Switch level modelling

Switch level modelling forms the basic level of modelling digital circuits . the switches are available as primitives in Verilog , they are central to design description at this level .

Basic gates can be defined in terms of such switches

By repeated and successive instantiation of such switches , more involved circuits can be modelled.

It provides a level of abstraction between the logic and analog transistor levels of abstraction and describing the interconnection of transmission gates which are abstraction of individual MOS and CMOS transistor

Basic transistor switches

- Consider an **N-MOS transistor of the depletion type** , N-MOS transistor is made up of n-type source and drain and a p- type substrate
- There are three modes of operation in N-MOS

1) VG<VS

   Where vg gate voltage and vs is source voltage

   When vg<vs , the transistor is off and offers a very high impedence across the source and drain. i.e. transistor is in Z state.

2) Vg=vs

   The transistor is in active region . it represents a resistance between the source and drain.

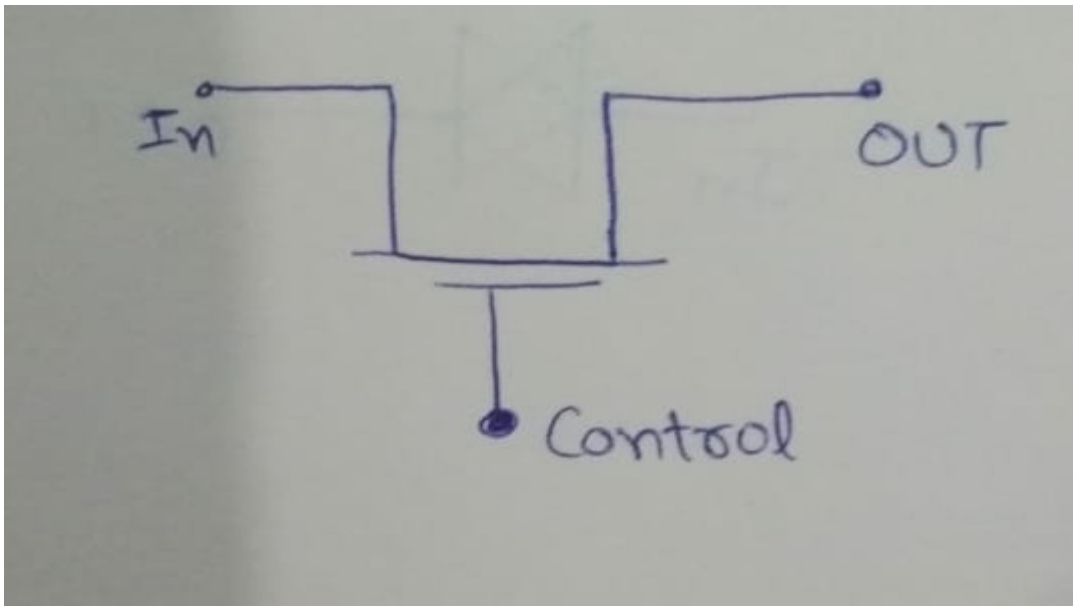
3) Vg>vs

   The transistor is fully turned on . it represents a very low resistance between source and drain.

N- MOS enhanced mode

- When vg=vs , transistor is in off state
- When vg<vs , transistor is moderately on or in active region
- When vg>vs , transistor is in on state representing very low resistance.

