折叠共源共栅放大器设计实验报告

杨修齐 201180044 集成电路与集成系统专业

**摘要：**折叠式共源共栅运算放大器因其较大的输出摆幅和偏置电压的较低，同时具有较高的输出阻抗，其电流利用率只有套筒式共源共栅的一半左右等优点。在ADC模块以及其他电路中具有重要的地位。在本课程中，选择使用折叠共源共栅放大器进行两级放大器的设计，并且满足要求的指标。

1. 2022实验设计要求

设计一个两级运算放大器，其性能指标为：

|  |  |
| --- | --- |
| 参数 | 性能要求 |
|  |  |
|  |  |
|  | *>3V/us* |
|  | *1.25V to 2.5V* |
|  | *45°* |
|  | *0.4V< <2.6V* |
|  | *≤5mW* |
|  | *> 5000* |
|  | *10pF* |

1. MOS管的工艺参数
2. NMOS的工艺参数

首先将NMOS管（n33）按如图方式接入电路中：

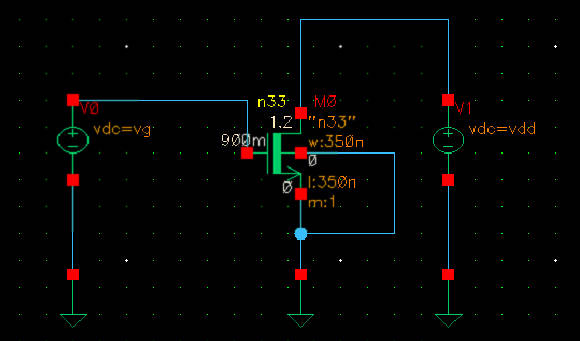


图1-1 NMOS\_TEST 测试n33 NMOS工艺参数

设置V0、V1（vdc）分别为vg，即为DC分析中的变量，并进行直流分析。再使用ADE中的Results-Print-Model Parameters，单击NMOS查看NMOS管的参数。

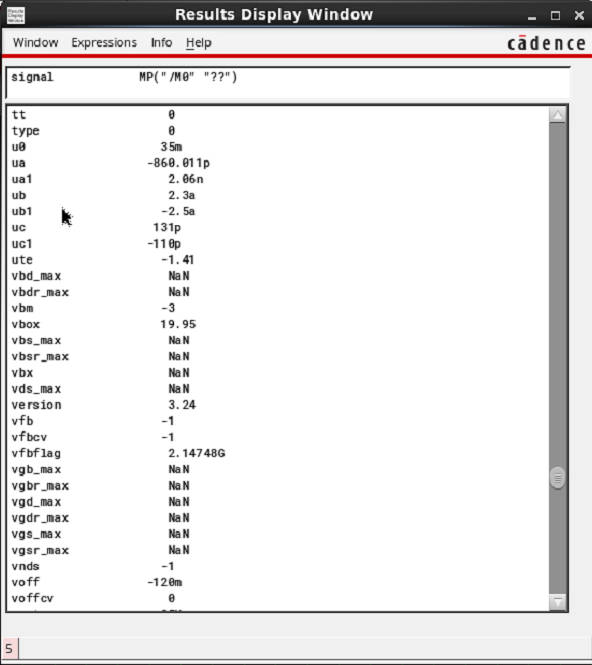


图1-2 在Results Display Window中查看相关参数







图1-3 NMOS管参数

从图中可以看出NMOS的等效栅氧厚度为，阈值电压，电子迁移率 。

经过计算可以得出：（以下全部使用国际单位制进行表示）。

其中为的介电常数，

1. PMOS的工艺参数

类似地，将PMOS管（p33）按如图方式接入电路中：

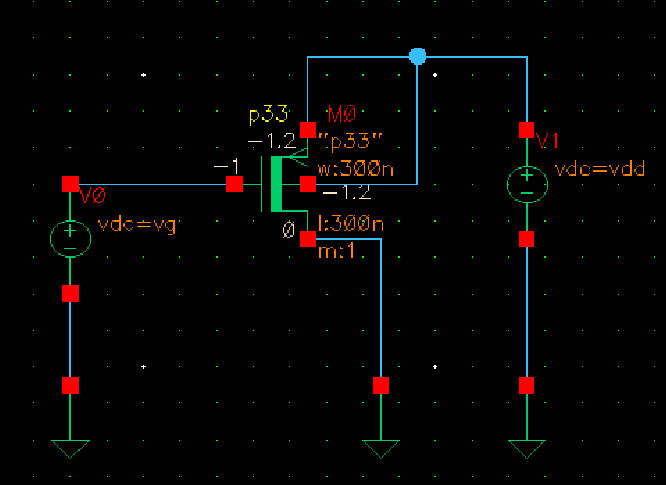
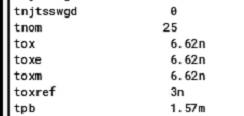
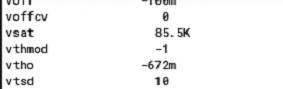


