DVD Project 1
Report

Nikhil 2020102021
Tarun 2020102056
Mitul 2020102026
Atharv 2020102025

```
Tech Node : 32nm MGK
Supply : 0.9 V
Temperature : 25℃
```

Contributions :

- Nikhil : Wrote netlist for the transistors in different modes, contributed in python script for finding leakages
- Tarun : Wrote conversion and formatting code for csv and .txt, stack python script
- Mitul : Contributed in python script for finding leakages.
- Atharv : Made individual python scripts for nmos, pmos.

Stage 1

Problem :

Generation of Matrices of single NMOS and PMOS for on-off conditions with varying widths.

Given :

Tech Node => 32nm MGK

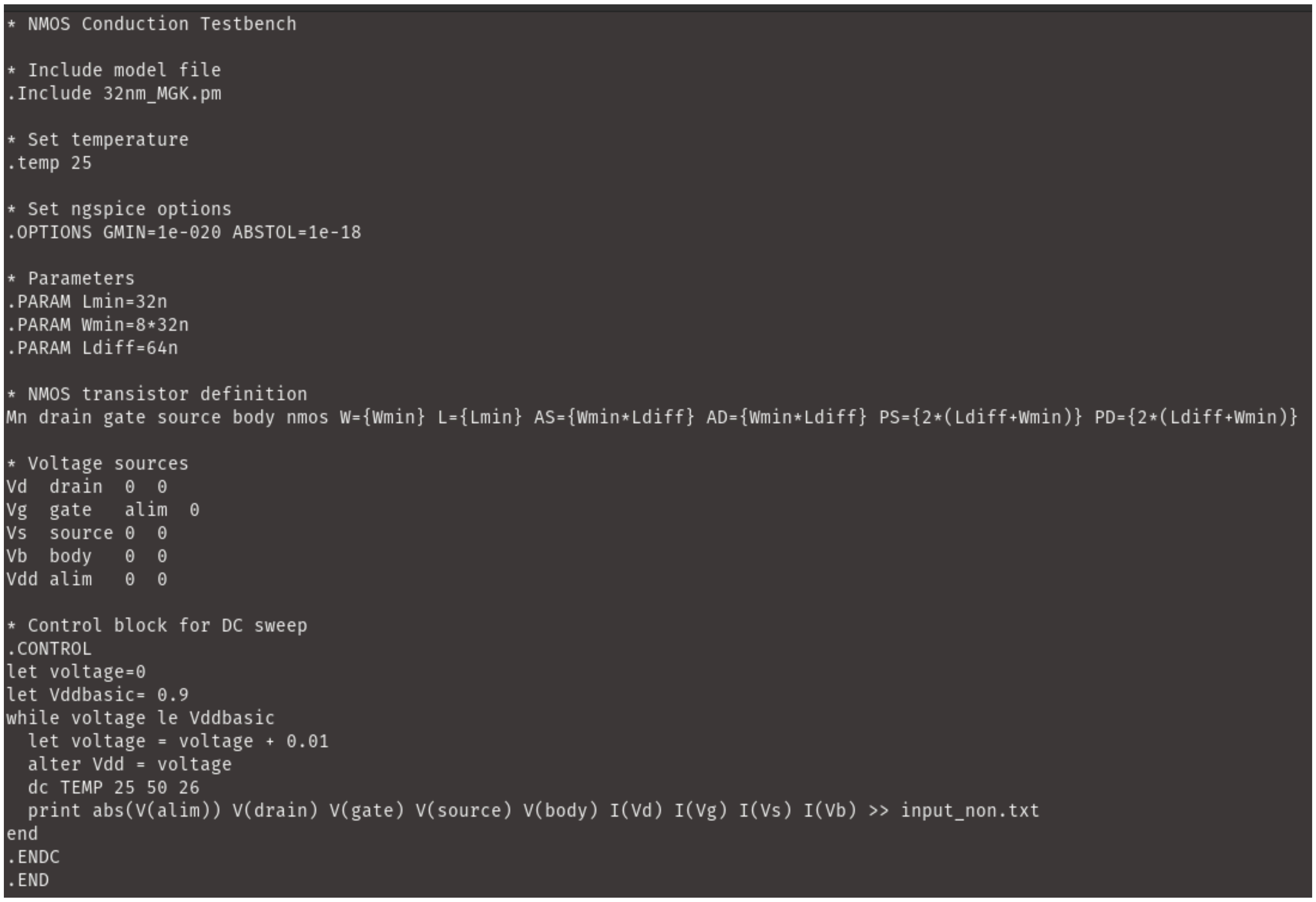
Temperature => 25 ℃

Supply => 0.9 V

Solution :

We understood that we had to systematically analyze the behavior of single NMOS and PMOS transistors under on-off conditions with varying widths. So we used SPICE to simulate the netlist and obtained the results.