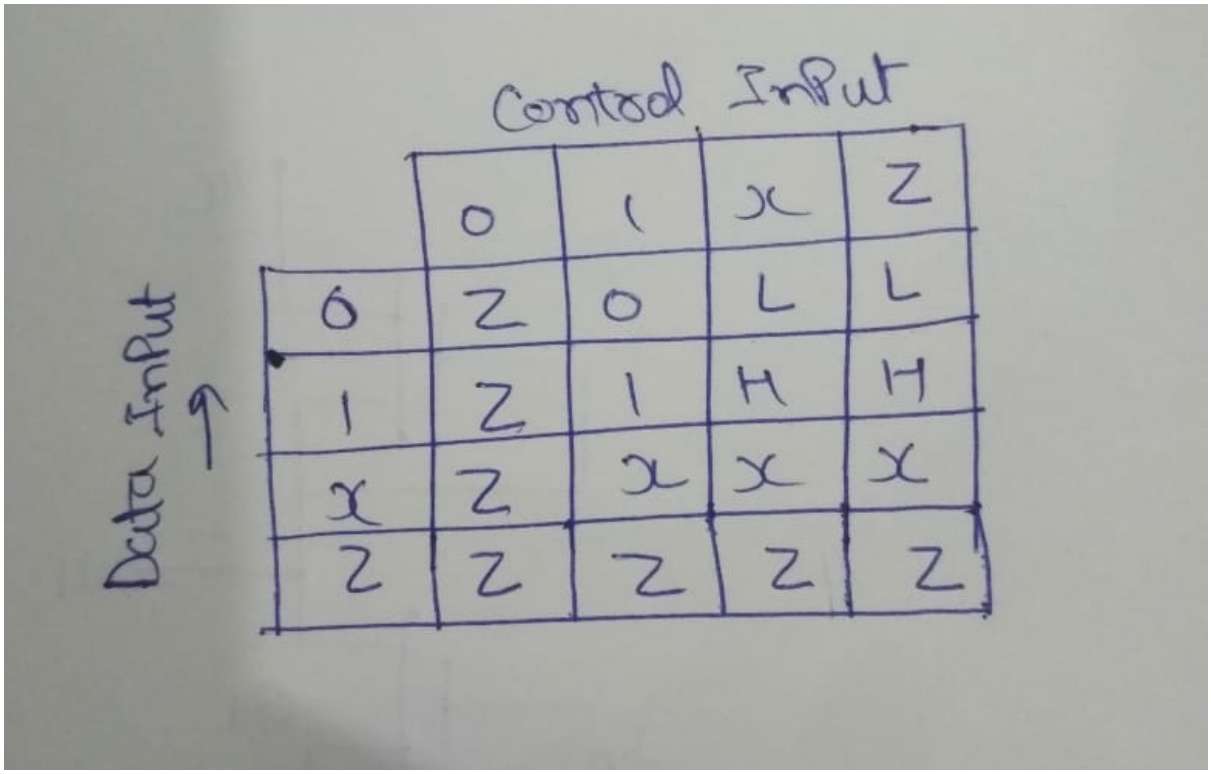
N-MOS switch



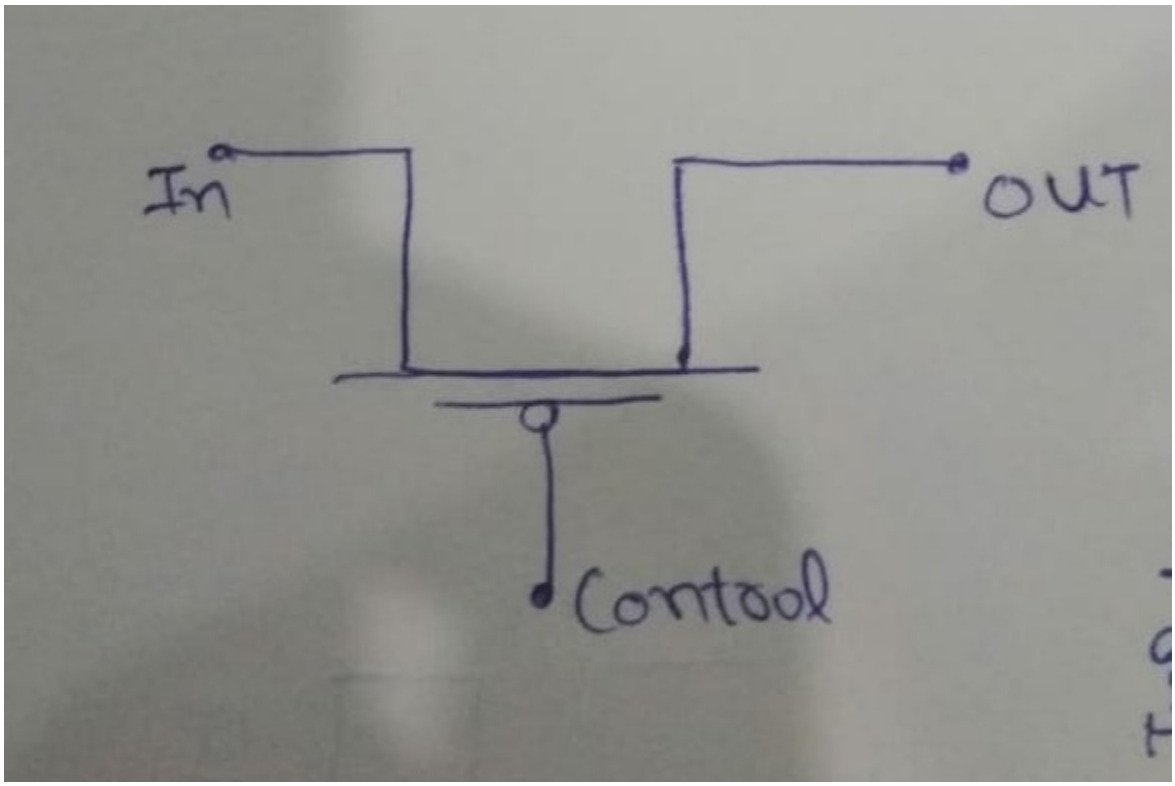
When control input is at (1) high state , the switch is on. It connects the input lead to the output side and offer zero impedence .

When control input is low , the switch is off and output is left floating i.e. is in Z state.

If the control is in Z or X state , the output may take the corresponding value.



P-MOS switch

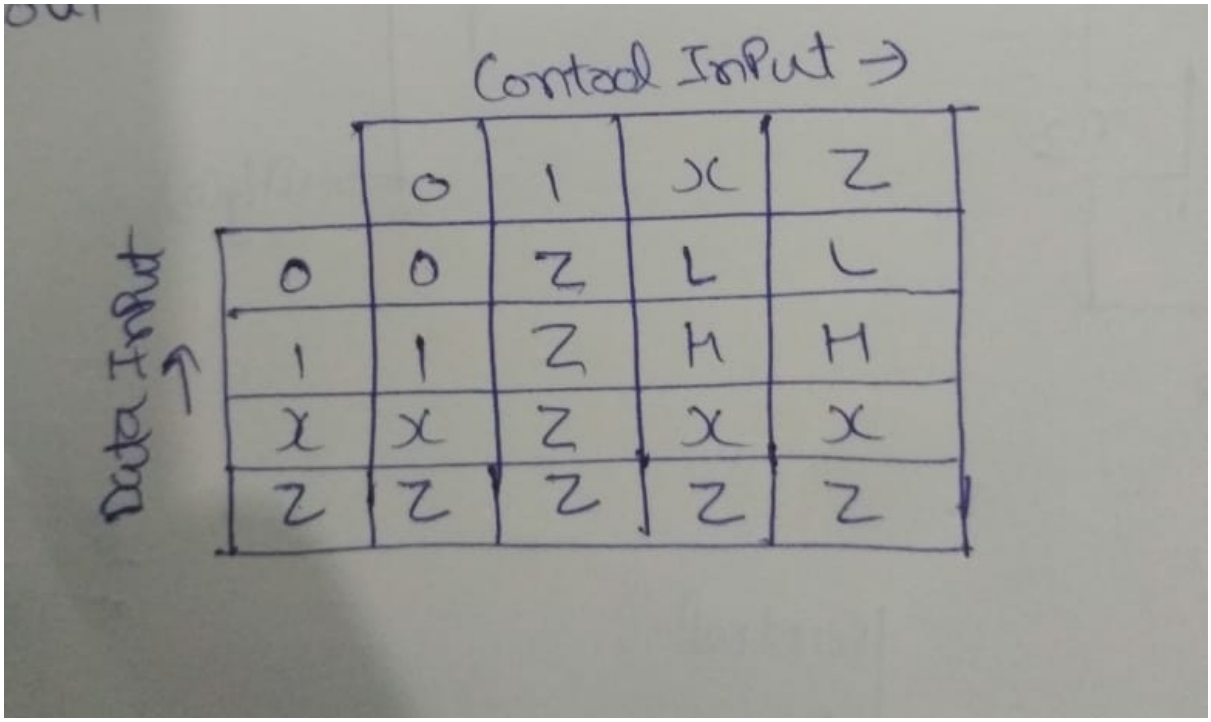


When control is at high state , the switch if off.

