Examples of Circuit Simulation

Editor's note: Section and figure references in this article are from the 2010-2013 editions of the ARRL Handbook. *This material was originally contributed to the* Handbook *by David Newkirk, W9VES.*

6.2.1 Example 1: A Two-JFET VFO and Buffer

In introducing this chapter, we "just built" — in our mind's eye — a receiver LO that "just worked," as real-world projects are intended and expected to do. Building and simulating the same two-JFET design in the circuit simulator called *SPICE* transports us right to the heart of fundamental circuit-modeling problems and their solutions. If we build through graphical schematic entry — through schematic capture — its oscillator-buffer circuit exactly according to its electrical schematic, *it cannot possibly work!*

Fig 6.1 shows the circuit for real-world duplication. In it, a J310 JFET Hartley oscillator drives a J310 buffer amplifier, which drives a 50-Ω load at +10 dBm via a trifilar broadband transformer that provides an impedance step-down ratio of 9:1. Any experimenter accustomed to building and using such circuits needs no more information than that provided in the Fig 6.1 schematic and its caption to make the circuit work as expected.

Fig 6.2 shows the same circuit successfully modeled in *OrCAD 16.0*, *SPICE* based simulator software from Cadence Design Systems. To understand the differences between Fig 6.1 and Fig 6.2, we'll examine the simulation's components type by type.

RESISTORS

The designer's output power specification (+10 dBm) assumes that the VFO is connected to the 50-Ω load afforded by the mixer system in the receiver it was designed to drive. To simulate this mixer load, we have added R5.

The value of R1, specified as 1 MΩ in the original circuit, is now specified as 1E6 — scientific/engineering notation for one million. We have done this to remind ourselves that *SPICE*'s use of unit suffixes — *scale factors* in *SPICE*-speak — differs from what we are generally accustomed to seeing in electrical schematics, and that we have multiple options for specifying values numerically using integer and decimal floating-point numbers. The scale factors available in *SPICE* include:

- F — 1E–15
- G — 1E9
- K — 1E3
- M — 1E–3
- MEG — 1E6
- MIL (0.001 inch) — 25.4E–6
- N — 1E–9
- P — 1E–12
- T — 1E12
- U — 1E–6

Specifying the value of R1 as 1M would declare its value as 1 milliohm (0.001 Ω), short-circuiting the JFET's gate to common

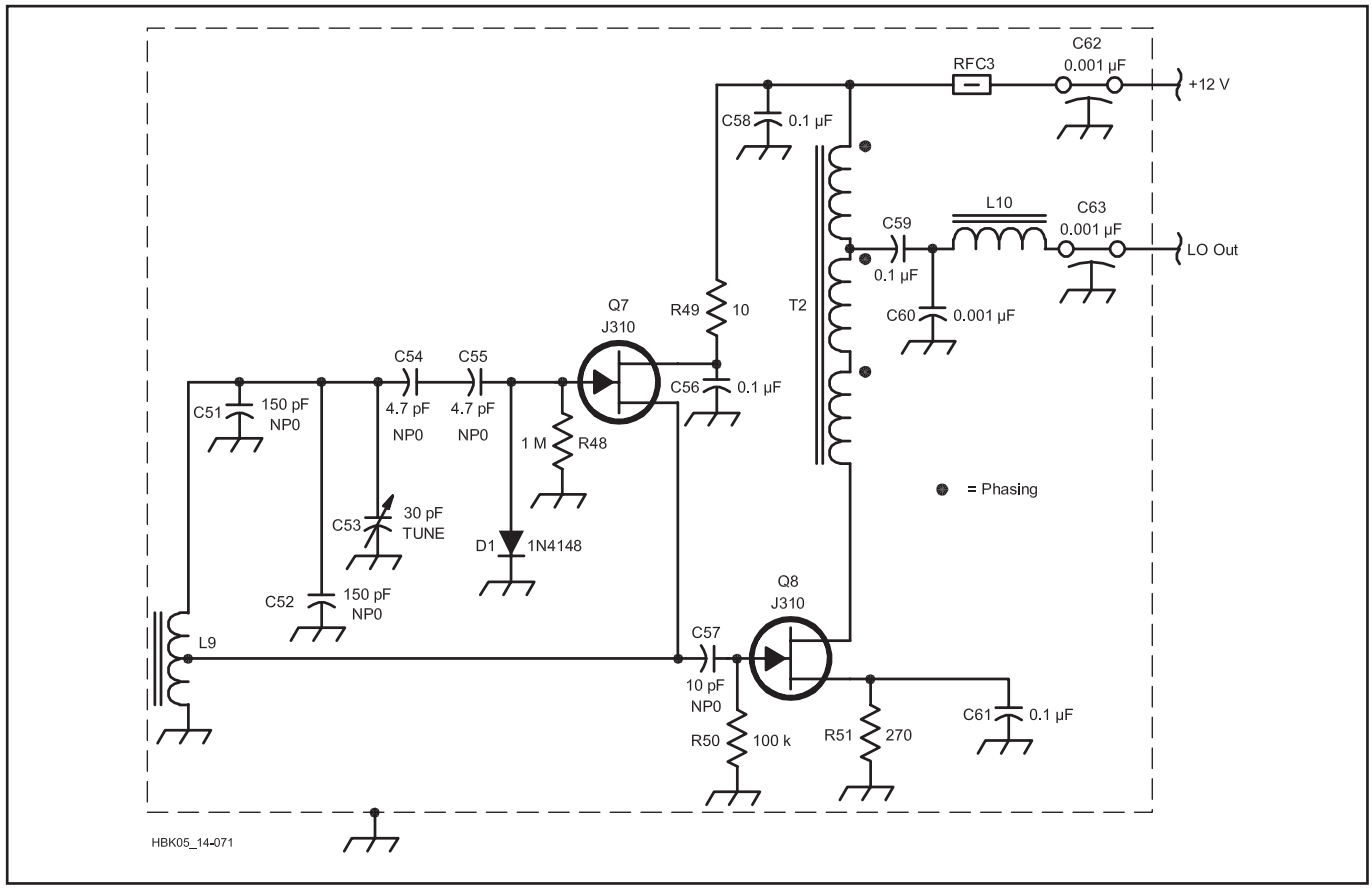


Fig 6.1 — Standard electrical representation of a 7-MHz VFO with buffer amplifier. JFET Q7 operates as a Hartley oscillator; D1, as a limiter that improves frequency stability by keeping Q7's gate voltage from going more positive than about 0.6 V; and Q8, as an amplifier that increases the oscillator output — obtained from the feedback tap on the oscillator inductor by capacitive coupling (C57) — to +10 dBm. The tapped inductor (L9), is 1.2 µH (22 turns of #28 wire on a T-30-6 toroidal core); the trifilar output transformer (T2), 10 trifilar turns of #28 wire on an Amidon FB-43-2401 ferrite bead).

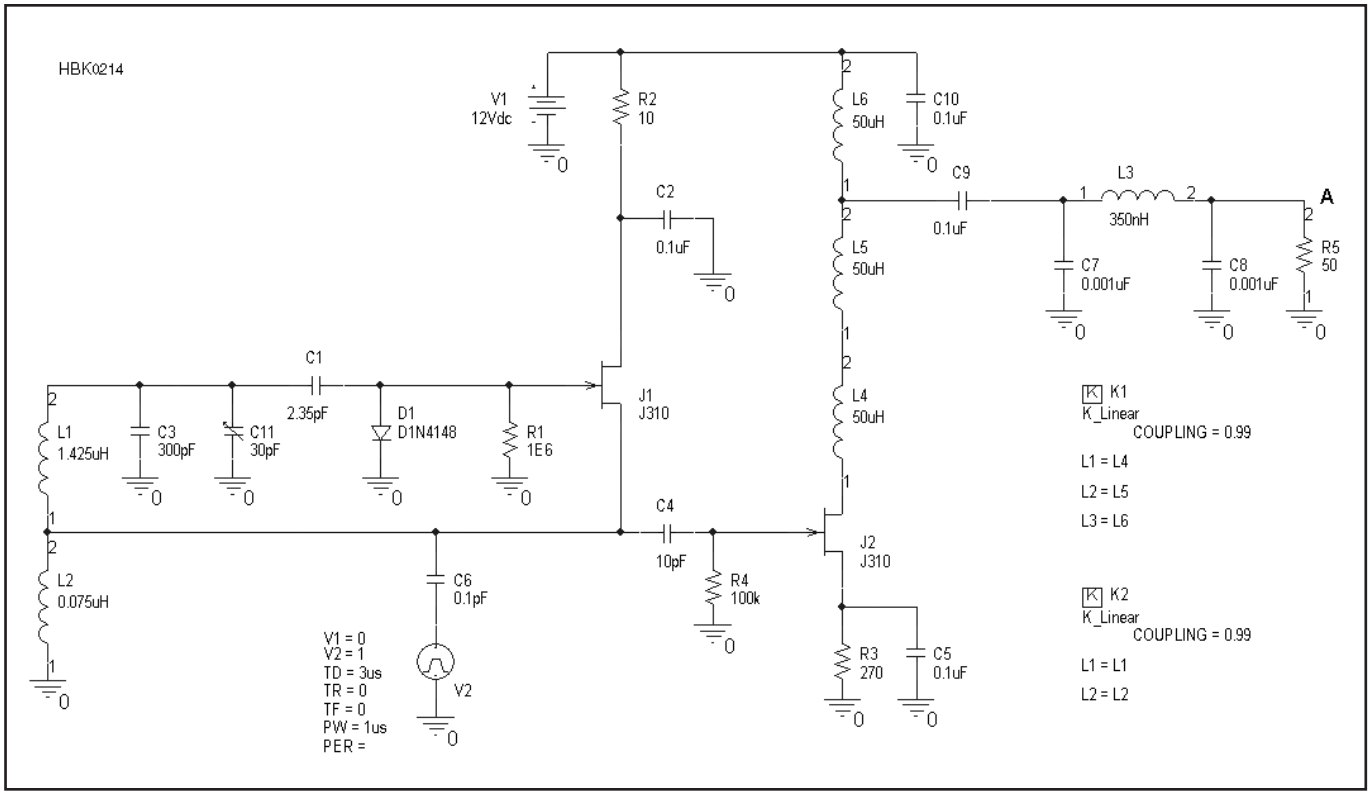


Fig 6.2 — The VFO-and-buffer circuit configured for successful *SPICE* simulation under the *OrCAD 16* CAD suite as drawn — "captured" — in the *OrCAD Capture CIS* schematic editor. To make the circuit work like the real thing, we added several components only implied in Fig 6.1, including a 12-V dc source (V1) and a 50-Ω load resistor (R5). Also new to the circuit, ac-coupled (via C6) to the oscillator JFET (J1) source, is a mysterious second voltage source (V2) — an addition without which the oscillator cannot oscillate. The numbers identifying the leads of each inductor are displayed by default in the *OrCAD* schematic editor to indicate phasing, knowledge of which is essential for properly modeling the behavior of the trifilar transformer formed by L4 through L6. We have also enabled pin-number display for R5 to help us determine the name of the circuit node — labeled A — we must probe to graph the circuit's output waveform.

Using Your Computer to Draw Schematics

The art of drawing circuit schematics predates electronic computers; the art of drawing schematics and PC-board layouts with computers predates the art and science of *simulating* circuits with computers. What if you only want to draw a circuit's schematic and design a PC board without simulating it? What computerized tools are available to *you*?

Almost any circuit-simulation program or electronic design automation (EDA) suite that uses schematic circuit capture can, within functionality limits imposed on the demoware version, serve as a first-rate schematic editor. Although demoware component library limitations usually restrict the *types* of components you can use — in CAD-speak, *place* — in a design, part-*count* limitations usually operate only at simulation time. Restrictions in physical size and layer count, and in the materials and metallizations available for substrate specification, will likely apply to whatever layout-design facilities may be available. After all, the main purpose of demoware is to let students and potential buyers taste the candy without giving away the store.

Excellent simulation-free schematic-capture and layout-design products exist, of course. The schematic style long standard in ARRL publications comes from the use of Autodesk *AutoCAD*, a fully professional product with a fully professional price. Long popular with radio amateurs and professionals alike is CadSoft *EAGLE*, a schematic-capture and layout-design product available in freeware and affordable full-version forms. You can even export *EAGLE* schematics to a *SPICE* simulator and back with Beige Bag Software's *B2 Spice*. The full-function freeware schematic and PCB-layout application *Kicad* and the EDA suite *gEDA* come to us from the open-source community.

Do-it-yourself schematic CAD can be as close as the basic drawing utility included with your computer's operating system. Some hobbyists find that cutting, copying, moving and pasting components snipped from favorite graphical schematic files into new configurations and adding new wires as graphical lines is enough for what they want to do. The connection between the world of modern EDA tools and this seemingly primitive approach to schematic creation can be as close as the operating system's clipboard: Some CAD schematic-capture programs represent circuit elements as metafile data during copying, cutting and pasting operations. Copying a schematic to the clipboard with such a program and then pasting the clipboard contents to a suitable drawing program creates a *picture* of the copied schematic — give it a try with *Windows Paint* and the schematic editor in the free demoware version of *OrCAD 16*.

and breaking our simulation. Specifying the value of R1 as 1MEG or 1000K would be correct alternatives. *SPICE* scale factors are case-insensitive.

Notice that *SPICE* assumes unit *dimensions* — ohms, farads, henrys and so on — from component-name context; in specifying resistance, we need not specify ohms. In parsing numbers for scale factors, *SPICE* detects only scale factors it knows and, having found one, ignores any additional letters that follow. This lets us make our schematics more human-readable by appending additional characters to values — as long as we don't confuse *SPICE* by running afoul of existing scale factors. We may therefore specify "100pF" or "2.2uF" for a capacitance rather than just "100p" or "2.2u" — a plus for schematic readability. (On the reduced readability side, however, *SPICE* requires that there be *no space* between a value and its scale factor — a limitation that stems from programming expediency and is present in many circuit-simulation programs.)

CAPACITORS

To develop the habit of keeping our simulated circuits' part counts down so we don't run up against the circuit-complexity restrictions of the *OrCAD 16.0* demoware (*all* demoware and student-version CAD packages have such limits) we have combined the seriesed and paralleled capacitor pairs in our real-world tuned circuit into single capacitances. Original C54 and C55 become C1; original C51 and C52, C3. That said, there are two reasons why we may not want to do so. If one of the objects of our simulation is to examine the voltage, current or power at the junction of C54 and C55 in the original circuit, combining them has disallowed achieving that aim. More generally, if we also intend to base a PC-board design on our captured circuit through *OrCAD 16.0 PCB Design Demo* — one of *OrCAD Capture CIS*'s sibling applications in the *OrCAD 16.0* demoware suite — we must engineer our simulation with that practical outcome in mind throughout.

INDUCTORS

Simulating the tapped coil of a Hartley oscillator immediately challenges us to learn more about our real-world circuit than we need to know to successfully build it. The original coil, 1.5 μH, consists of 22 turns of wire tapped at 5 turns, yet a tapped inductor is not available in the *SPICE* model library. Multiple approaches to simulating a tapped inductor can be used, including connecting two inductors in series (as we have done here) and proportioning their values intelligently, or basing the tuned circuit on one winding of an ideal transformer and proportioning the inductance of the secondary to simulate the tap. Either way, we must make the best educated guess we can about the inductance between the tap and ground, as its value directly affects the oscillator feedback, and hence its output.

Especially if your approach to building is more practical than theoretical, proportioning the values of the two coils in the 22:5 ratio reflected in the original's winding information might seem like a fair approximation — until we recall that a coil's inductance-versus-turns ratio is not linear. Taking that approach would give us a larger-than-life inductance value for the lower portion of the coil, resulting in more-than-realistic feedback and higher-than-realistic output. So what we have done for this simulation is calculate the inductance of 5 turns of wire on a T-30-6 core, taking the answer (0.075 μH) as the value of Fig 6.2's L2, and 1.5 μH – 0.075 μH (1.425 μH) as the value of L1, the upper portion of the coil.

To illustrate another feature of *SPICE* — and to provide one avenue for later experimentation with this simulation — we have also specified near-ideal ($K = 0.99$) coupling between L1 and L2 by means of K_Linear element K1 (in the lower right corner of Fig 6.2). *SPICE* allows us to specify coupling between any subset of inductors in a simulation, including *all* inductors in a simulation. The value of this feature in enabling greater realism in simulations of complex, cross-coupled connector, circuit-board and IC structures is profound — at the expense of requiring the realistic specification of coupling values if the power of this feature is to be realized.

Here we have coupled the two sections of our oscillator tank inductor because (1) we know that they actually *are* coupled in the real thing; (2) we want to experience specifying inductor coupling in *SPICE*; and (3) practical experiments with Hartley inductors consisting of separate toroidal cores nonetheless shows that such coupling is *not* necessary to make real Hartley oscillators work! Assuming we can get the circuit to oscillate as is, reducing the coupling between L1 and L2 would let us simulate the use of separate coils in a real-world oscillator.

Having specified coupling between the sections of the oscillator tank inductor, we are ready to welcome the similar challenge of simulating the trifilar broadband output transformer (T2) in Fig 6.1. Here, as with the tapped oscillator tank coil, no direct equivalent to this transformer topology is available in *SPICE*, but multiple alternatives can get us close enough. One option would be to use a conventional two-coil transformer, such as that available in the *OrCAD Capture CIS* component library. As with the tapped oscillator coil, however, we have decided to use three separate coupled inductors, specifying their coefficients of coupling with another K_Linear element, K2. For the inductance of each, we have drawn on our experience with the "10 multifilar turns on an FT-37-43 core"-class broadband transformers commonly used for just such applications in many ARRL RF projects, specifying an inductance (50 μH) close in value to that of a single such winding for each coil of our simulated transformer. (As a check on the intelligence of using this value, we recall that the rule of thumb for the inductance of a conventional broadband transformer winding calls for a winding reactance of at least 5 to 10 times the impedance at which the winding operates — and the reactance of 50 μH at 7 MHz equates to 2.2 kΩ, over 40 × 50 Ω.) In wiring the three inductors (L4, L5 and L6), we have also taken care to phase them properly, their **1** and **2** labels conveying the winding-sense information communicated by the phasing dots that accompany T2's windings in Fig 6.1.

Specifying for simulation the remaining inductor in Fig 6.1 — in our Fig 6.2 schematic, L3, 350 nH, one of three components in a π low-pass filter network — is straightforward enough to warrant no comment. But comment we shall, for in specifying the electrical performance of an inductor merely by setting a value for its inductance — as we have so far done for all of the inductors in our circuit — we have in no way specified its quality factor (Q). In simulation it will there act as an absolutely pure inductance — a component that cannot be built or bought. This will make a difference in how closely our simulation may approach the real thing — but how much of a difference?

All *real* inductors, capacitors, and resistors — all real components of any type — are non-ideal in many ways. For starters, as **Fig 6.3** models for a capacitor, every real L also exhibits some C and some R; every real C, some L and R; every real R, some L and C. These unwanted qualities may be termed *parasitic*, like the parasitic oscillations that sometimes occur in circuits that we want to act only as amplifiers, and in oscillators (which may simultaneously oscillate at multiples frequencies, including the frequency we intend). For experimental and proof-of-concept purposes at audio and HF radio frequencies, parasitic L, C and R can often be ignored. In oscillator and filter circuits and modeled active devices, however, and as a circuit's frequency of operation generally increases, neglecting to account for parasitic L, C and R can result in surprising performance shortfalls in real-world *and* simulated performance. In active-device modeling realistic enough to accurately simulate oscillator phase noise and amplifier phase shift and their effects on modern, phase-error-sensitive data-communication modes, device-equivalent models must even include *nonlinear* parasitic inductances and capacitances — Ls and Cs that vary as their associated voltages and currents change.

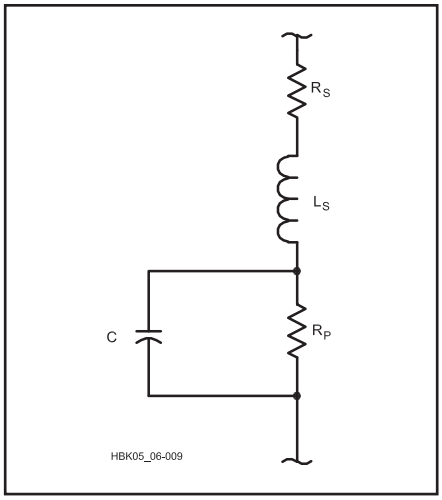


Fig 6.3 — A capacitor model that aims for improved realism at VHF and above. R_S models the net series resistance of the capacitor package; L_S, the net equivalent inductance of the structure. R_P, in parallel with the capacitance, models the effect of leakage that results in self-discharge. Intuiting the topology of this model is one thing; measuring and/or realistically calculating real-world values for R_S, L_S and R_P for application in a circuit simulator is a significant challenge. How and to what degree these parasitic characteristics may cause the electrical behavior of a capacitor to differ from the ideal depends on its role in the circuit that includes it and frequency at which the circuit operates. For simulating many ham-buildable circuits that operate below 30 MHz, the effects of component parasitic R, L and C can usually be ignored unless guidance or experience suggests otherwise.

As active-device operation moves from small-signal — in which the signals handled by a circuit do not significantly shift the dc bias points of its active devices — to large-signal — in which applied signals significantly shift active device dc bias and gain — the reality of *device self-heating* must be included in the device model. Examples: When amplitude stabilization occurs in an oscillator or gain reduction occurs in an amplifier as a result of voltage or current limiting or saturation.

Designers aiming for realism in simulating power circuits that include magnetic-core inductors face the additional challenge that all real magnetic cores are nonlinear. Their magnetization versus magnetic field strength (*B-H*) characteristics exhibit hysteresis. They can and will saturate (that is, fail to increase their magnetic-field strength commensurately with increasing magnetization) when overdriven. Short of saturation, the permeability of magnetic cores varies, hence changing the inductance of coils that include them, with the flow of dc through their windings. These effects can often be considered negligible in modeling ham-buildable low-power circuits (such as Fig 6.1), but designers using *SPICE* to simulate power supplies and electromechanical systems for mass fabrication and production must harness its ability to model nonlinear magnetics — a capability greatly limited in the demo version of *OrCAD 16.0*. See the **Power Supplies** chapter for more information on this topic.

So what of our non-specification of *Q* for the tuned-circuit inductors — and while we're at it, the tuned-circuit capacitors — in Fig 6.2? With all other factors ignored and with no steps taken to otherwise introduce realistic parasitics and losses into simulations, ideal tuned circuits will generally result in higher-than-expected output in amplifiers and heavier clipping than we might expect (and therefore only *maybe* higher output than expected) in oscillators. Ideal transformers and *LC* filters will exhibit lower-than-expected losses and sharper-than-actual resonances. Whether this matters depends on our simulation aims. If we merely want to get an idea of the frequency response of a filter, ideal *L*s and *C*s are fine; if we want to compare the efficiencies of competing matching networks or filters or realistically model filter insertion loss, ideal *L*s and *C*s will lead us astray. So, if we want to more realistically simulate the output power of an oscillator, setting a realistic *Q* for at least its tuned-circuit *L* would seem to be a good thing.

Yet we have not yet done so in Fig 6.2. Because we are new to circuit modeling in general and this circuit in particular, and because against all odds we have chosen as our first exercise the modeling of an *LC* oscillator — and merely getting a *SPICE*-simulated oscillator to start can be enough of a challenge — we will work with ideal *L*s and *C*s for now. We will go into issues of specifying realistic *Q* later as part of our exploration into evaluating circuit gain.

SOURCES

We would expect Fig 6.2 to include a 12-V dc source, and it does (V1). Unexpectedly, however, our simulated circuit includes a second source: V2, which we have configured to generate a 1-V, 1-ns-long pulse that occurs 3 µs after simulation begins. We have included this pulse source because our simulated oscillator has a high-*Q* tuned circuit and, simulated in *SPICE*, cannot start oscillating without it.

Real oscillators need help starting, too. The difference between Fig 6.2 and its real-world equivalent is that real-world transistors in well-designed oscillators can usually get themselves started oscillating without our building in any means of kick-starting them.. A real-world *LC* oscillator can do this because its active device or devices generate internal noise, which, fed back and filtered by its tuned circuit, and amplified again and again, becomes less and less noiselike and more and more sinusoidal as it builds, until some mechanism of voltage or current limiting allows the signal to increase no further. That an oscillator can non-self-destructively reach and maintain this condition of *amplitude stabilization* is no less momentous than the occurrence of oscillation startup.

SPICE can simulate active-device noise, but only during ac circuit analysis, in which any active devices present are treated as linear — that is, as operating under small-signal conditions — before circuit behavior with ac signal input is evaluated. As an oscillator starts and then reaches amplitude stabilization, its oscillatory device(s) move from small-signal to large-signal operation. We must therefore use *SPICE*'s time-domain (also known as transient) simulation capabilities to simulate Fig 6.2 if we want to observe its output power. In time-domain simulation, *SPICE* progressively calculates the voltages at and currents through each circuit node — each point of interconnection between components — as the time steps of a simulation advance from the initial conditions at Time Zero.

Our simulation of Fig 6.2 will begin as every *SPICE* time-domain simulation begins if we do nothing to adjust the initial conditions for any of its components away from their defaults: All voltage and current sources will be at their specified levels, all capacitors will not yet be charged, and all inductors and transformers will not yet be energized. Starting such an analysis is very much like suddenly connecting a 12-V battery to the real-world circuit in Fig 6.1.

Because we want to see what happens when we "just build" a real-world circuit in a simulator, we will only mention in passing the availability of the advanced technique of explicitly specifying non-zero initial conditions of key circuit components as an aid to oscillator startup. This technique requires that we first build a high-*Q* oscillator, such as a crystal oscillator, as low-*Q*, get it going well enough to determine steady-state voltage or current values for key components, rebuild it as high-*Q*, and then re-simulate it with the steady-state values in place. That said, rather than first trying Fig 6.2 without V2 only to have it fail to start, we choose with the writer's help to magically learn from that certain failure in advance by kick-starting our simulated oscillator with a voltage pulse from V2. (We also include V2 [and C6] in Fig 6.2 from the get-go as a service to your memory: Introducing the circuit without these components and adding them later in a small, separate schematic would encourage your image-memory capabilities to snapshot a picture of a simulatable circuit *that cannot work*.)

DEVICES, DEVICE MODELS AND DEVICE PARAMETERS

In building a real circuit that uses active devices, we pull the necessary parts out of storage, solder them in, and they "just work." Like the rest of the components in the projects we build, they operate to the full robustness of the intrinsic properties of their constituents and construction regardless of our knowledge of the details. We can, and do, count on it.

Not (likely) being degreed practitioners of either general circuit-simulator mathematics or of the more specialized disciplines of device manufacturing and/or modeling, we will naturally tend to do the same when using a circuit simulator. Just as with real-world devices, once we click **Run** and simulation begins, our modeled devices will act to the fullest degree allowed by their construction. Very differently, however, *absolutely every desired behavior exhibited by a simulated device must be explicitly built into the model*, mathematical atom by mathematical atom. A real-world device always "knows" exactly what to do with whatever conditions confront it (however the resulting behavior may alarm or confound us). A simulated device can reliably simulate real-world behavior only to the extent that it has been programmed and configured to do so. A 1N4148 diode from your junk box "knows" exactly what to do when ac is applied to it, regardless of the polarity and level of the signal. Mathematically *modeling* the forward- and reverse-biased behavior of the real thing is almost like modeling two different devices. Realistically modeling the smooth transition between those modes, especially with increasing frequency, is yet another challenge.

Mathematical transistor modeling approaches the amazingly complex, especially for devices that must handle significant power at increasingly high frequencies, and especially as such devices are used in digital-communication applications where phase relationships among components of the applied signal must be maintained to keep bit error rates low. The effect of nonlinear reactances — for instance, device capacitances that vary with applied-signal level — must be taken into account if circuit simulation is to accurately predict oscillator phase noise and effects of the large-signal phenomenon known as *AM-to-PM conversion*, in which changes in signal amplitude cause shifts in signal phase. In effect, different aspects of device behavior require greatly different models — for instance, a dc model, a small-signal ac model, and a large-signal ac model. Of *SPICE*'s bipolar-junction-transistor (BJT) model, we learn from the *SPICE* web pages that "The bipolar junction transistor model in *SPICE* is an adaptation of the integral charge control model of Gummel and Poon. This modified

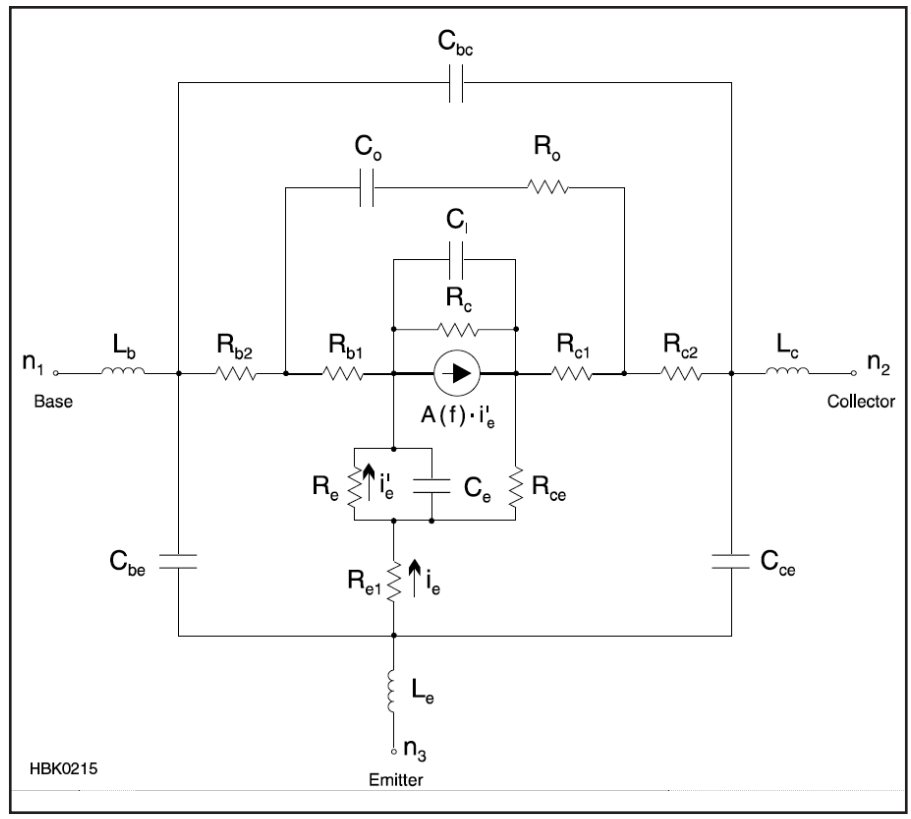


Fig 6.4 — The linear BJT model, BIP, from *ARRL Radio Designer*, a now-discontinued circuit-simulation product published by ARRL in the late 1990s. Frustratingly to users of *ARD*, real-world values for many BIP parameters could not be directly inferred from manufacturers' device datasheets. Scarcity of device parameters is much less of a problem for *SPICE* users, as the widespread use of the simulator by industry has compelled many device manufacturers to extract and publish — free for the downloading — real-world device parameter values that can be plugged directly into *SPICE*. Modern RF-fluent non-*SPICE* simulators like *Ansoft Designer SV 2* may be able to use *SPICE* device parameters directly or with a bit of parameter-renaming and value-resuffixing.