Sample code for NMOS-On



```
* NMOS Conduction Testbench

* Include model file
.Include 32nm_MGK.pm

* Set temperature
.temp 25

* Set ngspice options
.OPTIONS GMIN=1e-020 ABSTOL=1e-18

* Parameters
.PARAM Lmin=32n
.PARAM Wmin=8*32n
.PARAM Ldiff=64n

* NMOS transistor definition
Mn drain gate source body nmos W={Wmin} L={Lmin} AS={Wmin*Ldiff} AD={Wmin*Ldiff} PS={2*(Ldiff+Wmin)} PD={2*(Ldiff+Wmin)}

* Voltage sources
Vd  drain  0  0
Vg  gate   alim  0
Vs  source 0  0
Vb  body   0  0
Vdd alim   0  0

* Control block for DC sweep
.CONTROL
let voltage=0
let Vddbasic= 0.9
while voltage le Vddbasic
  let voltage = voltage + 0.01
  alter Vdd = voltage
  dc TEMP 25 50 26
  print abs(V(alim)) V(drain) V(gate) V(source) V(body) I(Vd) I(Vg) I(Vs) I(Vb) >> input_non.txt
end
.ENDC
.END
```

Logic :

- We loop in for all the given parameters by changing the ground voltage by a single step, where our step is 0.1 V and Vdd is 0.9 V. This Process is done for all the desired widths. All desired values like leakage currents are mentioned in the files.

- The results are saved in a file corresponding to the MOSFET type and configuration. The naming has been taken care of for all files.

Stage 2

Problem:

Generation of Matrices of NMOS and PMOS stacks of 2 transistors since the final circuit will contain gates with a maximum of 2-input.

Given :

Tech Node => 32nm MGK

Temperature => 25 ℃

Supply => 0.9 V

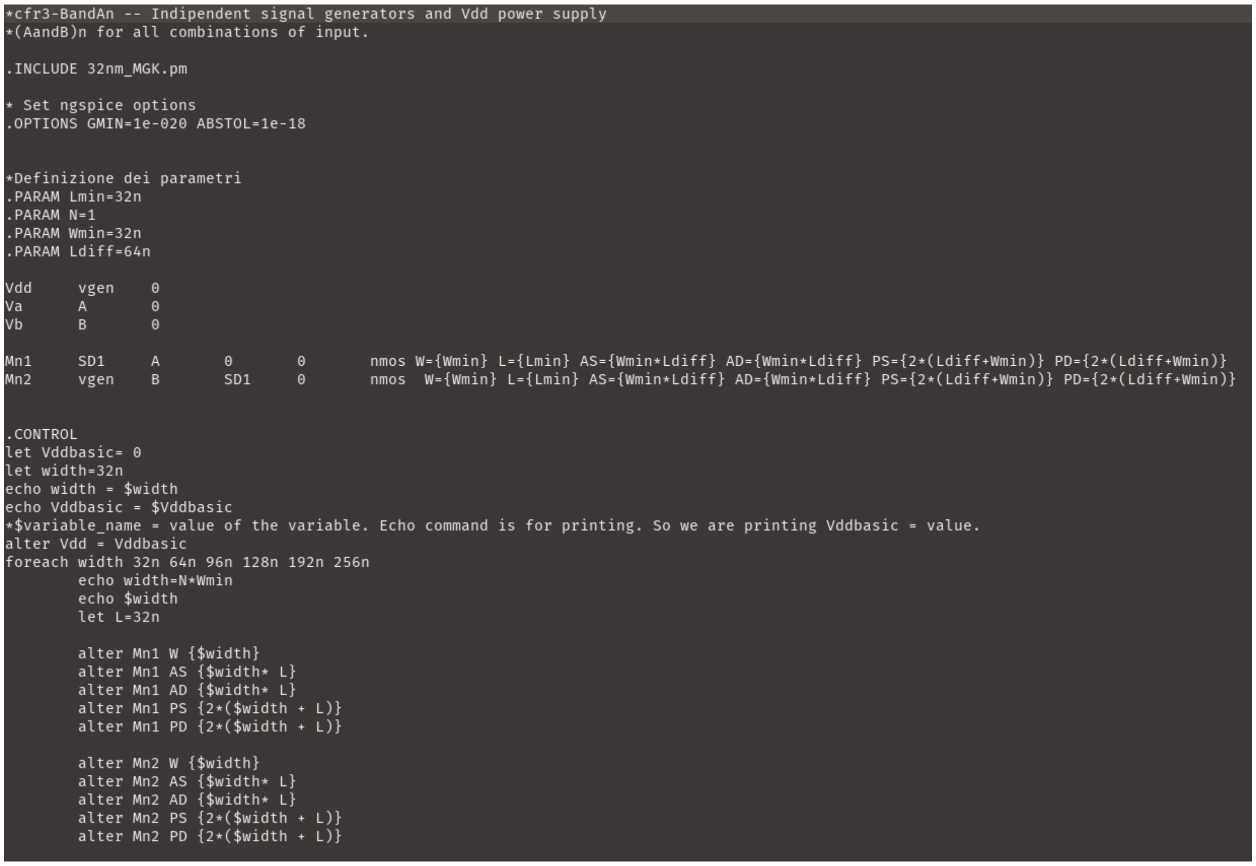
NMOS stacks and PMOS stacks netlist to be generated.