When control is at low state , the switch is on, input is connected to the

output and ouput is at same state as input.

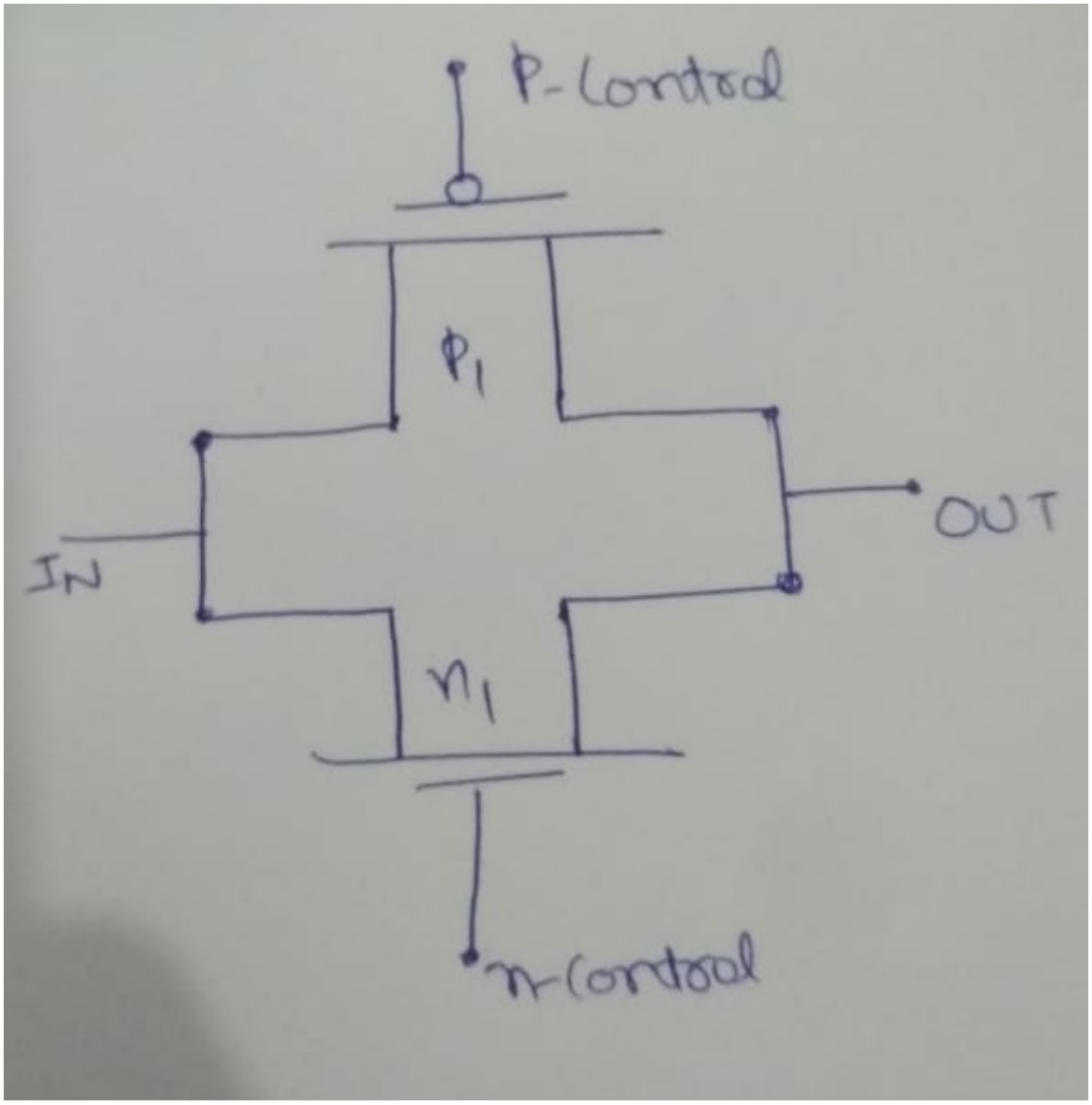
For other input values , output is at other values.



CMOS switch

A CMOS is basically an inverter logic that consist of p-mos at the top and n-mos at the bottom whose source and drain terminal are connected together.

Or we can say that , a cmos switch is formed by connecting pmos and an nmos switvh in parallel, the input leads are connected together on the one side and the output leads are connected together on the other side.



It has two control inputs

- N –control turns on the N-MOS transistor and keeps it on when it is in the 1 state.
- P –control turns on the P-MOS transistor and keeps it on when it is in the 0 state.

Syntax

Cmos switch instanitiated

cmos csw (out,in,N-control, P-control);