图1-4 PMOS\_TEST 测试n33 NMOS工艺参数





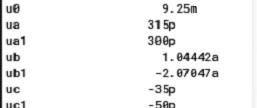


图1-5 PMOS管参数

从图中可以看出PMOS的等效栅氧厚度为，阈值电压，电子迁移率 。

经过计算可以得出：。

表1 NMOS、PMOS参数列表

|  |  |  |
| --- | --- | --- |
|  | NMOS | PMOS |
|  |  |  |
|  |  |  |
| / |  |  |
| / |  |  |

1. 放大器理论计算

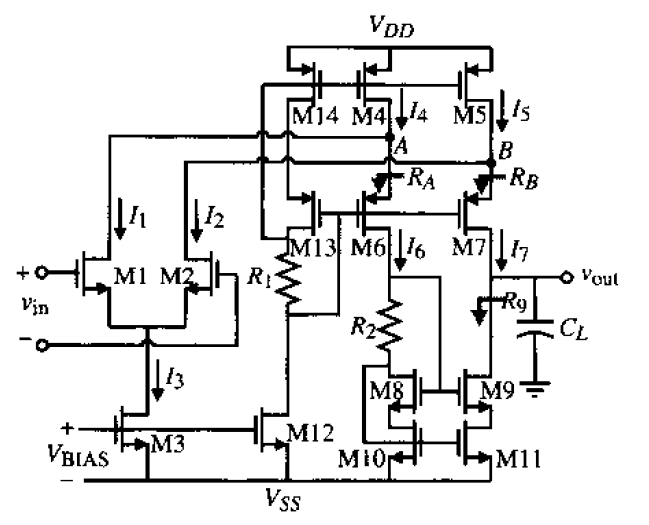


图3-1 折叠式共源共栅放大器结构图（Allen中文版P244）

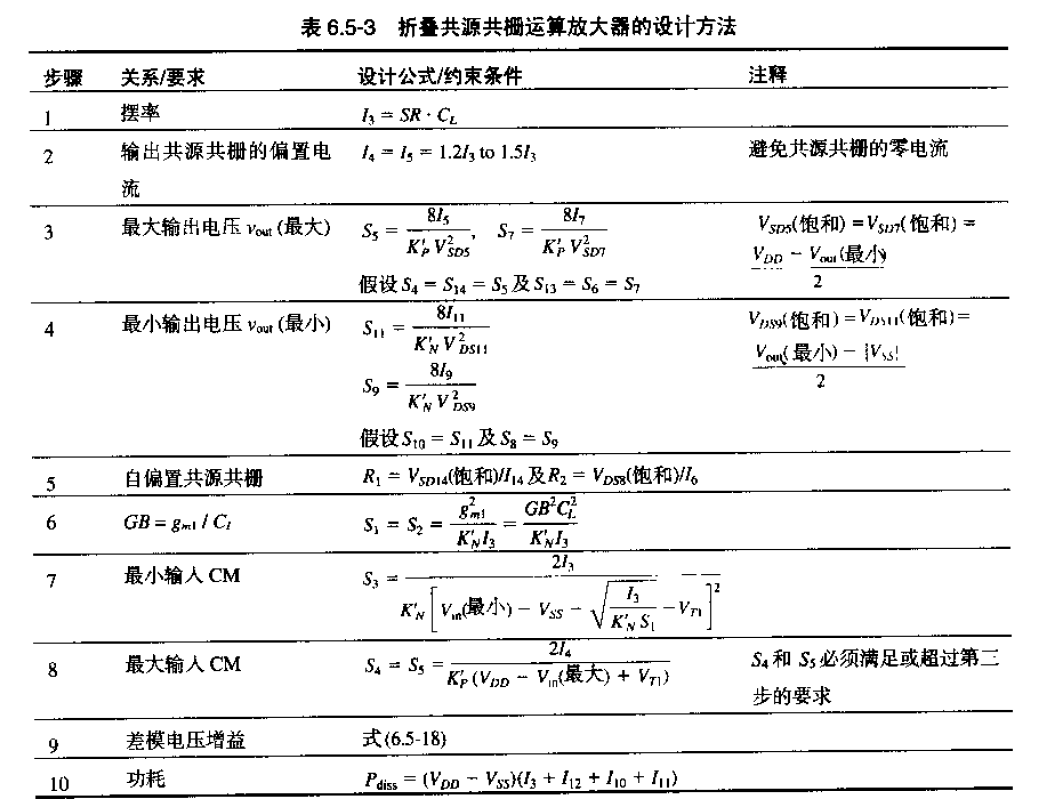


图3-2 折叠式共源共栅的设计方法（Allen中文版P247）

在实际计算中使用calculator.py进行整体调参，在calculator.py程序中 使用的都是国际单位制方便进行运算处理。

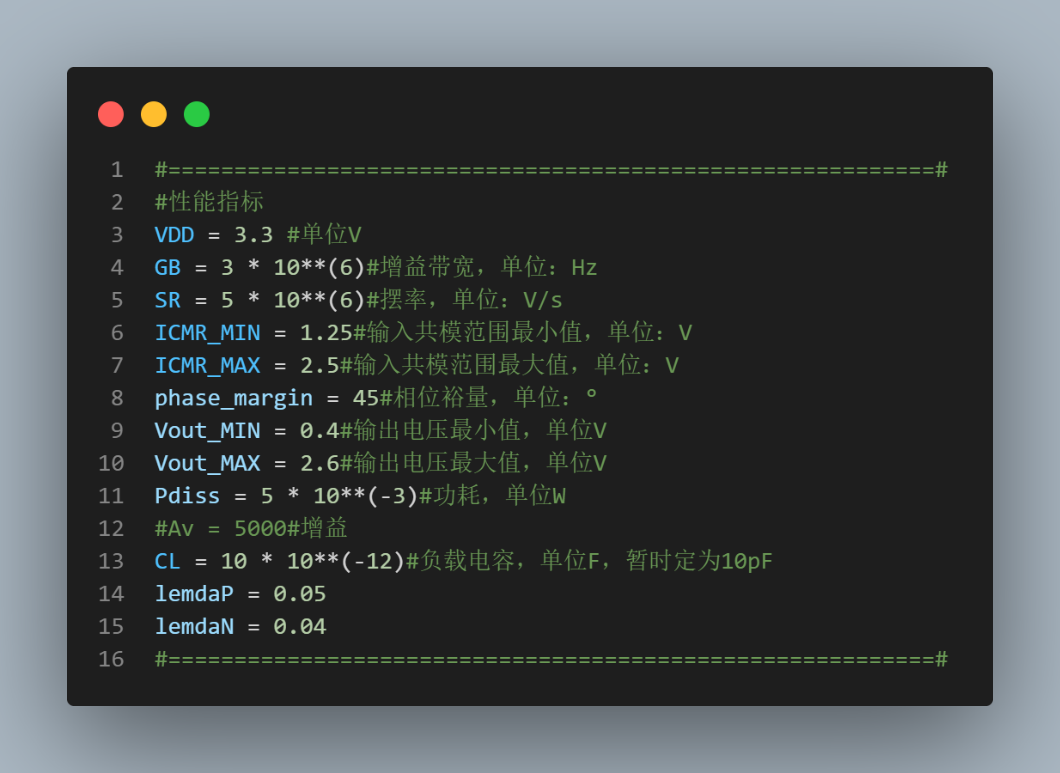


图3-3 设计参数

特别的，本次设计中由于，在实际设计中取,过小会导致后续设计中所有MOS管的宽长比过小。

1. **计算**
2. **确定**

取rate=1.25

则

1. **确定M4、M5、M14的参数**

由条件进行计算：

假设M6、M7电流处在最差的情况：

1. **确定M8、M9、M10、M11参数**

由条件进行计算：

1. **确定M1、M2管参数**
2. **确定M3管参数**

由最小共模输入电压可以确定, 进行运算：

1. **验证M4、M5是否满足最大输入共模电压**

由最大输入共模电压可以确定, 进行运算：

所以

1. **计算小信号模型下的跨导**
2. **计算R9、R11、k**

符合的要求

1. **验证功耗是否满足指标**

表3-1 各个MOS管的宽长比S以及对应的W、L设计

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  | 31.8 | 1.5 |
|  |  | 31.8 | 1.5 |
|  |  | 25.8 | 1.5 |
|  | 3.9 | 5.85 | 1.5 |
|  | 6.6 | 9.9 | 1.5 |
|  | 8.2 | 12.3 | 1.5 |