Solution:

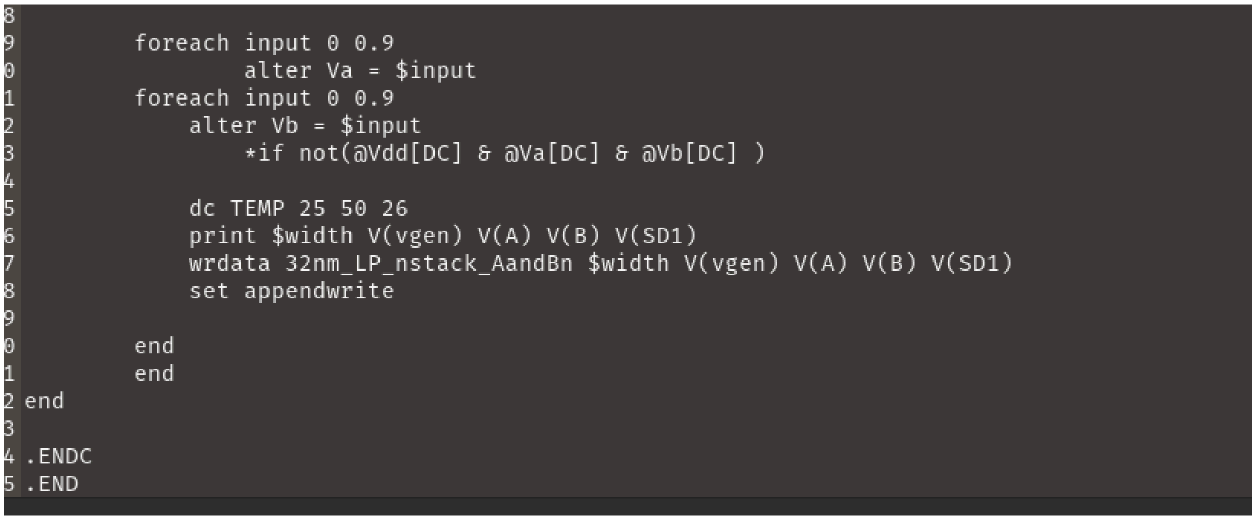
First we note the Pull Up (PMOS) and Pull Down (NMOS) about their working and how they behave on different inputs namely : 01,00,10,11, for each of the two gate inputs in both stacks. Now this process is done for all the required widths : 32n 64n 96n 128n 192n 256n. So our netlist delivers us a total of 4*2*6 different results for these configurations.

Our interest mainly lies in the **Intermediate node** and its voltage which will help us to derive at r**esults using previously noted results of stage-1.**