Where

Cmos : keyword for switch instantiation

Csw : name assigned to the switch

Out: output variable

In: input variable

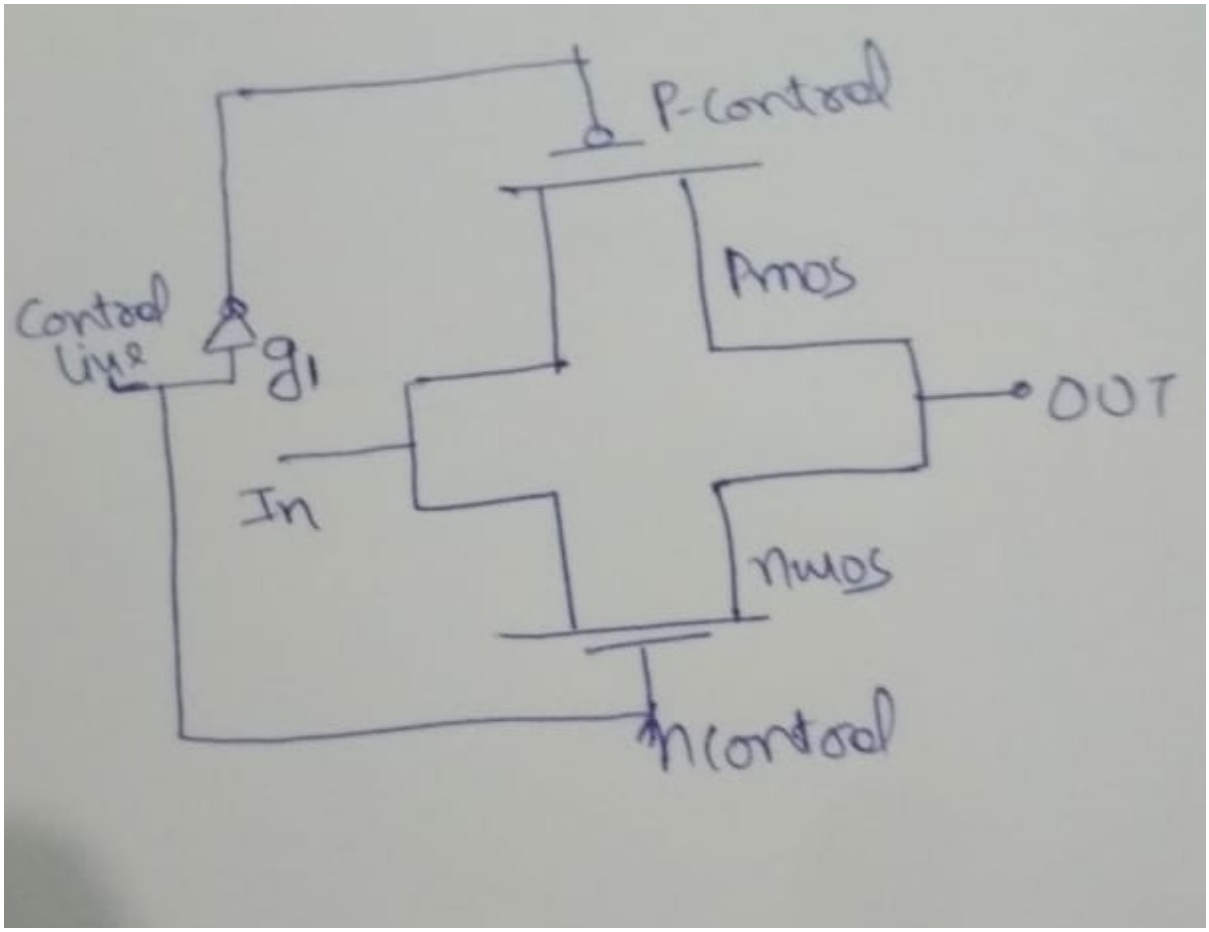
N-control, P-control : control variable

```
module cmos switch1(out,in,n-control, p-control);
output out;
input in, n-control, p-control;
nmos n1 (out,in,n-control);
Pmos p1 (out,in,p-control);
endmodule

module test_cmosswitch1();
reg in, n-control, p-control;
wire out;
initial
begin
in=1'b0;
n-control=1'b1;
p-control=~n-control;
end
always
begin
#3 in=~in;
#3 n-control=~n-control;
#3 p-control=~n-control;
end
initial$monitor($time,"in=%b,n-control=%b, p-control=%b, out=%b, in, n-control,p-control,out);
initial #39 $stop;
```

endmodule

CMOS Switch 2

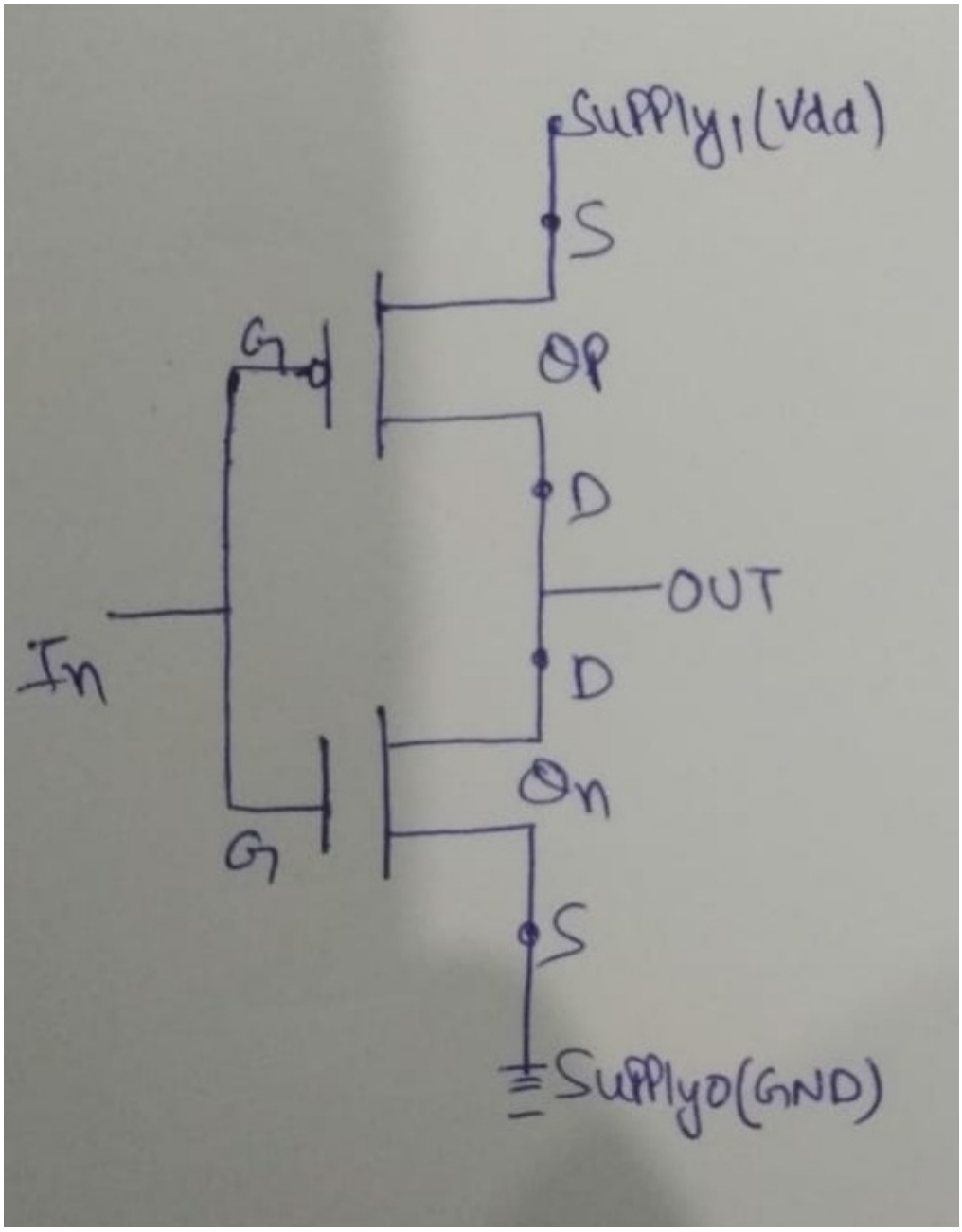



```
module cmosswitch2(out,in,control);
output out;
input in,control;
wire p-control;
not g1 (p-control,control);
Cmos(out,in,control,p-control);
Endmodule

module tst_cmos switch2();
reg in,control;
wire out;
```

```
cmos switch2(out,in,control);
initial
begin
in=1'b0;
control=1'b1;
end
always
#5in=~in;
always
#3control=~control;
initial$monitor($time,"in=%b,control=%b, out=%b, in, control,out);
intial #40 $stop;
endmodule
```