1. 电流漏的设计与仿真



图4-1 高摆幅共源共栅电流漏结构

要使得

在此电路中，通过将M4的栅宽栅长比变为其他管子的1/4，从而可以达到使减小到最低值的目的。则

所以可以最终推导得出：

根据计算出来的参数绘制出电流漏的电路图：

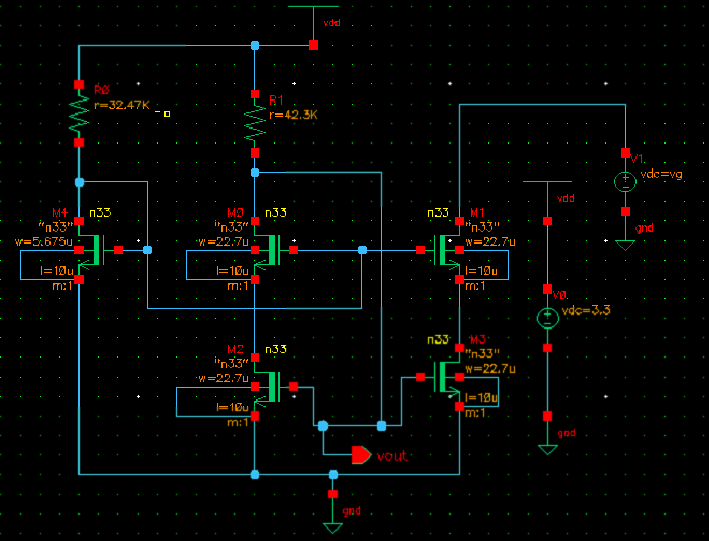


图4-2 电流漏仿真原理图

进行线性DC分析，将范围设置成，进行线性分析，步长选择为。

从途中可以看出提供的电压和计算值相符，约为，电压大于开启电压，可供正常使用，并且MOS管都工作在包河区

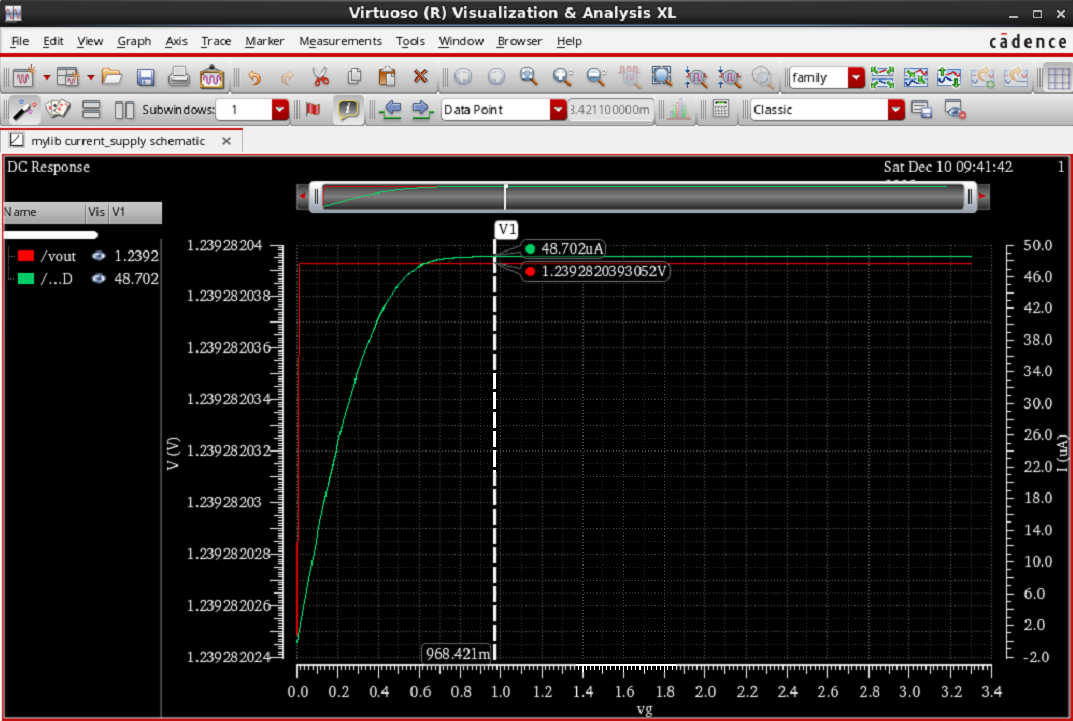


图4-3 对设计的电流漏进行DC分析并画图

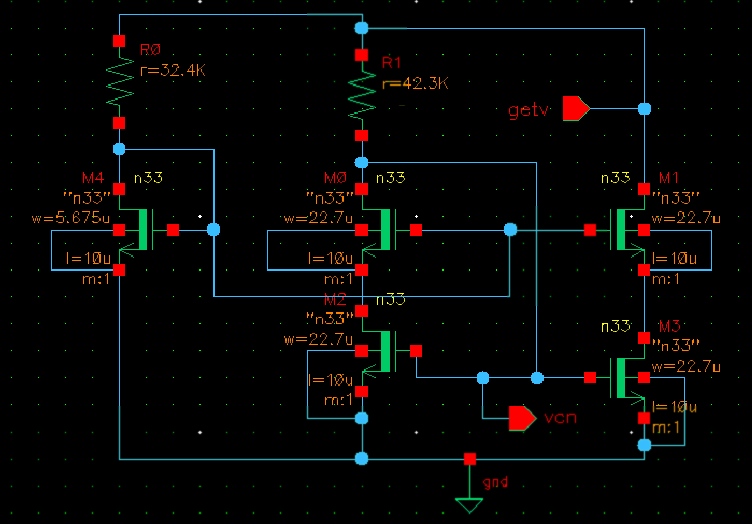


图4-4 最终的单端输入、单端输出的电流漏

如图4-4所示，对比测试电路，封装前的电路原理图去掉了变量以及,变成无源器件后更好地进行封装。



图4-5 电流漏封装symbol

如图所示，对测试好的电流漏电流漏进行封装，得到如图4-5的symbol。

1. 整体电路实际仿真以及性能测试
2. 放大电路的搭建

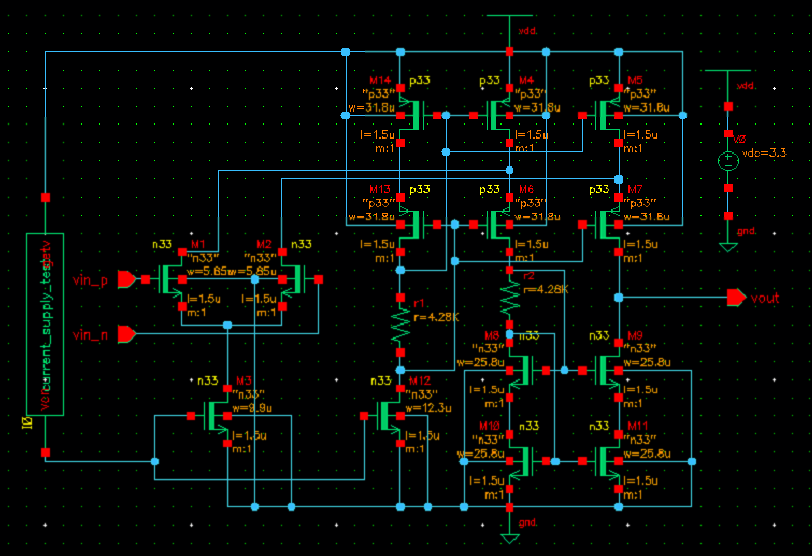


图5-1 放大器的原理图

经过初步的仿真测试，几乎所有MOS管都工作在饱和区，放大电路的搭建基本正确。对放大器进行封装之后，下面对放大器的性能进行测试。

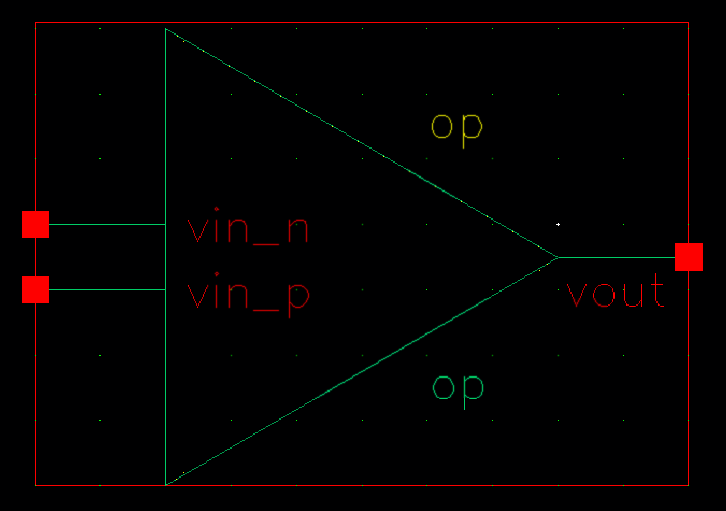


图5-2 放大器的封装symbol

1. 开环增益测试

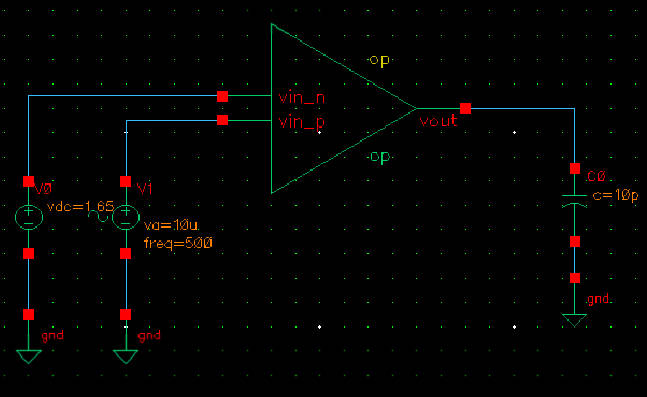


图5-2 开环电压增益测量原理图

放大器的正极接一个差分输入幅值为10uV的500Hz正弦电源，负极接⼀个1.65V的直流电源，分别设置如下：

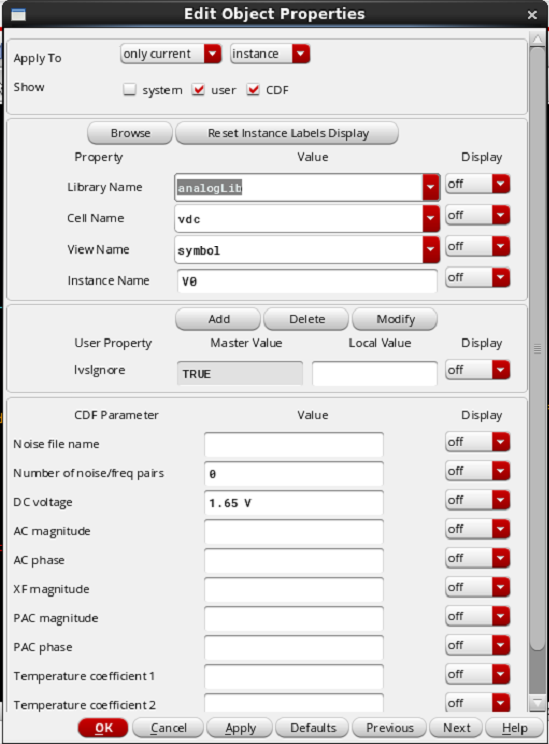
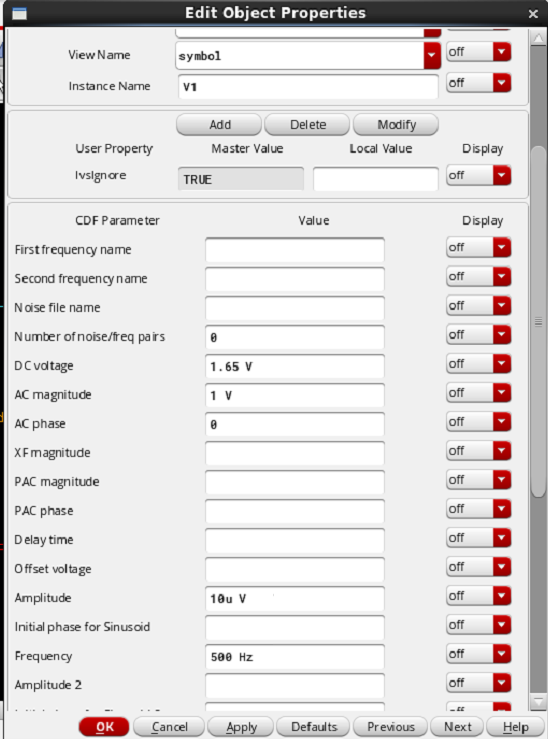
 

图5-3 开环电压增益测量时的vdc与vsin设置

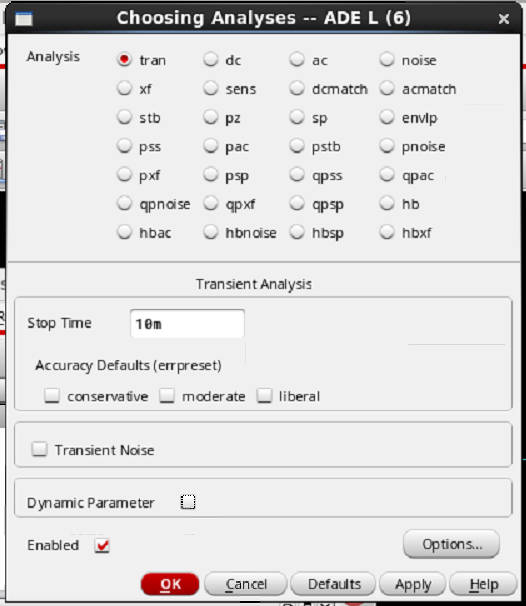
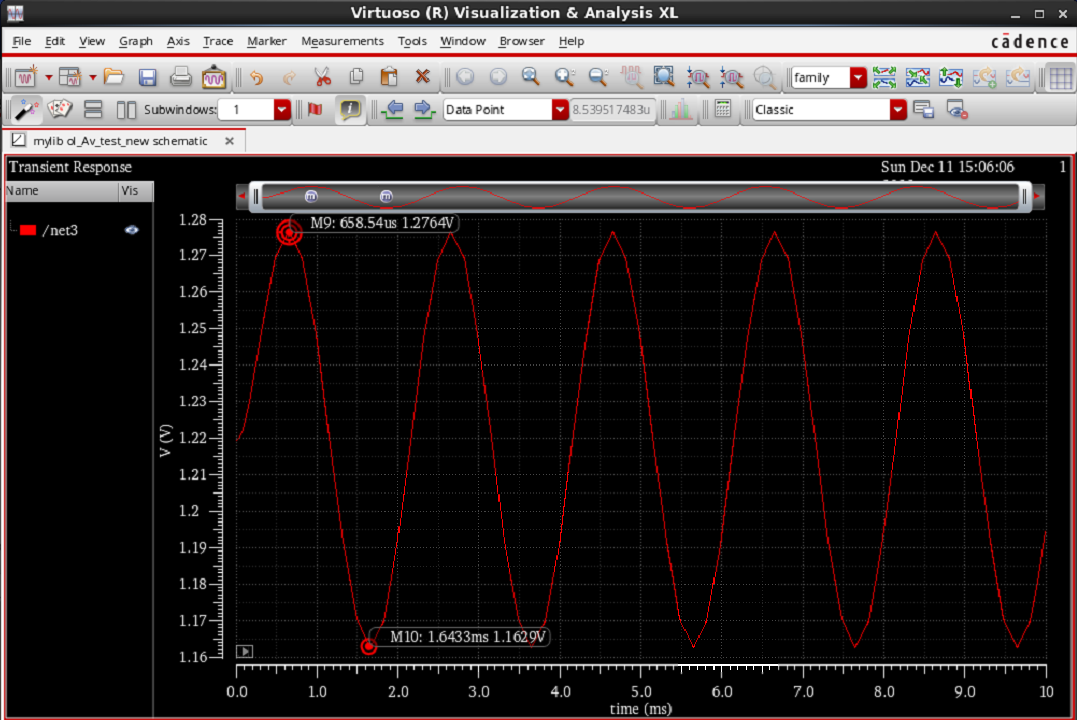
 