Sample Code for N-Stack.:



```
*cfr3-BandAn -- Indipendent signal generators and Vdd power supply
*(AandB)n for all combinations of input.

.INCLUDE 32nm_MGK.pm

* Set ngspice options
.OPTIONS GMIN=1e-020 ABSTOL=1e-18


*Definizione dei parametri
.PARAM Lmin=32n
.PARAM N=1
.PARAM Wmin=32n
.PARAM Ldiff=64n

Vdd     vgen    0
Va      A       0
Vb      B       0

Mn1     SD1     A       0       0       nmos W={Wmin} L={Lmin} AS={Wmin*Ldiff} AD={Wmin*Ldiff} PS={2*(Ldiff+Wmin)} PD={2*(Ldiff+Wmin)}
Mn2     vgen    B       SD1     0       nmos  W={Wmin} L={Lmin} AS={Wmin*Ldiff} AD={Wmin*Ldiff} PS={2*(Ldiff+Wmin)} PD={2*(Ldiff+Wmin)}


.CONTROL
let Vddbasic= 0
let width=32n
echo width = $width
echo Vddbasic = $Vddbasic
*$variable_name = value of the variable. Echo command is for printing. So we are printing Vddbasic = value.
alter Vdd = Vddbasic
foreach width 32n 64n 96n 128n 192n 256n
        echo width=N*Wmin
        echo $width
        let L=32n

        alter Mn1 W {$width}
        alter Mn1 AS {$width* L}
        alter Mn1 AD {$width* L}
        alter Mn1 PS {2*($width + L)}
        alter Mn1 PD {2*($width + L)}

        alter Mn2 W {$width}
        alter Mn2 AS {$width* L}
        alter Mn2 AD {$width* L}
        alter Mn2 PS {2*($width + L)}
        alter Mn2 PD {2*($width + L)}
```



```

        foreach input 0 0.9
                alter Va = $input
        foreach input 0 0.9
            alter Vb = $input
                *if not(@Vdd[DC] & @Va[DC] & @Vb[DC] )

            dc TEMP 25 50 26
            print $width V(vgen) V(A) V(B) V(SD1)
            wrdata 32nm_LP_nstack_AandBn $width V(vgen) V(A) V(B) V(SD1)
            set appendwrite

        end
        end
end

.ENDC
.END
```

Logic :
As shown in the code, we vary our gate parameters for each of the gate inputs which are then varied for each of the gate widths, resulting in a cluster of different values which we can use to estimate leakage currents for other states of a gate in different voltage configurations.

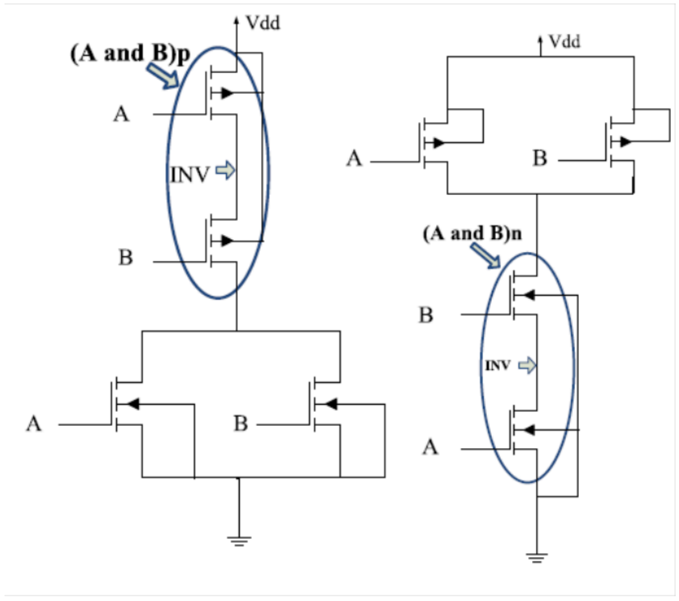
Stage 3:

Problem :
A script that can estimate the leakage of ISCAS 74182 circuit using the matrices generated in earlier two stages. ISCAS 74182 circuit consists of 3 or 4-i/p gates as well. Replace the 3 or 4-i/p gates with 2-i/p gates. The circuit netlist must also be designed in NGSPICE for cross verification.

Given :
Tech Node => 32nm MGK
Temperature => 25 ℃
Supply => 0.9 V
NMOS stacks and PMOS stacks netlist results and Stage 1 results.

Solution:

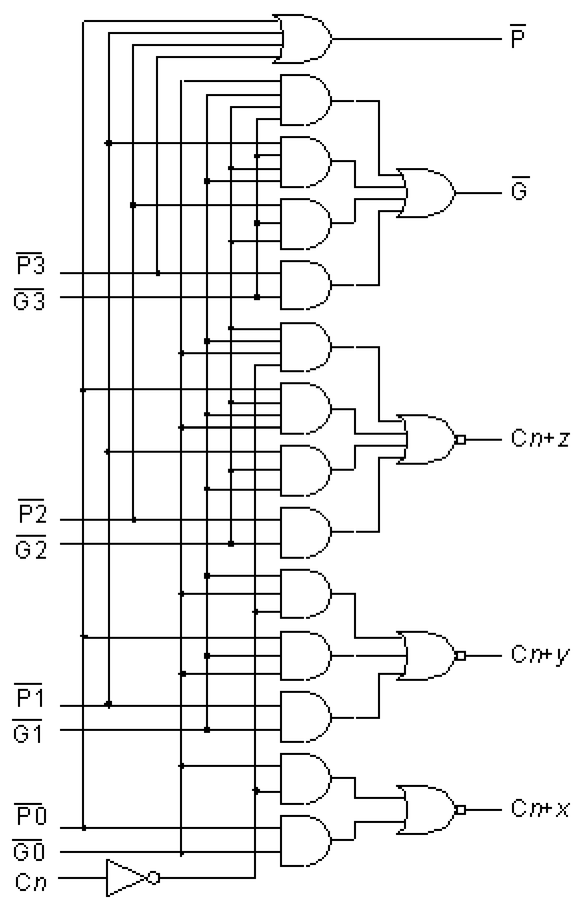
Here we can observe that for the given 2 logic gates, our PMOS and NMOS stacks act as a part and the results from them can be used for simplifying the calculations and solving the bigger circuit problem (ISCAS 74182).



We also note that the supply and Ground for all gates in our given problem circuit are independent which implies their leakage currents can also be considered independent. We use this conclusion to make our working more modular and solve smaller problems first and then build up to obtain the results.

Circuit :

74182 Gate-Level Schematic



Logic and Working:

Here the methodology used is we broke 3 input and gate into 2 -2 input and gate and 4 input and gate into 3-2 input and gate. Similarly for 4 input or gate we used 3 -2 input or gate.

We then calculated the leakage current for each gate by using previous stages where we calculated individual leakage currents and cumulative of them resulted in final leakage current.

Code for OR gate:

```python
def org(N, A, B):
   Isub=0
   Ib=0
   Ig=0
   LeakPower=0

   LeakPower_nor, Isub_nor, Ib_nor, Ig_nor = nor(N, A, B)
   if A == 0 & B == 0:
       LeakPower_inv, Isub_inv, Ib_inv, Ig_inv = inv( N,1)
   else:
       LeakPower_inv, Isub_inv, Ib_inv, Ig_inv = inv(N, 0)

   LeakPower = LeakPower_nor + LeakPower_inv
   Isub = Isub_nor + Isub_inv
   Ib = Ib_nor + Ib_inv
   Ig = Ig_nor + Ig_inv

   return LeakPower, Isub, Ib, Ig
```

In this above snippet it shows working of OR gate and written leakage current of 2 input or gate using values which we got in stage1,2.

Code for calculating Cy:

```
def circy(P0,G0,P1,G1,P2,G2,P3,G3,C,N):
   Isub=0
   Ib=0
   Ig=0
   LeakPower=0

   LeakPower_inv, Isub_inv, Ib_inv, Ig_inv = inv(N, C)

   LeakPower_and0, Isub_and0, Ib_and0, Ig_and0 = andg(N, P1, G1)
   LeakPower_and1, Isub_and1, Ib_and1, Ig_and1 = andg(N, G1, G0)
   LeakPower_and2, Isub_and2, Ib_and2, Ig_and2 = andg(N, G0 and G1, P0)
   LeakPower_and3, Isub_and3, Ib_and3, Ig_and3 = andg(N, G1, G0)
   LeakPower_and4, Isub_and4, Ib_and4, Ig_and4 = andg(N, G0 and G1, 1-C)


   LeakPower_nor0, Isub_nor0, Ib_nor0, Ig_nor0 = nor(N,P1 and G1, G0 and
G1 and P0)
   LeakPower_nor1, Isub_nor1, Ib_nor1, Ig_nor1 = nor(N,G1 and G0 and
(1-C), 1 - ((P1 and G1) or (G0 and G1 and P0) ))
   LeakPower =
LeakPower_inv+LeakPower_and0+LeakPower_and1+LeakPower_nor0+LeakPower_and3+
LeakPower_and4+LeakPower_nor1+LeakPower_and2
   Isub = Isub_and0+Isub_and1+Isub_inv+Isub_nor0 + Isub_and2 +Isub_and3
+Isub_and4 +Isub_nor1
   Ib = Ib_and0 + Ib_and1 + Ib_inv +
Ib_nor0+Ib_nor1+Ib_and2+Ib_and3+Ib_and4
   Ig = Ig_and0 + Ig_and1 + Ig_inv +
Ig_nor0+Ig_nor1+Ig_and2+Ig_and3+Ig_and4

   return LeakPower, Isub, Ib, Ig
```