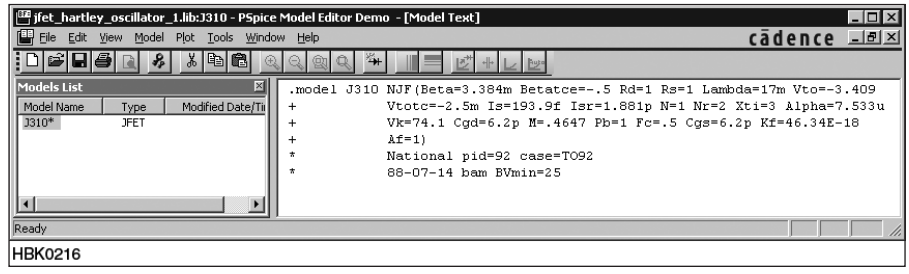


Fig 6.5 — Opening the circuit's J310 device for editing reveals that its model parameters were extracted at National Semiconductor in 1988. In the *OrCAD 16* demoware, both J310 instances in Fig 6.2 must use exactly this parameter-value set; non-demoware simulators allow separate customization of each instance of the same model. Lines that begin with asterisks are comments, ignored by the simulation engine.

Gummel-Poon model extends the original model to include several effects at high bias levels. The model automatically simplifies to the simpler Ebers-Moll model when certain parameters are not specified."

As an illustration of device-model complexity, **Fig 6.4** shows as a schematic the BIP linear bipolar junction transistor model from *ARRL Radio Designer*, a linear circuit simulator published by ARRL in the late 1990s. Luckily for those interested mainly in audio and relatively low-frequency RF applications, specifying values just for the parameters A (0.99 for average transistors) and RE (26 ÷ collector current in milliamperes, in which the *26* equates to 26 *millivolts*, the room-tem-

Table 6.2
Device Parameter Sources

Source	*Address*	*Resource*
Cadence Design Systems	**www.cadence.com/products/orcad/pages/downloads.aspx#cd**	*OrCAD*-ready libraries of manufacturer-supplied device models
California Eastern Laboratories	**www.cel.com**	data and models NEC RF transistors
Duncan's Amp Pages	**www.duncanamps.com/spice.html**	*SPICE* models for vacuum tubes
Infineon	**www.infineon.com**	data and models for Siemens devices
Fairchild Semiconductor	**www.fairchildsemi.com**	device data and *SPICE* models
National Semiconductor	**www.nsc.com**	data and *SPICE* models for National op amps
NXP	**www.nxp.com/models/index.html**	data and models for Philips devices

The National Semiconductor J310 parameters used in simulating Fig 6.2 came from a QRP-L posting ("Re: More JFET Models") archived at **qrp.kd4ab.org/1999/990616/0098.html**.

```
* FILENAME:        NE46134.CIR
* from http://www.cel.com/pdf/models/ne46134.cir
* NEC PART NUMBER: NE46134
* LAST MODIFIED:   11/97
* BIAS CONDITIONS: Vce=5V, to 12.5V,  Ic=50mA to 100mA
* FREQ RANGE:      0.1GHz TO 2.5GHz
*
*                           CCBpkg
*                      .-------||--------.
*                      |           CCB    |
*                      |       .--||---.  |           COLLECTOR
* BASE                 |       |        | |
*               __Tb_  |       |   /----o-o-------o----o
*  o----------|_____|-o-LB---o--|Q1   |          |
*                      |       B   \   --          |
*                     --           |   -- CCE     --
*                     -- CBEpkg E o----'          --CCEpkg
*                      |           |               |
*                      |          LE               |
*                      `----------o---------------'
*                                 |
*                                 -
*                                | |  Te
*                                | |
*                                 -
*                                 |
*                                 o EMITTER
*
*
*       CCB   = 0.03 pF       LB  = 1.2nH          Tb/Te:
*       CCE   = 0.5 pF        LE  = 1.2nH          z=60 ohms
*       CCBpkg= 0.18pF                             l=50 mils
*       CCEpkg= 0.18pF                             a=0.0001
*       CBEpkg= 0.01pF                             f=0.9GHz
*
*                   c b e
.SUBCKT NE46134/CEL 2 1 3
Q1 2 6 7 NE46100
CCB 6 2  0.03E-12
CCE 2 7  0.5E-12
LB  4 6  1.2E-9
LE  7 5  1.2E-9
TB 1 0 4 0 Z0=60 TD=9.63E-12
TE 5 0 3 0 Z0=60 TD=9.63E-12
CCBPKG 4 2 0.18E-12
CCEPKG 2 5 0.18E-12
CBEPKG 4 5 0.01E-12
.MODEL NE46100 NPN
+( IS=8.7e-16    BF=185.0         NF=0.959       VAF=30.0         IKF=0.20
+ ISE=5.70e-13   NE=1.80          BR=5.0         NR=1.0           VAR=12.4
+ IKR=0.018      ISC=1.0e-14      NC=1.95        RE=0.630         RB=6.0
+ RBM=4.0        IRB=0.004        RC=3.0         CJE=4.9e-12      VJE=0.60
+ MJE=0.450      CJC=2.50e-12     VJC=0.830      MJC=0.330        XCJC=0.20
+ CJS=0.0        VJS=0.750        MJS=0.0        FC=0.50          TF=12.9e-12
+ XTF=1.60       VTF=19.9         ITF=0.40       PTF=0.0          TR=1.70e-8
+ EG=1.11        XTB=0.0          XTI=3.0        KF=0.0           AF=1.0 )
.ENDS
*$
```

Fig 6.6 — ASCII art lives on in the depiction of package parasitics in the California Eastern Laboratories model of the NE46134 linear broadband transistor. In introducing us to the concept of subcircuits — a defined circuit chunk that, once simulated in a given simulation run, can be reused multiple times elsewhere in the simulation. This model also illustrates that *SPICE*, in common with other simulators, uses a network list — netlist — to communicate circuit topology, component values, and analysis instructions to its simulation engine.

perature value of V_T, the thermal equivalent of voltage in the transistor's semiconductor material) can suffice for good-enough-for-basic realism with the BIP model or linear BJT model equivalent to it.

Especially in the area of MOSFET and MESFET device modeling, and large-signal device modeling in general (of critical importance to designers of RF integrated circuits [RFICs] for use at microwave frequencies) *SPICE* and RF-fluent non-*SPICE* simulators include active-device models home experimenters are unlikely, even unable, to use. (Do BSIM3 and BSIM4 ring a bell? MEXTRAM? Statz, Curtice and TriQuint GaAs FET models [levels 1, 2, 3 and 6]? No points off if you can't already answer *yes* to these extra-credit questions. Several, if not most, of these models are of interest only to EE students and their teachers, those who work for a semiconductor foundry that uses them, and those who produce circuit-simulation products that implement them.)

Most of us will go (and need go) no further into the arcanities of device-modeling than using *SPICE*'s JFET model for FETs like the 2N3819, J310, and MPF102, and *SPICE*'s BJT model for bipolar transistors like the 2N3904. Getting the hang of the limitations and quirks of these models may well provide challenge enough for years of modeling exploration. (We'll encounter another device-modeling option — representing devices as black boxes characterized by manufacturer-supplied network parameter datasets — later as we consider the RF-fluent simulator *Ansoft Designer SV 2*.)

Obtaining real-world-useful device-parameter values to plug into even *SPICE*'s standard diode, BJT and JFET models is critically important if we are to creating simulations that work well. *OrCAD 16.0* includes preconfigured 1N914, 1N4148, 2N2222, 2N2907A, 2N3819, 2N3904 and 2N3906 devices, among others, and these will be sufficient for many a ham-radio simulation session. To get parameters for other devices, especially RF devices but also including the J310 JFETs of Fig 6.2, we must search the Internet in general and device-manufacturer websites in particular to find the data we need. The manufacturer sites listed in **Table 6.2** will get you started.

Fig 6.5 shows the Fig 6.2 J310 parameters in *OrCAD 16*'s component editor. Because the demoware version of *OrCAD 16.0* does not allow us to edit multiple instances of a device independently of each other, the J310s in our simulation are identical.

Fig 6.6 illustrates the level of detail involved in more-accurate device modeling for VHF and UHF. The device is a California Eastern Labs NE46134, a surface-mount BJT intended to serve as a broadband linear amplifier at collector currents up to 100

```
* source JFET_HARTLEY_OSCILLATOR_1
J_J2          N012082 N01236 N01504 J310
D_D1          N00091 0 D1N4148
Kn_K2          L_L1 L_L2     0.99
C_C8          0 N02935  0.001uF
J_J1          N00193 N00091 N00753 J310
C_C7          0 N09766  0.001uF
L_L6          N19502 N02415  50uH
V_V2          N07095 0
+ PULSE 0 1 3us 0 0 1us
R_R1          0 N00091  1E6
V_V1          N02415 0 12Vdc
C_C11          0 N15520 {30pF*0.5+.001p}
R_R2          N00193 N02415  10
C_C6          N07095 N00753  0.1pF
C_C9          N09766 N19502  0.1uF
L_L1          N00753 N15520  1.425uH
C_C4          N01236 N00753  10pF
L_L4          N012082 N18891  50uH
R_R3          0 N01504  270
C_C1          N00091 N15520  2.35pF
Kn_K1          L_L4 L_L5
+ L_L6    0.99
C_C2          0 N00193  0.1uF
R_R5          0 N02935  50
C_C5          0 N01504  0.1uF
C_C10          0 N02415  0.1uF
L_L5          N18891 N19502  50uH
L_L2          0 N00753  0.075uH
R_R4          0 N01236  100k
C_C3          0 N15520  300pF
L_L3          N09766 N02935  350nH
```

Fig 6.7 — The JFET VFO circuit in *SPICE* netlist form. The N*nnnn* declarations name circuit nodes or *nets* — points of interconnection between components. Each component statement names the part's *primitive* — the leading *J* (for JFET) in *J_J1* — and identifies its instance (*J1*) to form an instance name (*J_J1*). Net numbers, generated automatically and algorithmically by the *netlister* function associated with the schematic editor, are ordered in element statements such that element *pins* — points of electrical interconnection — are connected to the correct nets as graphically depicted in the schematic. With some tedium, display of net names may be toggable in a simulator's schematic editor; net name display in *OrCAD 16.0* must be enabled one net at a time. This can be useful because netlisters occasionally make mistakes, and comparing a circuit's netlist to its schematic may aid in troubleshooting simulation errors.

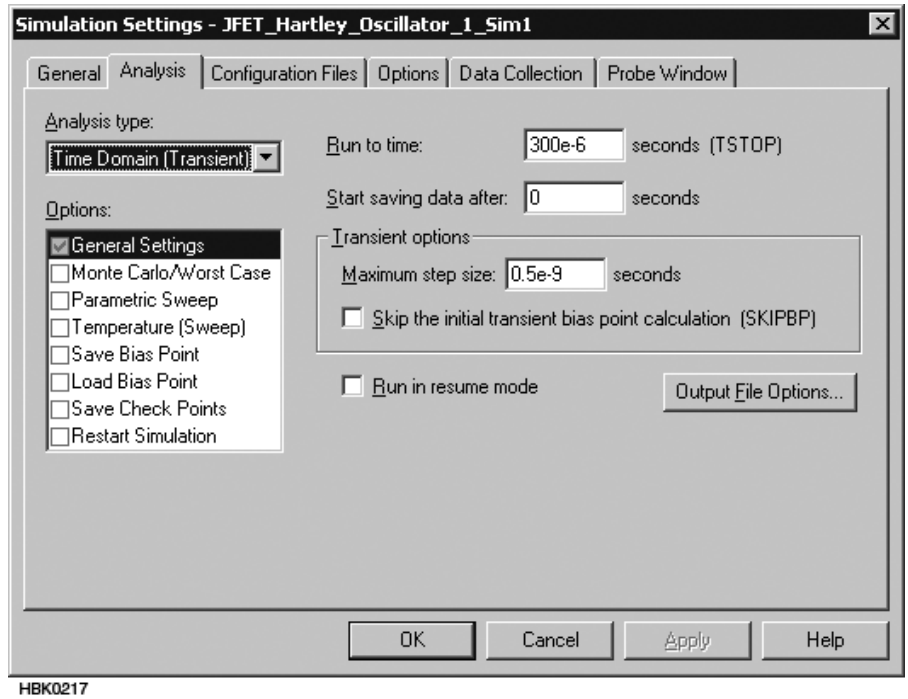


Fig 6.8 — *OrCAD 16 SPICE* setup for transient analysis of the JFET oscillator-buffer circuit. The behavior of the circuit will be progressively simulated every 0.5 ns (at most) from 0 to 300 µs.

Simulation Goal or Moving Target?

In simulating Fig 6.1, we set out to confirm the real-world circuit's output power, 10 dBm (10 mW). Fig 6.10 shows that we did just that. So, are we done? That depends.

Fig 6.A1 shows what the circuit's output waveform looks like when we zoom in on Fig 6.9 after the circuit has had time to amplitude-stabilize. It's clearly *not* a sine wave! Does our simulated circuit have a problem? What should we do next? That depends.

Circuit simulation allows us to explore circuits and what we know — and what we *think* we know — about them in a highly elastic and dynamic way. We may go into a simulation looking to find the answer to a simple question or questions, and find ourselves inventing new questions, even new modeling goals, along the way. This can bring good news and bad news. The good news is that our intuition and learning can be supercharged by whatever we may encounter. The bad news is that we can be misled by assumptions we may not even know we've been making.

An accurate RF power meter substituted for R5 in Fig 6.2 can indicate the absence or presence of RF at that point and its absolute level, and no more. The simulator's reporting function played the same role in providing us with the output-power curve shown in Fig 6.10. Whether the power-measured signal is spectrally dirty or clean, whether its frequency drifts or jumps — these characteristics, and more — cannot possibly be inferred from power measurement. Luckily — or maybe not — a simulator's reporter can tell us much more.

In real life, we would build our circuit expecting to see 10 mW output, measure that value with our RF power meter — if we even *have* a meter — connect the VFO to the receiver we built it to drive, and tune happily away. Little would we know that behind the deceptive simplicity of our measurement may lurk a signal that's other than a sine wave. (Even more arcane: The VFO output signal may be a sine wave when driving a resistive load but become non-sinusoidal when connected to the more reactive load presented by the receiver mixer's local-oscillator port.)

Perhaps the waveform in Fig 6.A1 really *does* generally reflect what real-world copies of Fig 6.1 do — but at this red-hot second we don't know enough to be sure. The description of the original circuit did not include a picture of its output waveform. A good high-frequency oscilloscope connected to the output of the real thing could tell us; a spectrum analyzer could also tell us, if a bit less directly.

Short of that, we can only intelligently speculate: Maybe the model data we used for the J310 JFET is insufficiently realistic. Maybe *SPICE*'s built-in JFET model folds up somehow as devices modeled with it move into the large-signal operation that oscillator amplitude stabilization involves. Maybe a signal like Fig 6.A1 happens whenever we diode-clamp the gate voltage of a JFET oscillator. Maybe the oscillator is overdriving the buffer amplifier. Maybe we or the author misspecified a value or left out a component. (Note to self: This is the first thing to check.) Maybe we have been unwarrantedly assuming for all of our radio-experimentation lives that unseen sources are sinusoidal until proven guilty.

Industrial-strength circuit simulators come to us as a result of engineering disciplines that seek to understand the world and predict its behavior toward the ultimate goal of achieving practical tools reproducible in quantity. As radio-hobbyist experimenter-builders, we may be just as satisfied with speculating about, and exploring of the quality and behavior of, a tool far beyond our having achieved its sufficiently practical function.

Computerized simulation can empower both approaches. The trick for us experimenters is to know when we've moved from solving a narrowly defined problem to dynamically redefining the problem such that enough is never enough. If you spend a half hour tweaking a bandpass-filter simulation for a –3-dB bandwidth of exactly 200 kHz through specification of capacitance values out to three decimal places, will you be able to achieve exactly that result with parts from your junk box? Will you even be able to know if you have? Will it even matter when you use your circuit on the air?

So what *about* that non-sinusoidality in Fig 6.A1? Is it a problem, an opportunity, or neither? Writing the rest of its story is up to you.

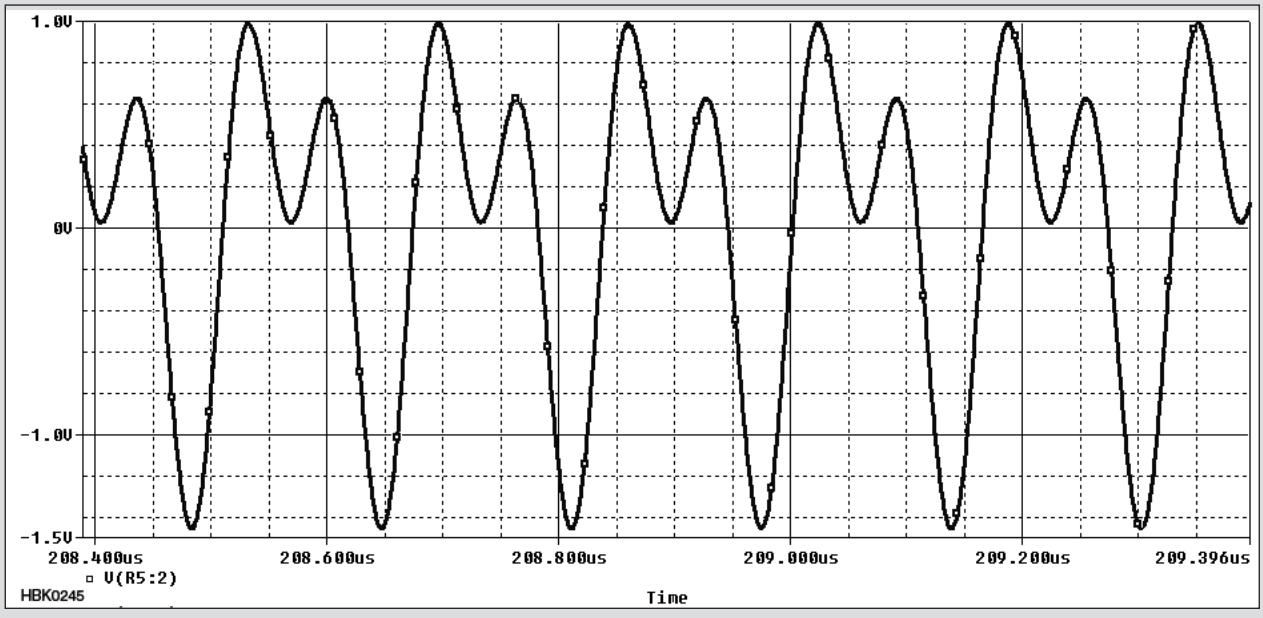


Fig 6.A1 — Zooming in on the JFET VFO's output signal confirms what the waveform asymmetry in Fig 6.9 and the second-harmonic spike in Fig 6.11 imply: that the VFO's output is not a pure sine wave. But is this a problem? That depends.

mA and collector voltages up to 12.5. This manufacturer-supplied model embeds the unpackaged device chip (NE46100) within a *netlist*-based subcircuit that models parasitic reactances contributed by the transistor package, including chip-to-lead connections. At MF and HF, where the NE46134 could serve well as a strong post-mixer amplifier, modeling the device with just its basic, bare-chip characteristics (declared in the .MODEL NE46100 statement) would likely be accurate enough for many applications.

That Fig 6.6 illustrates the device parasitics using ASCII art is a side attraction. The main show — aside from the conveyance of the *SPICE* parameters of the NE46100 chip in its .MODEL statement — is the depiction of the NE46134 device as an NE46100 chip embedded in a subcircuit defined in netlist — *network list* — form. A netlist is a specialized table that names the circuit's components, specifies their electrical characteristics, and maps in text form the electrical interconnections among them. Uniquely numbered nodes or *nets* — in effect, coordinates in the connectivity space walked by the simulated circuit — serve as interconnects between components, with each component defined by a statement comprising one or more netlist lines. Statements that must span multiple lines include *continuation characters* (+) to tell the netlist parser to join them at line breaks. Asterisk (*) or other non-alphanumeric characters denote comments — informational-to-human lines to be ignored by the simulator. In Fig 6.6, header information and the ASCII-art portrayal of the device-package parasitics are *commented out* in this way.

The netlist served as the original means of circuit capture for all simulators known to the writer, including *SPICE* and the now-discontinued *ARRL Radio Designer* simulation product; *schematic* capture came later. Further reflecting *SPICE*'s pre-graphical heritage is the fact that, to this day a *SPICE* netlist may be referred to by long-time *SPICE* hands as a *SPICE deck*, as in "deck of Hollerith punch cards." In *SPICE*'s early days, circuit definitions and simulation instructions (netlist statements that begin with a period [.]) were commonly conveyed to the simulation engine in punched-paper-card form. All of the circuit simulators known to the writer still use a netlist as a means, if not *the* means, of conveying circuit topology and simulation instructions to the simulator; simulation based on schematicless user-created netlists may be possible with some.

Fig 6.7 shows the VFO circuit rendered as a partial netlist for simulation. This netlist is "partial" in the sense that it omits simulation and output commands we would expect to see in a full *SPICE* deck. At analysis time, the netlist contents we're shown by *OrCAD*

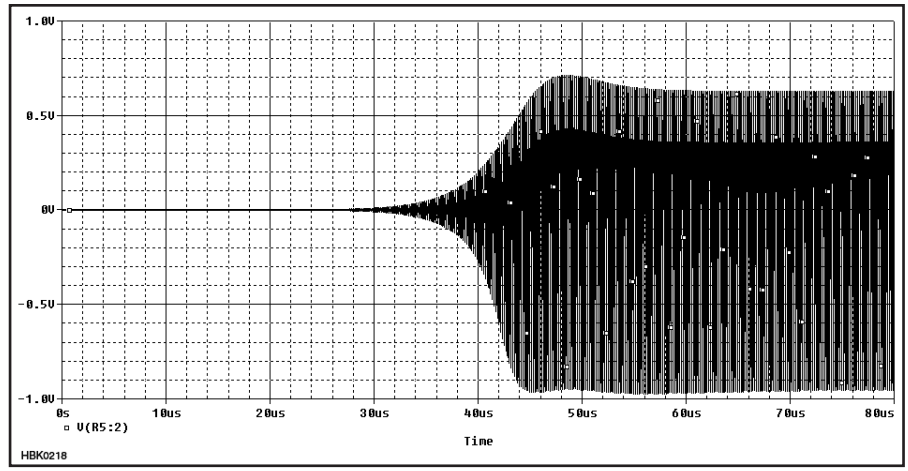


Fig 6.9 — At last: The spectacle of oscillator startup and amplitude stabilization rewards our attention to modeling detail. This graph shows the voltage at point A in Fig 6.2 — that is, the voltage, referred to ground (node 0), at node 2 of R5 [in this simulator's reporter-terminology, V(R5:2)], the circuit's 50-Ω load. The square dots scattered throughout the plot are trace markers. Note that the waveform's peak-to-peak span is not symmetrical around 0 V.

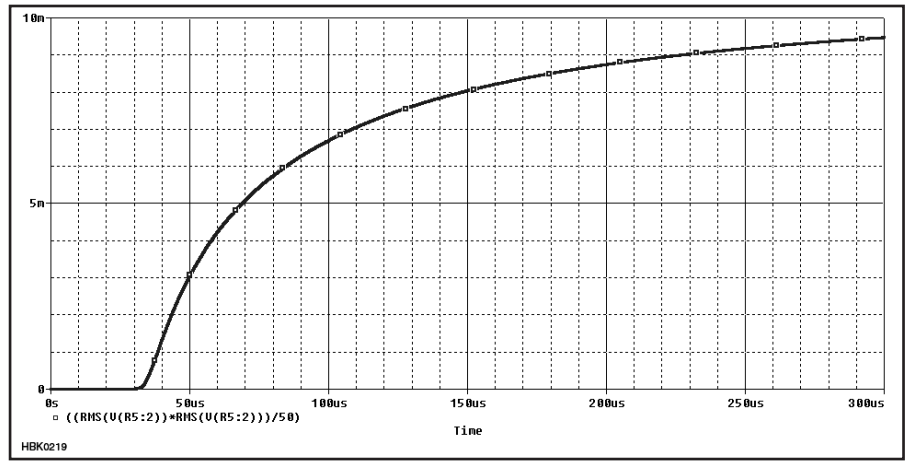


Fig 6.10 — To graph the circuit's power output, we render the voltage across R5 as RMS with the reporter's built-in RMS function, square the result by multiplying it by itself, and then divide the result by 50, the resistance of R5. By the end of the simulation, the output has reached 9 mW (9.5 dBm) — realistically close to the 10 dBm reported for the real-world circuit.

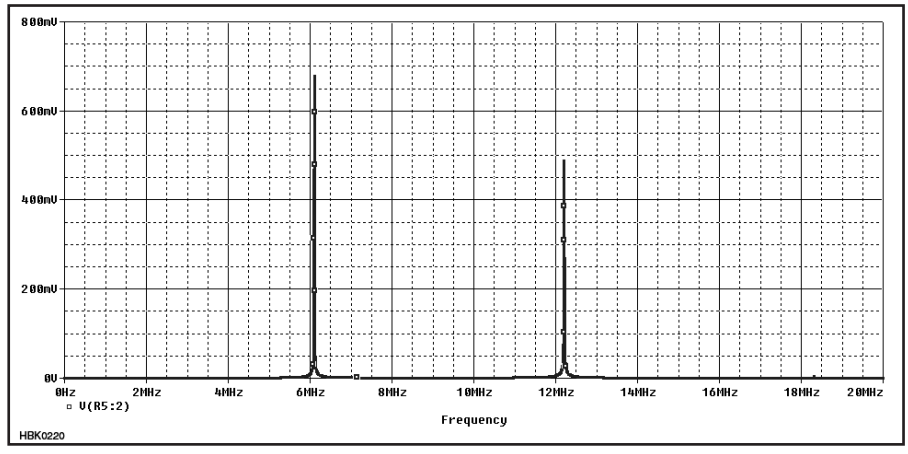


Fig 6.11 — Output spectrum of the simulated VFO. To make this clean spectral plot, we told the reporter's fast Fourier transformer (FFT) function to use only data beyond 60 microseconds after the start of simulation. Including data up through startup and stabilization data would cause the FFT to generate noise and discrete-frequency artifacts that diverge from real-life behavior.

16.0 are concatenated in memory with other data, including analysis setup information.

ANALYSIS SETUP

Many of us know from our smattering of learned theory that a complex waveform must be sampled at a frequency at least twice that of the highest-frequency component present if the samples taken are to be acceptably representative of reality. In doing transient (time-domain) simulations in *SPICE*, we have a similar concern: We must sample circuit behavior over time often enough to show us accurately the highest-frequency effects we want to see. Present-day *SPICE* simulators can intelligently size the maximum time step used in transient analysis to a value appropriate for useful representation of the highest-frequency fundamental source in a simulation. That said, manually setting the maximum step to a smaller-than-automatic value can provide smoother-looking waveform graphs more acceptable to the eye. Smaller time steps come at the expense of longer simulation times and larger simulation-data files — issues more important in the earlier days of *SPICE* than nowadays, when gigahertz-class CPUs, gigabytes of RAM and even terabyte-class hard drives are standard. In this case, to simulate Fig 6.2 we set up a transient analysis out to 300 microseconds and a minimum step size of 0.5 ns (**Fig 6.8**). And then we click **Run**.

The *OrCAD 16.0* reporter opens when the analysis has finished. What we want to see is the circuit's output waveform across R5, the 50-Ω load we added. In reporter-terminology for this simulator, that's V(R5:2). **Fig 6.9** shows the resulting graph, rescaled a bit to center the instant of startup in the frame.

As a goal within our more general goal of getting a feel for "just building" simulated circuits as we often build them in the real world, we undertook this simulation with the aim of confirming the real-world circuit's claimed output power of +10 dBm (10 mW). That we have done, as **Fig 6.10** shows. Interestingly, the power level rises relatively slowly (and actually is still edging higher — ultimately to 9.5 mW — even after 300 µs). Does this reflect the behavior of the real thing? The same question may occur to us after we view the circuit's output spectrum, which **Fig 6.11** shows on a linear voltage scale. Shouldn't the output of our VFO be a pure sine wave? For a discussion of where such assumptions, unconscious and otherwise, may lead us, see the sidebar, "Simulation Goal or Moving Target?"

6.2.2 Example 2: Modeling a Phase-Lead RC Oscillator

The small maximum time step (0.5 ns) and long simulation period (300 µs) we set in simulating Fig 6.2 result in simulation runs that take nearly two minutes on an 868-MHz Pentium III computer with 512 MB of RAM. We chose those settings to give the circuit time to amplitude-stabilize and to allow for smoother display of waveform graphs when we zoom in. If our simulating computer has sufficient RAM and disk space to generate and handle the resulting large data file (170 MB), *OrCAD 16.0* is ready to graph simulation results from Fig 6.2 in the time it takes to start a cup of tea steeping.

As an illustration of a class of oscillators that can simulate much more rapidly, **Fig 6.12** presents an RC phase-lead circuit used as a CW sidetone generator in popular Amateur Radio transceivers of the 1970s and 1980s. In this case, the addition of a kick-start pulse is not required because charging currents in the circuit's 0.012-µF capacitors provide the stimulus for startup; **Fig 6.13** shows its startup to 40 ms.

As we did with the JFET VFO example, we can use the reporter's FFT function to display the circuit's output waveform as an amplitude-vs-frequency (spectral) graph. We

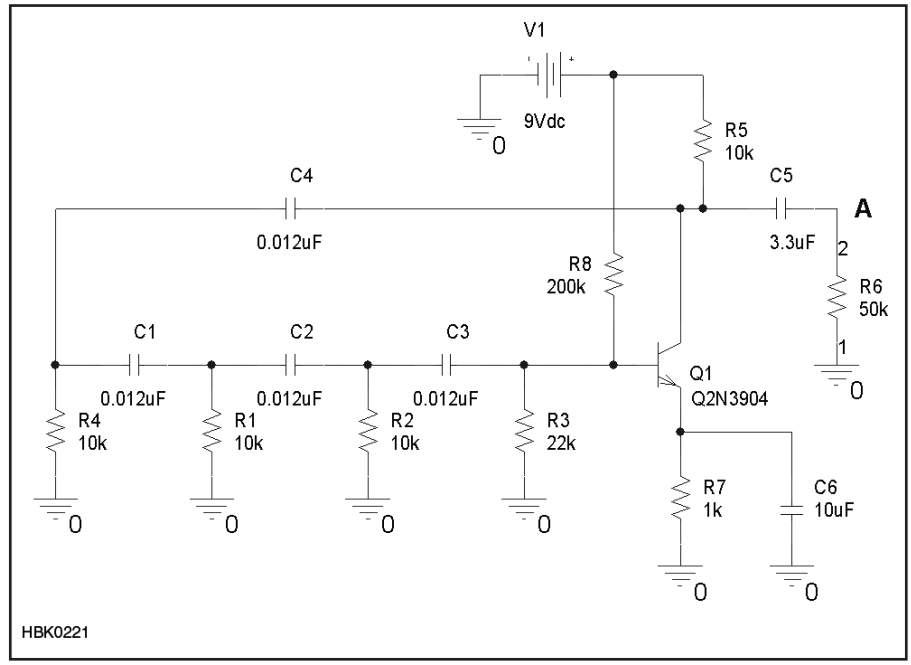


Fig 6.12 — Popular ham transceivers of the 1970s and 1980s included an RC phase-lead oscillator very much like this one as a CW sidetone generator. This modeled oscillator does not need a kick-start pulse, as the cascading disturbance of charging currents progressing through its *RC* phase-shift network is sufficient to start oscillation. Real-world builders may find that the lossiness of the circuit's feedback loop may require careful selection of the 2N3904 to ensure reliable starting.