图5-4 瞬态响应和ac响应设置

在仿真时，分别选择瞬态仿真与ac仿真，瞬态响应截止时间为10ms，ac响应选择频率Freq，范围从1Hz~1GHz进行扫频。在实际测试之中，将和分别调整为10，略微调大1、2这两个MOS管的宽长比可以具有更好的放大效果。

图5-5 输出的函数曲线图

从图中可以看出输出的峰峰值为

可以计算得到放大倍数

符合设计的要求。

1. 增益带宽积测试（GB）

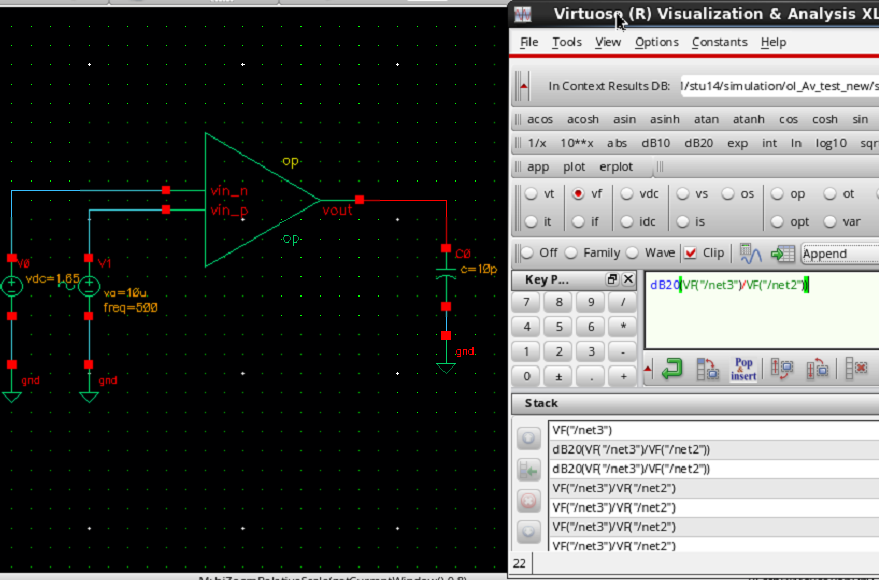


图5-6 使用Tools中的calculator可以画出放大器的增益曲线

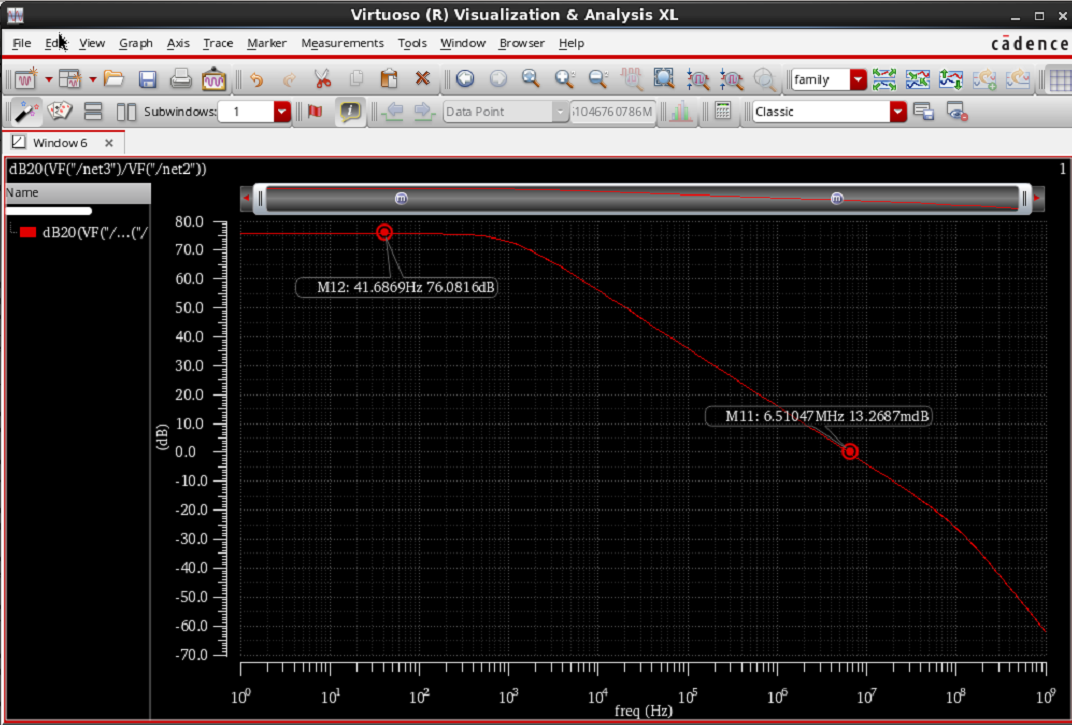


图5-7 放大器增益曲线

从图5-6中看出，放大器的低频增益在76dB左右。

在13.27mdB时（≈0dB）的频率值即为增益带宽积，从图中可以读出增益带宽积为6.51Mhz，满足设计要求的3MHz。

1. 相位裕量测试（Phase margin）

选择Results->Direct Plot->AC Gain&Phase，分别点击输出线和输出线，可以画出AC Gain&Phase曲线，结果如下：

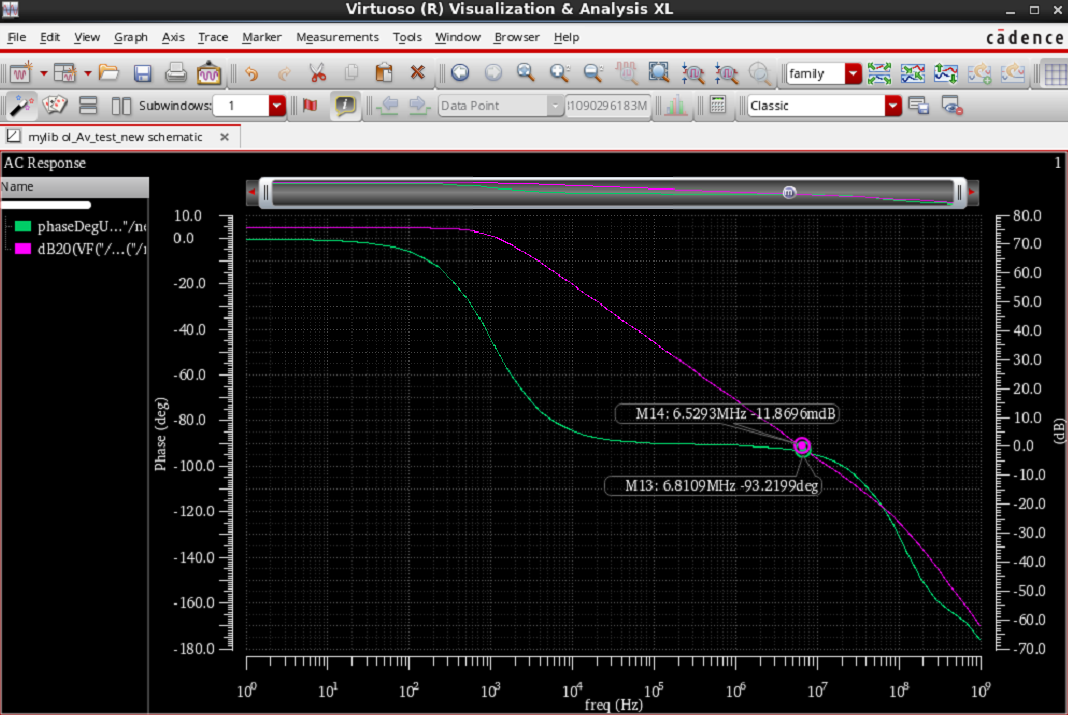


图5-8 AC分析AC Gain&Phase曲线

从图中可以看出，在-11mdB时（≈0dB），此时对应的相位值为-93.22deg，所以相位裕量为180deg-93.22deg=86.78deg>45deg，符合设计要求。