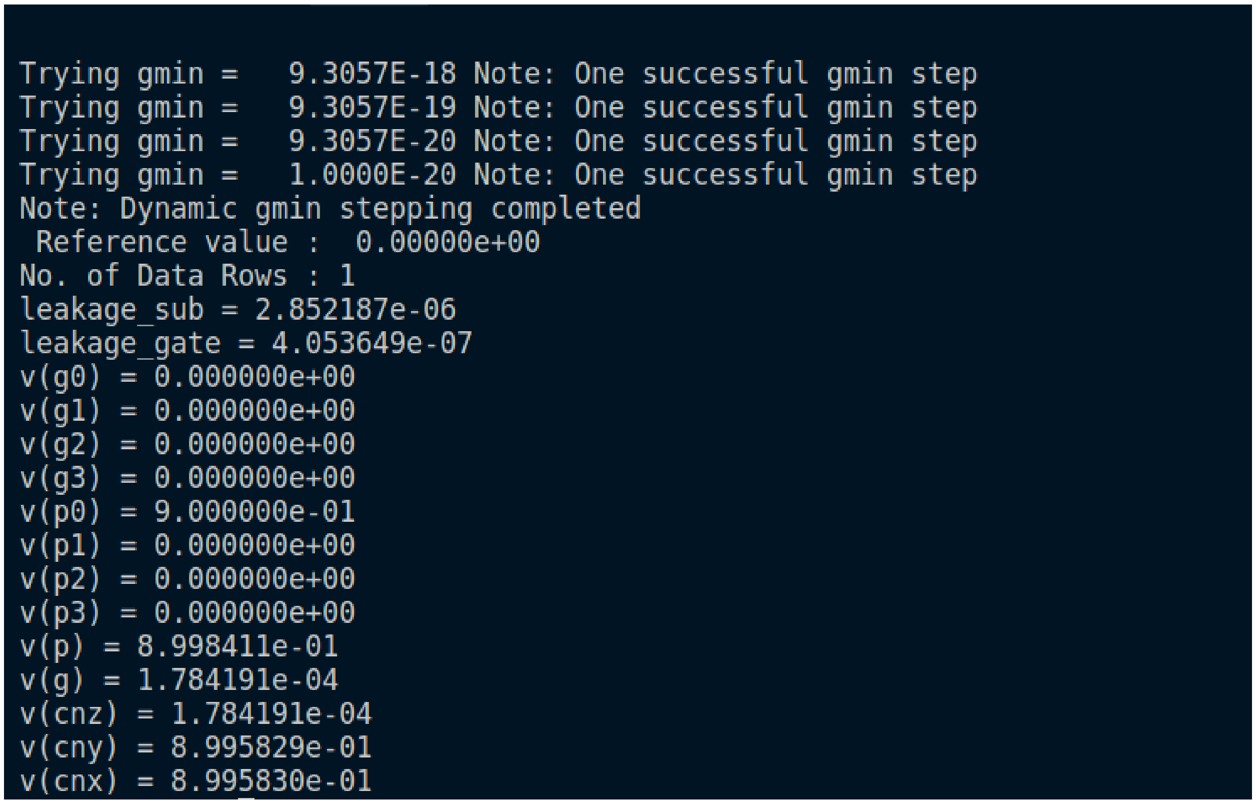
For calculating Cy we can see we have used 8 gates and calculated individual leakage current and cumulative of them represents total leakage current for getting Cy.

Final Code for whole circuit:

```
def circuit(P0,G0,P1,G1,P2,G2,P3,G3,C,N,voltage):
   Isub=0
   Ib=0
   Ig=0
   LeakPower=0

   LeakPower_P, Isub_P, Ib_P, Ig_P = cirp( P0,P1,P2,P3,N)
   LeakPower_G, Isub_G, Ib_G, Ig_G = cirg(P0,G0,P1,G1,P2,G2,P3,G3,N)
   LeakPower_Cx, Isub_Cx, Ib_Cx, Ig_Cx =
circx(P0,G0,P1,G1,P2,G2,P3,G3,C,N)
   LeakPower_Cy, Isub_Cy, Ib_Cy, Ig_Cy =
circy(P0,G0,P1,G1,P2,G2,P3,G3,C,N)
   LeakPower_Cz, Isub_Cz, Ib_Cz, Ig_Cz =
circz(P0,G0,P1,G1,P2,G2,P3,G3,C,N)

   LeakPower = LeakPower_P + LeakPower_G + LeakPower_Cx + LeakPower_Cy +
LeakPower_Cz
   Isub = Isub_P + Isub_G + Isub_Cx + Isub_Cy + Isub_Cz
   Ib = Ib_P + Ib_G + Ib_Cx + Ib_Cy + Ib_Cz
   Ig = Ig_P + Ig_G + Ig_Cx + Ig_Cy + Ig_Cz

   return LeakPower, Isub, Ib, Ig
```