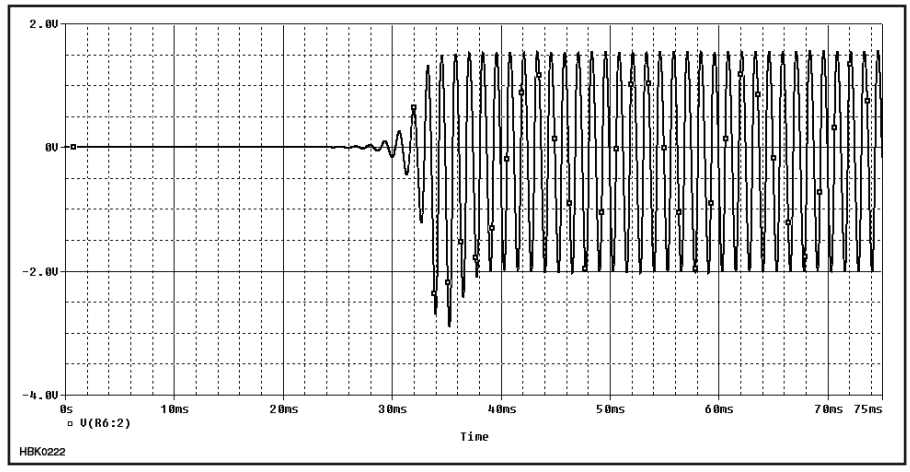


Fig 6.13 — Oscillator startup as embodied by the *RC* oscillator. This simulation (10-µs steps to 75 ms) takes only a few tens of seconds in a relatively slow computer; the Fig 6.2 simulation, *minutes*.

must do so with care, however, as the results may vary significantly with the amount and type of data used in the transform. To illustrate this, **Fig 6.14** shows the circuit's output spectrum based on analysis data from 40 ms to 500 ms, and **Fig 6.15** shows us what the FFT reports when we tell it to use only data from 480 to 500 ms. In effect, an FFT becomes surer of its results — the spectral components it reports sharpen — as we give it more data to work with; on the other hand, once the FFT has shown us all there is to see, we encounter diminishing returns — insufficient improvement in resolution as the dataset grows — if we make the simulation longer than it needs to be.

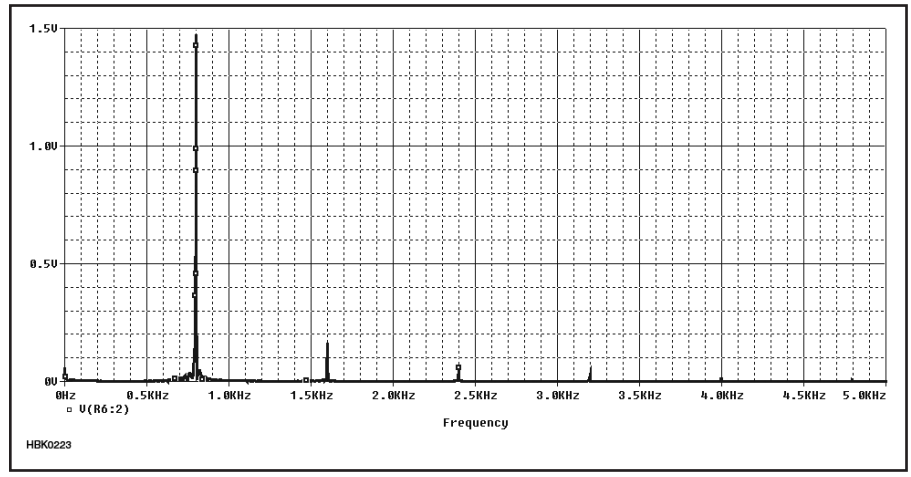


Fig 6.14 — Output spectrum of the RC phase-lead oscillator on a linear voltage as returned by the *OrCAD 16* reporter's fast Fourier transform (FFT) function. For a cleaner response — that is, to avoid displaying mathematical resultants from the circuit's rapidly changing spectral characteristics before and through startup and amplitude stabilization, and to give the FFT a larger periodic dataset to digest — we have extended the analysis to 0.5 s and excluded from the FFT analysis data between 0 and 40 ms.

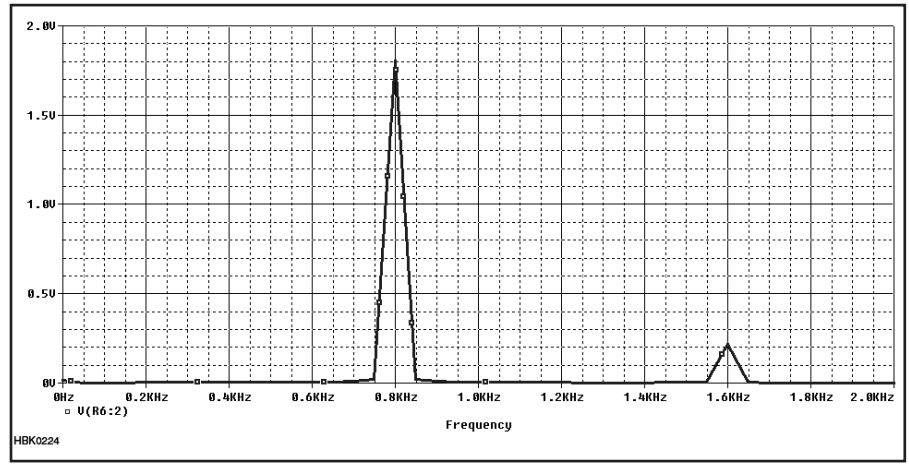


Fig 6.15 — Greatly restricting the dataset digested by the FFT degrades the amplitude-vs-frequency resolution of the report to the point of unrealism.

6.2.3 Example 3: Exploring Issues of Modeling Gain, Link Coupling and Q with a 7-MHz Filter

In evaluating the simulated oscillators in Figs 6.2 and 6.12, we had little difficulty in identifying the schematic junction — *node* — at which to probe the signal we wanted to evaluate. Other than working through the issue of graphing RMS power rather than peak power for our simulated VFO, we were able to graph our simulations' output without difficulty. Choosing the circuit point at which we would monitor our simulated circuits' behavior was simplified because an oscillator has only one *port* — excluding power and bias sources and control lines, only one point of interaction with the outside world.

Obtaining simulation results that we can both understand and trust becomes more complex when we simulate circuits with two or more ports. This is so because reporting the behavior of a simulated circuit is form of Q & A: Through the simulator's reporting functions we formulate a question, receiving as our answer a graph or table of circuit responses as numerical values, manipulated by such additional mathematical operations as we may specify.

From life experience we know that asking the wrong question will necessarily give us a wrong — though not necessary useless — answer. Especially if we are not electrical or electronics engineers, however, asking the wrong question of a simulator's reporting functions — its *reporter* — is deceptively easy. Our real-world experience in evaluating circuit behavior may not have prepared us for the subtleties of correctly posing even the most basic question to our simulator's reporter. The most immediate and far-reaching example of this is the evaluation of circuit gain.

The *concept* of gain as a ratio of output power, voltage or current to input power, voltage or current, perhaps expressed in decibels, is straightforward enough: Gain is at base a ratio that expresses the difference between the level of a signal at a circuit's output and the level of the same signal at the circuit's input. Sometimes we express gain numbers directly ("a voltage gain of 10"); sometimes we express gains terms of decibels ("a gain of 20 dB"). And we usually, but not always, call negative gain *loss*.

Measuring gain in real life is straightforward as well — at least gain as we are accustomed to thinking about it when we evaluate mixers, amplifiers, and filters for many ham-radio purposes. **Fig 6.16** shows a test-bench setup for measuring and displaying the gain (or loss), of a two-port circuit — a *two-port*, to put it generically — across a range of frequencies. In it, a signal generator — the tracking generator — applies a test signal to the input of the device under test (DUT), and an output-level-calibrated receiver — the spectrum analyzer in Fig 6.16 — receives the test signal as modified by the DUT and displays the results as an output-vs-frequency graph (**Fig 6.17**). To determine the gain or loss of the DUT at a given frequency, we'd compare that graph to the graph we get when we connect the tracking generator directly to the spectrum analyzer. When in ARRL publications we say that an RF circuit has "2 dB of loss" or "16 dB of gain," we almost always mean the type of gain derivable from measurements made by this process or its equivalent.

Type of gain? There's more than one? Yes: See the sidebar, "Defining Gain." It turns out

Defining Gain

Although the concept of gain as the ratio of the voltage, current or power at the output of a circuit to the power, voltage or current at the input of a circuit may seem to require no qualification, exactly where we measure these values in a system under test can greatly affect the gain value returned. Determining the point at which to measure circuit output is relatively straightforward: We measure output power in the load, output voltage across the load, and output current through the load. But what about input power, voltage, or current? Considering the problem only in terms of power, **Fig 6.A2** lays the groundwork for an understanding of this issue by depicting a gain-measurement setup in generic form.

Writing in NTIA Publication TR-04-410, *Gain Characterization of the RF Measurement Path*, J. Wayde Allen shows that four possible definitions can be proposed for the power gain of the 2-port in Fig 6.A2:

$$G_1 = \frac{P_{2a}}{P_{1a}} \quad (1)$$

$$G_2 = \frac{P_{2a}}{P_{1d}} \quad (2)$$

$$G_3 = \frac{P_{2d}}{P_{1a}} \quad (3)$$

$$G_4 = \frac{P_{2d}}{P_{1d}} \quad (4)$$

where *a* denotes available power and *d* denotes delivered power.

Equation 1 is commonly considered as describing the *available gain* (G_a) of the 2-port; equation 3 as describing the 2-port's *signal gain* (G_s) or transducer gain (G_t); and equation 4 as describing *power gain* (G). Further qualification of measurement conditions leads to additional, more specialized definitions for gain.

When we simulate circuits with the aim of directly comparing the results with real-world measurements, we need to simulate measurements made with a test set like that shown in Fig 6.16. To do that, we must know exactly which circuit points to probe to give a ratio that, expressed in decibels, gives the same results we'd see if we tested our simulation's real-world counterpart in the Fig 6.16 setup. Keeping Fig 6.A2 in mind, knowing which of the equations above corresponds to the gain measured by the Fig 6.16 setup could help us determine where to probe in our simulations. (By the way, considering that throughout our example simulation discussions we have chosen to generate graphs in terms of signal voltage, here we should mention that equations 1 through 4 are just as valid for voltage and current as they are for power; it's only when we want to render G in decibels that we must remember whether to multiply the logarithm of *G* by 10 or 20.)

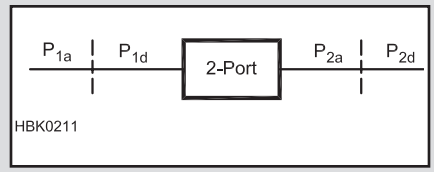


Fig 6.A2 — Generalized two-port gain evaluation in terms of available (subscript *a*) and delivered (subscript *d*) power (P) at port 1 (subscript *1*) and port 2 (subscript *2*). The power available from a source can differ from the power delivered to its load as a result of impedance mismatch between source and load. Although the subtleties of the issues involved are beyond the scope of this chapter, we must note that measuring gain becomes even more complex in the presence of sources and loads that are not purely resistive — that is, when voltage and current are not in phase.

Reading further in Allen, it turns out that none of those equations will suffice, for what the Fig 6.16 setup actually measures is *insertion gain* (G_i), which, thinking in terms of power, can be defined as

$$G_i = \frac{P_d}{P_r} \quad (5)$$

where P_d is the power delivered to the load when the 2-port under test is connected between the signal generator and the load, and P_r, the reference power, is the power delivered to the load when the 2-port under test is absent and the signal generator is connected directly to the load.

Working with a spectrum analyzer to determine gain or loss involves exactly this two-step operation. What a test setup like that shown in Fig 6.16 determines is insertion gain. As we'll see in simulating a 7-MHz band-pass filter, having access to the internals of the test-signal generator lets us report a circuit's insertion gain in just *one* step with the help of a bit of math.

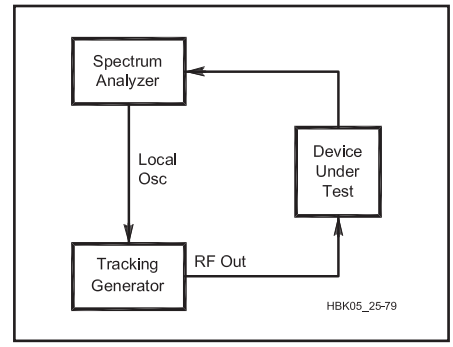


Fig 6.16 — One possible test setup for measuring the gain-versus-frequency response of a two-port device under test (DUT). This simplified diagram does not show the additional input and/or output attenuation that would be present in many actual measurement scenarios. For example, the presence of an active DUT (one expected to exhibit positive gain, as opposed to negative gain [loss]) would compel us to add attenuation between its output and the spectrum-analyzer input — not only to keep the spectrum-analyzer receiver from overloading and giving us false results, but also to protect the analyzer from damage if the DUT were to start oscillating rather than just amplifying.

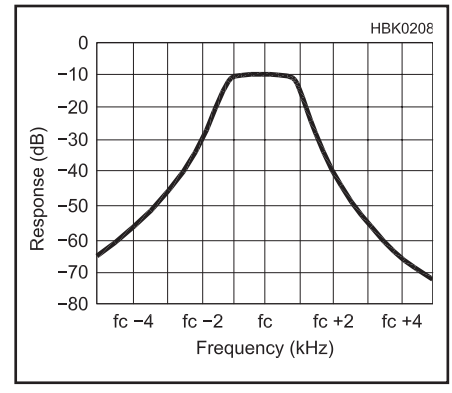


Fig 6.17 — The passband response of a crystal ladder filter as displayed by a spectrum analyzer.

that the gain we measure with a system like that shown in Fig 6.16 returns values that correspond to just one among multiple possible definitions of gain. In this case, our test system measures *insertion gain*: the difference between output measurements made at the load with the source connected directly to the load and with the device under test inserted between source and load. If we want to be able to compare the gain of a CAD-simulated two-port device with the gain of a counterpart real-life device measured in a test setup like Fig 6.16, we must tell our CAD program to report our circuit's insertion gain.

Using the Fig 6.16 system to measure in-

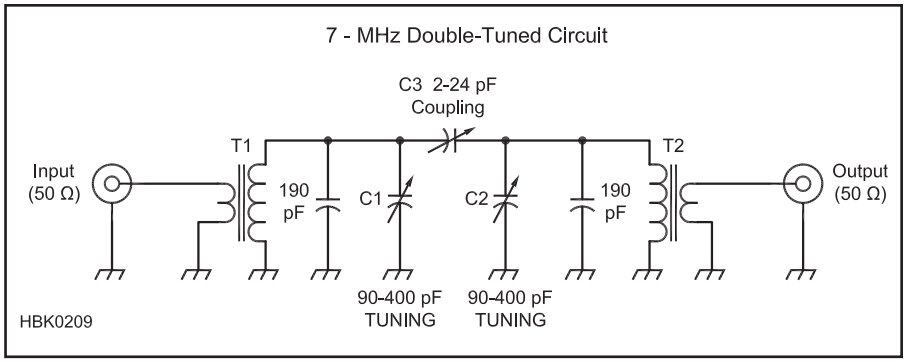


Fig 6.18 — A 7-MHz double-tuned-circuit filter as drawn for real-world builders. Each resonator consists of 17 turns of #22 wire on a T-50-6 toroidal, powdered-iron core; the coupling links consist of 3 turns of insulated wire. Per the article that described this circuit, the filter is intended to have "a 7.1-MHz center frequency and a 200-kHz bandwidth."

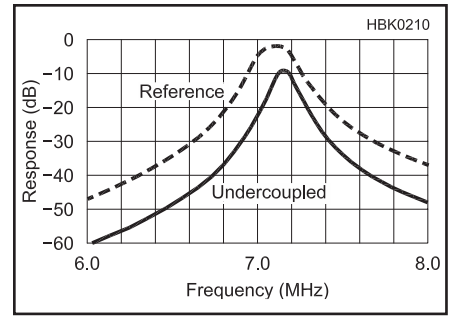


Fig 6.19 — Published response (Reference curve) of the 7-MHz filter. Careful reading reveals that these and other response graphs in the source article were generated by CAD software ("a computer generated the data for this and the other graphs in this article; experiment confirmed the data"); additional reading provides measurement results that provide real-world modeling goals: "The result: a critically coupled filter that's 178 kHz wide, with just over 2 dB of insertion loss."

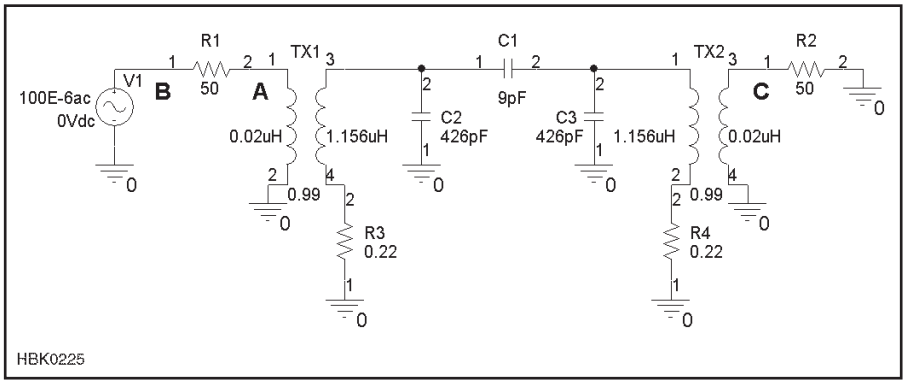


Fig 6.20 — The 7-MHz filter ready for SPICE simulation in *OrCAD 16.0 Capture CIS*. Transformer (TX) components simulate the circuit's link-coupled resonators (note the coupling coefficient, *K*, of 0.99), and 0.22-Ω resistors added in series with the tuned-circuit windings to model realistic *Q*. (No attempt is made to simulate realistic *Q* in the coupling links [0.02 µH] as real-world data is unavailable for that characteristic.) Paralleled capacitances in Fig 6.18 are represented by single values — an approach worth cultivating as a habit to stay within the component-count limitations of feature-limited demoware. Sinusoidal voltage source V1 (100 µV) and R1 serve as the test-signal generator, with R1 also serving as the filter's input termination; R2 serves as the filter load. Although intuition tells us that probing node R2:1 (labeled C) will give the circuit's output voltage, real-world test-bench experience does not immediately suggest where (point A or B?) to obtain the input-voltage value necessary for calculating the circuit's insertion gain as a real-world test setup would report it.

sertion gain is straightforward because it is hard-configured to produce two-port insertion-gain measurements. Its spectrum analyzer can report only the signal level present at the load, and its TEST GENERATOR OUTPUT and SPECTRUM ANALYZER INPUT jacks enforce the insertion of the DUT at a particular point between source and load. We must merely take care not to connect the DUT backwards — unless doing so might return some other value of interest.

Reporting the insertion gain of circuit simulated in *SPICE* (even a very simple one) is fundamentally trickier because the relationships between source, DUT and load, and between signal-level metering and load, that can be safely assumed to exist in Fig 6.16 are neither predefined nor enforced in *SPICE* simulations. We must build a test-signal generator into our circuit, and we may report the signal level at one circuit node as easily as another. To explore how this complication can affect the results we see, we'll examine a simple 7-MHz filter in *OrCAD 16.0* — that is, as simulated by *SPICE* — and in a more-RF-friendly simulator, *Ansoft Designer SV 2*.

Fig 6.18 shows the circuit, a top-coupled double-tuned-circuit (DTC) design intended to provide a 3-dB bandwidth of 200 kHz centered at 7.1 MHz. From its description, we learn that its inductors consist of 17-turn windings on T-50-6 powdered-iron cores, with 2-turn links for input and output coupling. Per the inductance-from-turns equation associated with the toroidal core properties tables in this *Handbook*'s **Component Data and References** chapter, we calculate the inductance of the resonators as 1.156 µH; of the coupling links, 0.02 µH. **Fig 6.19** shows its response as published by Hayward.

Because we want the response of our simulated filter to approach Hayward's results as closely as possible, we must somehow specify the quality factor, *Q*, of its resonator inductors —their *loaded Q* (Q_U). In this case, the most direct approach would be to construct the coils and measure their *Q* on a *Q* meter. Not having access to one, we do the next best thing and extrapolate from another's careful *Q* measurements on a similar coil. From the **Measured Toroidal Inductor *Q* Values** table in Zack Lau's "RF" column in March 1995 *QEX*, we estimate our resonators' *Q* as 238 based on his data for a very similar 1.165-µH coil.

Having convinced ourselves of the importance of working inductor *Q* into our simulation and having obtained a trustworthy *Q* value for our inductors, we face a more fundamental challenge: *SPICE*'s inductor (and capacitor) models afford *no* built-in means of specifying *Q*! Recalling that the *Q* of a coil can be equated to its reactance, *X*, divided by its (equivalent series) resistance, R_S, we realize further that R_S can be determined by dividing *X* by *Q*. For a *Q* of 238 in a 1.156-µH coil at 7.1 MHz, R_S works out to 0.22 Ω. It seems that all we must do to model realistic resonator-inductor *Q* in our simulation of Fig 6.18 is add a 0.22-Ω resistance in series with

each tuned-circuit coil.

In this case, yes — but a few sentences ago we said "*equivalent* series resistance" for good reason. The less-than-infinite *Q* of inductors at ham-band frequencies is a result not of ohmic resistance, but ac resistance due to *skin effect,* the tendency for ac at frequencies higher than a few hundred kilohertz to flow mainly at and near the *surface* of a conductor. Adding 0.22 Ω in series with each of our resonators therefore has the unwanted side-effect of making the coils unrealistically lossy at dc and low frequencies. In this simple HF-filter-modeling case, dc accuracy doesn't matter because our coils carry no dc. If, however, we wanted to model realistic *Q* in a coil that also carried dc to, say, a transistorized power amplifier, the voltage drop across any inductor R_S added merely as a *Q*-modeling workaround might well make our simulation unacceptably inaccurate at dc. One workaround to this new problem might be to shunt the *R*-equipped *L* with a high-value ideal *L* — a dc-bypass choke — but doing so would likely introduce yet other side-effects, including parasitic resonances and reduced realism in simulating the circuit in the time domain. A better approach would be to use an inductor model that implements skin-effect-based *Q*, such as that available in *Ansoft Designer SV2*.

Fig 6.20 shows the filter in *OrCAD Capture CIS*, ready for modeling in *SPICE*, with our *Q*-modeling resistances included as R3 and R4. Three more additional components, V1, R1 and R2, provide our test setup for measuring the circuit's insertion gain. V1 and R1 form the test-signal generator, with R1, which corresponds to the internal impedance of the generator, serving as the filter's input termination, and R2 serving as the filter's output termination and test load.

That the test-signal generator we build in treats as separate the signal source and its internal impedance gives us access to generator internals that we can't access in the real thing. Specifically, we can probe point B, the node at which the test generator's full output is available without modification by loss in the generator's internal impedance, R1 — a loss that will vary inversely with the impedance presented by the filter and its load. This means that we always have access to the reference level we need for calculating the insertion gain of whatever two-port circuit we connect between the test generator (V1R1) and its load (R2). Because R1 = R2, we can determine the insertion gain of whatever we connect between R1 and R2 with the equation

$$\text{Gain (dB)} = 20 \log\left(2\left[\frac{V_C}{V_B}\right]\right) \qquad (1)$$

where V_C is the voltage at point C of Fig 6.20 and V_B is the voltage at point B. Assuming that R1 = R2 — exactly the case in a real-world gain-measurement setup, in which the values of both will usually be 50 Ω — equation 1 returns a value of 0 when the test generator (V1R1) is directly connected to its load (R2).

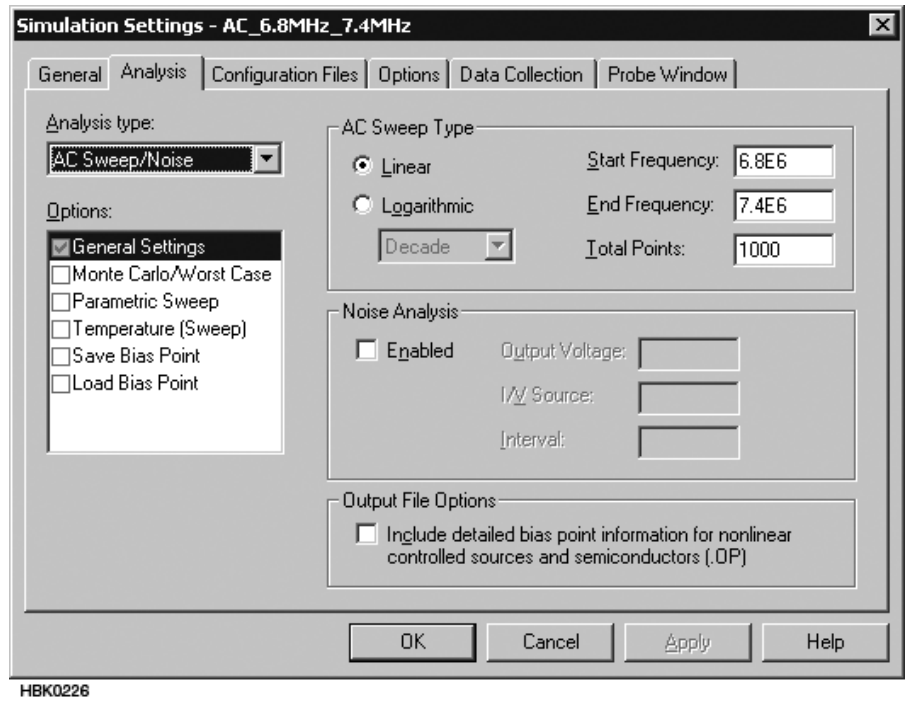


Fig 6.21 — Setup for a *SPICE* ac analysis at 1000 evenly spaced points from 6.8 MHz to 7.4 MHz.

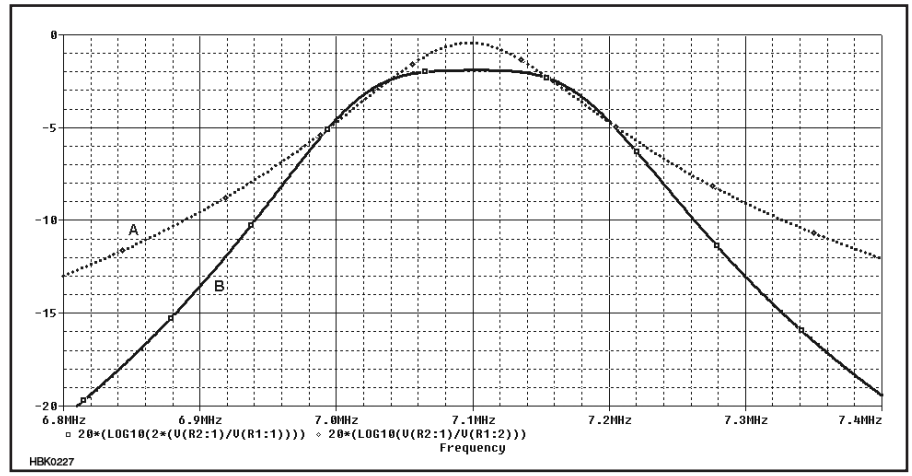


Fig 6.22 — Comparison of right and wrong approaches to reporting the filter gain in decibels. The dotted curve, A, merely plots the ratio, expressed in decibels, of the voltages at the filter output and input — points C and A — in Fig 6.20. The solid curve, B, plots values based on the ratio of the voltages at points C and B as determined by the trace definition 20*(LOG10(2*(V(R2:1)/V(R1:1)))) — the main text's Eq insertion gain calculation rendered in a form understandable by the reporter. Curve B depicts the circuit's insertion gain in decibels as the Fig 6.16 test setup would report it.

Fig 6.21 shows the analysis setup settings that tell *SPICE* to do *ac* analysis, in which *SPICE* first determines a circuit's dc characteristics before determining its ac characteristics in response to whatever ac sources it may contain. In this case, our ac source puts out 100E–6 volts — 100 µV, a level that corresponds to a strong on-the-air signal.

Fig 6.22 graphs the analysis results — as returned by equation 1 (curve B) and (in curve A) as returned by merely expressing in decibels the ratio of the voltage across the filter output to the voltage across the filter input. Curve B reports the circuit's insertion gain as a real-world gain-measurement setup would display it. As we shall also see, Curve

Symbols, Circuits and Hierarchies

We discover something quite interesting if we happen to view the *SPICE* netlist for Fig 6.20: Its linear transformer (TX) components (**Fig 6.A3**) are actually symbols that represent subcircuits — in this case, subcircuits that consist of two inductor (L) components coupled with a K_Linear coupling component. **Fig 6.A4** displays the evidence.

Because it includes subcircuits, Fig 6.20 is a *hierarchical* circuit — a circuit with multiple levels. Although support for hierarchies is usually limited in the demoware versions of CAD software that radio amateurs are likely to encounter, subcircuits are a powerful tool for creating and simulating large-scale circuits that contain other circuits, and circuits that use multiple copies of groups of components — individual identical transistor cells used many times throughout an RFIC, switching transistors with built-in bias resistors, bias and decoupling networks — throughout a design. Creating a schematic symbol that represents a subcircuit lets you in turn add the symbolized subcircuit as a new component for subsequent point-and-click placement in schematics like any other part — just as we do whenever we place a linear transformer component in the *OrCAD 16.0* schematic editor.

The power of symbolized subcircuits also operates at analysis time: However many identical instances of a given subcircuit may be present in an analysis, the simulation engine need evaluate its structure only once, recalculating its electrical behavior as it calculates the behavior of circuits that contain it.

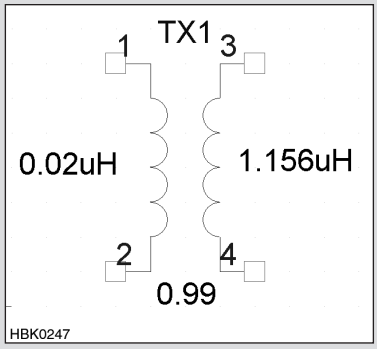


Fig 6.A3 — A linear transformer (TX) component in the *OrCAD 16.0* schematic editor. This part is actually a symbol for a subcircuit that contains three parts.

```
* source ZOI_40_METER_FILTER
V_V1          N07792 0 DC 0Vdc AC 100E-6ac
R_R1          N07792 N07820  50
R_R2          N04311 0  50
X_TX1     N07820 0 N01324 N00387 zoi_40m_filter_orcad16_TX1
X_TX2     N01606 N00537 N04311 0 zoi_40m_filter_orcad16_TX2
R_R3          0 N00387  0.22
R_R4          0 N00537  0.22
C_C1          N01324 N01606  9pF
C_C2          0 N01324  426pF
C_C3          0 N01606  426pF

.subckt zoi_40m_filter_orcad16_TX1 1 2 3 4
K_TX1          L1_TX1 L2_TX1 0.99
L1_TX1          1 2 0.02uH
L2_TX1          3 4 1.156uH
.ends zoi_40m_filter_orcad16_TX1

.subckt zoi_40m_filter_orcad16_TX2 1 2 3 4
K_TX2          L1_TX2 L2_TX2 0.99
L1_TX2          1 2 1.156uH
L2_TX2          3 4 0.02uH
.ends zoi_40m_filter_orcad16_TX2
```

Fig 6.A4 — Inspecting the *SPICE* netlist for Fig 6.20 reveals the reality behind the TX symbol: Placing a TX element actually places two inductors (L) and a coupling element (K).