CMOS inverter



Cmos inverter formed by connecting an n-mos and a p-mos switch in series across the supply.the output terminal are joined together to form the common output

Similarly the input is used as the common control input to both the switches

- When the input is low (0) , transistor Q_n is off, but Q_p is on.

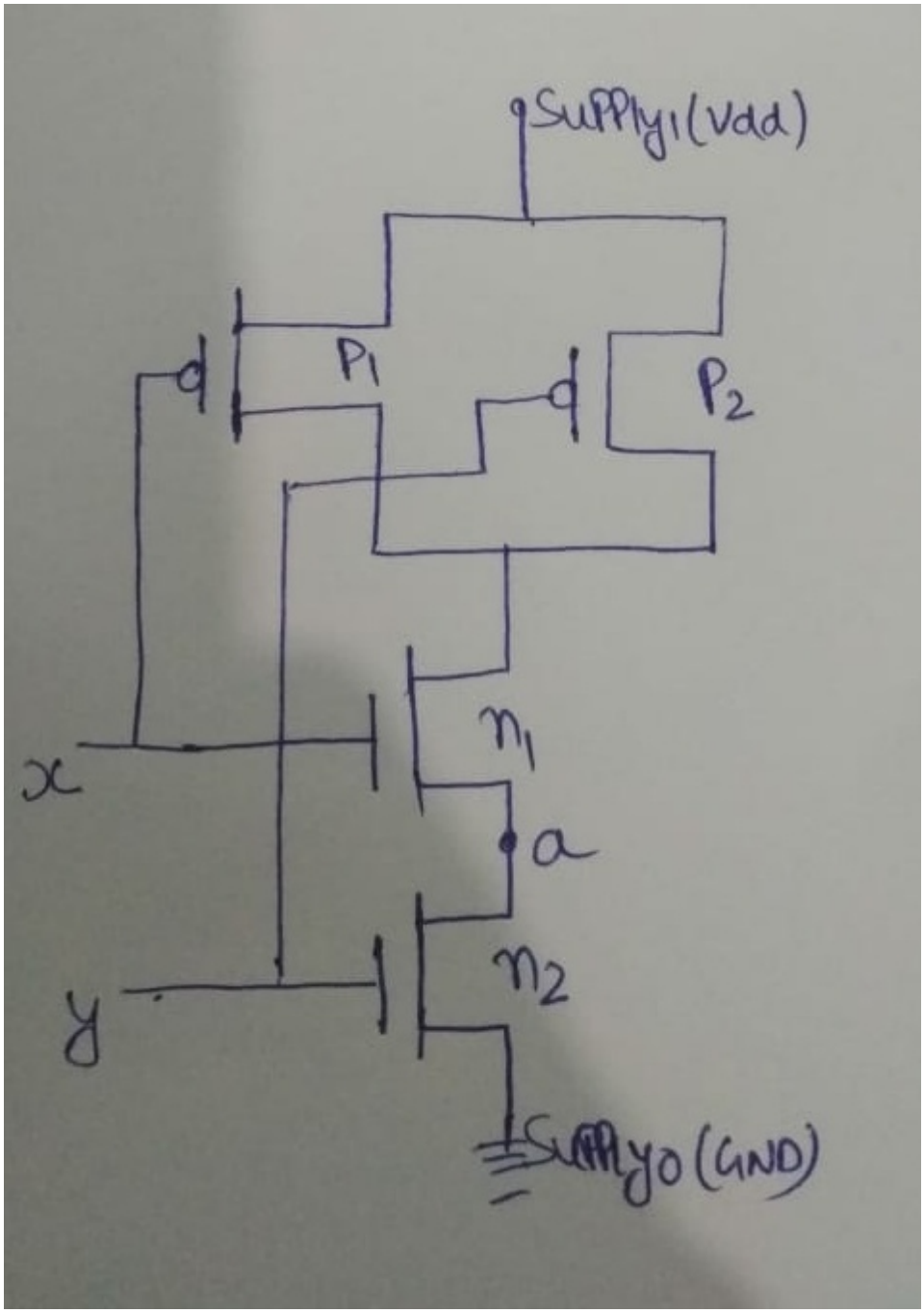
  Supply1 is connected to the output. Hence the output is high

- When the input is high (5v), transistor Q_p is off , but Q_n is on

  Supply 0 is connected to the output hence the output is low.

```
module inv(in,out);
output out;
input in;
//internal declaration a,b declared as nets connected to supply 0 and supply1
Supply 0 a;
Supply 1 b;
nmos(out,a,in);
pmos(out,b,in);
endmodule
```

CMOS nand gate



Cmos nand gate consist of two pmos and two nmos gates

When x and y are highi.e.(5v), then the two pmos p1 ,p2 will be in off condition i.e. open circuited and the two nmos will be on . the output out will be shorted to ground and produce zero output.