1. 共模抑制比测量（CMRR）

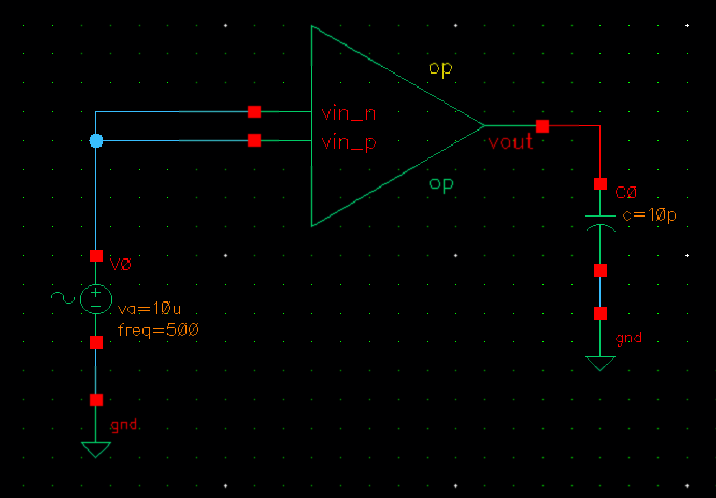


图5-9 放大器共模抑制比CMRR测量

将正弦信号vsin同时接入到正极负极，设置测量增益时的相同，对要输出进行瞬态分析，设置截止时间为30ms，画出如下的瞬态分析图：

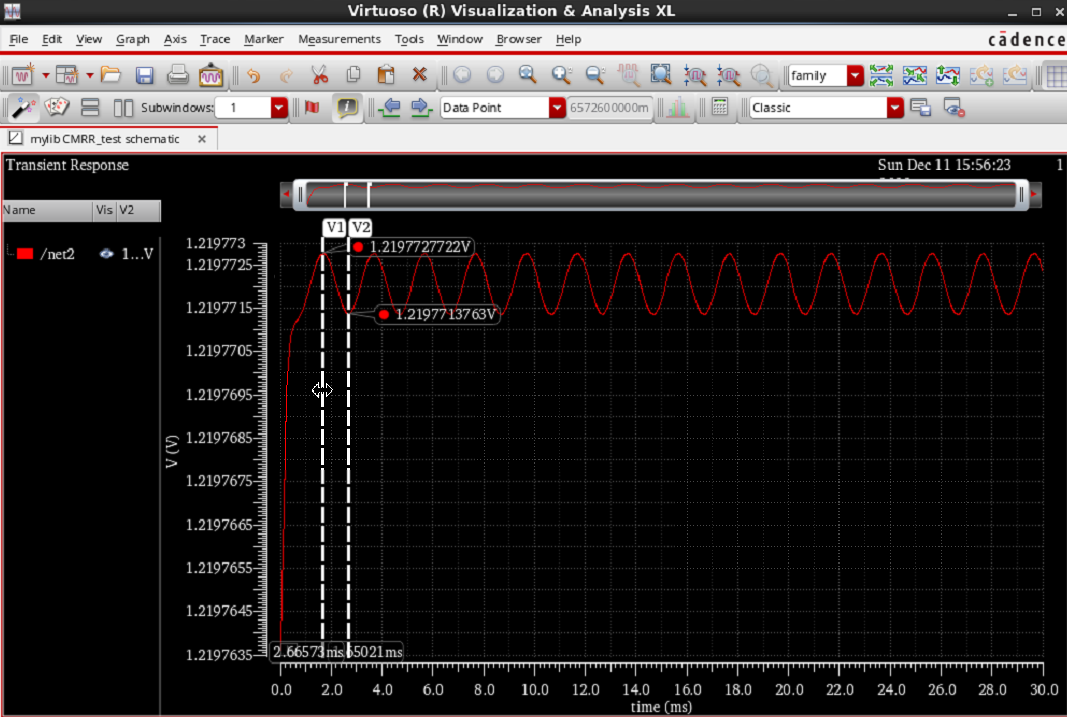


图5-10 放大器共模抑制比CMRR测量

可以看出画出图像的峰峰值为,所以共模增益为=0.07，因此：

可以看出，该放大器的共模抑制比，可以看出该放大电路对于差模信号的增益约为共模信号增益的10000倍左右，有很好的共模抑制效果。

1. 共模输入范围测量（ICMR）

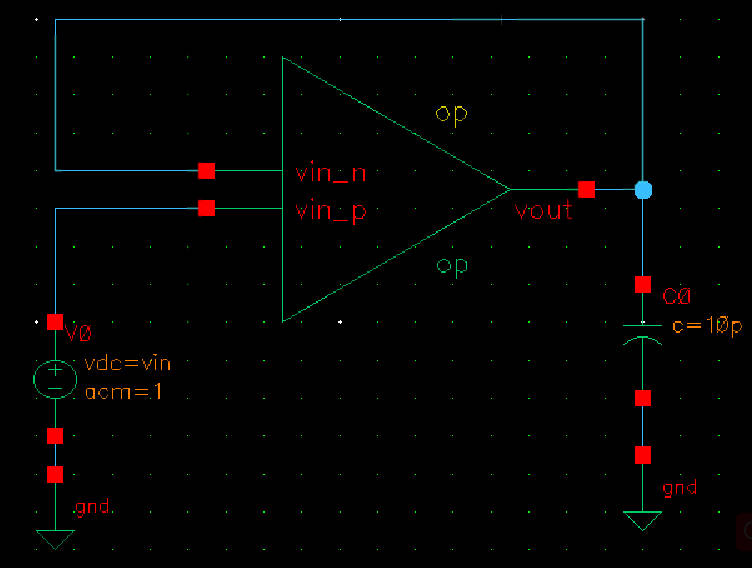


图5-9 放大器共模输入范围ICMR测量

将运算放大器的负极vin\_n接入到输出端，正极输入一个vdc，交流分量acm=1V，选择Results->To be saved->Select on design，分别点击输入输出，将vdc设置成变量vin进行DC分析，扫描范围设置成0-3.3V，步长设置成0.01V，运行仿真可以画出如下曲线：

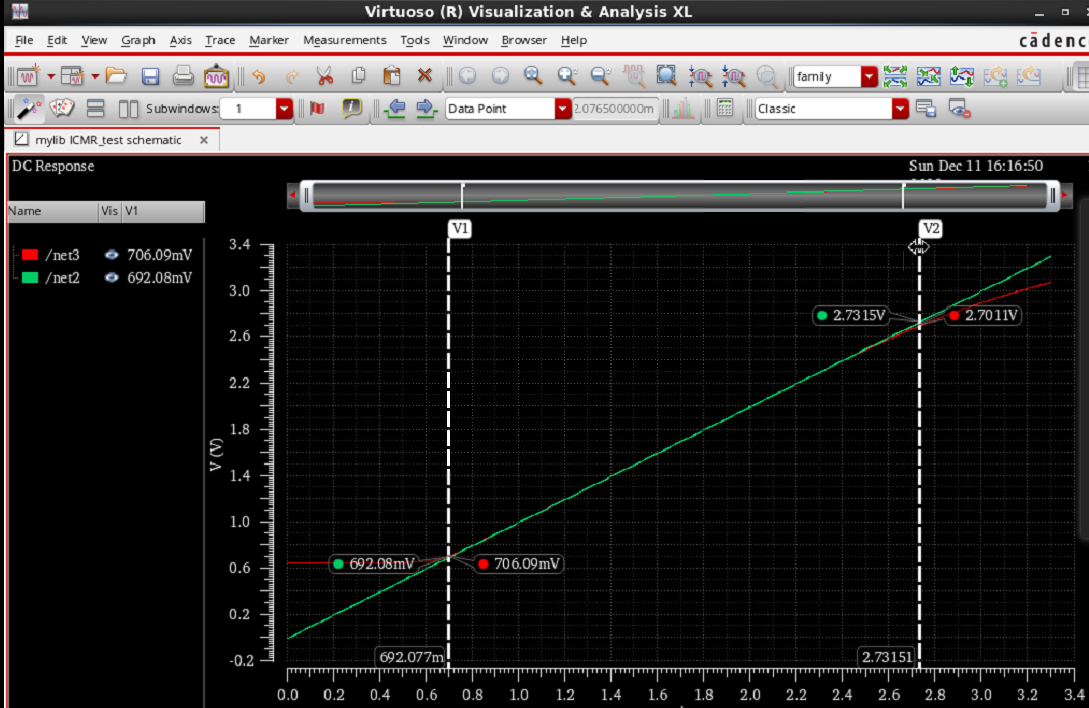


图5-10 ICMR DC分析

在该图中，红色为输出曲线，绿色为输入曲线，可以看到两条曲线在0.692V~2.732V范围内保持重合，由于设计要求共模输入范围为1.25V~2.5V，该范围包含在本放大器电路的共模输入范围内，所以符合设计要求。