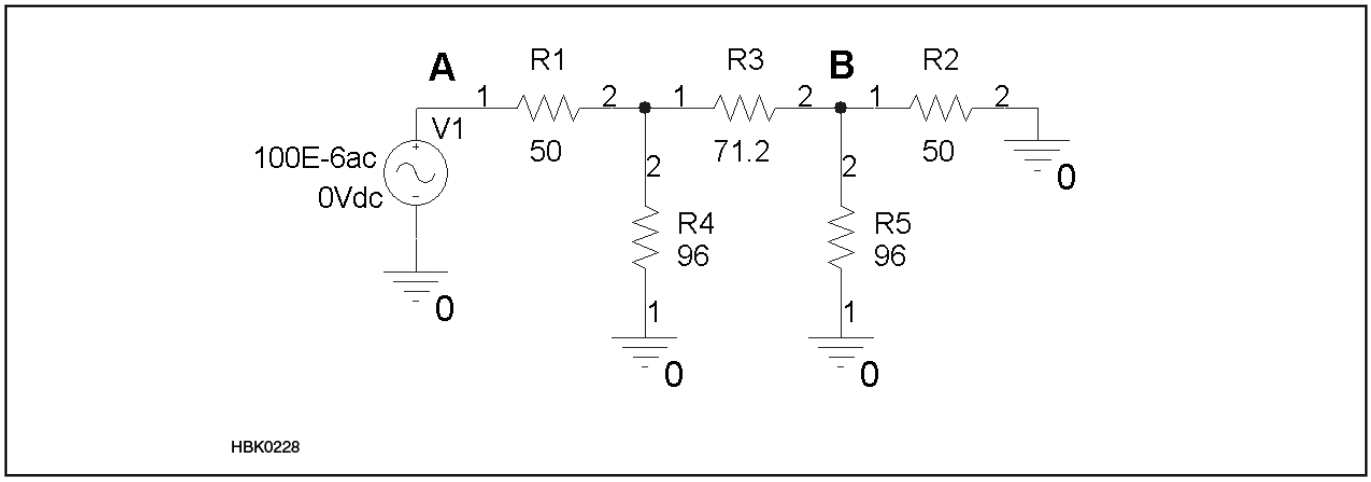


Fig 6.23 — To confirm what we think we now understand about modeling insertion gain with *SPICE*, we model a 10-dB attenuator using the same test generator and load arrangement as in Fig 6.18. Once again, we will calculate the gain of the circuit based on voltages probed between R1:1 (point A) and common (node 0) and R2:1 (point B) and common.

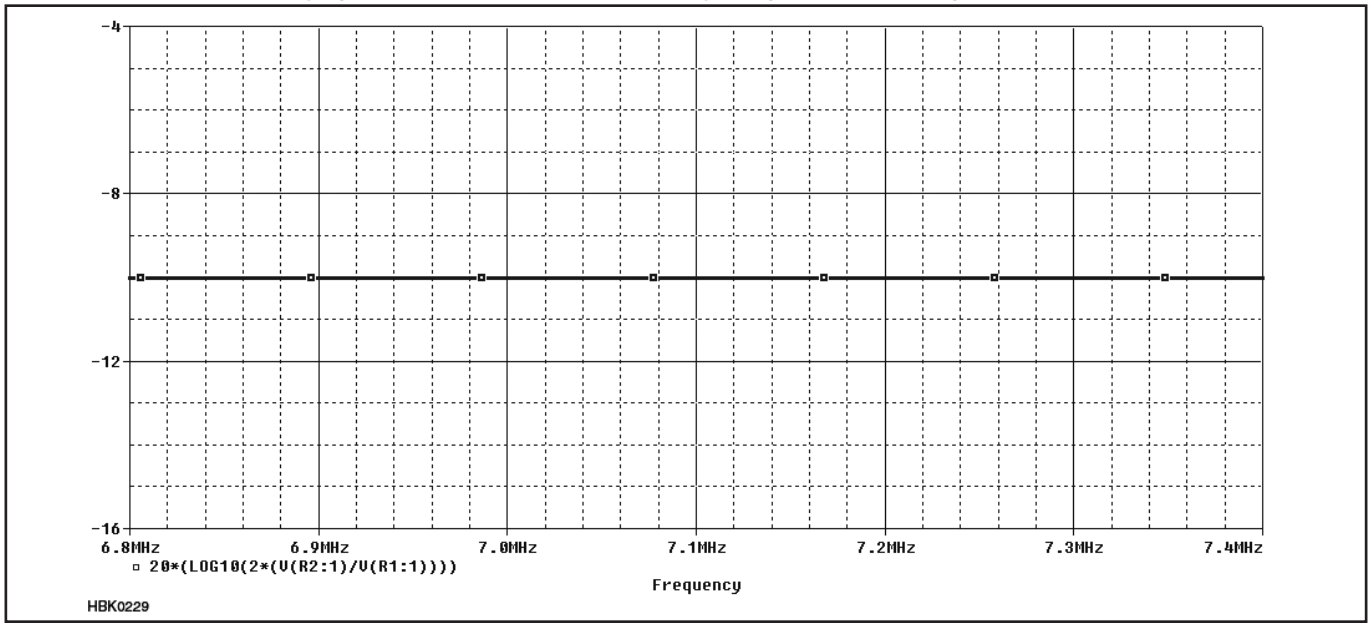


Fig 6.24 — As intended, our simulated 10-dB attenuator exhibits 10 dB of loss. In achieving this result, we have confirmed that our method of probing the circuit at R1:1 and R2:1 and common, in conjunction with reporting the circuit's gain as 20*(LOG10(2*(V(R2:1)/V(R1:1)))), validly returns the insertion gain we would measure in a real-world setup like that of Fig 6.16. In confirming this, we have also confirmed our finding of insertion gain through simulating Fig 6.20.

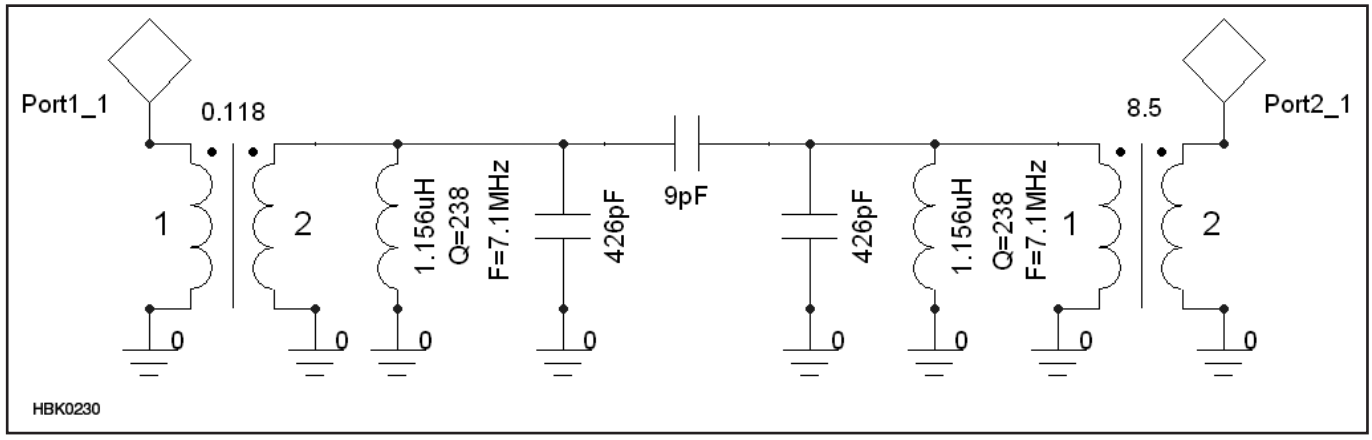


Fig 6.25 — Simulating the 7-MHz filter in *Ansoft Designer SV 2*, lets us use inductors that realistically model *Q*. Doing so then requires us to model the filter's input and output links with ideal transformers characterized in terms of turns ratio. The resonators in the real-world circuit are 17-turn toroidal coils with 2-turn links. The test generator and terminations necessary for *SPICE* analysis are absent for the reasons described in the text.

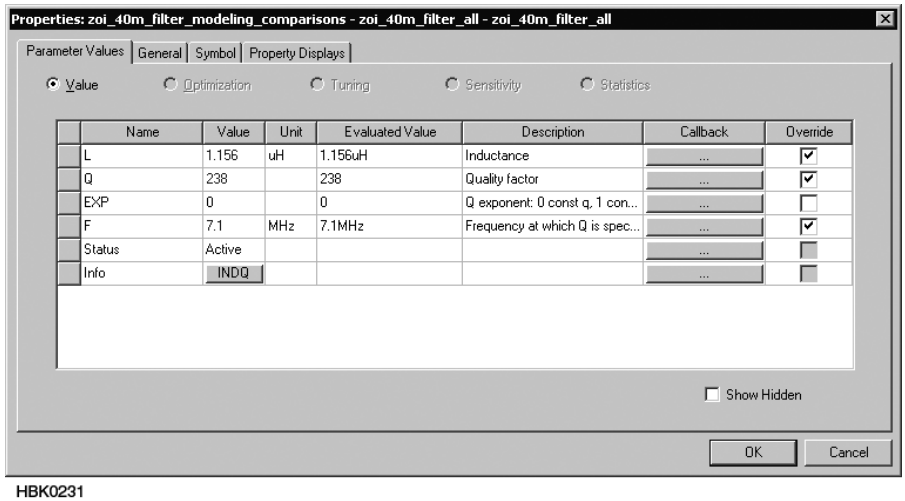


Fig 6.26 — Properties dialog for the *Ansoft Designer SV 2* INDQ component, showing settings for *Q*.

B reports the circuit's insertion gain as an RF-fluent circuit simulator reports it "out of the box" as one of a very important set of tools called *S parameters*. Before we meet such a simulator in the form of *Ansoft Designer SV 2*, we'll use *SPICE* to reality-check our virtual insertion-gain-measurement setup.

6.2.4 Example 4: Checking Reality with a 10-dB Attenuator

To confirm in a general way that we're on the right track with the insertion-gain probing and reporting regime we arrived at for our simulation of Fig 6.20, we'll simulate a simple circuit of known gain and frequency response. The circuit, **Fig 6.23**, consists of little more than a pi-network attenuator with its resistances configured to exhibit 10 dB of loss in a 50-Ω system. As in Fig 6.20, we add 50-Ω source and load resistors, and a voltage source as a test-signal generator. **Fig 6.24** shows the attenuator's gain: –10 dB. We have indeed built and calibrated our virtual insertion-gain test setup to duplicate the behavior of the real thing, confirming as well the validity of the results we obtained for the insertion gain of the filter in Fig 6.20.

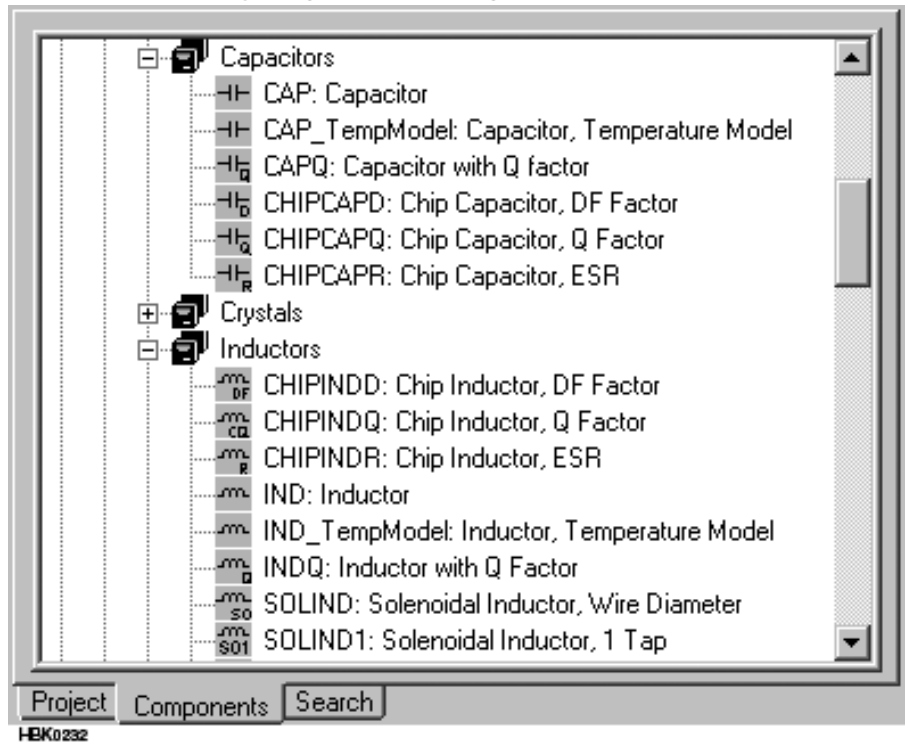


Fig 6.27 — The *Ansoft Designer SV 2* component chooser includes inductor and capacitor models that realistically model lossiness at RF.

6.2.5 Example 5: Simulating the 40-Meter Filter in an RF-Fluent Simulator

Realistically simulating Fig 6.18 in *SPICE* required the addition of resistors to set its resonators' unloaded *Q* to 238. Such workarounds are not necessary when we use a circuit simulator specialized to speak RF, as we'll illustrate by simulating Fig 6.18 in *Ansoft Designer SV 2*, the student version of *Ansoft Designer*, a linear, nonlinear, electromagnetic and system EDA CAD suite engineered for modern RF, MMIC and RFIC design. **Fig 6.25** shows the filter in the *Ansoft Designer SV 2* schematic editor. Now, instead of using ideal transformer windings, we can model the filter's resonators as inductors with realistic *Q* based on skin effect (**Fig 6.26**). But using stand-alone inductors rather than transformers presents a new challenge: How will we model the filter's input and output coupling links?

In addition to including many realistically non-ideal components (**Fig 6.27**), the *Ansoft Designer SV 2* component library includes ideal transformers — transformers with a coupling coefficient (*K*) of 1 — that can be characterized by turns ratio. The resonators in the real-world circuit consist of 17-turn coils with 2-turn links, so we can simulate the links by using transformers with turns ratios of 2:17 (0.118) and 17:2 (8.5) at the filter input and output, respectively.

Absent from our schematic are a signal

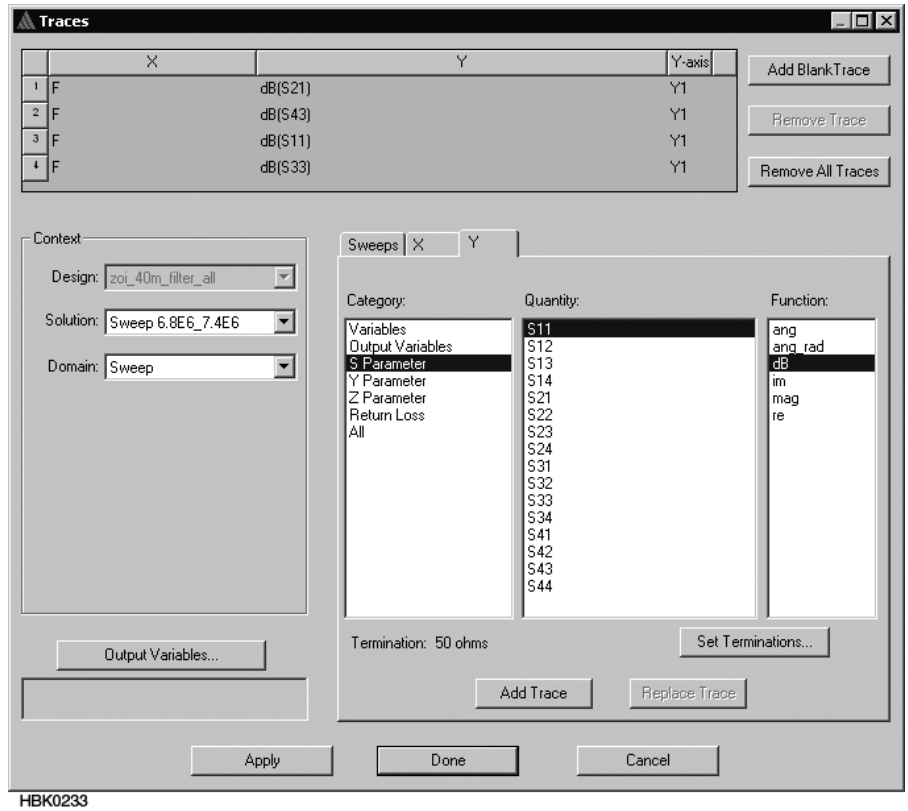


Fig 6.28 — The reporter of the *Ansoft Designer* RF-fluent linear simulator lets us evaluate circuit behavior not in terms of voltages and currents but rather in terms of *network parameters* — scattering parameters (*S*), admittance parameters (*Y*), and impedance parameters (*Z*) — and return loss. These reporter settings, used to produce the comparative graph in Fig 6.29, show responses for a four-port circuit rather than a two-port because the full analysis run included as a separate circuit copy the *Q*-from-added-resistors circuit from Fig 6.20.

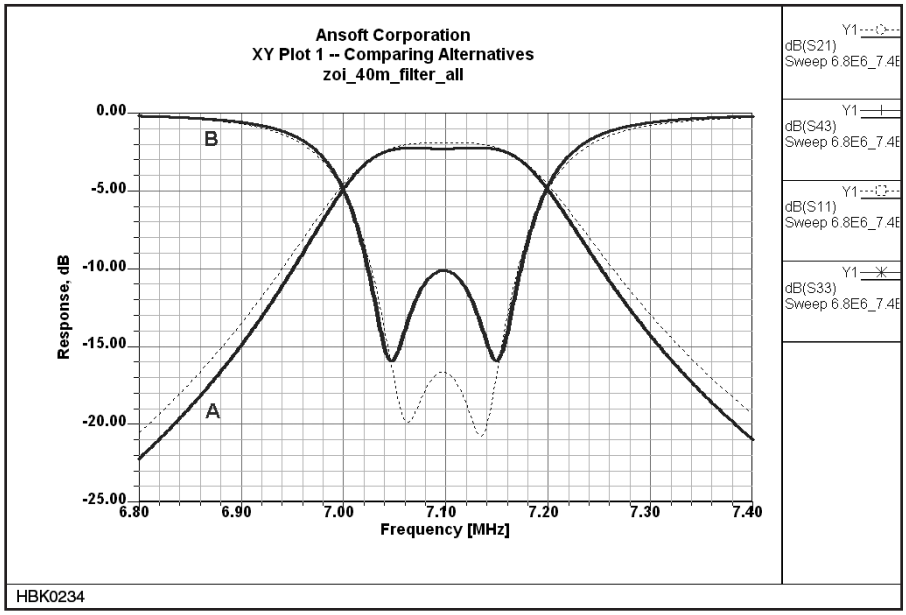


Fig 6.29 — *Ansoft Designer SV 2* report for the realistic-*L* filter showing (A) insertion gain and (B) return loss, both in decibels. (Return loss, the magnitude of S_{11}, is a positive value in dB although in this plot the Y-axis is calibrated in negative dB.) The dotted lines show the same responses for the *Q*-from-series-*R* circuit of Fig 6.20. Graphing the filter's return loss provides information about how closely the impedance of the terminated filter matches the impedance terminating the filter's input. The higher the return loss, the more closely the impedance presented by the input of the terminated filter approaches the impedance of its input termination (in this case, 50 Ω). Return Loss associated with a passive circuit, such as a filter, is always positive.

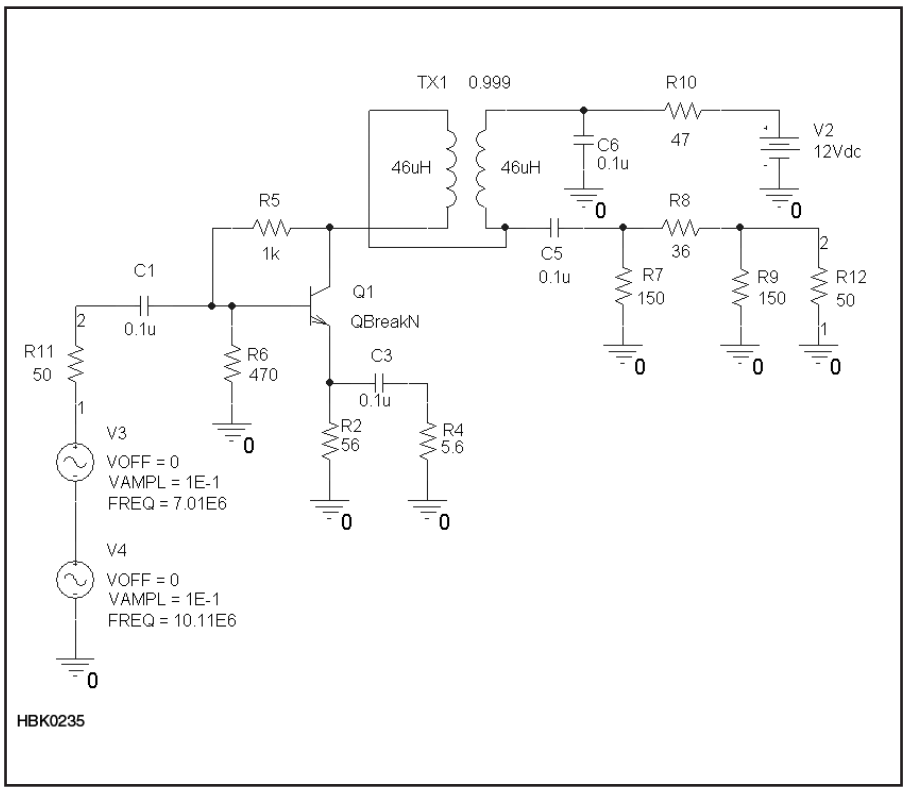


Fig 6.30 — The Progressive Communications Receiver post-mixer amplifier configured for two-tone IMD analysis in *OrCAD 16.0 SPICE*. The level of both test signals is 0.1 V — very strong, "other hams in the immediate neighborhood"-class signals. The transistor is modeled with *SPICE* data for the California Eastern Laboratories NE46134. Using a value of 47 Ω for R10 sets its collector current to 40 mA; 100 Ω, 31 mA.

source and hardwired input and output terminations. Sources are unnecessary for linear simulation using *Ansoft Designer SV 2*. Terminations, including the 50-Ω default set within the circuit's port elements, are applied at reporting time, *after* analysis has concluded.

Fig 6.28 shows the responses available for our simulation in the *Ansoft Designer SV 2* reporter. Four-port responses are shown because the full circuit analyzed actually consists of the Fig 6.18 circuit (ports 1 and 2) and Fig 6.25 circuit (port 3 and 4) place side by side. **Fig 6.29** compares the two circuit's responses, with the responses of the Fig 6.20 circuit shown as dotted lines.

Unsurprisingly, the two modeling approaches produce slightly different responses, both of which meet the goals of the filter designer. Which one is "better" can therefore be considered to be academic; discerning the difference between two real-world filters that exhibit exactly these responses would be a measurement challenge. On the air we could tell them apart only with difficulty.

Our true purpose in reevaluating the Fig 6.20 filter in *Ansoft Designer SV 2* is to illustrate the difference between RF-fluent CAD and the general-purpose *SPICE*-based CAD tools to which CAD-minded hams tend to default for circuit design. In graphing its responses in *Ansoft Designer SV2* we stand at the threshold of a class of muscular CAD tools that bring great power to hobbyists interested in RF CAD. Rather than plotting just gain, Fig 6.29 actually plots gain and *return loss*, a value of practical importance in many RF-design applications. (The higher the return loss, the more input-signal energy the filter accepts, and the less input-signal energy is reflected back to its source. For a filter of this design, we want and expect return loss to be high in its passband and low in its stopband.) Rather than merely evaluating the circuit's voltage gain, we report its response in terms of standardized *network parameters* — in this case the *S* (scattering) parameters S_{21} and S_{11} (for Fig 6.20, and S_{43} and S_{33} for Fig 6.25 in our two-two-ports-at-once simulation) expressed in decibels. The sidebar "*S*-Parameter Basics" explains more about why *S* parameters are important and how RF engineers use them.

The ability to handle network parameters is a profound differentiator between *SPICE* and more RF-fluent simulators. Although it's possible to derive *S* parameter from *SPICE* analysis through post-processing and/or the use of special subcircuits that stimulate an *n*-port for the purpose of more network-parameter-literate reporting, fluency in small-signal network parameters is not among *SPICE*'s simulation and reporting strengths. As we'll see in the next section, *SPICE* can only limitedly simulate intermodulation distortion, a signal-handling flaw in which

S-Parameter Basics

The tool called *S parameters* provides a standardized way of characterizing how a device behaves in response to signal energy applied to its ports — its signal inputs and outputs — usually with all of its ports terminated in identical, standard impedances (commonly, 50 Ω, resistive). A transistor, for instance, is a two-port device. By convention, the ports are labeled with numbers, Port 1 being the input and Port 2 being the output.

Signal energy applied to one port of a two-port device comes out two places: at the same port (because the device reflects some of the energy back to the generator) and at the other port. How much signal comes out relative to the applied signal tells us the device's gain (which can be negative — that is, a loss); how much signal reflects back out tells us something about the impedance match between that port's impedance and our signal generator. Determining the phase of the output or reflection signals relative to the phase of the applied signals tells us even more about the device or subcircuit under test.

Fig 6.A5 shows this idea graphically. An *S* parameter is a voltage ratio (commonly, but not always, expressed in decibels) annotated with two subscript numbers that indicate the ports involved. For instance, a device's forward transmission gain, S_{21}, ("S sub two one"), is the ratio of the voltage at Port 2 to the voltage applied to Port 1 — a value that must be expressed as a vector to convey the two signals' relative phase. To discuss *S* parameters more readily and to communicate them in tabular form, we can split each of the four basic parameters into separate components — real and imaginary parts, or, especially useful for device modeling with *S* parameters, magnitude and phase: MS_{11} (magnitude of input reflection) and PS_{11} (phase of input reflection); MS_{21} (magnitude of forward transmission gain) and PS_{21} (phase of forward transmission gain); MS_{12} (magnitude of reverse transmission gain) and PS_{12} (phase of reverse transmission gain); and MS_{22} (magnitude of output reflection) and PS_{22} (phase of output reflection).

From the standpoint of circuit evaluation and modeling, the great power of *S* parameters is that they can convey usefully realistic information about the ac behavior of a device or subcircuit through relatively few standardized numerical values. For instance, if we know the *S* parameters of a given transistor operating under known conditions of power- and bias-supply voltage and current, we have a very useful picture of how it looks to the outside world — a picture we can paste directly into an *S*-parameter-fluent linear circuit simulator. Of great value to the modeling efforts of professionals and radio amateurs alike is the fact that RF-device manufacturers commonly make their products' *S* parameters freely available in data formats widely used by industry-standard simulators. **Table 6.3** shows *S*-parameter data for the NE46134 transistor in an *S*-parameter format that most RF-fluent simulators can handle.

To use device *S*-parameter data in a simulation, we place a generic *black-box* component in our circuit and tell the simulator to read the *S*-parameter data when it calculates the behavior of the black box. Using black-box *n*-port parts in place of detailed mathematical device models extends the power of linear simulation for use in modeling the network responses, noise, and stability characteristics of, and developing matching networks for, devices that might otherwise not be modelable without recourse to nonlinear simulation and detailed mathematic device models configured with realistic mathematical parameter values. A significant limitation of black-box modeling is that an *S*-parameter dataset is static, and most accurately reflects real-world device behavior only under the conditions of voltage and current used in generating it.

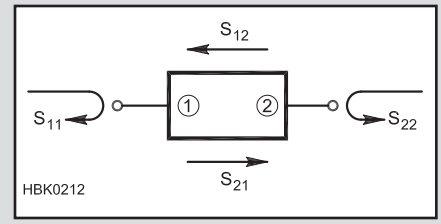


Fig 6.A5 — *S* parameters corresponding to input reflection, forward transmission gain, reverse transmission gain and output reflection can quite closely characterize a small-signal linear device — in this case, a two-port device. Expressing a two-port's forward transmission gain, S_{21}, in decibels returns the same number we would report for its insertion gain.

Table 6.3
Two-Port *S*-Parameter Data Equivalent to an NE46134 Transistor (Operating at V_{CE}=12.5 and I_C=50 mA) from the California Eastern Laboratories File *NE46134G.S2P*

! FILENAME: NE46134G.S2P VERSION: 8.0
! NEC PART NUMBER: NE46134 DATE: 07/94
! BIAS CONDITIONS: VCE=12.5V, IC= 50mA
GHZ S MA R 50

0.050	0.432	–91.9	34.140	125.6	0.024	63.0	0.586	–56.7
0.100	0.372	–129.4	19.834	106.3	0.036	63.5	0.362	–78.4
0.200	0.348	–161.2	10.444	92.4	0.058	65.4	0.223	–97.1
0.300	0.344	–175.7	7.086	85.1	0.081	67.7	0.183	–106.7
0.400	0.343	173.5	5.332	79.3	0.104	68.1	0.170	–112.8
0.500	0.341	164.1	4.282	74.5	0.127	67.5	0.168	–116.6
0.600	0.343	157.1	3.610	70.1	0.150	66.3	0.172	–119.3
0.700	0.344	149.7	3.114	65.7	0.173	64.7	0.179	–121.0
0.800	0.349	143.8	2.755	61.8	0.196	63.1	0.188	–122.4
0.900	0.347	137.4	2.471	57.9	0.218	61.2	0.198	–123.4
1.000	0.352	132.4	2.254	54.4	0.239	59.2	0.210	–124.2
1.100	0.351	126.5	2.075	50.7	0.260	57.2	0.223	–125.0
1.200	0.353	121.2	1.927	47.1	0.280	55.0	0.236	–125.8
1.300	0.351	116.3	1.803	43.7	0.300	52.9	0.251	–126.5
1.400	0.350	111.6	1.710	40.4	0.320	50.7	0.267	–127.4
1.500	0.346	106.6	1.607	37.2	0.337	48.4	0.283	–128.0
1.600	0.343	102.0	1.523	34.0	0.354	46.2	0.300	–128.9
1.700	0.339	97.0	1.447	31.4	0.369	44.1	0.317	–130.2
1.800	0.339	93.1	1.388	29.0	0.385	42.4	0.330	–131.5
1.900	0.338	88.4	1.340	26.2	0.402	40.4	0.343	–132.1
2.000	0.336	83.3	1.295	23.5	0.418	38.2	0.357	–132.7
2.100	0.331	78.4	1.252	20.7	0.432	36.1	0.372	–133.4
2.200	0.325	73.3	1.206	18.1	0.445	34.0	0.386	–134.0
2.300	0.320	68.4	1.172	16.1	0.458	32.0	0.399	–134.7
2.400	0.318	63.0	1.140	13.6	0.471	29.9	0.411	–135.3
2.500	0.314	57.3	1.109	11.7	0.482	28.0	0.423	–135.9

Each of these lines conveys frequency and the *S* parameter magnitude and phase values M*S*11, P*S*11, M*S*21, P*S*21, M*S*12, P*S*12, M*S*22 and P*S*22. This S*n*P format (where *n* is the number of ports in the device characterized) was originally used by the EESOF Touchstone circuit simulator, and is one of several *S*-parameter data formats now widely used in the RF engineering community. An S*n*P file may also contain noise-modeling data.

multiple signals present at a circuit's input interact in circuit components — in active devices, especially — to produce output spectral components not present at a circuit's input.