If any one of the input is low(0v) corresponding pmos will be shorted and nmos will be opened, output (out) is shorted to V_{dd} i.e high output.

```
Module nand (x,y,out);

Input x,y;
```

output out;

supply 1 V_{dd} ;

supply 0 gnd;

wire a;

pmos p1 (out, V_{dd} ,x);

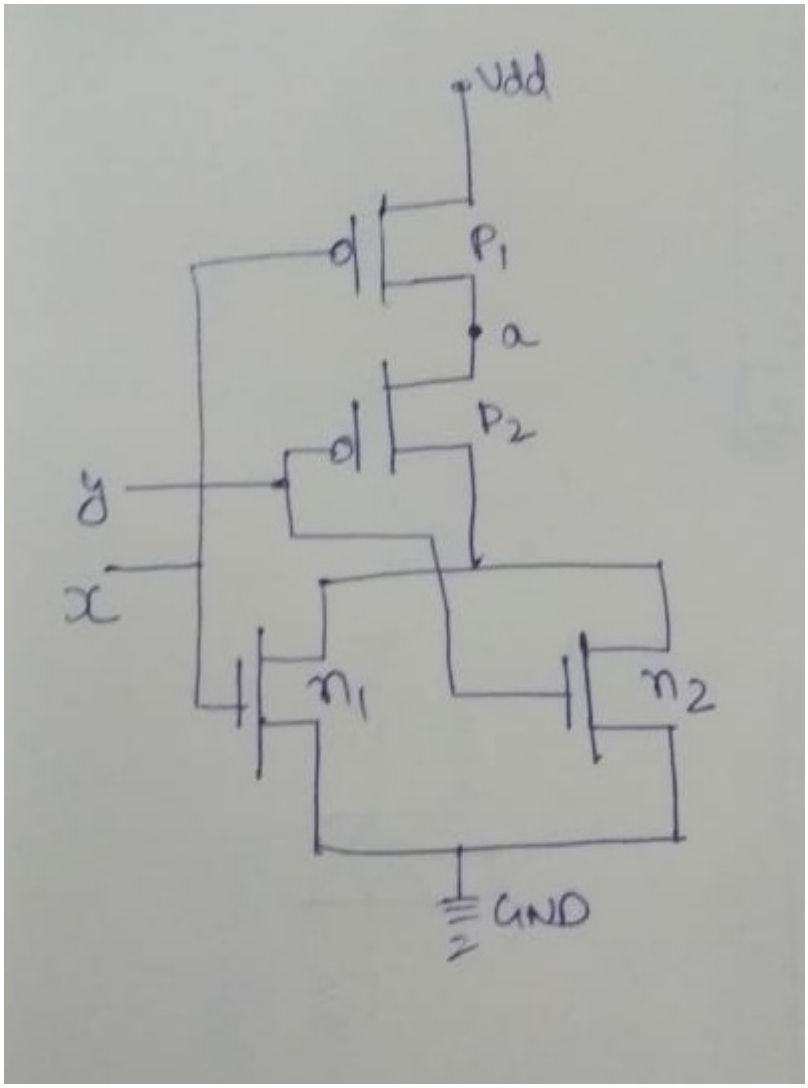
pmos p2(out, V_{dd} ,y);

nmos n1 (out, a ,x);

nmos n2(a,gnd,y);

endmodule

CMOS nor gate



Module nor (x,y,out);

Input x,y;

output out;

supply 1 V_{dd} ;

supply 0 gnd;

wire a;

pmos p1 (a, V_{dd} ,x);

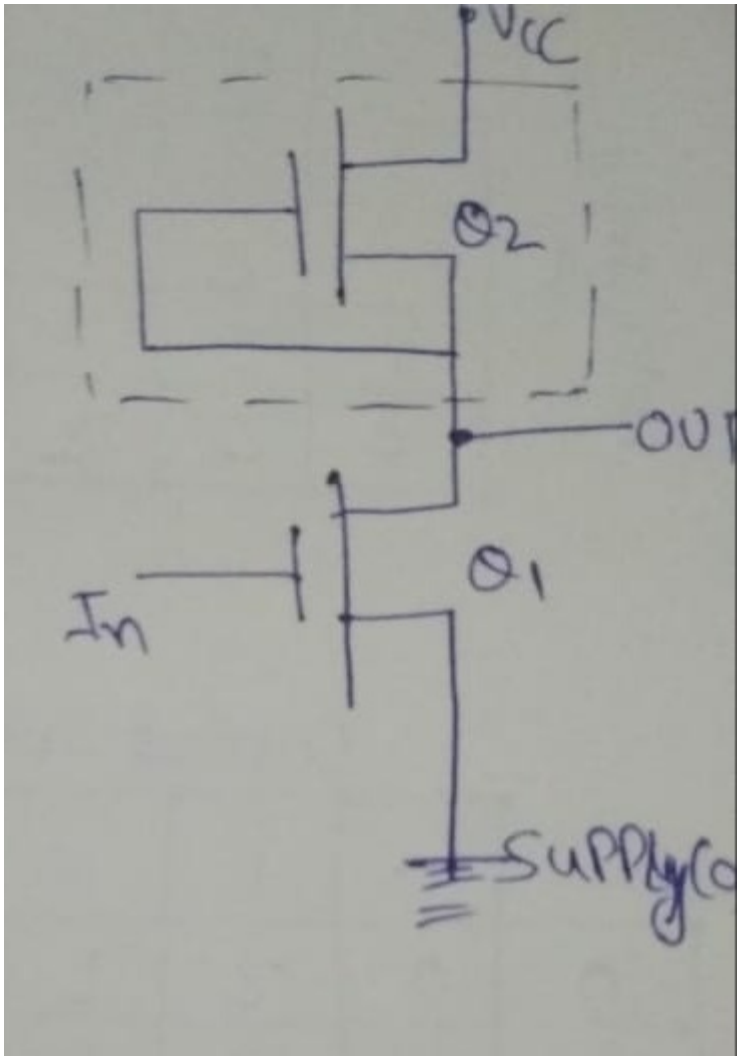
pmos p2(V_{out}, a ,y);