1. 测量摆率（SR）

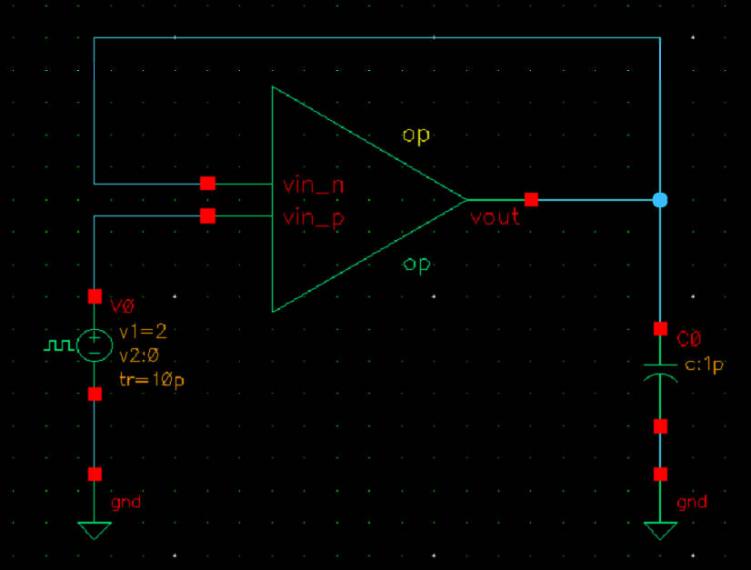


图5-11 摆率（SR）测量电路

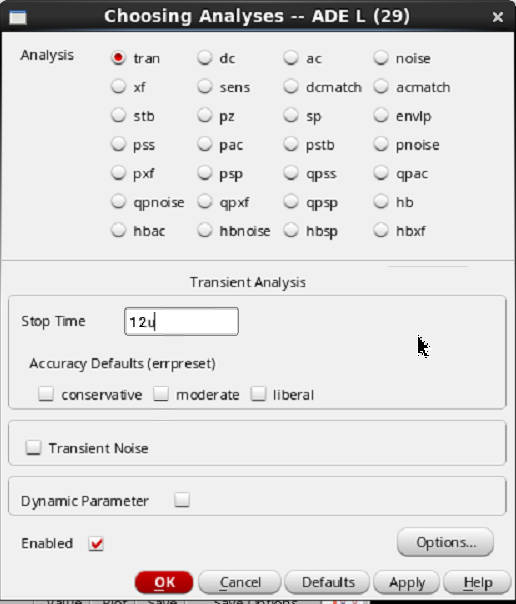
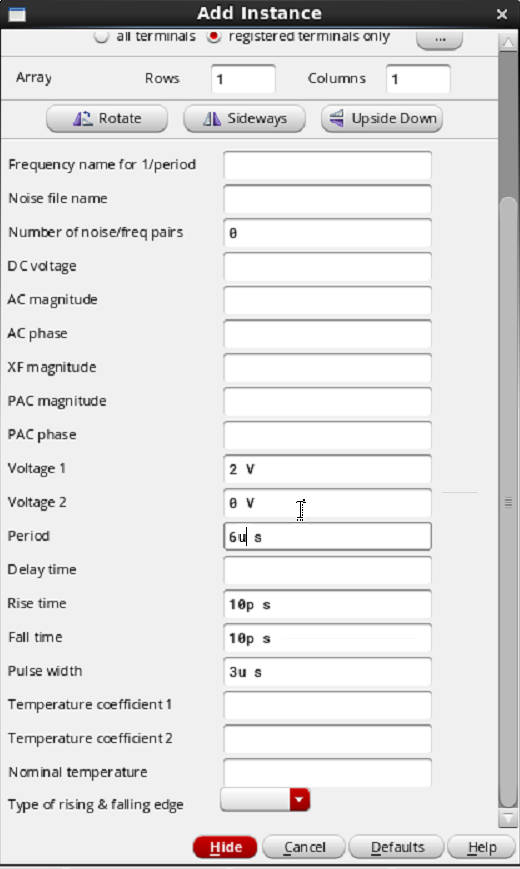


图5-12 方波源设置与瞬态分析

将电路的负极连接到输出，正极接入一个方波源，周期为6us，高低电源分贝设置为2V和0V，设置瞬态分析的截止时间为两个周期，也就是12us。

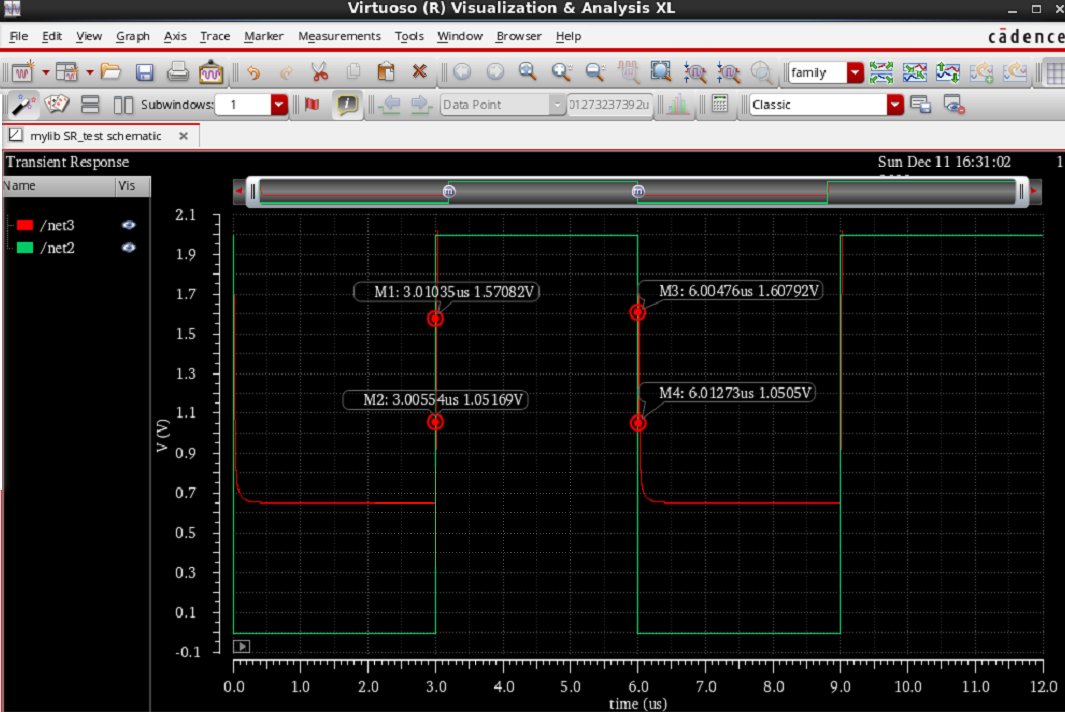


图5-13 SR瞬态分析

其中绿色线为输入端口（标准的方波曲线），红色曲线为输出曲线。

对于上升沿：

对于下降沿：

所以上升沿和下降演的摆率都大于，因此也是满足设计要求。

1. 电源纹波抑制比测量（PSRR）

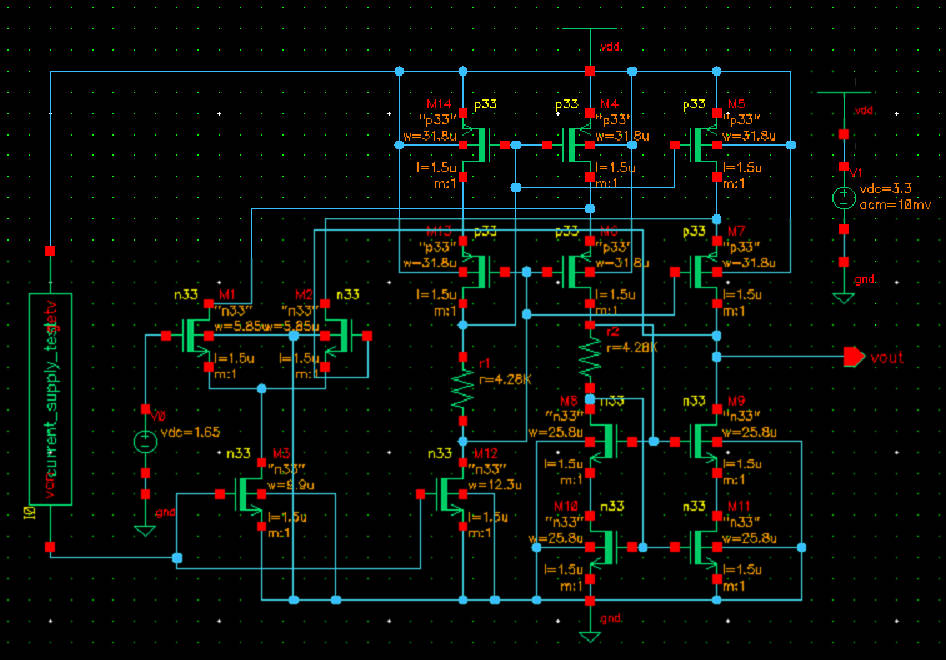


图5-14 纹波抑制比PSRR测量电路

在图5-15的电路中测试PSRR，在PSRR设计测量中不需要加入电容，在vdc（3.3V）中加入acm=10mV的交流分量，并进行AC分析，扫频范围从1Hz~1MHz。电压源设置与AC分析设置如5-15图所示：