6.2.6 Example 6: *SPICE* and Intermodulation Modeling

As an example of the limitations of *SPICE* for critical RF-fluent analyses, we will attempt to simulate intermodulation distortion (IMD) with *OrCAD 16.0*. IMD is of great importance to RF engineers because the span of signal levels — the *dynamic range* — we expect modern communications circuitry to handle is so wide that communication possible by means of weak legitimate signals can easily be made impossible by the weak false signals produced by IMD.

Fig 6.30 shows the circuit: the widely used and well-characterized post-mixer feedback amplifier introduced by Hayward and Lawson in their 1981 Progressive Communications Receiver. This implementation uses the *SPICE* parameters shown in Fig 6.6 for the California Eastern Laboratories NE46134 transistor and includes two signal sources in series for the purpose of generating IMD products.

Simulating the circuit's gain vs frequency response (**Fig 6.31**) returns realistic numbers; turning on nodal voltage and current display in the schematic editor (**Fig 6.32**) confirms that the device's bias point (for a BJT, collector current) is realistic and that we are not exceeding the device's collector-to-emitter voltage rating (15).

Fig 6.33 shows the FFTed output spectrum on a linear scale. Spectral components other than those attributable to the two test signals are absent. **Fig 6.34** displays the same data on a logarithmic voltage scale, with the X-axis zoomed in on the 0- to 40-MHz span. If we know what to look for, we can see responses at frequencies attributable to harmonics of the input tones; to second-order IMD (the frequency of each tone plus and minus the frequency of the other); and third-order IMD (twice the frequency of each tone plus and minus the frequency of the other), but the graph is complicated by higher-order products — arguably good from the standpoint of realism — a high noise floor, and a rising response toward 0 Hz.

As a comparison of simulation techniques, **Fig 6.35** shows the output spectrum for the same circuit as predicted by the harmonic-balance nonlinear simulator in *Serenade SV 8.5*, the now-discontinued predecessor of *Ansoft Serenade SV 2*. Harmonic-balance simulation treats the linear and nonlinear portions of a circuit as separately solvable subsystems, analyzing the linear portion in the frequency domain and the nonlinear portion in the (steady-state) time domain. Harmonic-balance analysis offers significant speed and dynamic-range advantages over *SPICE* for circuits that include transmission lines, long time constants relative to operating frequency, and many reactive components (such as RF circuits and systems commonly contain). Alas, at this writing, harmonic-balance analysis is unavailable to hobbyists and students in free demoware form as discussed in the sidebar, "RF-fluent CAD: What We're Missing."

6.2.7 Circuit CAD in the Radio Amateur's Toolbox

This chapter emphasizes the use of *SPICE* for circuit simulation because radio amateurs interested in circuit CAD will likely first experience it with a *SPICE*-based simulator and keep using *SPICE*. At this writing, *SPICE* is the only *nonlinear* simulator freely available to hobbyists. Radio amateurs seeking to enhance their knowledge of RF design techniques through circuit simulation will want to use an RF-fluent simulator instead of or in addition to *SPICE*, and *Ansoft Designer SV 2* can serve as a linear-simulation workhorse for this purpose. (Here we must differentiate between trialware and demoware: Although RF-fluent nonlinear simulation may be available in the feature-limited *trialware* versions of some EDA products, hobbyists need CAD capabilities that won't stop working in 30 to 90 days. We hasten to add that radio amateurs do not expect that such capabilities need be free, just affordable; the full versions of the RF-fluent simulators known to the author sell for thousands to tens of thousands of dollars.)

An expensive full-version simulator can mislead as, or more, easily than its freeware version in the absence of designer know-how teamed with *carefully and fully characterized performance data describing the actual behavior of simulatable real-world circuits*. Without tempering by experimental experience and constant comparison with real-world performance data, results obtained through CAD can lose their necessary real-world anchoring. Well-applied, however, computerized circuit simulation can greatly accelerate one's acquisition of the intuition and RF "street smarts" that make RF design an art *and* science.

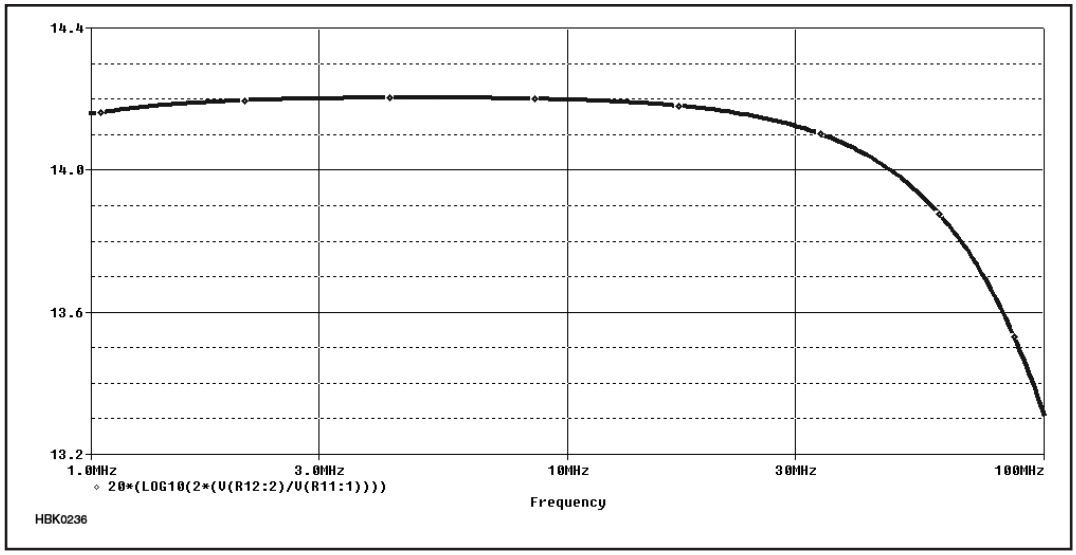


Fig 6.31 —Simulating the gain of the post-mixer amplifier with *OrCAD 16.0 SPICE* produces a realistic gain vs frequency response.

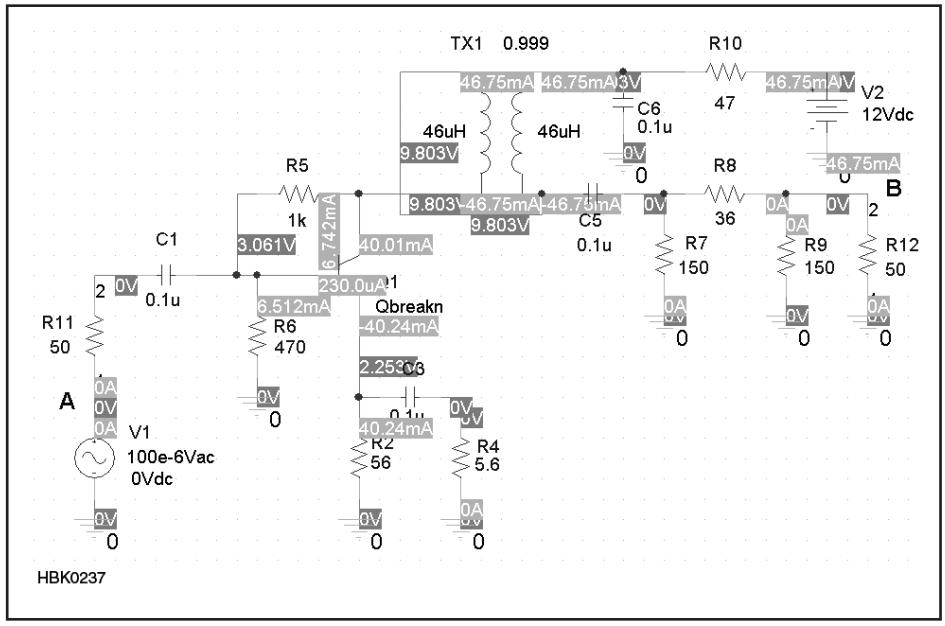


Fig 6.32 — After running at least one analysis, we can turn on voltage and current display in the schematic editor to reality-check the device bias point (for a BJT, collector current). Comparing the voltages displayed for the simulated BJT's collector, base, and emitter lets us ensure that we're not exceeding the published ratings of the real-world device.

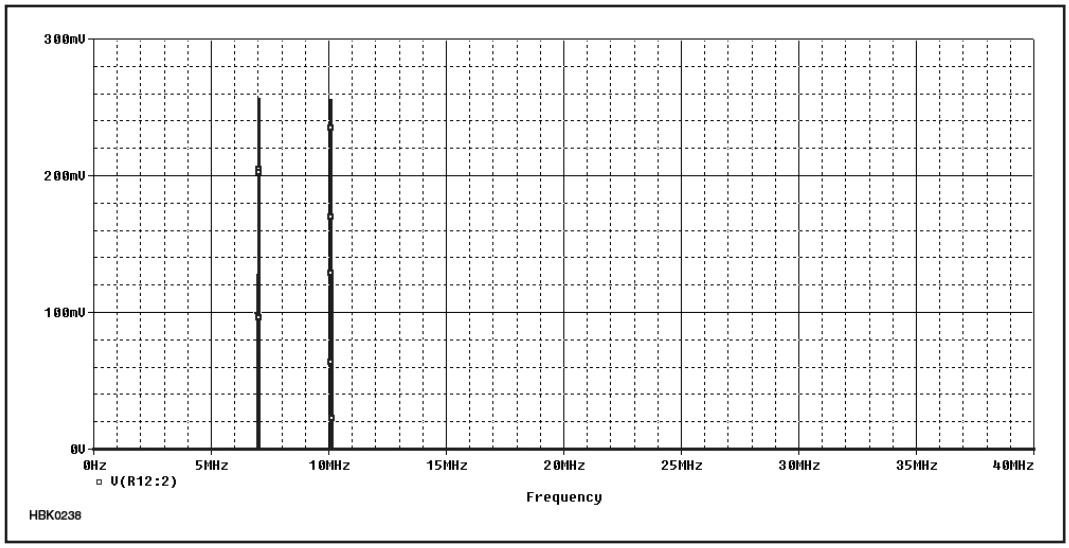


Fig 6.33 — Output spectrum of the feedback amplifier on a linear voltage scale. To generate this graph, we analyzed the Fig 6.30 circuit for 100 µs with a maximum time step of 10e–10 s, and used the *OrCAD 16.0* reporter's FFT function. Components attributable to IMD are superficially absent because of the linear *y*-axis scale.

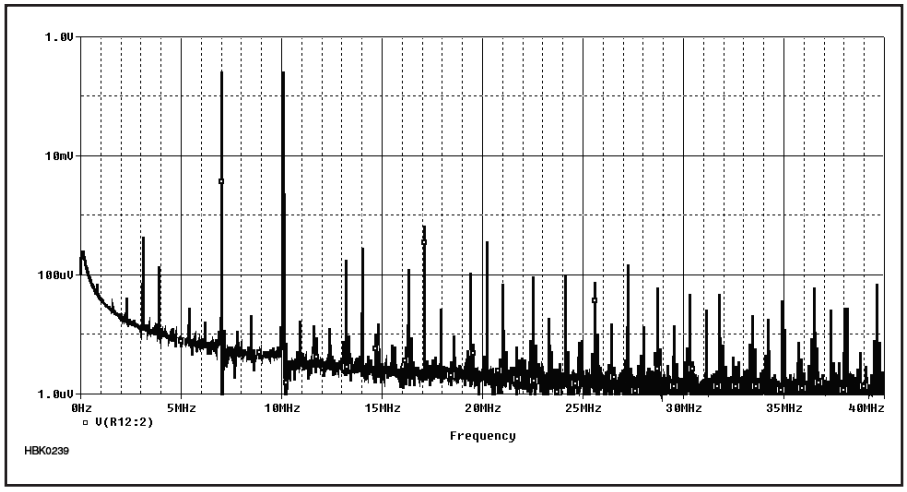


Fig 6.34 — Switching to a logarithmic voltage scale and zooming in on the 0- to 40-MHz range lets us discern spectral components at frequencies corresponding to harmonics and second- and third-order IMD products, but significant artifacts — a relatively high noise floor and a rising response toward time zero — are apparent.

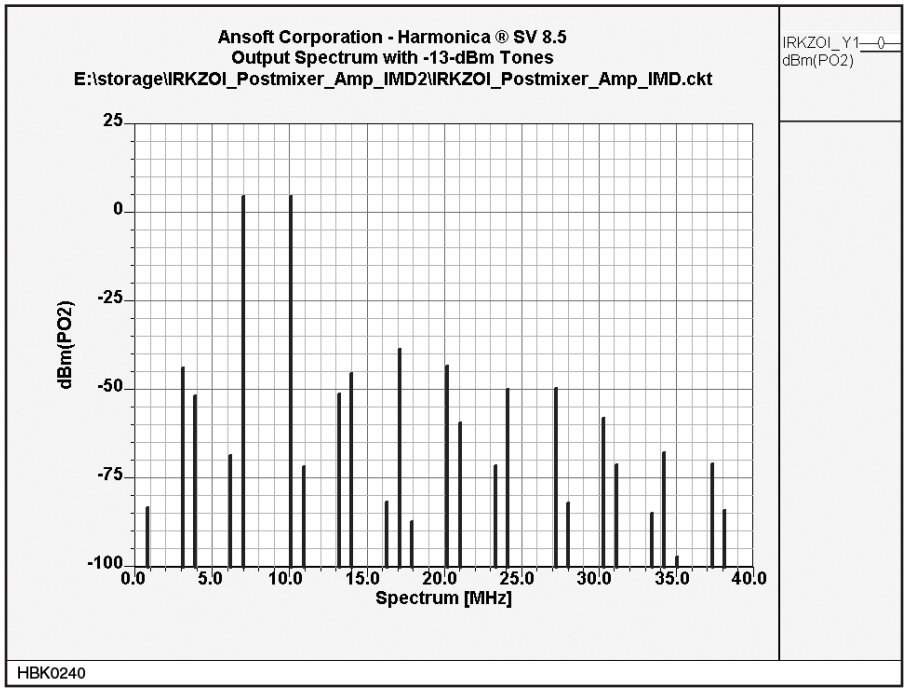


Fig 6.35 — Output power spectrum of the same circuit as predicted by the harmonic-balance nonlinear simulator in the no-longer-available Ansoft *Serenade SV 8.5.* This spectrum is simpler compared to Fig 6.34 because IM calculations were limited to fifth-order products in the student version of *Serenade 8.5*. Most striking is the large simulation dynamic range achieved without the appearance of math-noise artifacts, and the ability to report output power in decibels relative to a milliwatt (dBm). One more thing: The *SPICE* analysis for Figs 6.33 and 6.34 took over five minutes; the *Serenade SV 8.5* run that produced this graph, *under two seconds*.

Simulating Keying with a *SPICE* Behavioral Model

This chapter begins by illustrating that electronic circuits "just do math," responding to and processing electronic signals — also describable mathematically — by, in effect, performing mathematical operations on them. If, for instance, what you require in a simulation is, say, the mathematically idealized behavior of a comparator or 555 timer IC rather than detailed, step-by-time-step nodal analysis of the behaviors of its internal circuitry, you can use (after building it yourself, if necessary) a *behavioral model* of the part instead. Behavioral modeling can be used to simulate analog and digital parts.

The *OrCAD 16.0 Capture CIS* demoware component library includes quite a few behavioral models — mostly 7400-series TTL ICs, but also several analog ICs, including LF411, LM324 and 741 op amps, an LM111 comparator, and a 555 timer. As an example of what behavior models allow us to do, **Fig 6.A6** presents an analysis designed by John Seboldt, KØJD, to simulate on-off keying of a broadband feedback amplifier (Q1, QBreakN, an *OrCAD Capture CIS* "breakout" device characterized with data for an Infineon BFG135 transistor) by means of a series dc-supply switch (Q2, a 2N2907A) keyed via a 2N3904 switch by a 555 timer configured as a "ditter." So realistic is this simulation (**Fig 6.A7**) that it even models "short first dit" and backwave!

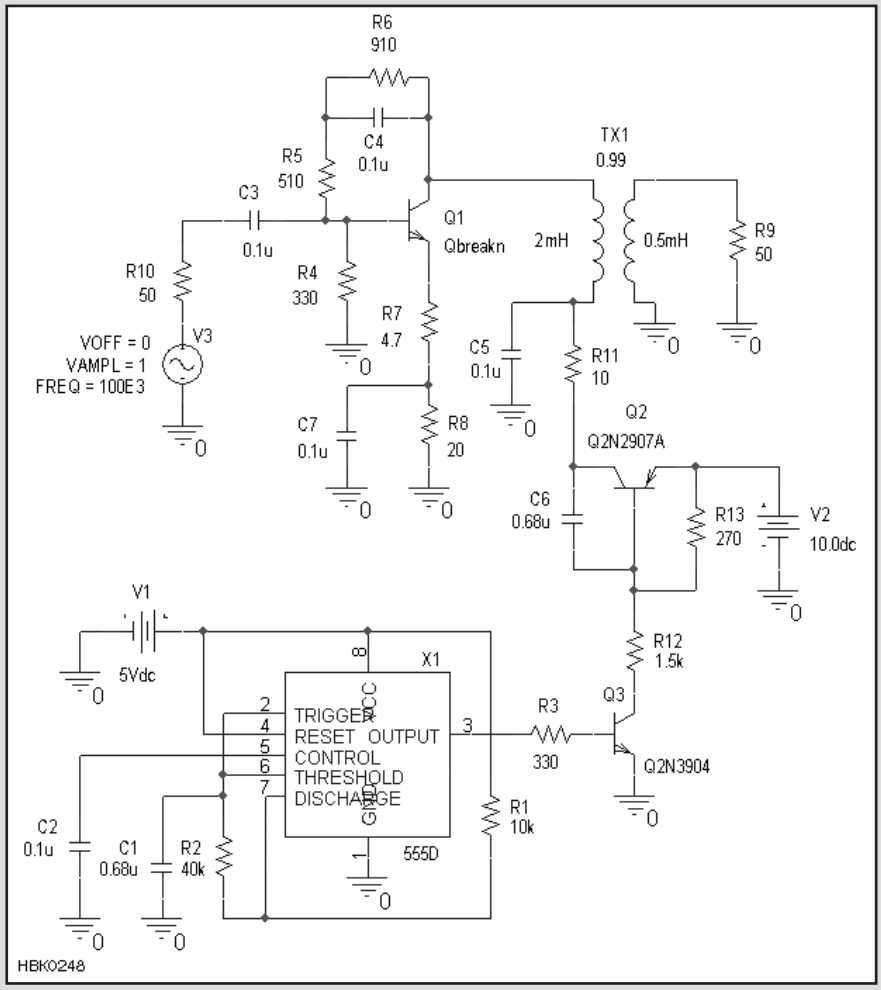


Fig 6.A6 — To simulate the keying waveform of a QRP transmitter stage in *SPICE*, John Seboldt, KØJD, built a ditter — a circuit that, powered up, sends a continuous string of Morse code dots — using a SPICE behavioral model for the 555 timer IC. The timer output drives 2N3904 and 2N2907A switches to interrupt the collector power supply of amplifier transistor QBreakN, an *OrCAD Capture CIS* "breakout" device characterized by NXP Semiconductor data for a Philips BFG135 (BFR194 chip) broadband transistor. Although a low-frequency (100-kHz) source is used to reduce the analysis time and datafile size (and we have increased the inductances in transformer TX1 appropriately relative to their values at HF), coupling and bypass capacitances are kept at their HF-appropriate values to keep RC-time-constant-related settling times consistent with the behavior of the real-world circuit at HF. To make the keyed signal's rise and fall times slower and more easily discerned in a graph, we have also increased the value of shaping capacitor C6 from Seboldt's value of 0.47 µF.

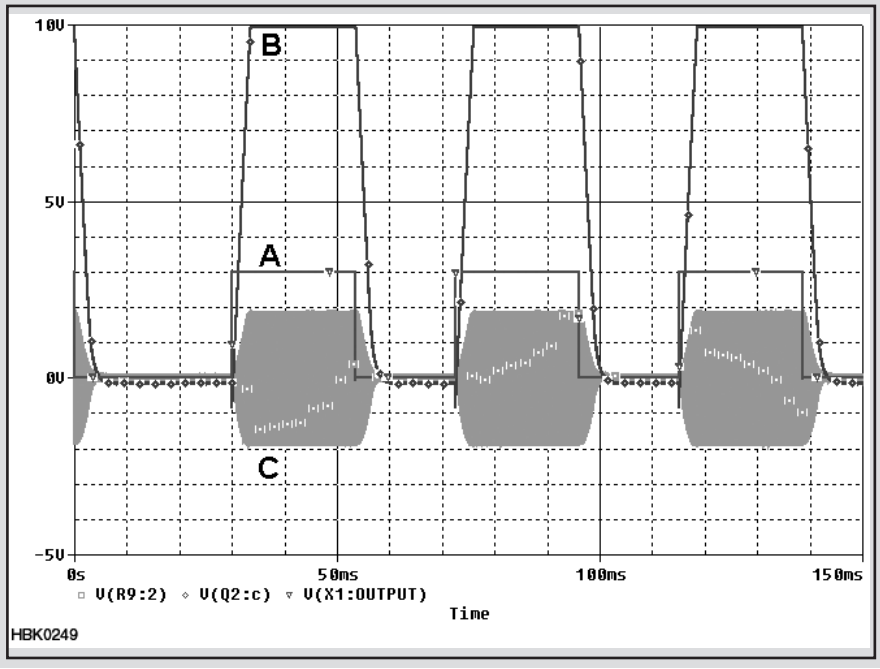


Fig 6.A7 — Results of the keying simulation, showing (A), the voltage at the 555 OUTPUT pin; (B), the keyed collector supply applied to the amplifier; and (C), the keyed amplifier output; time steps are 0.2 µs. So realistic is this simulation that it also reflects the "short first dit" problem shared by some real-world transmitters (in this case, it's an artifact of powering up the keyer and amplifier together as the analysis begins) and significant *backwave* (discernible output during key-up periods). In a real-world transmitter, we would reduce backwave to an acceptable level by keying multiple amplifier stages, a mixer, or — at the risk of degrading signal quality in additional ways — an oscillator.

RF-Fluent CAD: What We're Missing

SPICE-based simulators can do wonders in many classes of circuit simulation. For RF use, however, *SPICE* has significant drawbacks. For starters, *SPICE* is not RF-fluent in that it does not realistically model *physical* distributed circuit elements — microstrip, stripline, and other distributed circuit elements based on transmission lines. It cannot directly speak network parameters (*S*, *Y*, *Z* and more), stability factor and group delay. It cannot simulate component *Q* attributable to skin effect. It cannot simulate noise in nonlinear circuits, including oscillator phase noise. It cannot realistically simulate intermodulation and distortion in high-dynamic-range circuits intended to operate linearly. This also means that it cannot simulate RF mixing and intermodulation with critical accuracy.

The feature-unlimited version of *Ansoft Designer* and competing RF-fluent simulation products can do these things and more excellently — but many of these features, especially those related to nonlinear simulation, are unavailable in the student/demoware versions of these packages where such versions exist.

For awhile, from 2000 to 2005, the free demoware precursor of *Ansoft Designer SV 2*, Ansoft *Serenade SV 8.5*, brought limited use of nonlinear-simulation tools to students and experimenters. With *Serenade SV 8.5*, you could simulate mixers, and you could simulate amplifier IMD — IMD from two tones only, to be sure — to a maximum of four nonlinear ports, meaning that *Serenade SV 8.5* could simulate mixers with up to four diodes or up to two transistors (or one transistor and two diodes — you see the strategy of the limitation). See **Figs 6.A8**, **6.A9** and **6.A10**. You could simulate the conversion gain and noise figure of a mixer. Optimization was enabled. Realistic nonlinear libraries were included for several Siemens — now Infineon — parts. You could accurately predict whether or not a circuit you hoped would oscillate would *actually* oscillate, and assuming that it would, you could accurately predict its output power and frequency.

The harmonic-balance techniques used by Ansoft's nonlinear solver — and by the nonlinear solvers at the core of competing RF-fluent CAD products, such as Agilent *Advanced Design System* (ADS) — allowed you to simulate crystal oscillators as rapidly as you can simulate lower-*Q* oscillators based on LC circuits. (In *SPICE*, getting a crystal oscillator to start may be impossible without presetting current and/or voltages in key components to nonzero values, even if you try kick-starting it with a pulse as we do in this chapter's JFET VFO simulation.)

Students and CAD-minded radio amateurs alike miss the features made available in *Serenade SV 8.5* and hope that they will one day return in some form to the world of free demoware CAD software. In the meantime, if you're serious about pushing into RF CAD beyond what *SPICE* can do and someone you know has no use for their copy of *Serenade SV 8.5*, see if you talk them out of it!

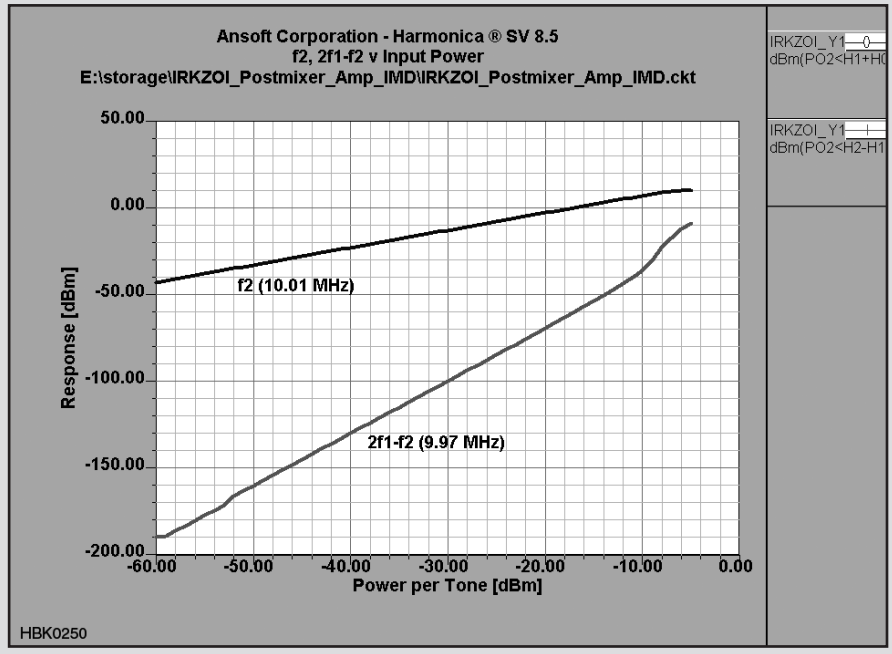


Fig 6.A8 — At this writing, the two-tone nonlinear simulation capabilities necessary to model third-order IMD were unavailable in free-demoware form. This simulation was done by Ansoft *Serenade Designer SV 8.5*, available from 2000 to 2005.

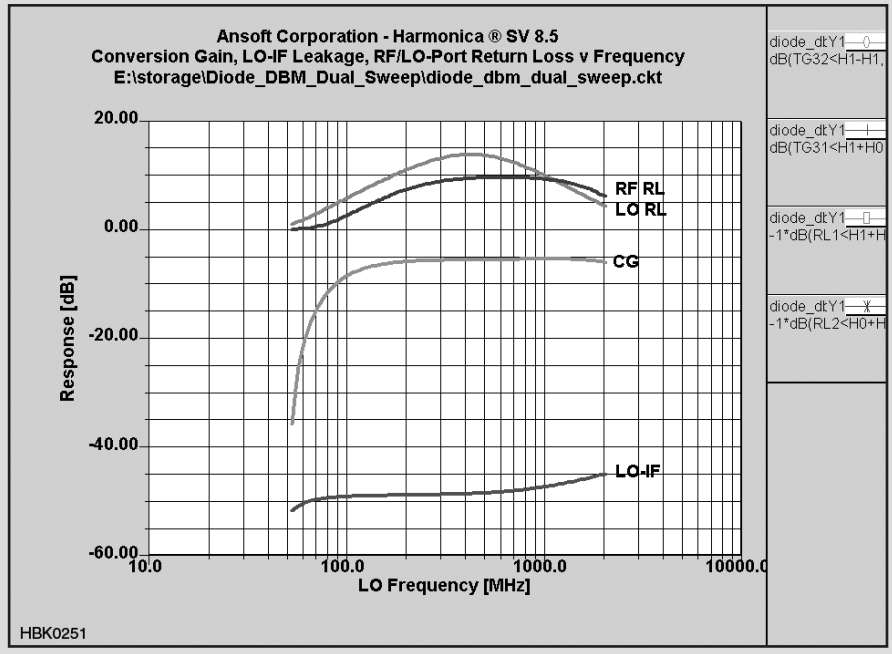


Fig 6.A9 — Professional RF circuit simulators can also simulate mixing and the small-signal characteristics of mixers, such as port return loss, conversion gain, and port-to-port isolation. (Serenade SV 8.5 *simulation*)

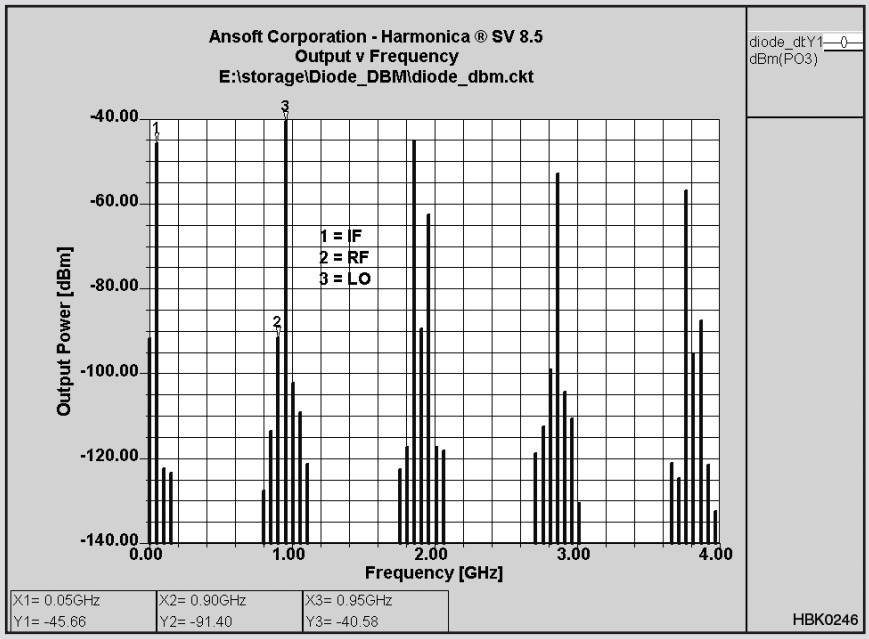


Fig 6.A10 — Output spectrum of a diode-ring doubly balanced mixer as simulated by *Serenade SV 8.5*. Note the dynamic range implicit in this graph: In a simulation that includes a local-oscillator (LO) signal at 7 dBm, we're seeing accurate values for IMD products nearly *140 dB weaker* without encountering mathematical noise — an achievement unapproachable with *SPICE*-based simulators.