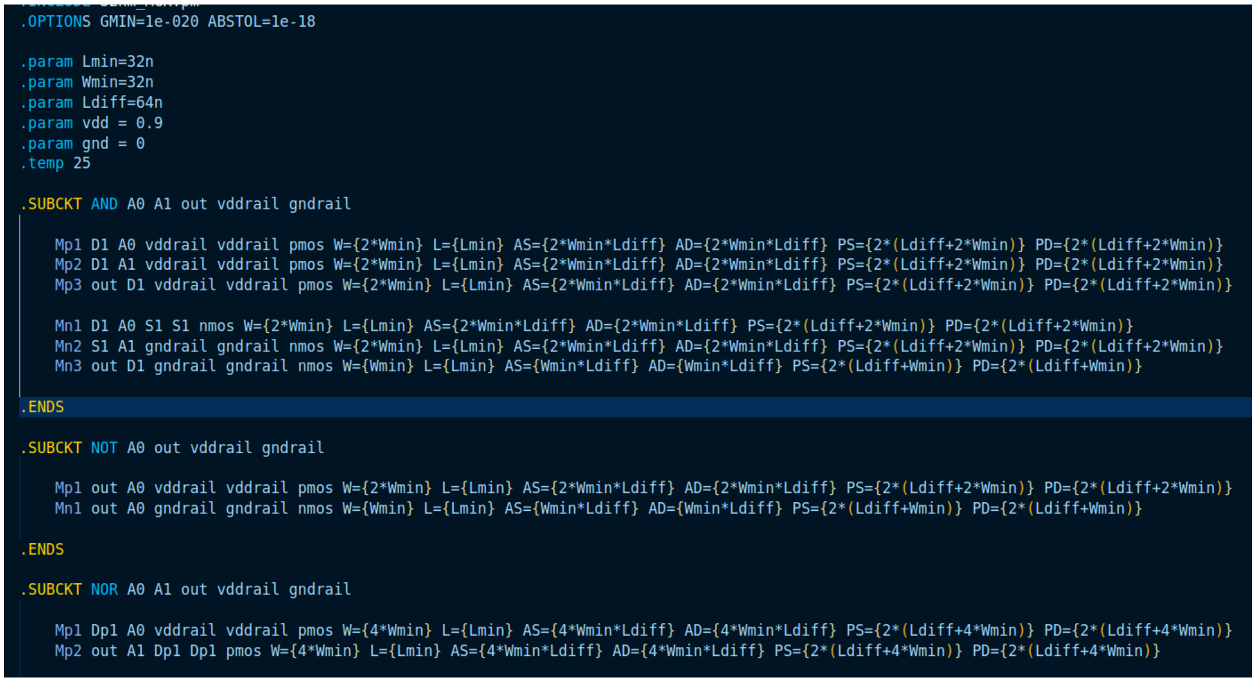
In this snippet we can see that we have divided the circuit into 5 smaller circuit and calculated leakage current individualy. We have written modular code so that it is easy to understand.

final.net output



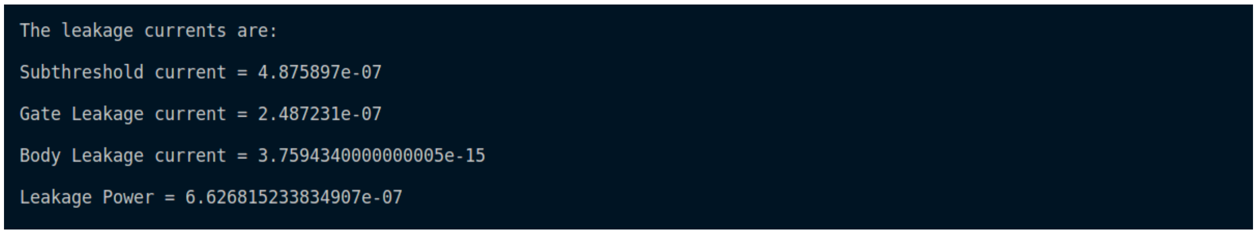
```
Trying gmin =   9.3057E-18 Note: One successful gmin step
Trying gmin =   9.3057E-19 Note: One successful gmin step
Trying gmin =   9.3057E-20 Note: One successful gmin step
Trying gmin =   1.0000E-20 Note: One successful gmin step
Note: Dynamic gmin stepping completed
 Reference value :  0.00000e+00
No. of Data Rows : 1
leakage_sub = 2.852187e-06
leakage_gate = 4.053649e-07
v(g0) = 0.000000e+00
v(g1) = 0.000000e+00
v(g2) = 0.000000e+00
v(g3) = 0.000000e+00
v(p0) = 9.000000e-01
v(p1) = 0.000000e+00
v(p2) = 0.000000e+00
v(p3) = 0.000000e+00
v(p) = 8.998411e-01
v(g) = 1.784191e-04
v(cnz) = 1.784191e-04
v(cny) = 8.995829e-01
v(cnx) = 8.995830e-01
```