nmos n1 (V_{out},x,gnd);

nmos n2(V_{out},y,gnd);

endmodule

NOS inverter with pull up load



Q1: nmos transistor

Q2 properly biased , forms an active resistance and is load on Q1

When input(in) is zero, Q1 is off, out is pulled up and is at state 1

i.e. out=1

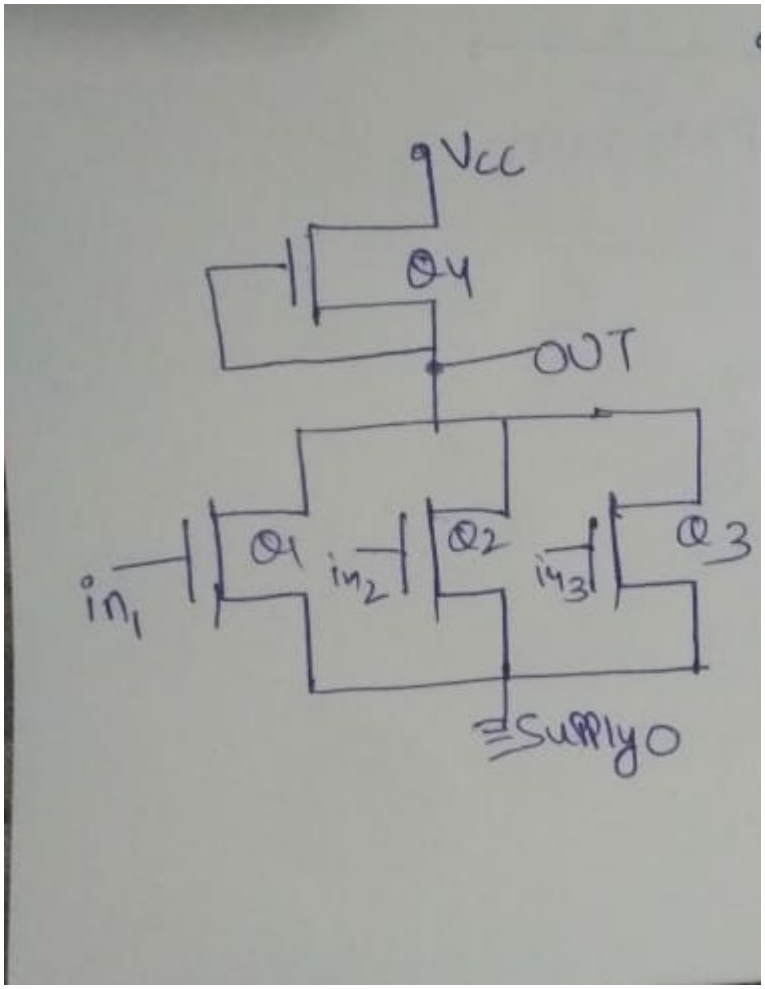
when input (in) is one , Q1 is on , out is pulled down to zero

i.e. out=0

```
module nmosinv(out,in);
output out;
input in,supply 0;
pull up(out);
nmos(out,supply0,in);
endmodule

module tst_nmosinv();
reg in;
wire out;
nmosinv(out,in);
initial
in=1'b1;
always#3 in=~in;
initial $monitor ($time," in= %b, out=%b", in,out);
initial #30 $stop;
endmodule
```

NMOS nor gate with active pull up




```
Module nor3nmos(in1,in2,in3,out);
Input in1,in2,in3;
Output out;
Wire out;
Supply 0 a;
nmos Q1(out,a,in1);
nmos Q2(out,a,in2);
nmos Q3(out,a,in3);
pull up (Out);
endmodule
```

```
module tst_ nor3nmos();
reg in1,in2,in3;
wire out;
```

```
nor3nmos(in1,in2,in3,out);
initial begin
in1=1'b1;
in2=1'b1;
in3=1'b1;
end
always
#2 in1=~in1;
always
#3 in2=~in2;
always
#5 in3=~in3;
Intial $monitor ($time ," in1=%b, in2=%b, in3=%b, out=%b", in1, in2, in3, out);
Initial #24 $stop;
endmodule
```

Bidirectional gates

There are six different switch primitives used in Verilog

i.e.

nmos

pmos

cmos

rnmos

rpmos

rcmos

last three are resistive switches

these all are unidirectional gates , when turned on , the gates establishes a connection and makes the signal at input side available at the output side

Verilog has a set of primitives for bidirectional switches as well.

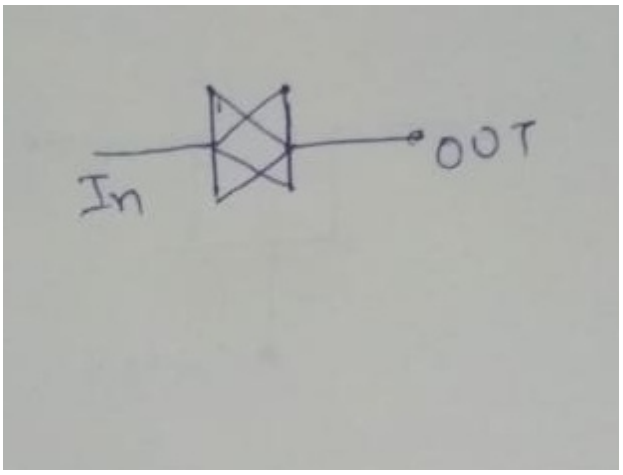
They connect the net on either side when on and isolate them when off

The signal flow can be in either direction

There are six types of bidirectional gates

- tran
- rtran
- tranif1
- rtranif1
- tranif0
- rtranif0

tran



tran gate is bidirectional gate of two ports , when instaniated it connect the two ports directly

thus the instantiation is tran([instance_name](in,out,inout);

tran(s1,s2);