The Dangers of simple usage of Microwave Software

Ulrich L Rohde *, Hans L. Hartnagel **

* Synergy Microwave Corp

201 McLean Blvd. Paterson, NJ, 07504, USA

e-mail: ulr@synergymwave.com

web: www.synergymwave.com:8080/

**Fachgebiet Mikrowellenelektronik, Technische Universitäet Darmstadt,

Merckstr.25, D-64283 Darmstadt, Germany

e-mail: hartnagel@ieee.org

web: www.mwe.tu-darmstadt.de

Introduction:

We have seen many dissertations about the implementation of microwave circuits, where a student has built an oscillator or another circuit, measured it, ran a simulation, obtained different answers and then tried to explain the reasons. Actually there are two main sources of inaccuracy, one being the measurements and the other the simulation. In the case of an oscillator the important parameters are output power, harmonic content and, most important, phase noise.

These three critical parameters are determined under large-signal conditions. Using CAD introduces automatically two weaknesses. The device used for the application needs to be characterized, many times by curve fitting, and needs to match a model of the simulator which itself is mostly an analytical model rather than physics based.

In a simple oscillator case we would like to show that using a rigorous mathematical model the educational benefit outweighs the simplicity of a CAD analysis and subsequent optimization.

The two test cases are a Driscoll oscillator with the crystal resonator in the emitter, which was taken from the literature and the design could have never worked because of errors in the publication data. A CAD tool would not have found the problem but an understanding of the operation allows to find the correction. The next case is the Colpitts oscillator, which offers many choices of design but only the large signal approach will work. This is more analytical rather than trial and error.

This effort is based on using Bessel functions and a calculation in the time and frequency domain. The added benefit is that all physics-based noise models will be used and therefore the student gains much more insight in all the concurrencies. Once the basic set of equations is derived, the first derivative of the feedback components vs. phase noise allows exact optimization.

This type of circuit analysis, which can be applied to many other designs such as amplifiers and mixers, give the best insight into the functionality of circuits in the time domain where we discover such new things like time average loaded Q and noisy feedback or noise-contributing support circuits.

In this paper we will use a simple but in the end highly non-linear circuit, where we will demonstrate the accuracy of our approach using simulations, sets of analytical time domain equations and of course accurate measurements using test equipment from two established manufacturers, Agilent and R&S. Each step of this design provides much better insight in the functionality than the standard teaching approach of this topic resorting to too much CAD. In the following we will show three cases, which will highlight the problems.

Case study of a: Twin T-oscillator using an Infineon BFR93aw, microwave transistor, showing that the linear and the non-linear analysis for the resonant frequency gives a significant different results [1], a analysis of a Driscoll oscillator where the Cad prediction is far too optimistic because it does not have good data for the flicker corner frequency [not provided by the manufacturer] and flicker noise contribution of the crystal and finally the third case is the mathematical analysis of the Colpitts oscillator using the large signal parameter and the Bessel function to get a very close result to both the measured and the CAD simulation [2].

Case 1:

In general and until recently transistor simulations used linear analysis, which turns out to give fairly inaccurate results. To show the base line, here is the example using a RC example. It is based on [1] and operates at 1.6MHz. Figure 1 shows the actual circuit diagram.

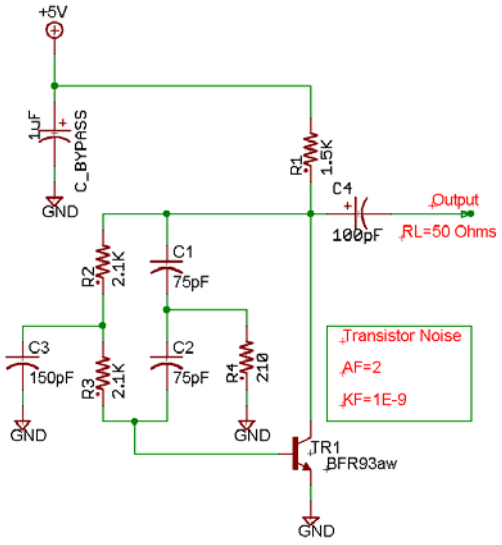


Figure 1: the actual circuit diagrams for 1.6MHz

The literature is full of RC oscillators but very little information is available on its phase noise and the difference between the linear and the non-linear operation. So we analyzed [Figure 1] this oscillator and scaled it to about 1MHz and using a linear simulation determined the following resonance frequency.

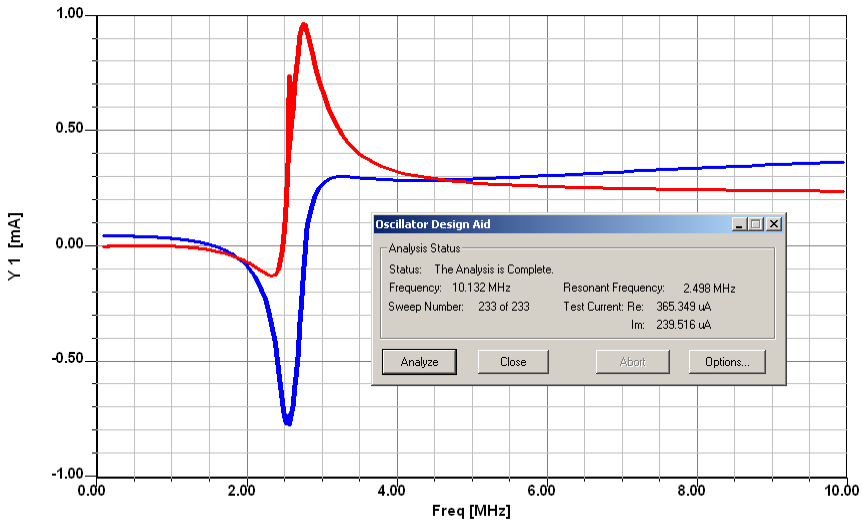


Figure 2: Linear simulation result of the schematic in figure 1.

The linear case indicates the resonance frequency around 2.5GHz. The Y-axis is the RF current in milliamps at the junction between the two resistors, 2.1Kohms and the 150pF capacitors to the ground. This assumes a total linear system and the purpose of this example is to show that the linear simulator can mislead you totally. After this result we used the Ansoft serenade harmonic

balance simulator 8.7V and a validated model for the siemens transistor BFR93aw. The initial DC analysis provides the operating point.

Bias Point Values

Voltage	Current
Vp() = -0.232686 uV	Ip() = 0 A
Vbe(_lib1) = 0.734765 V	Ib(_lib1) = 36.3168 uA
Vce(_lib1) = 0.862989 V	Ic(_lib1) = 2.70409 mA

The results are 2.7mA for 0.86V V_{ce}. The output waveform is slightly distorted and shown in figure 3. Figure 4 shows the harmonic contents. The output frequency as seen in figure 4 is different from the linear prediction and is 1.6MHz. The harmonic suppression is about 14dB. The loaded output terminated into 50 ohms is –19dBm.

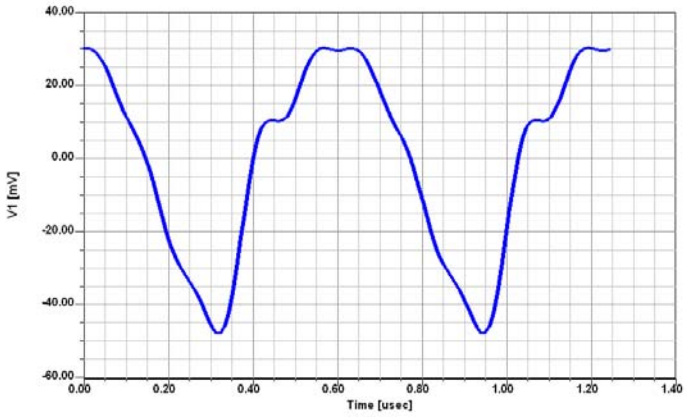


Figure 3: Simulated output waveform.

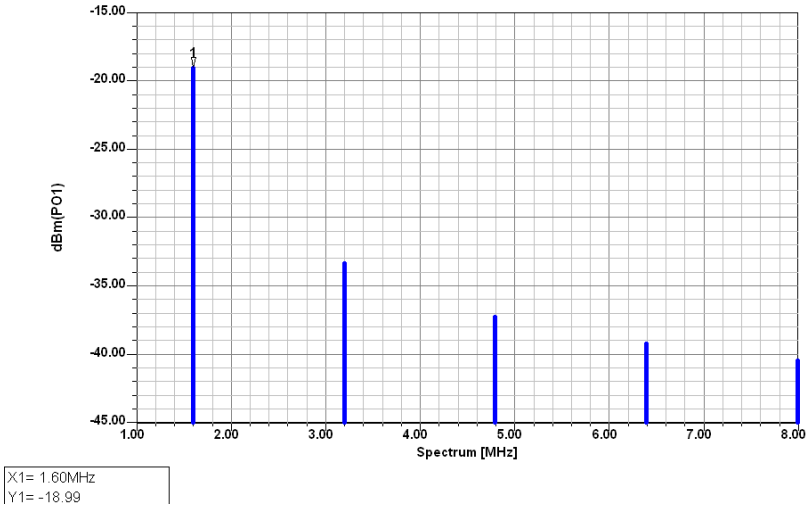


Figure 4: Simulated output power for the schematic in figure 1

We must keep in mind that this is a RC oscillator consisting of a notch filter and does not have a Q in the traditional sense. These types of oscillators typically do not operate into 50 ohms but into some CMOS gates, which are voltage and not power driven. IF we assume that the practical load is 9Kohms then the voltage swing at the output increase to ±900mV, this is 1.8Vp-p at the end to drive the gate.

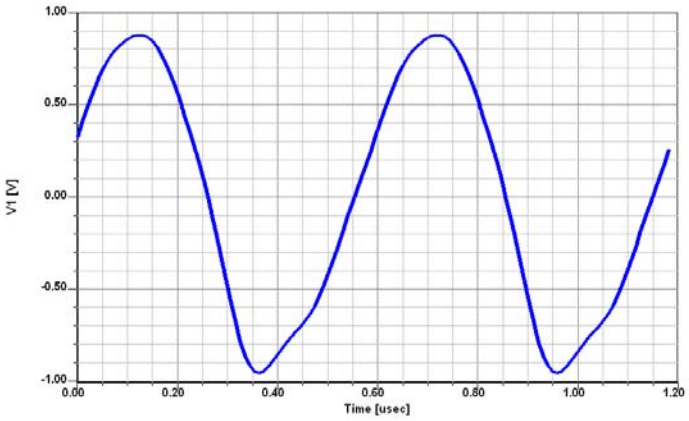


Figure 5: Simulated output waveform with a high impedance termination of 9Kohms.

Now to our surprise the resonant frequency is 1.679MHz a huge difference from the linear approximation. So far we have shown output power harmonic contents and now how about the phase noise.

This information is rarely found in the literature, but here it is shown in figure 6.

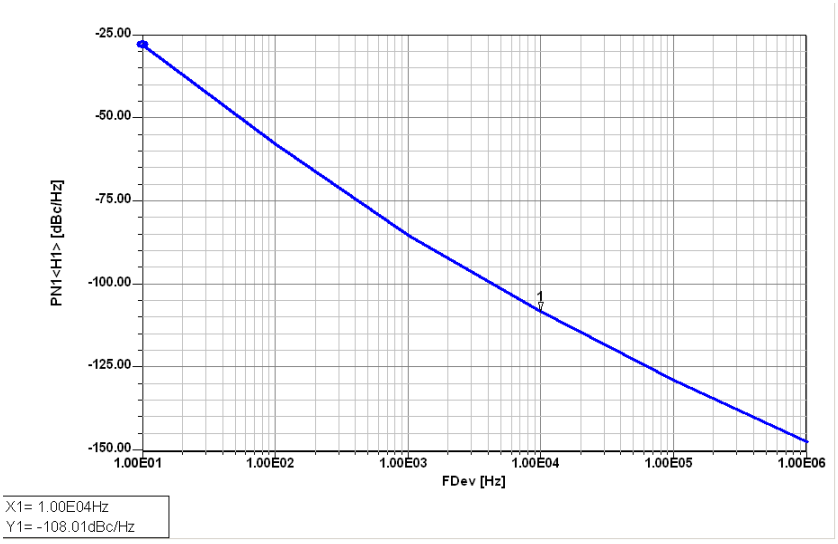


Figure 6: Predicted phase noise in dBc/Hz offset from the carrier frequency ranging from 10Hz to 1MHz of this RC oscillator.

By itself it is not overwhelming but if it is used as a part of the synthesizer loop divided down by the 100 a 40dB improvement, then it looks much better. The 10 KHz offset would be at –148dBc/Hz. It is mixed into a synthesizer it is a good performer.

Again why is this barely found in literature?

1: Most of the CAD tools cannot analyze this accurately. An important test is to validate the existence of the flicker corner frequency. In our case it is at 1 kHz. This is typical for a microwave transistor at this DC current, an audio type transistor or a FET to show much smaller number.

2: Majority of phase noise setups does not operate below 10MHz; Measurements of 5MHz are typically done using a diode multiplier at higher frequency.

Case 2:

One of the promising oscillator is the circuit discover by Driscoll; Its schematic shown in the figure 7.

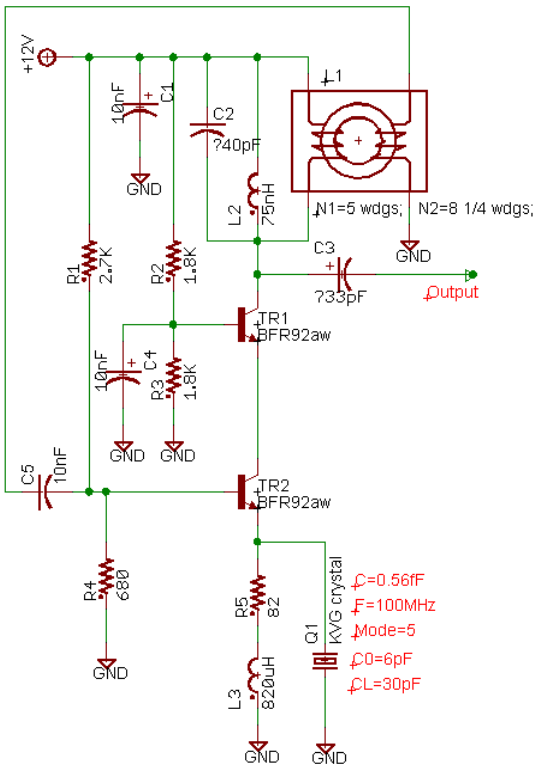


Figure 7: 100MHz crystal oscillator using the Driscoll Schematic.

Essentially it is a cascode amplifier where the output from the second transistor is inverted by 180 degrees and drives the lower transistor. At its resonance frequency the Crystal, grounds the emitter via a small resistor (C of the crystal) and the oscillations starts.

The measured results first. They were obtained using the FSUP.

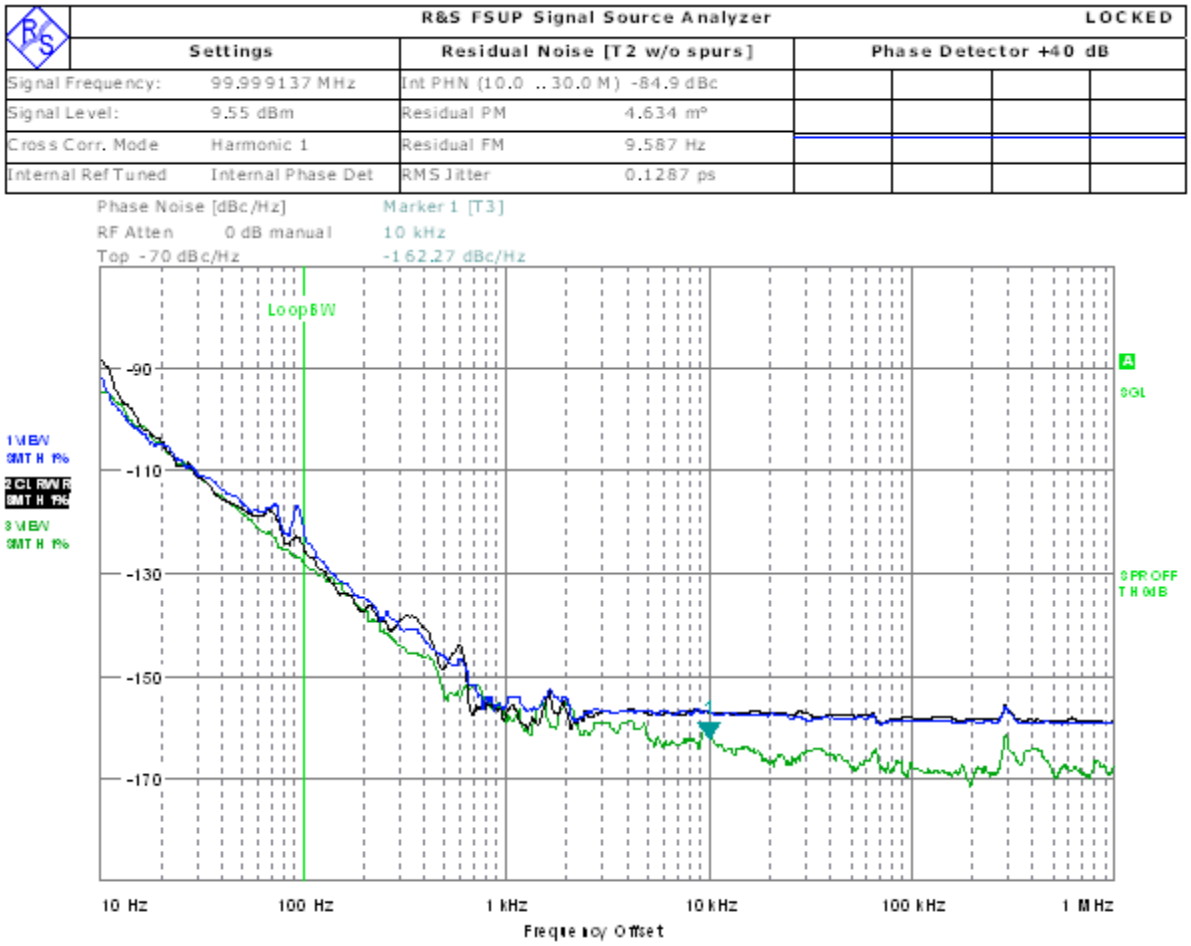


Figure 8: measured result for the 100MHz Crystal oscillator.

The difference between the blue and the green curve is the measurement taken without the buffer (green) and with the buffer amplifier, BGA614, dual Darlington amplifiers (blue curve). We really would like to point out that choice of right buffers is extra ordinarily important and the CAD tools may not give the right answer.

Now let's do the simulation.

The linear simulation tells us 99.998MHz. Because this is a very high Q device and it also maintains its high Q we can expect the circuit simulator to give a similar answer in the nonlinear mode.

The transistor cascode is operated at 17.4mA. The lower transistor TR2, model BFR92aw, has a V_{ce} of 3.46V and the upper transistor TR1 same model has the V_{ce} of 6.94V. The output power is 9.58dBm with the predicted harmonic suppression of 22 dB. The output waveform is shown in the figure 10 and it shows on the upper right corner the harmonic contents.

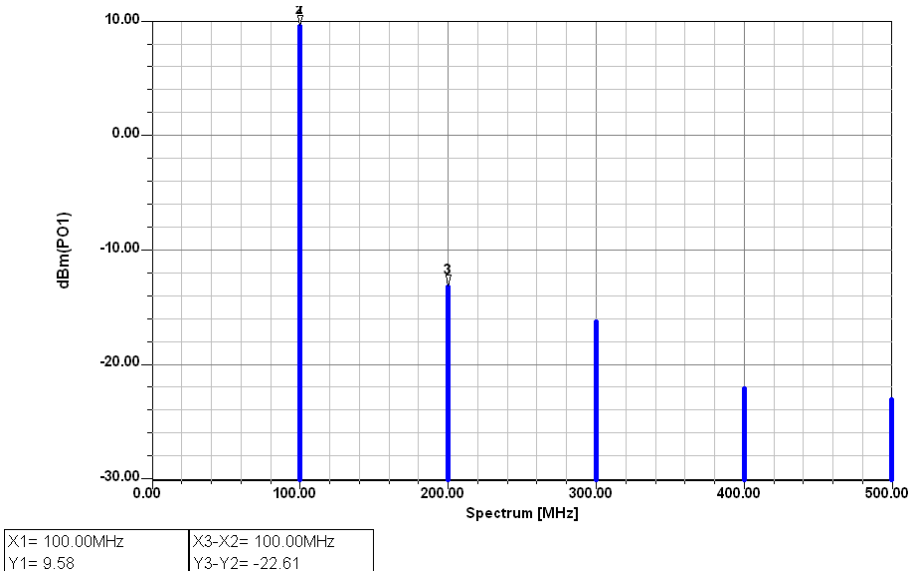


Figure 9 Predicted harmonic contents of the schematic in figure 7.

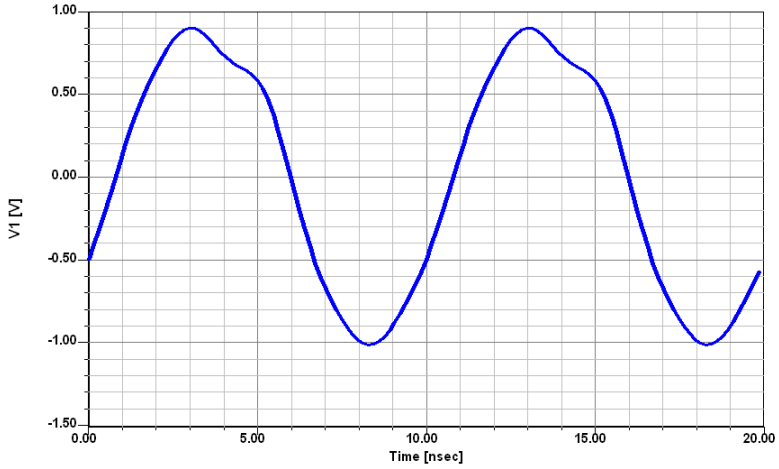


Figure 10: The estimated output waveform for 100MHz crystal oscillator

Now we are curious what the phase noise simulation tells us compared to the measured results.

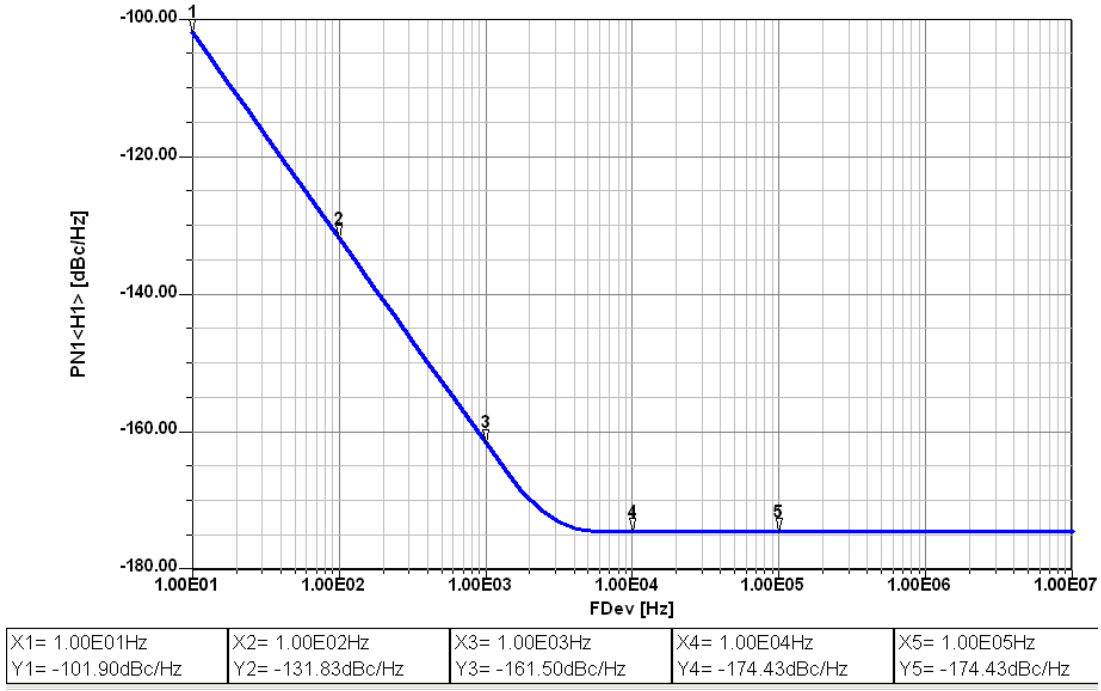


Figure 11: Predicted phase of the 100MHz crystal oscillator from Driscoll.

As we had already anticipated the CAD solution is somewhat erroneous. Since this is a crystal oscillator or a frequency reference it cannot be phase locked easily and there is no easy way to improve it. Here are the results and their deviations.

Frequency offset	10Hz	100Hz	1KHz	10KHz	100Khz
Simulation result	-102dBc	-132dBc	-161.5dBc	-174dBc	-174dBc
Measurement result	-90dBc	-125dBc	-155dBc	-162dBc	-170dBc

Having spent $50,000 for the simulator and $80,000 for the test equipment the simulator is too optimistic. The reason for this lays in the uncertainty of the flicker frequency which none of the manufacturers are willing to give guarantee for, and a type of flicker noise that the crystal has itself. The standard crystal models are not sufficiently accurate for the good modeling.

In case three we like to develop an analytic formula, which greatly eliminates the CAD cost, and its problems and yet gives results were calculations, not simulation agrees with the measurement.

Case 3:

A colpitts oscillator is an attractive oscillator, as it uses the capacitive divider and is essentially an emitter follower, which results in phase shift in transistor much less than a grounded emitter circuit has. For further details on Colpitts oscillator see reference [2].

The colpitts oscillator schematic is as shown below:

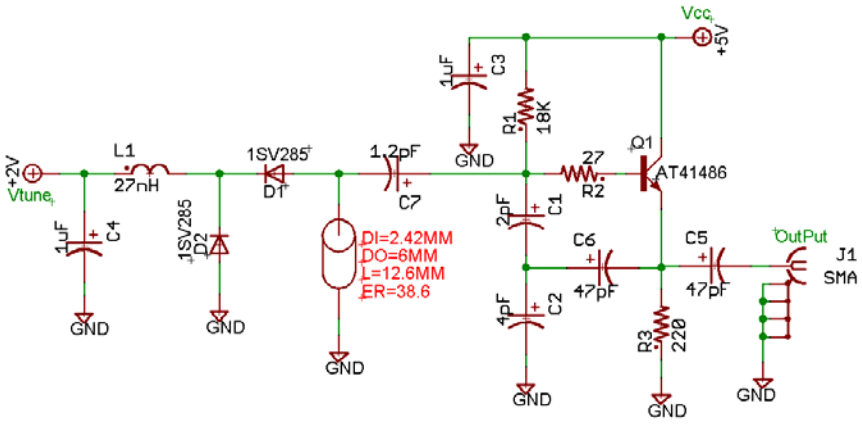


Figure 12: Colpitts oscillator design for 800-900MHz

What we have is a voltage controlled oscillator with Toshiba's 1Sv285 varactor. This fulfills the requirement the author notes for the tuning diodes, page (314) [1]. The simulated results in the book strongly disagree with the simulation [1, page 315 figure 7.19]

Frequency offset	100Hz	1KHz	10KHz	100Khz	1Mhz
Simulation result with diode	-22dBc	-52dBc	-82dBc	-111dBc	-136dBc
Simulation result without diode	-51.5dBc	-81dBc	-110dBc	-137dBc	-158dBc

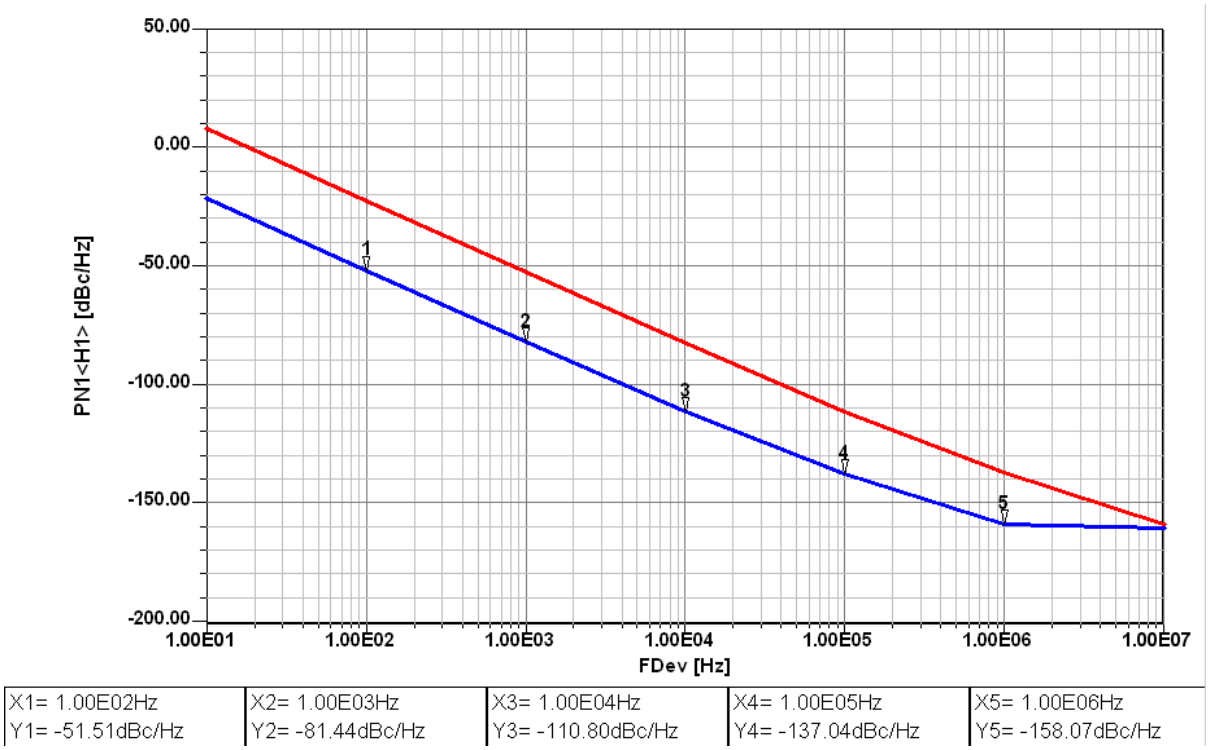


Figure 13: Predicted phase noise of the schematic in figure 12. The blue curve is with fixed capacitors, and the red curve is the phase noise with diodes.