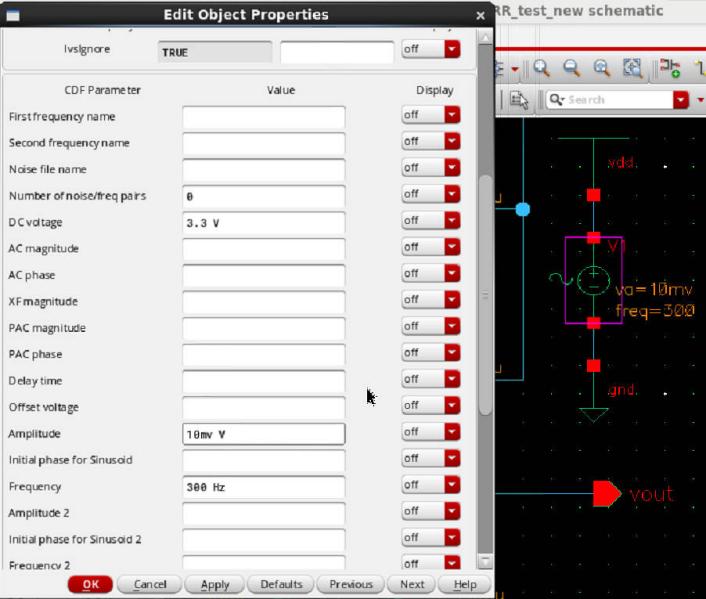
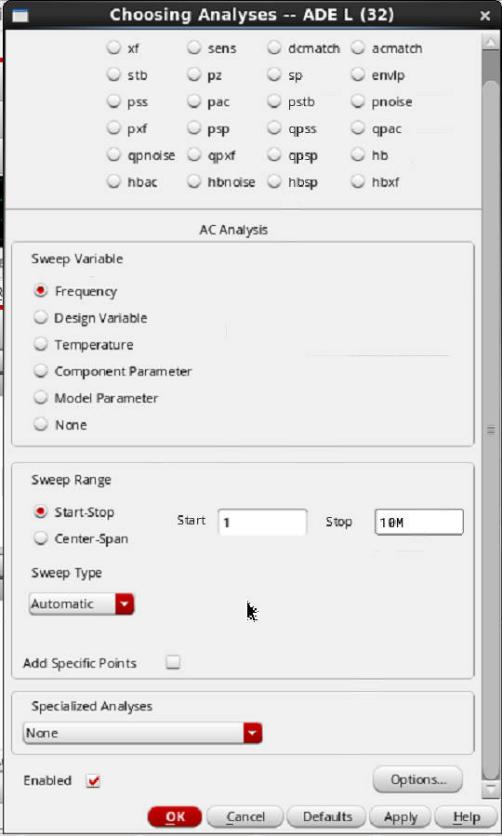


图5-15 PSRR测量ac分析设置与电压源设置

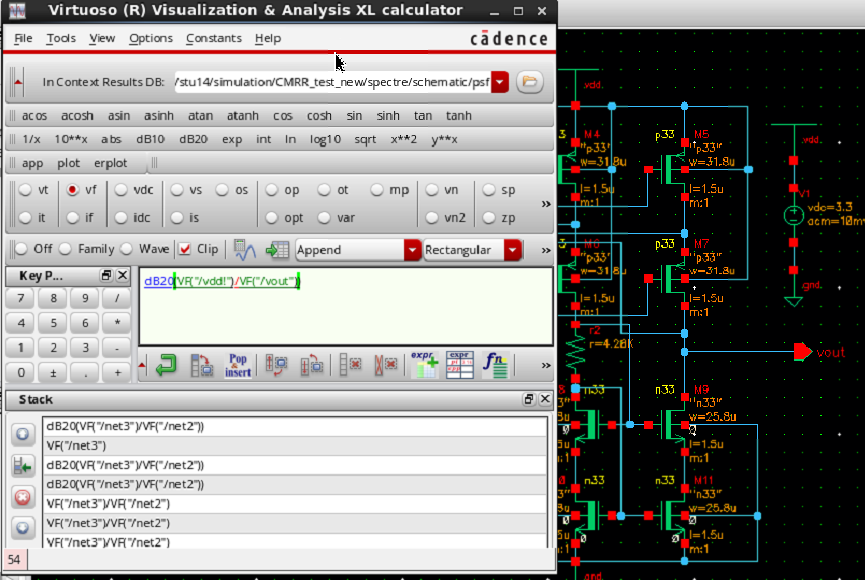


图5-16 按照如图所示方式设置calculator

对于电路进行AC分析，可以画出如下曲线图：

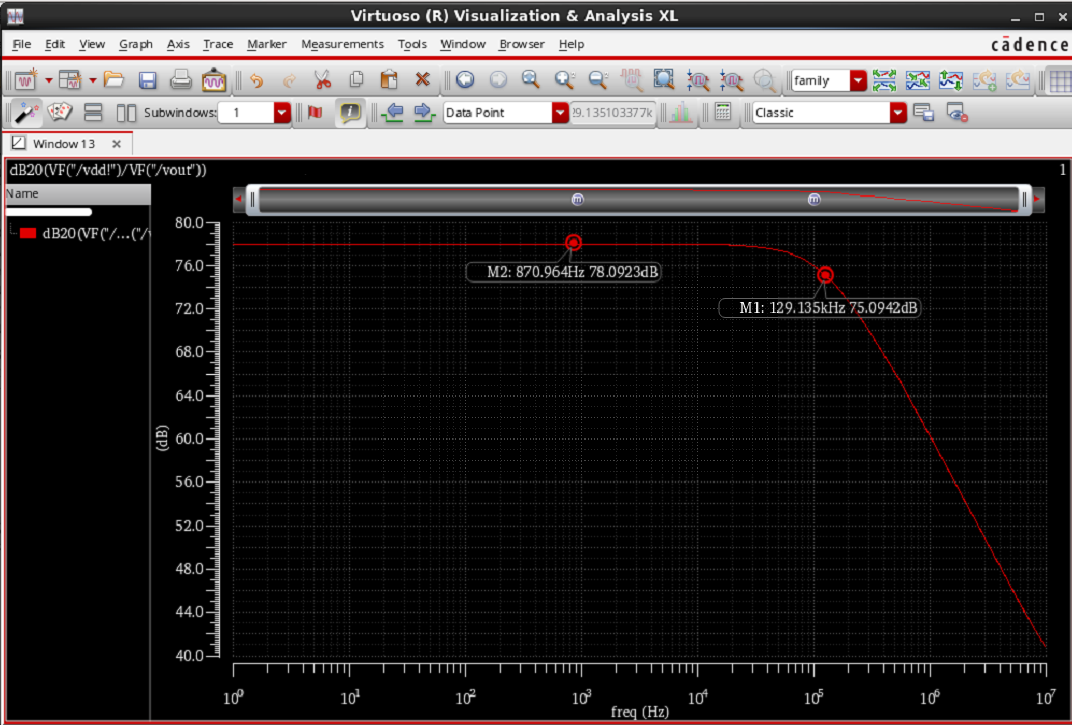


图5-16 PSRR AC分析

从图中可以看出，PSRR在低频时约为78.09dB，可以看出低频时，电源输出纹波是输入的约为1000倍，证明该电路有较好的电源抑制效果。在-3dB处，对应的频率为129.14kHz。

1. 测试功耗

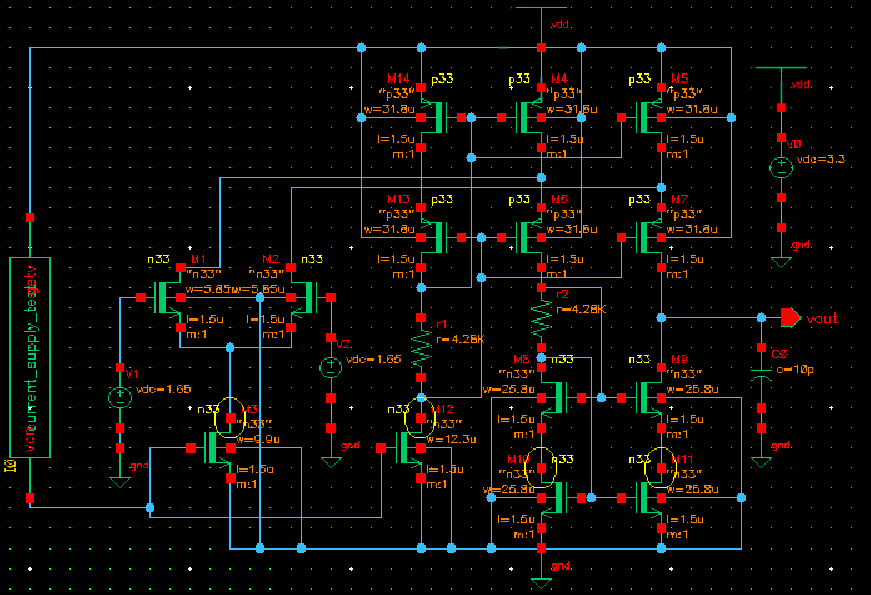


图5-16 瞬态分析电路图

将正极负极都接入1.65V的vdc，使用瞬态分析测试，截止时间设置为10ms。



图5-16 瞬态分析

所以放大器满足设计要求！

1. 实验总结与反思
2. 实验中出现的错误

在实验中，在绘制完电路之后点击保存并检查电路，常常会出现问题，并且会在电路中用方块和打叉的方式标注明。检查电路往往会发现连接线头没有接上。

1. 关于摆率SR

在实验中老师要求的摆率，但是实际测试中由公式可以发现过小的SR容易造成电流镜的MOS管宽长比过小，影响电流镜的效果，所以在本次实验中把摆率调整为，电流镜效果明显得到改善！