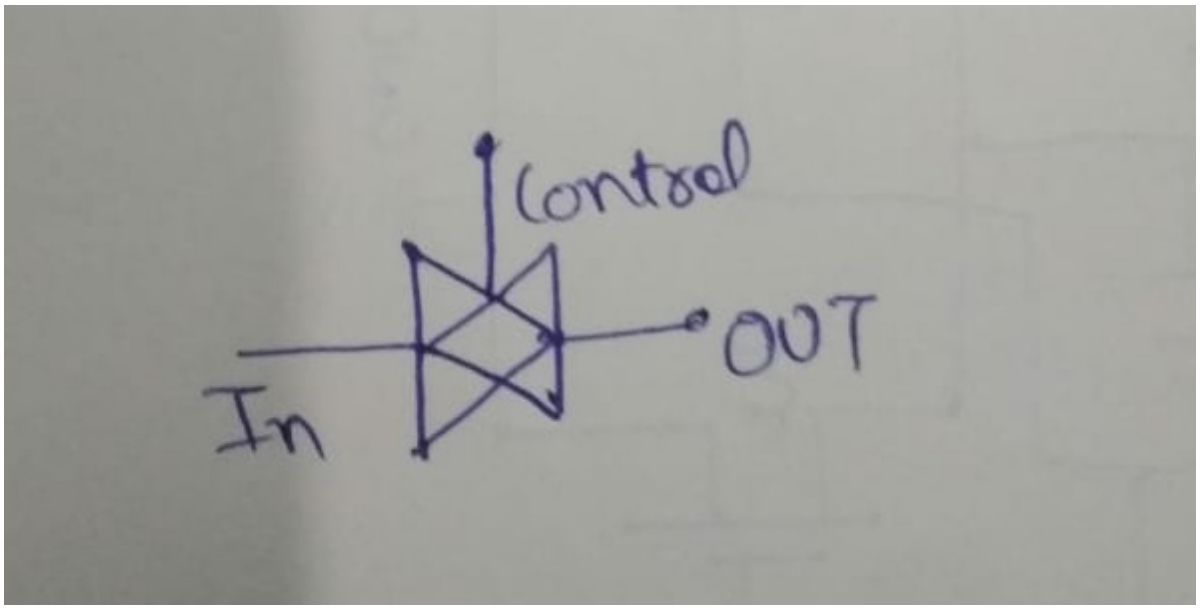
it connect the signal line s1 and s2

either line can be input,inout,output.

rtran

it is resistive counter part of tran.

tranif1



It is bidirectional switch , turned on/off through a contro, line when control signal is 1, it is in on state

And when control signal=0 , switch is in off state

Syntax

tranif([instance_name](in,out,control);

tranif(s1,s2,c);

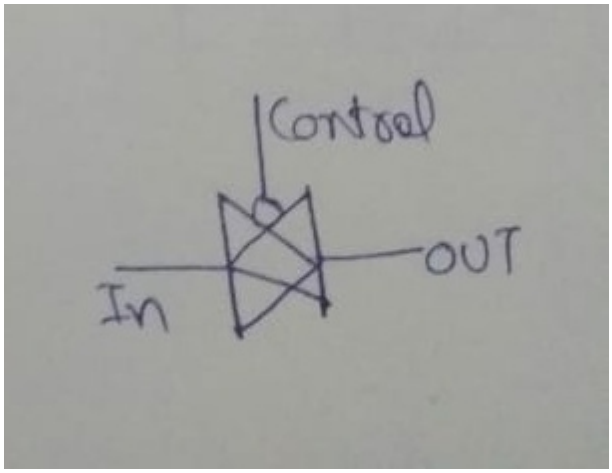
If c=1

S1,s2 are connected and signal transmission can be in either direction

rtranif1

Resistive counter part of tranif1.

tranif0



The action of this switch is reverse of tranif1.

It is off if control=1 and on if control =0

Syntax

tranif0[instance_name](in,out,control);

tranif0(s1,s2,c):

If c=0

S1,s2 are connected and signal transmission can be in either direction

If c=1

Switch is off, s1 and s2 are isolated from each other

TIME DELAYS WITH SWITCH PRIMITIVES

- nmos g1 (out, in, ctrl); has no delay associated with it.
- The instantiation
- nmos (delay1) g2 (out, in, ctrl);

  has delay1 as the delay for the output to rise, fall, and turn OFF.
- nmos (delay_r, delay_f) g3 (out, in, ctrl);

  has delay_r as the rise-time for the output. delay_f is the fall-time for the

output. The turn-off time is zero.

- nmos (delay_r, delay_f, delay_o) g4 (out, in, ctrl);

  has delay_r as the rise-time for the output. delay_f is the fall-time for the output delay_o is the time to turn OFF when the control signal ctrl goes from 0 to 1.

- Delays can be assigned to the other uni-directional gates in a similar manner.

- Bi-directional switches do not delay transmission – their rise- and fall-times are zero. They can have only turn-on and turn-off delays associated with them.

- tran has no delay associated with it.

- tranif1 (delay_r, delay_f) g5 (out, in, ctrl);

  When control changes from 0 to 1, the switch turns on with a delay of delay_r. When control changes from 1 to 0, the switch turns off with a delay of delay_f.

- transif1 (delay0) g2 (out, in, ctrl); represents an instantiation with delay0 as the delay for the switch to turn on when control changes from 0 to 1, with the same delay for it to turn off when control changes from 1 to 0

INSTANTIATIONS WITH STRENGTHS AND DELAYS

nmos (strong1, strong0) (delay_r, delay_f, delay_o) gg (s1, s2, ctrl) ;

rnmos, pmos, and rpmos switches too can be instantiated in the general form in the same manner. The general instantiation for the bi-directional gates too can be done similarly.

STRENGTH CONTENTION WITH TRIREG NETS

nets declared as trireg can have capacitive storage. Such storage can be assigned one of three strengths – large, medium, or small.

Driving such a net from different sources can lead to contention