final.net snippet



```
.OPTIONS GMIN=1e-020 ABSTOL=1e-18

.param Lmin=32n
.param Wmin=32n
.param Ldiff=64n
.param vdd = 0.9
.param gnd = 0
.temp 25

.SUBCKT AND A0 A1 out vddrail gndrail

    Mp1 D1 A0 vddrail vddrail pmos W={2*Wmin} L={Lmin} AS={2*Wmin*Ldiff} AD={2*Wmin*Ldiff} PS={2*(Ldiff+2*Wmin)} PD={2*(Ldiff+2*Wmin)}
    Mp2 D1 A1 vddrail vddrail pmos W={2*Wmin} L={Lmin} AS={2*Wmin*Ldiff} AD={2*Wmin*Ldiff} PS={2*(Ldiff+2*Wmin)} PD={2*(Ldiff+2*Wmin)}
    Mp3 out D1 vddrail vddrail pmos W={2*Wmin} L={Lmin} AS={2*Wmin*Ldiff} AD={2*Wmin*Ldiff} PS={2*(Ldiff+2*Wmin)} PD={2*(Ldiff+2*Wmin)}

    Mn1 D1 A0 S1 S1 nmos W={2*Wmin} L={Lmin} AS={2*Wmin*Ldiff} AD={2*Wmin*Ldiff} PS={2*(Ldiff+2*Wmin)} PD={2*(Ldiff+2*Wmin)}
    Mn2 S1 A1 gndrail gndrail nmos W={2*Wmin} L={Lmin} AS={2*Wmin*Ldiff} AD={2*Wmin*Ldiff} PS={2*(Ldiff+2*Wmin)} PD={2*(Ldiff+2*Wmin)}
    Mn3 out D1 gndrail gndrail nmos W={Wmin} L={Lmin} AS={Wmin*Ldiff} AD={Wmin*Ldiff} PS={2*(Ldiff+Wmin)} PD={2*(Ldiff+Wmin)}

.ENDS

.SUBCKT NOT A0 out vddrail gndrail

    Mp1 out A0 vddrail vddrail pmos W={2*Wmin} L={Lmin} AS={2*Wmin*Ldiff} AD={2*Wmin*Ldiff} PS={2*(Ldiff+2*Wmin)} PD={2*(Ldiff+2*Wmin)}
    Mn1 out A0 gndrail gndrail nmos W={Wmin} L={Lmin} AS={Wmin*Ldiff} AD={Wmin*Ldiff} PS={2*(Ldiff+Wmin)} PD={2*(Ldiff+Wmin)}

.ENDS

.SUBCKT NOR A0 A1 out vddrail gndrail

    Mp1 Dp1 A0 vddrail vddrail pmos W={4*Wmin} L={Lmin} AS={4*Wmin*Ldiff} AD={4*Wmin*Ldiff} PS={2*(Ldiff+4*Wmin)} PD={2*(Ldiff+4*Wmin)}
    Mp2 out A1 Dp1 Dp1 pmos W={4*Wmin} L={Lmin} AS={4*Wmin*Ldiff} AD={4*Wmin*Ldiff} PS={2*(Ldiff+4*Wmin)} PD={2*(Ldiff+4*Wmin)}
```

Results of python current leakages



```
The leakage currents are:

Subthreshold current = 4.875897e-07

Gate Leakage current = 2.487231e-07

Body Leakage current = 3.7594340000000005e-15

Leakage Power = 6.626815233834907e-07
```

Conclusion:
Here in python we separately called all the gates and added their leakages to get the final leakage and by ngspice we got exact leakage for certain width and 000000000 case, now we observe that the difference in gate leakage is of order 10^-7 which is very less and hence our calculations are correct.