附录

Calculator.py源代码

#==========================================================#

#性能指标

VDD = 3.3 #单位V

GB = 3 \* 10\*\*(6)#增益带宽，单位：Hz

SR = 5 \* 10\*\*(6)#摆率，单位：V/s

ICMR\_MIN = 1.25#输入共模范围最小值，单位：V

ICMR\_MAX = 2.5#输入共模范围最大值，单位：V

phase\_margin = 45#相位裕量，单位：°

Vout\_MIN = 0.4#输出电压最小值，单位V

Vout\_MAX = 2.6#输出电压最大值，单位V

Pdiss = 5 \* 10\*\*(-3)#功耗，单位W

#Av = 5000#增益

CL = 10 \* 10\*\*(-12)#负载电容，单位F，暂时定为10pF

lemdaP = 0.05

lemdaN = 0.04

#==========================================================#

s\_si = 3.9 \* 8.85 \* 10\*\*(-12) #硅的介电常数，单位F/m

#==========================================================#

tnox = 6.65 \* 10\*\*(-9) #NMOS栅氧厚度，单位m

uno = 35 \* 10\*\*(-3) #NMOS沟道迁移率，单位m^2/V/s

Vthn = 0.695 #NMOS阈值电压，单位V

tpox = 6.62 \* 10\*\*(-9) #PMOS栅氧厚度，单位m

upo = 9.25 \* 10\*\*(-3) #PMOS沟道迁移率，单位m^2/V/s

Vthp = -0.672 #PMOS阈值电压，单位V

#==========================================================#

Kn = uno \* s\_si / tnox

Kp = upo \* s\_si / tpox

print('Kn:{:.6e}'.format(Kn))

print('Kp:{:.6e}'.format(Kp))

#米勒补偿电容Cc > 0.22CL

Cc = 10 \* 10\*\*(-12) #单位F

S = np.zeros((15,), dtype=float)#S[0]不用

I = np.zeros((15,), dtype=float)#I[0]不用

uo = np.zeros((15,), dtype=float)#uo[0]不用

uo[14] = uo[4] = uo[5] = uo[13] = uo[6] = uo[7] = upo

uo[1] = uo[2] = uo[3] = uo[8] = uo[9] = uo[10] = uo[11] = uo[12] = uno

#确定尾电流I3

I[3] = SR \* Cc

print('I3:{:.6e}'.format(I[3]))

#==========================================================#

#设置一个合适的比例，需在1.2到1.5之间

rate1 = 1.25

#==========================================================#

#输出共源共栅的偏置电流

I[4] = I[5] = rate1 \* I[3]

print('I4、I5:{:.6e}'.format(I[4]))

Vsd5 = Vsd7 = (VDD - Vout\_MAX) / 2

S[4] = S[5] = S[14] = (2 \* I[5]) / (Kp \* Vsd5\*\*(2))

print('S4、S5、S14:{:.6e}'.format(S[4]))

#假设M6、M7电流处于最坏情况

I[6] = I[7] = I[5]

S[6] = S[7] = S[13] = (2 \* I[7]) / (Kp \* Vsd7\*\*(2))

print('S6、S7、S13:{:.6e}'.format(S[6]))

Vsd9 = Vsd11 = (Vout\_MIN - 0) / 2

I[8] = I[9] = I[10] = I[11] = I[6]

S[8] = S[9] = S[10] = S[11] = (2 \* I[9]) / (Kn \* Vsd9\*\*(2))

print('S8、S9、S10、S11:{:.6e}'.format(S[8]))

S[1] = S[2] = ((2 \* np.pi \* GB)\*\*(2) \* CL\*\*(2)) / (Kn \* I[3])

print('S1、S2:{:.6e}'.format(S[1]))

S[3] = (2 \* I[3]) / (Kn \* (ICMR\_MIN - np.sqrt(I[3] / (Kn \* S[1])) - Vthn)\*\*(2))

print('S3:{:.6e}'.format(S[3]))

#验证S4、S5是否足够大以满足最大输入共模电压

S\_temp = (2 \* I[4]) / (Kp \* (VDD - ICMR\_MIN - Vthp)\*\*(2))

print('S\_temp:{:.6e}'.format(S\_temp))

if(S[3] >= S\_temp):

    print("满足条件")

else:

    print("不满足条件，需重新设计S3、S4")

S[12] = (S[3] \* I[4]) / I[3]

print('S12:{:.6e}'.format(S[12]))

#功耗计算

P = VDD \* (I[4] \* 3)

print('P:{:.6e}'.format(P))

if(P < Pdiss):

    print("功耗满足要求")

else:

    print("功耗不满足要求，请重新设计")

#计算小信号模型下的跨导

g = np.zeros((15,), dtype=float)#g[0]不用

print('I4、I5:{:.6e} A'.format(I[4]))

#Kp、Kn的单位是A/V^2 I的单位是A S标量 开根量纲是A/V 也就是S

g[4] = g[5] = g[13] = g[14] = np.sqrt(2 \* I[5] \* Kp \* S[5])

g[6] = g[7] = np.sqrt(2 \* (I[5] - I[3] / 2) \* Kp \* S[7])

g[8] = g[9] = g[10] = g[11] = np.sqrt(2 \* (I[5] - I[3] / 2) \* Kn \* S[11])

g[1] = g[2] = np.sqrt(2 \* (I[3] / 2) \* Kn \* S[1])

print('g[4]、g[5]、g[13]、g[14]：{:.6e} S'.format(g[4]))

print('g[6]、g[7]：{:.6e} S'.format(g[6]))

print('g[8]、g[9]：{:.6e} S'.format(g[8]))

print('g[1]、g[2]：{:.6e} S'.format(g[1]))

gds = np.zeros((15,), dtype=float)#gds[0]不用

gds[4] = gds[5] = gds[13] = gds[14] = I[5] \* lemdaP

gds[6] = gds[7] = (I[5] - I[3] / 2)  \* lemdaP

gds[8] = gds[9] = gds[10] = gds[11] = (I[5] - I[3] / 2) \* lemdaN

gds[1] = gds[2] = (I[3] / 2) \* lemdaN

print('gds[4]、gds[5]、gds[13]、gds[14]：{:.6e} S'.format(gds[4]))

print('gds[6]、gds[7]：{:.6e} S'.format(gds[6]))

print('gds[8]、gds[9]、gds[10]、gds[11]：{:.6e} S'.format(gds[8]))

print('gds[1]、gds[2]：{:.6e} S'.format(gds[1]))

#计算R9、R11、k

rds = np.zeros((15,), dtype=float)#rds[0]不用

R = np.zeros((15,), dtype=float)#R[0]不用

rds[4] = rds[5] = rds[13] = rds[14] = 1/gds[4]

rds[6] = rds[7] = 1/gds[6]

rds[8] = rds[9] = rds[10] = rds[11] = 1/gds[8]

rds[1] = rds[2] = 1/gds[1]

print('rds[4]、rds[5]、rds[13]、rds[14]：{:.6e} Ω'.format(rds[4]))

print('rds[6]、rds[7]：{:.6e} Ω'.format(rds[6]))

print('rds[8]、rds[9]、rds[10]、rds[11]：{:.6e} Ω'.format(rds[8]))

print('rds[1]、rds[2]：{:.6e} Ω'.format(rds[1]))

R[9] = g[9] \* rds[9] \* rds[11]

#r[1]= g[6] \* rds[6] \* (rds[1] \* rds[4]/(rds[1] + rds[4]))

r = g[6] \* rds[6] \* (1/(gds[1] + gds[4]))

R[11] = (R[9] \* r)/(R[9] + r)

k =R[9] \* (gds[2] + gds[4])/(g[7] \* rds[7]) #k

Avd = (2 + k)/(2 + 2 \* k) \* g[1] \* R[11] #Avd

print('r：{:.6e} Ω'.format(r))

print('R[9]：{:.6e} Ω'.format(R[9]))

print('R[11]：{:.6e} Ω'.format(R[11]))

print('k:{:.6e}'.format(k))

print('Avd:{:.6e}'.format(Avd))

#电流漏的设计计算,取最小的作为偏置电压

Vbias = np.zeros((15,), dtype=float)#Vbias[0]不用

#Slou = np.zeros((15,), dtype=float)#Slou[0]不用

#Von = np.zeros((15,), dtype=float)#Slou[0]不用

Rref = np.zeros((15,), dtype=float)#Slou[0]不用

Vbias[3] = np.sqrt(2 \* I[3] / (Kn \* S[3])) + Vthn

Vbias[12] = np.sqrt(2 \* I[4] / (Kn \* S[12])) + Vthn

print('Vbias[3]：{:.6e} V'.format(Vbias[3]))

print('Vbias[12]：{:.6e} V'.format(Vbias[12]))

#去Vbias=1.1V 则Von=0.55V ID=I3

Von = Vbias[3]/2

#Slou[1] = Slou[2] = Slou[3] = Slou[5] = 2 \* I[3] / (Kn \* Von[1] \* Von[1])

Slou = 2 \* I[3] / (Kn \* Von \* Von)

print('Slou[1]、Slou[2]、Slou[3]、Slou[4]：{:.6e} '.format(Slou))

Rref[1] = (3.3 - Vthn - 2\*Von)/I[3]

Rref[2] = (3.3 - Vthn - Von)/I[3]

print('Rref[1]：{:.6e} Ω'.format(Rref[1]))

print('Rref[2]：{:.6e} Ω'.format(Rref[2]))