

Conditional Optimization Using MMICAD

Designers often attempt to optimize circuits with many simultaneous goals. This application note will discuss the pitfalls and problems encountered by the designers when using CAD software to accomplish this task, and how MMICAD may be used to alleviate them.

Problem #1 **"The optimizer is putting too much emphasis on one of my design goals."**

Modern CAD software algorithms often weigh the optimization goals by the order in which they are specified in the OPT block. This leads to the situation where the optimizer will preferentially try to satisfy the first few goals in the list, at the expense of the others. This problem can be addressed through application of individual weights for each of the goals, as well as an appropriate choice of an optimization algorithm. Unfortunately, this solution is quite dependent upon the nature of the circuit being analyzed, rendering the problem very much "an exercise best left to the reader"!

Problem #2 **"The optimizer has met most of the goals, but is taking forever to converge on the last few."**

This happens when the optimizer falls into a hole and oscillates the values of some circuit variable over a narrow range, leading to no net improvement in the error function. To correct this, the designer is forced to pause the optimization, change the weighting on some of the goals and restart the optimization.

Problem #3 **"I know that some of my design goals are independent of the others - how do I optimize these first?"**

The answer to this question is to optimize the independent goals first, followed by optimization of the rest of the goals.

The MMICAD Solution

All of the above problems and their respective solutions require a high degree of manual intervention on the part of the designer during the CAD optimization cycle. This is a costly, time-consuming exercise.

All of these shortcomings are due to a lack of "intelligence" in the software optimization algorithm. Fortunately, MMICAD offers an elegant solution to this problem. Using conditional circuit variables, designers can now specify "intelligent" optimization goals. For example, MMICAD can be told not to start the optimization of one goal until some other condition is met, or to dynamically change the weighting on individual goals during optimization.

Design Example

As an example of the possibilities, consider the design of a MMIC amplifier with the following specifications:

FREQ: 2-4 GHz
Gain: 15 ± 0.7 dB
NF: < 3 dB
S11: < -11 dB
S22: < -15 dB

Since noise figure is primarily affected by the input match, while gain is affected by the output conditions of the amplifier, it makes sense to optimize the input match for noise figure first, followed by gain optimization.

The MMICAD circuit file is shown in Figure 1 overleaf. Please note that the weighting on the gain of the LNA is assigned to be equal to variable GW, which is in turn dependent upon the PROC block variable NT, assigned the difference of the measured noise figure and 3.05 dB. During optimization, if the noise figure rises above 3.05 dB, this gain weighting variable GW is set to 1, and the optimizer emphasizes improvement in input match and noise figure. When the noise figure conditions are met, the noise weighting variable NW is assigned to be 1, the gain weighting variable is assigned to be 10 and emphasis is then given to gain during optimization. The optimizer will continue in this manner until optimization is complete.

```

! NOTES: A low-noise 2-4 GHz amplifier based on MMICs      ! OPT
GOALS: GAIN: 15dB+-0.7 NF:<3dB S22:<-15 S11:<-11 dB
! Conditional Noise Optimization for low noise amplifiers  ! During
Optimization, the NOISE goals will be met before the
! GAIN goals.

```

```
MODE FREQ NOISE
```

```

GLOBAL
DIM FREQ=1e+009 RES=1 COND=0.001 CAP=1e-012 IND=1e-009
LNG=1e-006 TIME=1e-012
MSUB ER=12.8 H=200 T=3 RHO=1.3 TAND=0 @SUB0

```

```

FILES
C:\MMICAD\EXAMPLES\F3002.S2P F3002 101 2P FREQ

```

```

VAR
LRFB=? 50 100 400 ?
NIN1=? 1 4 8 ?
LOUT1=? 50 70 400 ?
NIN2=? 1 1.5 8 ?
NOUT2=? 1 4 8 ?

```

```

CKT
!Square Spiral Inductor Model with N=Number of Turns as Parameter:
MODVAR N=3
SRL 1 2 R={ .233*N + .0927*N*N } L={ -.164*N + .286*N*N }
CAP 1 2 C={ .002*N - 6E-5*N*N }
CAP 1 0 C={ .0175*N+ 1.15E-4*N*N }
CAP 2 0 C={ .0121*N+ 1.86E-4*N*N }
DEF2P 1 2 SPIR ( N )

```

```

!Distributed Thin Film Resistor Model with W, L, and RSQ as Parameters:
MODVAR W=20 L=100.413 RSQ=100
RES 1 2 R={ RSQ*L/W }
MTRLND 2 3 W={ W } L={ L } @SUB0
DEF2P 1 3 TFRES ( W L RSQ )

```

```

! Drain Bias Network for Stage#1 and Stage#2:
TFRES 1 2 0 W=20 L=40 RSQ=50
SPIR 2 3 0 N=3
SLC 3 0 L=0.1 C=1000
DEF1P 1 DBIAS

```

```

!Stage#1 of the Amplifier:
SRC 1 0 R=5000 C=10
SPIR 1 2 0 N=NIN1
F3002 2 3 10 M=4
MTRL 10 11 W=20 L=400 @SUB0
RES 11 0 R=25
CAP 11 0 C=20
MTRL 3 4 W=12 L=LOUT1 @SUB0
TFRES 2 20 0 W=20 L=LRFB RSQ=100
TFC 4 20 W=225 L=200 CS=310 @SUB0
DBIAS 4 0 M=1
DEF2P 1 4 STAGE1

```

```

!Stage#2 of the Amplifier:
MTRL 1 2 W=12 L=250 @SUB0
SRC 2 0 R=5000 C=4
SPIR 2 3 0 N=NIN2
F3002 3 4 10 M=2
MTRL 10 11 W=100 L=50 @SUB0
RES 11 0 R=50
CAP 11 0 C=20
SPIR 4 5 0 N=NOUT2
DBIAS 5 0 M=1
DEF2P 1 5 STAGE2

```

```

! Final Amplifier Construction:
STAGE1 1 2 0
TFC 2 3 W=200 L=135 CS=310 @SUB0
STAGE2 3 4 0 M=1
DEF2P 1 4 LNA

```

```

FREQ
STEP 1 1.5
SWEEP 2 4 0.1
STEP 4.5 5

```

```

MARKER
STEP 2 2.5 3 3.5 4

```

```

PROC
NT = LNA DB[NF] - 3.05      !Set up the test variable.

```

```

VAR
!IF LNA NOISE < 3.05 DB THEN
! GAIN OPTIMIZATION WEIGHT = 10
! ELSE
! GAIN OPTIMIZATION WEIGHT = 1
GW/NT = 10 : 1 : 1
NW/NT = 1 : 10 : 10

```

```

OPT
RANGE 2.0 4.0
LNA DB[S21] IN 15.5 0.7 W=GW
LNA DB[S22] LT -15 W=1
LNA DB[S11] LT -11 W=1
LNA DB[NF] LT 3.0 W=NW

```

```

OUT
LNA DB[S21] AMP
LNA DB[NF] AMP
LNA DB[S11] AMP R
LNA DB[S22] AMP R
LNA SM[S11] AMP2
LNA VSWR[S11] VSWR
LNA VSWR[S22] VSWR
LNA RE[K] k
LNA DB[NF] NFIG
LNA AMP STAB
LNA SPAR LNA_SPAR
F3002 SPAR FET_SPAR

```

```

GRID
AMP 1 5 0 20 R 0 -40
VSWR 1 5 0 20
NFIG 1 5 0 8
k 1 5 0 40

```

Figure 1