As stated in the beginning a correct analysis of the oscillator, which is a transistor operating in the large signal stage, requires large signal parameters. The amplifier circuit applies enough negative feedback, to compensate the losses of the tuned circuit and the parasitic elements. Figure 14 shows the typical block diagram of conventional feedback oscillator circuit.

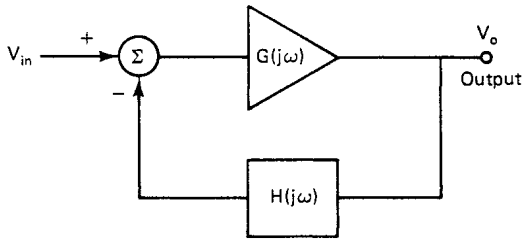


Figure14: A typical block diagram of feedback oscillator circuit [2]

Barkausen in 1935 was the first to state that for this case the product of forward voltage gain and the feedback voltage gain had to be > 1.

$$\frac{V_o}{V_{in}} = \frac{\mu}{1-\mu\beta} > 1 \qquad (1)$$

In recent years engineers used a linear approach and stipulated that the positive loss resistance and to be compensated by a parallel or series negative resistance.

Figure 15 shows a Colpitts oscillator, its input impedance with the feedback capacitors C1 and C2 connected, is calculated and to be seen later.

In the practical case, the device parasitics and loss resistance of the resonator will play an important role in the oscillator design. Figure 15 incorporates the base lead-inductance L_p and the package-capacitance C_p.

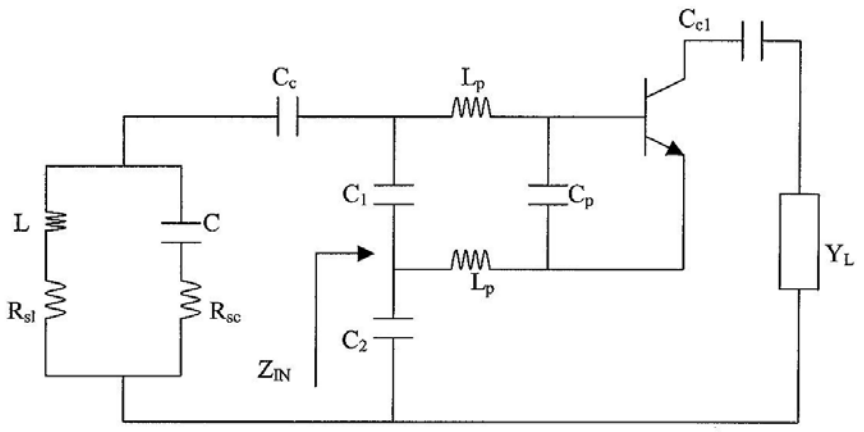


Figure 15: Colpitts oscillator with base-lead inductances and package capacitance. C_C is neglected.

The expression of input impedance is given as [2]

$$Z_{IN}\big|_{pacakage} = -\left[\frac{Y_{21}}{\omega^2(C_1+C_p)C_2}\frac{1}{(1+\omega^2 Y_{21}^2 L_p^2)}\right] - j\left[\frac{(C_1+C_P+C_2)}{\omega(C_1+C_P)C_2} - \frac{\omega Y_{21} L_P}{(1+\omega^2 Y_{21}^2 L_p^2)}\frac{Y_{21}}{\omega(C_1+C_P)C_2}\right] \quad (2)$$

$$Z_{IN}\big|_{without-pacakage} = -\left[\frac{Y_{21}}{\omega^2 C_1 C_2}\right] - j\left[\frac{(C_1+C_2)}{\omega C_1 C_2}\right] \quad (3)$$

Where L_p is the base-lead inductance of the bipolar transistor and C_p is base-emitter package capacitance. All further circuits are based on this model. From the expression above, it is obvious that the base lead-inductance makes the input capacitance appear larger and the negative resistance appears smaller.

The equivalent negative resistance and capacitance can be defined as [2]

$$R_{NEQ} = \frac{R_N}{(1+\omega^2 Y_{21}^2 L_p^2)} \qquad (4)$$

The assumptions in the past were if R_{NEQ} was sufficiently negative then stable oscillation occurs. However the oscillator is inherently a linearized non-linear circuit and the assumption that this R_{NEQ} was sufficient was not always correct.

The value of R_{NEQ} is the starting value before oscillation, and as the large signal condition takes over, Y21 decreases!

This large signal effect will be analyzed and will become part of the noise analysis und large signal condition.

Large Signal Analysis:

In order to better understand the noise generation in an oscillator, we need to first leave the traditional small signal analysis and consider the actual large signals conditions. So instead of using the familiar linear S parameter, we now resort to their large signal equivalent,

Large Signal *S*-Parameter Measurements

Assume S_{11} and S_{21} are functions only of incident power at port 1 and S_{22} and S_{12} are functions only of incident power at port 2. Note: the plus (+) sign indicates the forward wave (voltage) and the minus (-) sign would be the reflected wave (voltage).

$$S_{11} = S_{11}\left(\left|V_1^+\right|\right) \quad S_{12} = S_{12}\left(\left|V_2^+\right|\right) \qquad (5)$$

$$S_{21} = S_{21}\left(\left|V_1^+\right|\right) \quad S_{22} = S_{22}\left(\left|V_2^+\right|\right) \qquad (6)$$

The relationship between the traveling waves now becomes

$$V_1^- = S_{11}(V_1^+)V_1^+ + S_{12}(V_2^+)V_2^+ \qquad (7)$$

$$V_2^- = S_{21}(V_1^+)V_1^+ + S_{22}(V_2^+)V_2^+ \qquad (8)$$

Measurement is possible if V_1^+ is set to zero,

$$S_{12}(V_2^+) = \frac{V_1^-(V_2^+)}{V_2^+} \qquad (9)$$

Check the assumption by simultaneous application of V_1^+ and V_2^+

$$\begin{bmatrix} V_1^- \\ V_2^- \end{bmatrix} = \begin{bmatrix} F_1(V_1^+, V_2^+) \\ F_2(V_1^+, V_2^+) \end{bmatrix} \qquad (10)$$

If harmonics are neglected, a general decomposition is

$$\begin{bmatrix} V_1^-(V_1^+, V_2^+) \\ V_2^-(V_1^+, V_2^+) \end{bmatrix} = \begin{bmatrix} S_{11}(V_1^+, V_2^+) & S_{12}(V_1^+, V_2^+) \\ S_{21}(V_1^+, V_2^+) & S_{22}(V_1^+, V_2^+) \end{bmatrix} \begin{bmatrix} V_1^+ \\ V_2^+ \end{bmatrix} \qquad (11)$$

Figure 16 shows the R&S vector analyzer and the test fixture for the transistor of choice.

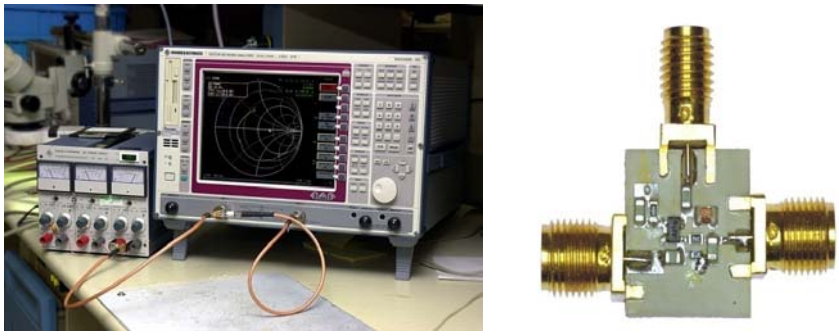


Figure 16: Typical measurement setup for evaluation of large signal parameters (R&S vector analyzer and the test fixture for the transistor of choice)

The bias, drive level, and frequency dependent S parameters are then obtained for practical use. Since we did not have an access to AT41486, we used the infineon transistor BFP520 as an example.

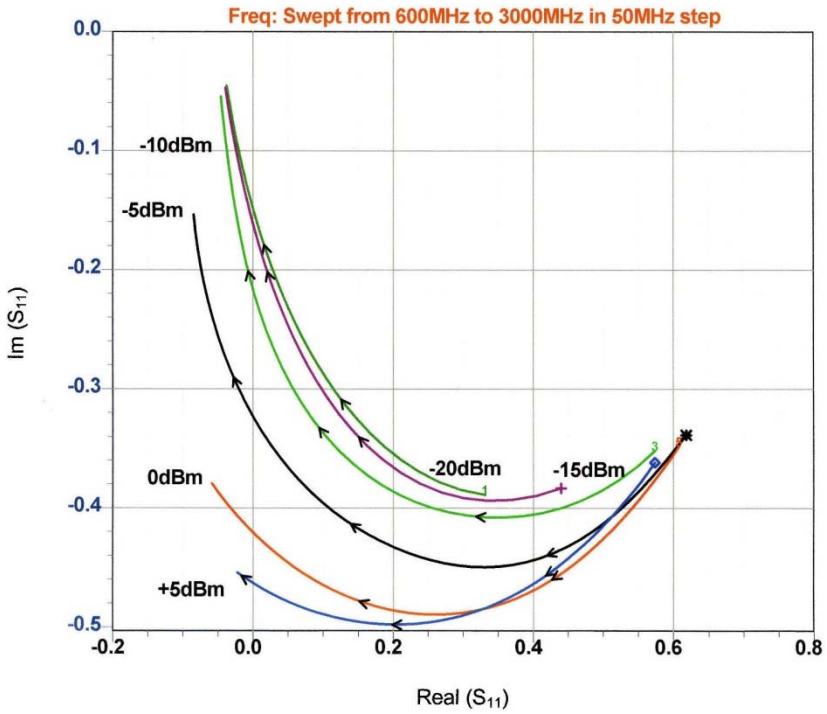


Figure 17: Measured large-signal S_{11} of the BFP520 [2, pp. 68].

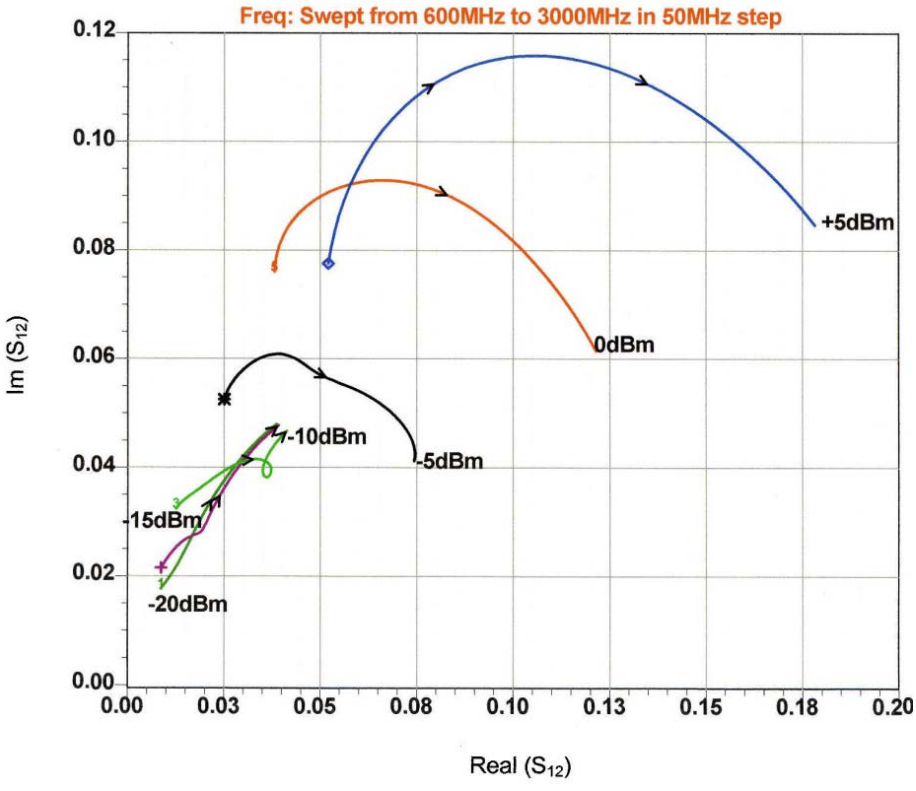


Figure 18: Measured large-signal S_{12} of the BFP520 [2, pp. 68]

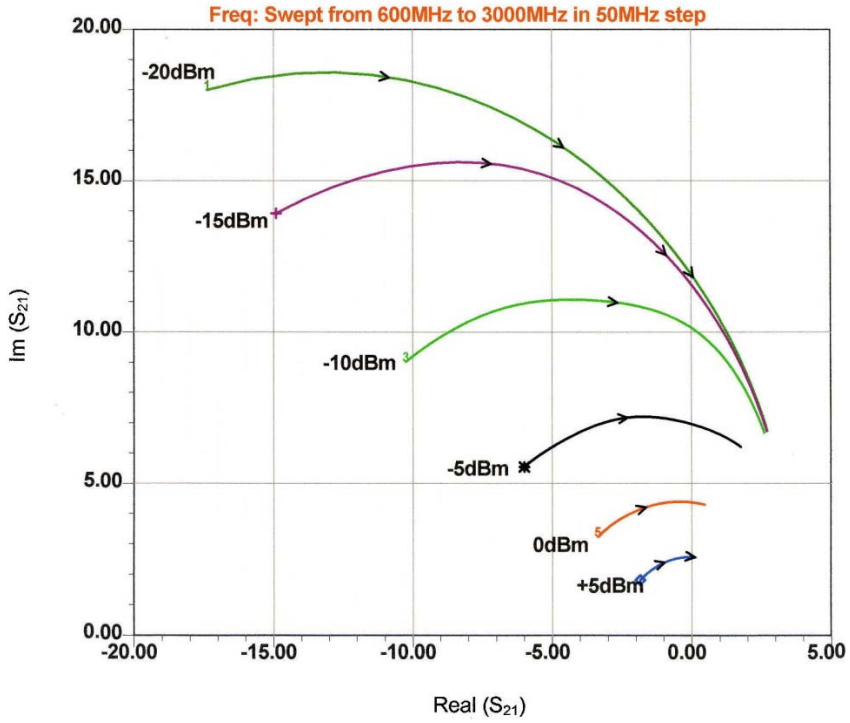


Figure 19: Measured large-signal S_{21} of the BFP520 [2, pp. 69].

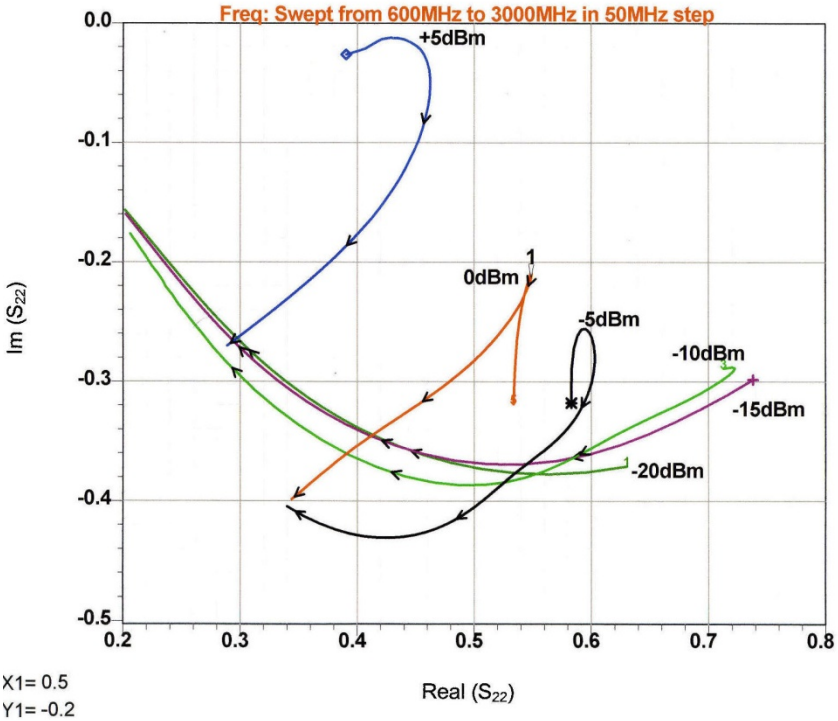


Figure 20: Measured large-signal S_{22} of the BFP520 [2, pp. 69].

The legal definitions of large signal S parameter apply only for a 50-Ohm termination. In our case, an oscillator, the harmonic related S parameters could be neglected. Otherwise the load pull technique applies.

Why are these parameters of interest for us?

They show the dramatic change of S_{11} and S_{21} as a function of frequency and bias level. For the Colpitts oscillator, where the collector is separated S22 is less relevant and since the feedback is external, S12 also less important, depending on the frequency. If calculating the negative resistance to compensate the losses, we must insert the large signal frequency depending value for Y_{21}.

Large-Signal Oscillator Design and Start-Up Condition

As a basic requirement for producing a self-sustained near-sinusoidal oscillation, an oscillator must have a pair of complex-conjugate poles on the imaginary axis i.e. in the right half of s-plane with α>0.

$$P(p_1, p_2) = \alpha \pm j\beta \qquad (12)$$

When the *Barkhausen criterion* is met, the two conjugate poles of the overall transfer function are located on the imaginary axis of the s-plane. Any departure from that position will lead to an increase or a decrease of the amplitude of the oscillator output signal in time domain, which is shown in Figure 21. Figure 22 shows the typical transient simulation of a ceramic resonator-based high-Q oscillator, where node of the voltage is taken from the emitter.

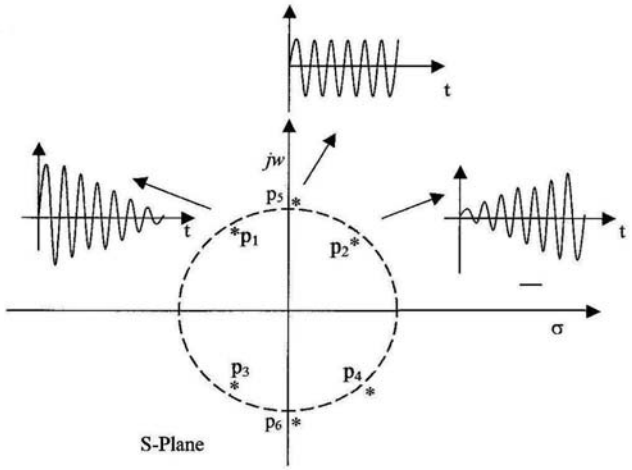


Figure 21: Typical frequency domain root locus and the corresponding time domain response [2, pp. 96].

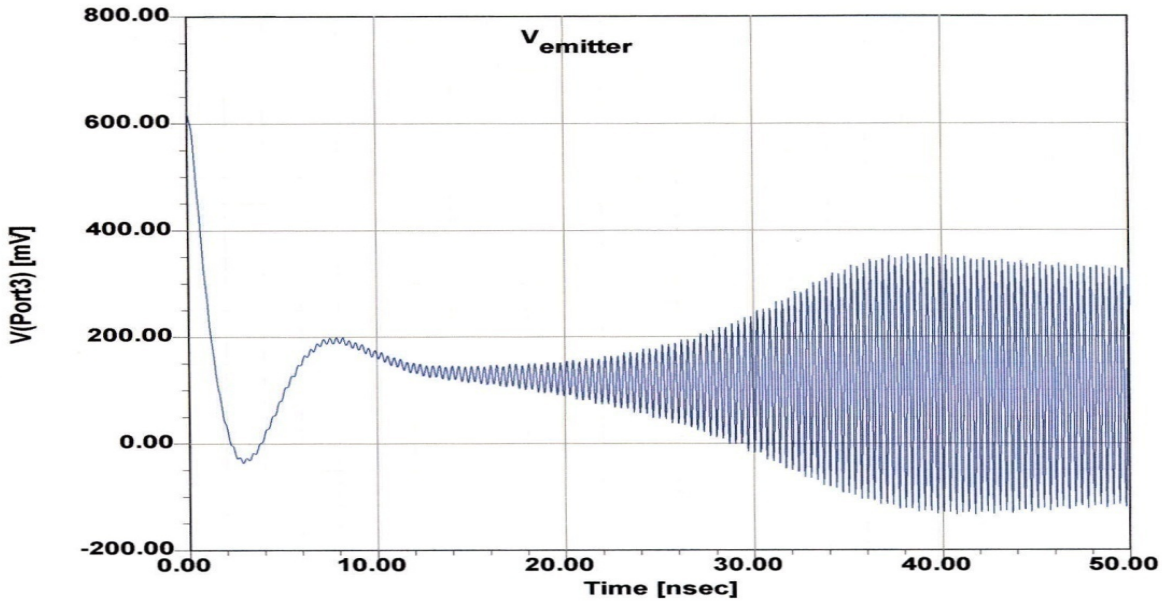


Figure 22: Typical transient simulation of a ceramic resonator-based high-Q oscillator (node of the voltage for display is taken from the emitter) [2, pp.100].

The steady state oscillation condition can be expressed as

$$\Gamma_a(A, f)\Gamma_r(f)\big|_{f=f_0} \Rightarrow \Gamma_a(A_0, f_0)\Gamma_r(f_0) = 1 \qquad (13)$$

For brief insights about the negative resistance oscillator, a block diagram of one-port negative reflection model is shown in Figure 23.

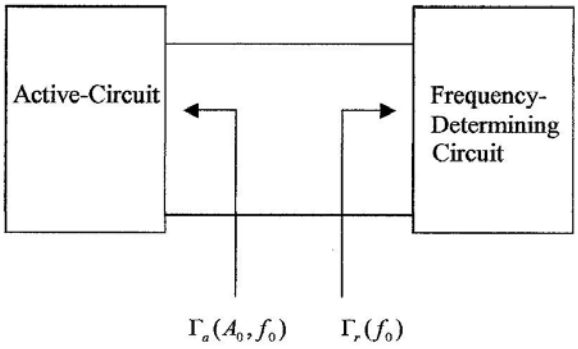


Figure 23: Schematic diagram of a one-port negative reflection model.

Figure 24 illustrates the start and steady-state oscillation conditions.

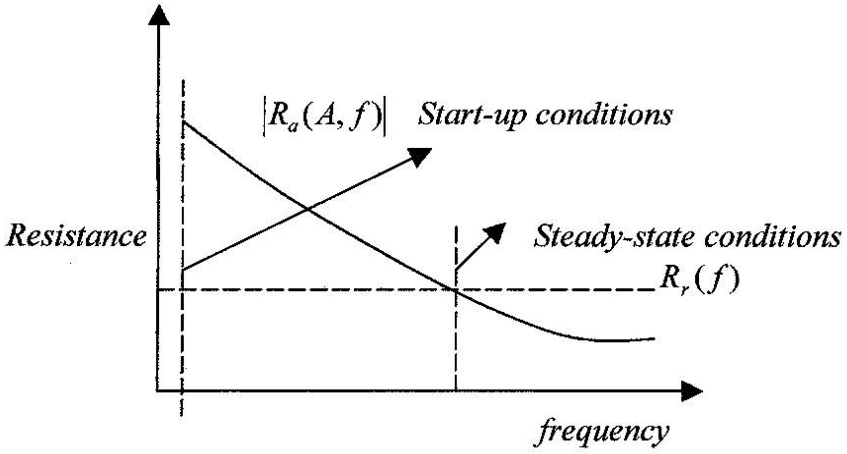


Figure 24: A typical start and steady-state oscillation conditions.

As described in Figure 24, R_a (*A*, *f*) is the starting negative Resistance, which gets lower as the amplitude increases. Therefore, feedback must be sufficient to maintain enough negative resistance to sustain oscillating.

Time-Domain Behavior

The large-signal transfer characteristic affecting the current and voltage of an active device in an oscillator circuit is nonlinear. It limits the amplitude of the oscillation and produces harmonic content in the output signal. The resonant circuit and resulting phase shift sets the oscillation frequency. The nonlinear, exponential relationship between the voltage and current of a bipolar transistor is given as

$$i(t) = I_s e^{\frac{qv(t)}{kT}} \tag{14}$$

$$v(t) = V_{dc} + V_1 \cos(wt) \tag{15}$$

$$i_e(t) = I_s e^{\frac{qv(t)}{kT}} \tag{16}$$

$$i_e(t) = I_s e^{\frac{qV_{dc}}{kT}} e^{\frac{qV_1 \cos(wt)}{kT}} \tag{17}$$

$$i_e(t) = I_s e^{\frac{qV_{dc}}{kT}} e^{x\cos(wt)} \tag{18}$$

assuming, $I_c \approx I_e$ (β>10)

The normalized drive level is

$$x = \frac{V_1}{(kT/q)} = \frac{qV_1}{kT} \qquad (19)$$

$i_e(t)$ is the emitter current and x is the drive level which is normalized to kT/q.

From the Fourier series expansion, $e^{x\cos(wt)}$ is expressed as

$$e^{x\cos(wt)} = \sum_n a_n(x)\cos(nwt) \qquad (20)$$

$a_n(x)$ is a Fourier coefficient and given as

$$a_0(x)\big|_{n=0} = \frac{1}{2\pi}\int_0^{2\pi} e^{x\cos(wt)}d(wt) = I_0(x) \qquad (21)$$

$$a_n(x)\big|_{n>0} = \frac{1}{2\pi}\int_0^{2\pi} e^{x\cos(wt)}\cos(nwt)d(wt) = I_n(x) \qquad (22)$$

$$e^{x\cos(wt)} = \sum_n a_n(x)\cos(nwt) = I_0(x) + \sum_1^{\infty} I_n(x)\cos(nwt) \qquad (23)$$

$I_n(x)$ is the modified Bessel function.

As $x \to 0 \Rightarrow I_n(x) \to \frac{(x/2)^n}{n!}$ (24)

$I_0(x)$ are monotonic functions having positive values for x≥0 and n≥0; $I_0(0)$ is unity, whereas all higher order functions start at zero.

The short current pulses are generated from the growing large-signal drive level across the base-emitter junction, which leads to strong harmonic generation. The emitter current represented above can be expressed in terms of harmonics as [2].

$$i_e(t) = I_s e^{\frac{qV_{dc}}{kT}} I_0(x)\left[1 + 2\sum_1^{\infty}\frac{I_n(x)}{I_0(x)}\cos(nwt)\right] \qquad (25)$$

$$I_{dc} = I_s e^{\frac{qV_{dc}}{kT}} I_0(x) \tag{26}$$

$$V_{dc} = \frac{kT}{q}\ln\left[\frac{I_{dc}}{I_s I_0(x)}\right] \Rightarrow \frac{kT}{q}\ln\left[\frac{I_{dc}}{I_s}\right] + \frac{kT}{q}\ln\left[\frac{1}{I_0(x)}\right] \tag{27}$$

I_s = collector saturation current

$$V_{dc} = V_{dcQ} - \frac{kT}{q}\ln I_0(x) \tag{28}$$

$$i_e(t) = I_{dc}\left[1 + 2\sum_1^{\infty}\frac{I_n(x)}{I_0(x)}\cos(nwt)\right] \tag{29}$$

αI_e and Y_{21} are the current source and large-signal transconductance of the device given by the ratio of the fundamental-frequency component of the current to the fundamental-frequency of the drive voltage.

$$Y_{21} = \left.\frac{I_{1peak}}{V_{1peak}}\right|_{fundamental-frequency} \tag{30}$$

$$I_1\big|_{n=1} = I_{dc}\left[1 + 2\sum_1^{\infty}\frac{I_1(x)}{I_0(x)}\cos(wt)\right] \Rightarrow I_{1peak} = 2I_{dc}\frac{I_1(x)}{I_0(x)} \tag{31}$$

x = normalized drive level

$$V_1\big|_{peak} = \frac{kT}{q}x \tag{32}$$

$$Y_{21}\big|_{large-signal} = G_m(x) \tag{33}$$

$$Y_{21}\big|_{small-signal} = \left.\frac{I_{dc}}{kT/q}\right| = g_m \tag{34}$$

$$Y_{21}\big|_{large-signal} = G_m(x) = \frac{qI_{dc}}{kTx}\left[\frac{2I_1(x)}{I_0(x)}\right]_{n=1} = \frac{g_m}{x}\left[\frac{2I_1(x)}{I_0(x)}\right]_{n=1} \tag{35}$$

$$\frac{[Y_{21}\big|_{large-signal}]_{n=1}}{[Y_{21}\big|_{small-signal}]_{n=1}} = \frac{G_m(x)}{g_m} \Rightarrow \frac{2I_1(x)}{xI_0(x)} \tag{36}$$

$$|Y_{21}|_{small-signal} > |Y_{21}|_{large-signal} \Rightarrow g_m > G_m(x) \qquad (37)$$

This allows us to calculate the frequency dependent transconductance, which is needed to optimize the circuit for best noise performance.

The following picture (Figure 25) shows the collector current as a function of time and the normalized base drive voltage *x*. For larger values of *x*, the current and voltage peaks may require a larger transistor. As a result, the time the tuned circuit during less time gets loaded, is reduced and the time average Q is higher.

Figure 26 shows the phase noise of an LC-based 1GHz oscillator as a function of X. For higher values of X the phase noise improves significantly.

The dependency of x can be expressed as

$$x = \frac{R_P G_m C_2}{C_1} \qquad (38)$$

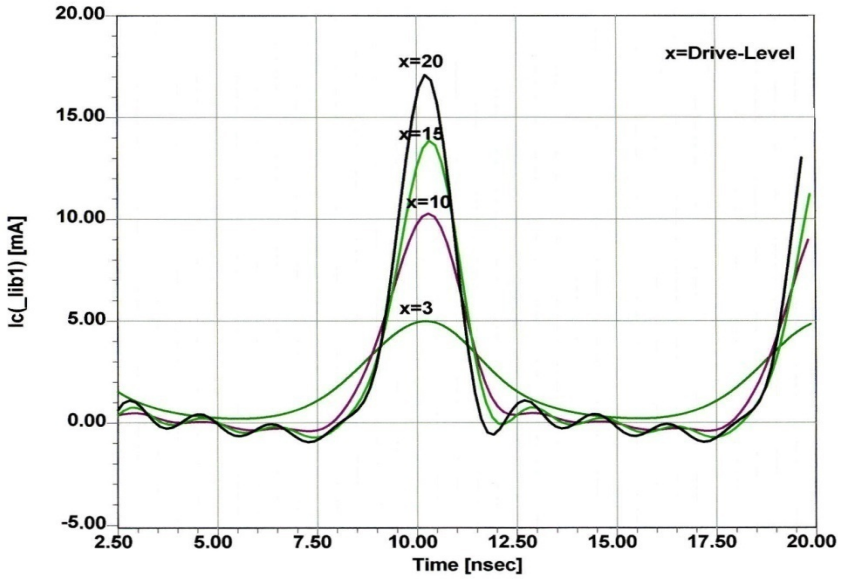


Figure 25: Plot shows the collector current as a function of time with respect to normalized base drive Voltage *x*.

