

in = 1001 sel = 10 out = 0

Stopped at time : 41 ns : [File "D:/Projects/GitHub/Verilog-simple-logic-elements/testMUX4\_UDP.v" Line 56](|D:/Projects/GitHub/Verilog-simple-logic-elements/testMUX4_UDP.v|%20Line%2056%20)