For large drive level, $x \propto C_2$, and the corresponding conduction angle of the output current is given as

$$\varphi = \cos^{-1}\left[1 + \frac{\ln(0.05)}{x}\right] \Rightarrow \varphi \approx \cos^{-1}\left[1 - \frac{3}{x}\right] \qquad (39)$$

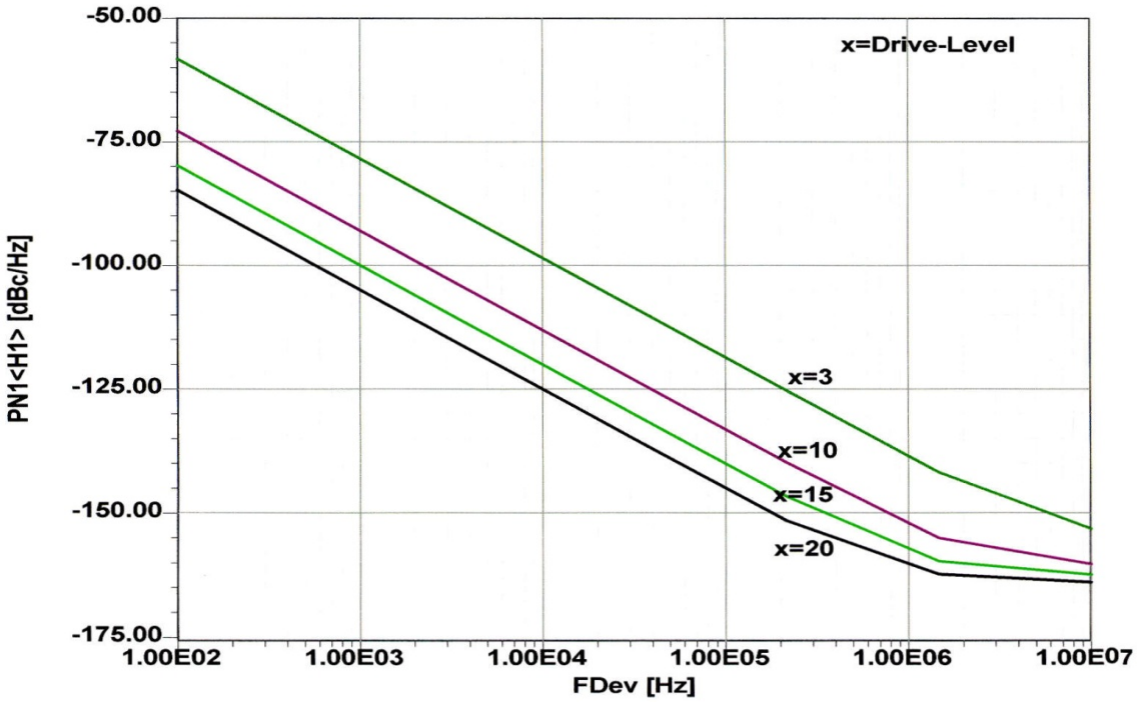


Figure 26: A typical phase noise plot of 1GHz oscillator as a function of x

There is a limit for x, not only due to the limits of voltage and current but also because of reverse biasing of the base collector diode which then makes the circuit really noisy.

Having learned how to design the feedback circuit and introduced the conducting angle and its calculation we have simulated and confirmed the influence on the phase noise, but have not really introduced the oscillator phase noise.

Phase noise in Oscillators

A Linear approach:

In 1965 Leeson developed a model for a noisy transistor oscillator based on a phase modulator, an amplifier, a low pass filter and a resonator, see (Figure 27). In general, oscillator can be viewed as a mixer, where the sum of all inputs is collected and superimposed on the oscillator. Figure 28 shows the components where oscillator acts like a mixer circuit.

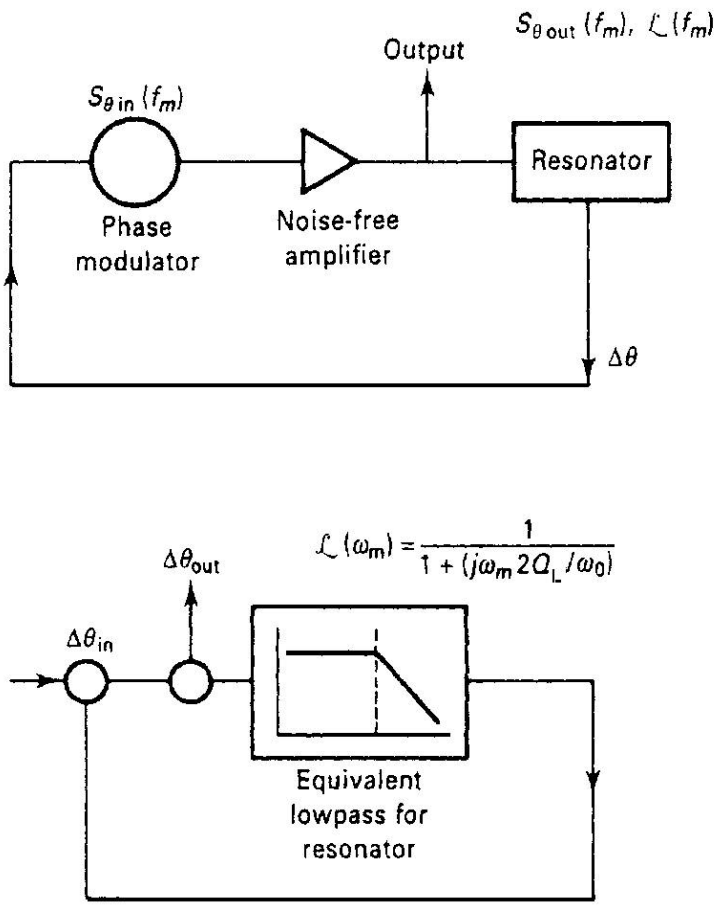


Figure 27: A typical linear oscillator phase noise model (block diagram)

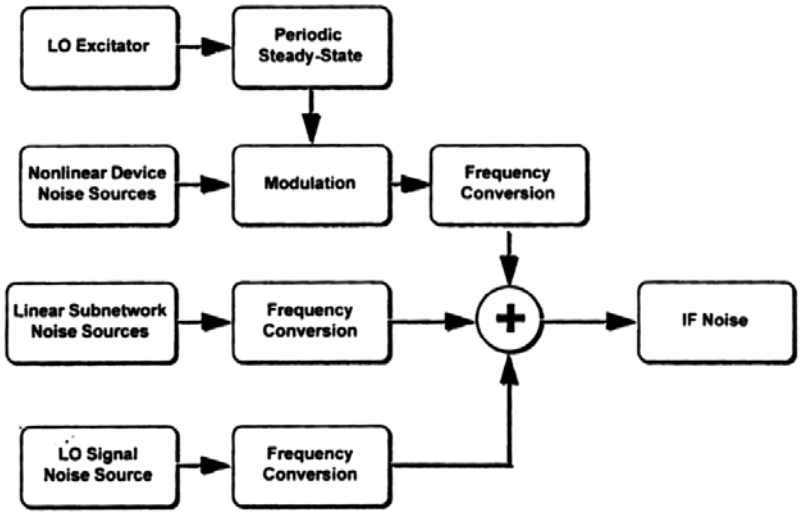
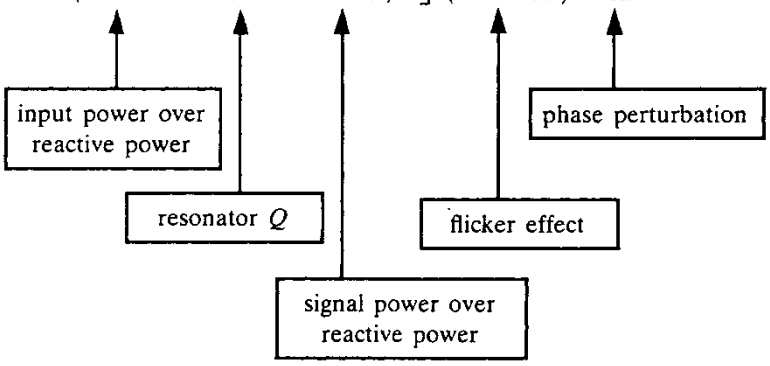


Figure 28: A typical block diagram of mixer circuit, where the oscillator acts like a mixer.

From [2], the resulting signal in linear terms can be calculated as

$$\mathcal{L}(f_m) = \frac{1}{2}\left[1 + \frac{\omega_o^2}{4\omega_m^2}\left(\frac{P_{\text{in}}}{\omega_o W_e} + \frac{1}{Q_{\text{unl}}} + \frac{P_{\text{sig}}}{\omega_o W_e}\right)^2\right]\left(1 + \frac{\omega_c}{\omega_m}\right)\frac{FkT_o}{P_{\text{sav}}}$$



(40)

Equation (40) is the linear Leeson equation, with the pushing effect omitted and the flicker term added by Dieter Scherer (Hewlett Packard, about 1975), the final version with the pushing (VCO effect) added by Rohde, is

$$L\ (f_m) = 10\log\left\{\left[1 + \frac{f_0^{\,2}}{(2f_m Q_L)^2}\right]\left(1 + \frac{f_c}{f_m}\right)\frac{FkT}{2P_{sav}} + \frac{2kTRK_0^{\,2}}{f_m^{\,2}}\right\} \qquad (41)$$

Where

$L\ (f_m)$ = ratio of sideband power in a 1 Hz bandwidth at f_m to total power in dB

f_m = frequency offset

f_0 = center frequency

f_c = flicker frequency

Q_L = loaded Q of the tuned circuit

F = noise factor

$kT = 4.1 \times 10^{-21}$ at 300 K_0 (room temperature)

P_{sav} = average power at oscillator output

R = equivalent noise resistance of tuning diode (typically 50 Ω - 10 kΩ)

K_o = oscillator voltage gain

The problem with this is that key values like loaded Q, large signal NF and output power are not known a priory and the effect of transistor distortion are not included. In some way this provides sometimes an unrealistic good phase noise. On the other hand, it shows the limitation for reasonable values and this presentation will show some mechanism to overcome this.

Figure 29 shows the plot for an ideal 1 GHz LC-based oscillator phase noise of about -140dBc/Hz at offset of 10 kHz offset, assuming unloaded Q of 1E6, loaded Q of 500, noise factor 6 dB, flicker frequency 1kHz, oscillator voltage gain 1Hz/V, equivalent noise resistance of tuning diode 1Ohm and average power at oscillator output 10dBm. Even today this is very much state of the art designer can achieve.

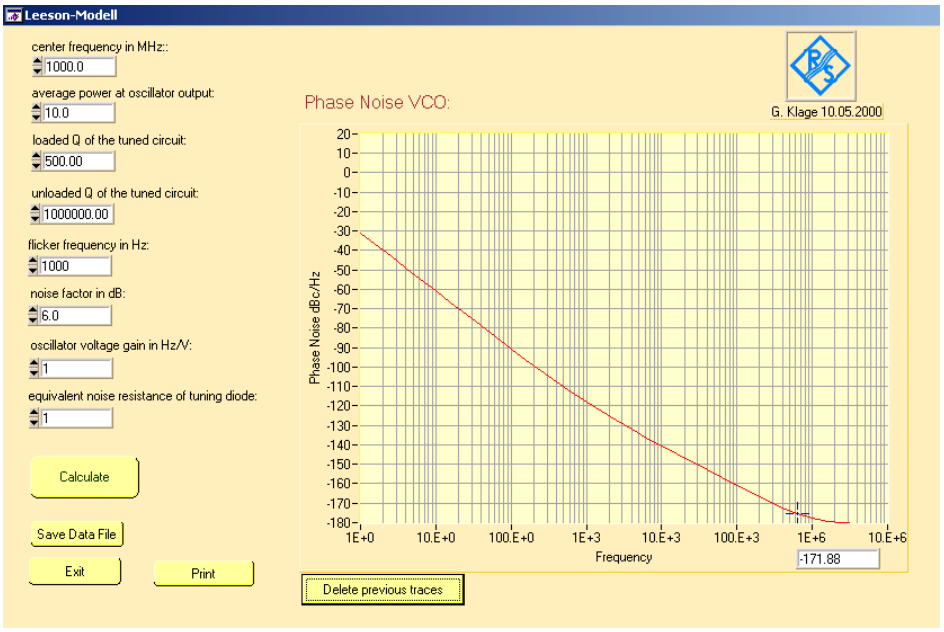


Figure 29: A typical phase noise plot for an ideal 1 GHz oscillator phase noise of about –140dBc/Hz at offset of 10 kHz offset, assuming unloaded Q of 1E6, loaded Q of 500, noise factor 6 dB, flicker frequency 1kHz, oscillator voltage gain 1Hz/V, equivalent noise resistance of tuning diode 1Ohm and average power at oscillator output 10dBm.

The non-linear noise approach [2, Ch-8]:

The equations (41) use a linearized system and are too simplified.

To start the nonlinear noise calculation, we look at the noise sources. The resonator noise is [2, Ch-8, pp. 159-232]

$$\left|\overline{e_R^2(f)}\right|_{\omega=\omega_0} = 4kTBR_s \qquad (42)$$

R_s is the series equivalent noise resistance, based on the losses of the resonator.

The circuit equation for the oscillator with the negative resistance present is [2, Ch-8, pp. 159-232]

$$L\frac{di(t)}{dt}+(R_L-R_N(t))i(t)+\frac{1}{C'}\int i(t)dt=e_N(t) \qquad (43)$$

This is a non-homogeneous differential equation, which can be simplified to [2, Ch-8, pp. 159-232]

$$L\left[-I_1(t)(\omega+\frac{d\varphi_1(t)}{dt})\sin[\omega t+\varphi_1(t)]+\frac{dI_1(t)}{dt}\cos[\omega t+\varphi_1(t)]\right]+\left[(R_L-R_N(t))I(t)\right]+$$

$$\frac{1}{C}\left\{\left[\frac{I_1(t)}{\omega}-\frac{I_1(t)}{\omega^2}\left(\frac{d\varphi_1(t)}{dt}\right)\right]\sin[\omega t+\varphi_1(t)]+\frac{1}{\omega^2}\left(\frac{dI_1(t)}{dt}\right)\cos[\omega t+\varphi_1(t)]\right\}=e_N(t) \qquad (44)$$

Further

where $\overline{R_N(t)}$ is the average negative resistance under large signal condition.

$$\overline{R_N(t)}=\left[\frac{2}{T_0 I}\right]\int_{t-T_0}^{t}R_N(t)I(t)\cos^2[\omega t+\varphi)]dt$$

Contrary to common publications, this is a time variant resistance; ideally it does not degrade the Q outside the on condition. This resistance however is “noisy”.

Since the negative resistance is related to the large signal transconductance and the feedback capacitors of the Colpitts oscillator, we can insert this in the equation above and after a lengthy set of calculation the phase noise under large signal conditions become [2, Ch-8, pp. 159-232]

$$\pounds(\omega)=10\times\log\left[\left[k_0+\left(\frac{k^3k_1\left[\frac{Y_{21}^+}{Y_{11}^+}\right]^4[y]^{4p}}{\left[Y_{21}^+\right]^6[y]^{6q}}\right)\left(\frac{1}{(y^2+k)}\right)\right]\left[\frac{[1+y]^2}{y^2}\right]\frac{Q_{\max}^2}{Q_0^2}\right] \qquad (45)$$

Where, $y=\frac{C_1}{C_2}$, $k_0=\frac{kTR}{\omega^2\omega_0^2L^2V_{cc}^2C_2^2}$, $k_1=\frac{qI_cg_m^2+\frac{K_fI_b^{AF}}{\omega}g_m^2}{\omega^2\omega_0^4L^2V_{cc}^2}$, $k_2=\omega_0^4(\beta^+)^2$, $k=\frac{k_3}{k_2C_2^2}$

This results in the phase noise values as a function of the large signal parameters. These are identified and the term k_1 adds the semiconductor noise contributions, which are now bias dependent.

The following is a first in the sense, that we calculate the exact solution of the phase noise of the transmission line, using a smaller than quarter wavelength resonator (inductive) and substitute this for the inductor. This uses a tangent function and if the losses would be applied a hyperbolic tangent function. In this case we assume that the Q is sufficiently high that the value of the $\cosh(\alpha l) \approx \sinh(\alpha l) \approx e^{\frac{\alpha l}{2}}$

The characteristic impedance of most of the coaxial resonator is approximately 10ohms and can be calculated by the following equation. D is the outer diameter or side length of the coaxial resonator, d is the inner diameter of the coaxial resonator and εr is the dielectric constant.

$$Z = \frac{60}{\sqrt{\varepsilon r}} \ln\left(\frac{D}{d}\right) = \frac{60}{\sqrt{38.6}} \ln\left(\frac{6}{2.42}\right) = 8.768\Omega$$

We know that, $L = \frac{Z_l}{\omega}$ [3]

Where, $Z_l(f) = jZ\tan(\beta l)$; $Z_l(f) = jZ\tan\left(\frac{\omega}{v_P} l\right)$;

Therefore $L = j\frac{Z}{\omega}\tan\left(\frac{\omega}{v_P} l\right)$ where, v_P is the Phase velocity and l is the length of the coaxial resonator.

As seen from this equation L is the function of frequency and needs to be calculated for each computation of frequency sweep.

So the modified equation for phase noise calculation is as follows.

$$\pounds(\omega) = 10 \times \log\left[\left[k_0 + \left(\frac{k^3 k_1 \left[\frac{Y_{21}^+}{Y_{11}^+}\right]^4 [y]^{4p}}{\left[Y_{21}^+\right]^6 [y]^{6q}}\right)\left(\frac{1}{(y^2+k)}\right)\right]\left[\frac{[1+y]^2}{y^2}\right]\right]\frac{Q_{\max}^2}{Q_0^2} \qquad (46)$$

Where, $y = \frac{C_1}{C_2}$, $k_0 = \frac{kTR}{\omega^2 \omega_0^2 \left(j\frac{Z}{\omega}\tan\left(\frac{\omega}{v_P} l\right)\right)^2 V_{cc}^2 C_2^2}$, $k_1 = \frac{qI_c g_m^2 + \frac{K_f I_b^{AF}}{\omega} g_m^2}{\omega^2 \omega_0^4 \left(j\frac{Z}{\omega}\tan\left(\frac{\omega}{v_P} l\right)\right)^2 V_{cc}^2}$, $k_2 = \omega_0^4 (\beta^+)^2$,

$k = \frac{k_3}{k_2 C_2^2}$

The phase noise equation (46) above can be differentiated to determine the best possible phase noise. This is a better approach then to depend on the optimizer of the HB simulator.

$$\frac{\partial \left| \phi^2(\omega, y, k) \right|}{\partial y} \Rightarrow 0 \qquad (47)$$

$$\frac{\partial}{\partial y}\left[\left\{\left[\left[k_0 + \left(\frac{k^3 k_1 \left[\frac{Y_{21}^+}{Y_{11}^+}\right]^4 [y]^{4p}}{\left[Y_{21}^+\right]^6 [y]^{6q}}\right)\left(\frac{1}{(y^2+k)}\right)\right]\left[\frac{[1+y]^2}{y^2}\right]\right\}\frac{Q_{max}^2}{Q_0^2}\right]_{y=m} \Rightarrow 0 \qquad (48)$$

For minimization of noise and regime of y, we leave this task and its calculation and validation to the interested reader; detailed information can be found in [2, pp. 181].

The next step is to think about improving quality factor of resonator tank circuit and techniques to minimize the phase noise for modern oscillators (narrowband and wideband voltage controlled oscillator) for current and later generation of communications systems.

Validation

Modern expensive harmonic balance based simulators such as ADS from Agilent and Serenade from Ansoft, part of Ansys, can be used to determine the resulting phase noise with a high degree of accuracy, about 2dB typical error. By introducing a novel mathematical method as shown above, based on measured large signal parameters, the correct phase noise can be calculated, relative to the simulation. Our test case is figure 12.

A similar 800MHz VCO from the standard listing of Synergy microwave was used to further validate the method. The analysis with the Harmonic balance program indicates the predicted phase noise (Figure 30). The flicker corner frequency is about 1kHz, though it is not distinctly visible due to high Q resonator in use and the the phase noise at 10 KHz offset is -132.14 dB/Hz.

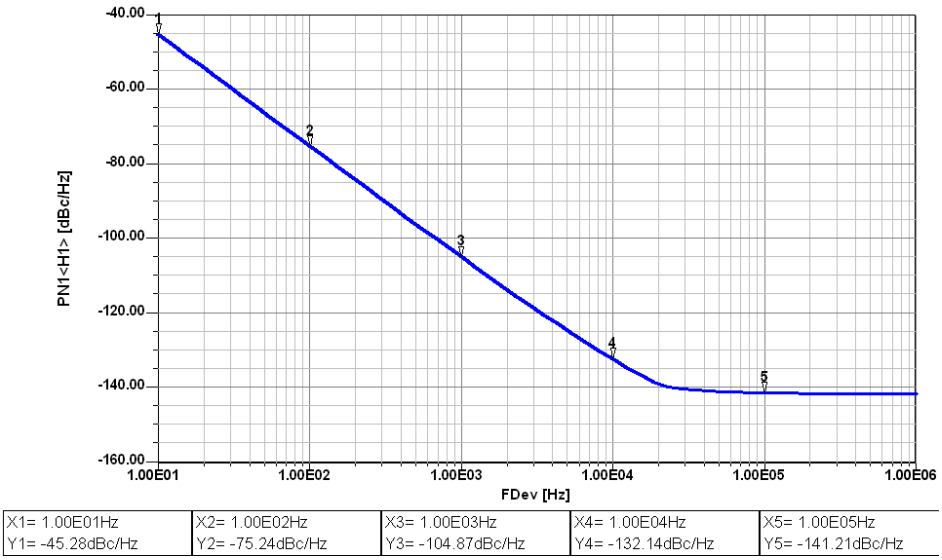


Figure 30: A CAD Simulated (Ansoft Designer) phase noise plot for 1 GHz oscillator

The simulator fails to give any change in phase noise above 1MHz offset due to numerical problems of the simulator. The calculation based on equation 46 predicted the phase noise shown in figure 31. It can show a flicker corner at 1KHz and the predicted phase noise at 10KHz is around 130.5dBc/Hz. The 10MHz offset phase noise calculates to about -170dBc/Hz.

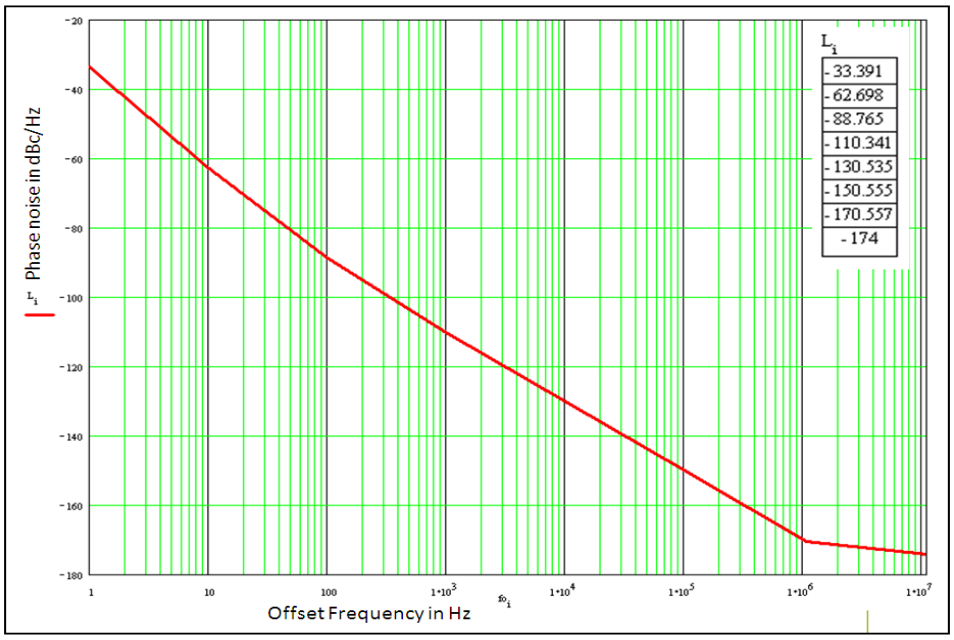


Figure 31: Predicted phase noise based on equation 46 using MathCad.

The measured response of this unit is shown verified on R&S FSUP network analyzer and the Agilent Network analyzer E5052A. Shown in Figure 32-A (R&S FSUP measurement) and Figure 32-B (Agilent E5052A measurement). This data matches well with the calculation but does not agree with the simulation at 1MHz and further out.

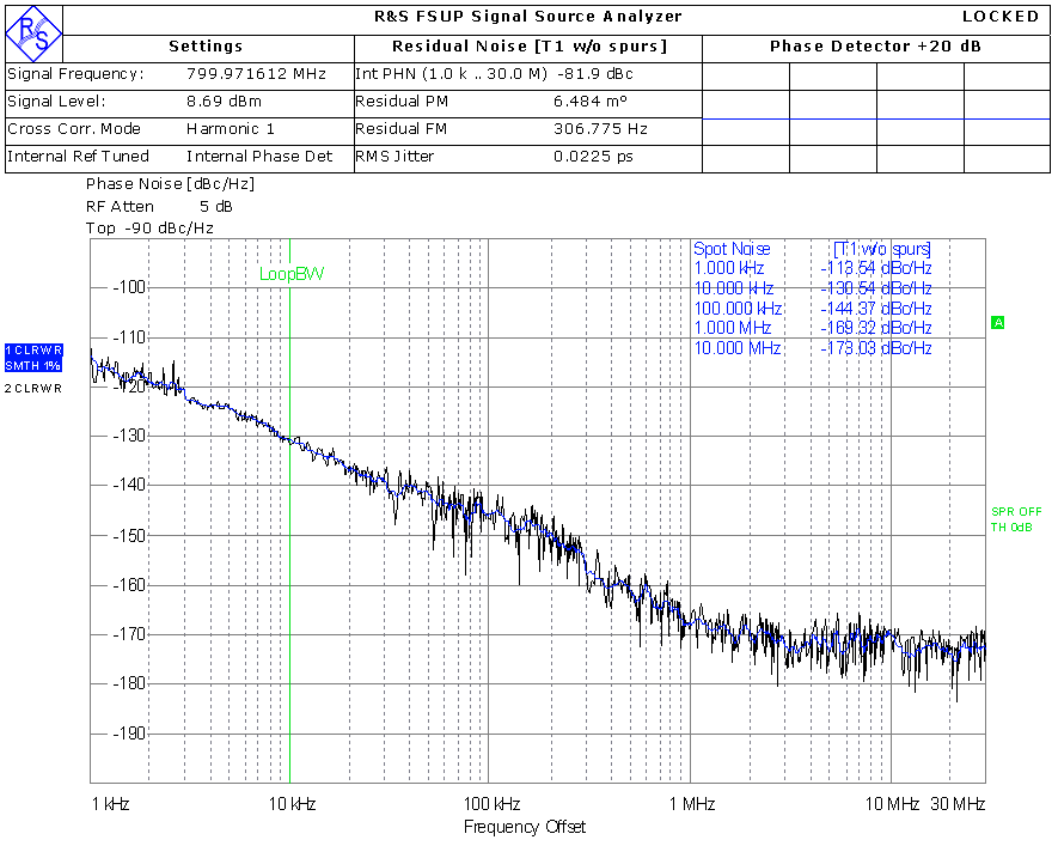


Figure 32-A: Measurement of the unit with an R&S FSUP analyzer.

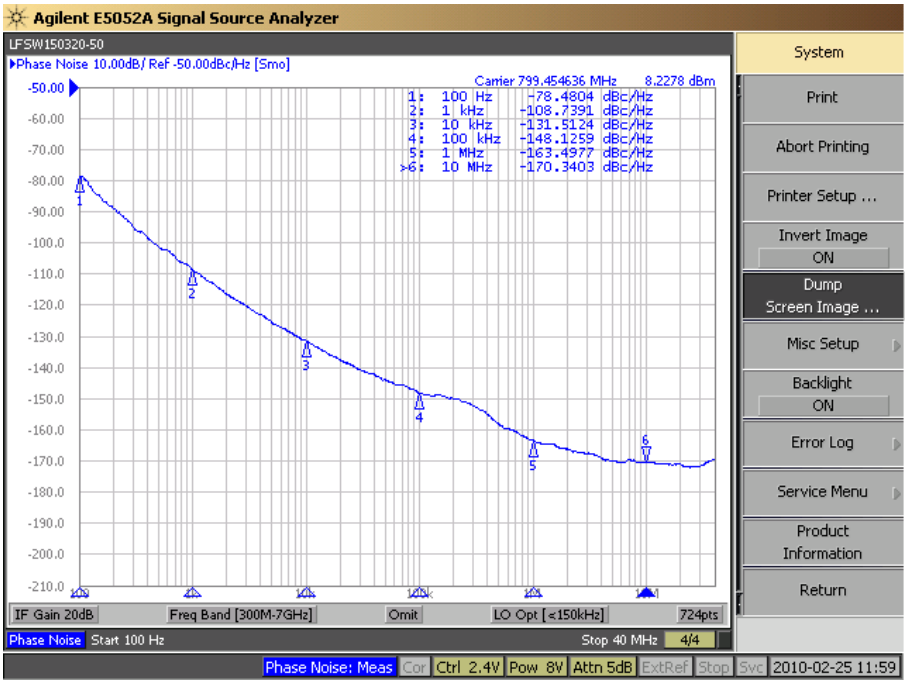


Figure 32-B: Measurement of the Unit with an Agilent E5052A analyzer.

Acknowledgements:

We would like to thank Rucha Lakhe for helping us to put this report together.

Reference:

1. Randall W. Rhea, *Discrete Oscillator Design,* Artech House, Boston, 2009.
2. U.L. Rohde, A.K. Poddar, and G. Boeck, *The Design of Modern Microwave Oscillators for Wireless Applications: Theory and Optimization*, Wiley, New York, 2005.
3. U. L. Rohde, *Transistoren bei höchsten Frequenzen,* Verlag Für Radio-Foto-Kinotechnik, Germany 1965, page 92.
4. H. Zinke and Brunswig, Volume 1 *Lehrbuch der Hochfrequenz-technik,* New York 1986 page 77.
5. G. Vendelin, A. Pavio and U. L. Rohde, *Microwave circuits design using Linear and nonlinear techniques,* Wiley, New York, 2005 page 577.
6. Michael M.Driscoll, Two-Stage Self-Limiting Series Mode Type Quartz Crystal Oscillator Exhibiting Improved Short-Term Frequency Stability, IEEE Transactions on Instrumentation and Measurement June 1973, pp130 – 138, modified by Chris Bartram, *Notes on the Driscoll VHF Overtone Crystal Oscillator and a New Low-Noise VHF Crystal Oscillator Topology* 2008.

Mathematical stability problems in modern non-linear simulations programs

Ulrich L. Rohde , Rucha Lakhe

Synergy Microwave Corporation

e-mail: ulr@synergymwave.com

The use of nonlinear components such as bipolar transistors, GaAs FETs, and microwave diodes makes it necessary to predict large signal-handling performance. The traditional tools to do this were the SPICE approach and Volterra series expansion. The SPICE program is a program operating solely in the time domain. SPICE is an outstanding workhorse for dc analysis as a function of bias and temperature and transient analysis. The drawbacks of SPICE are (1) the lack of an optimizer; (2) the lack of distributed elements such as tee junctions, crosses, and others; and (3) the slow execution speed related to the time-domain approach. [1-3]

Another approach that has been tried is Volterra series expansion. This approach is a simulation where the actual computation time is somewhat independent of the values of the components used in a circuit. However, once the number of harmonics goes up, Volterra series expansion also becomes very time consuming. The Volterra series can be regarded as a nonlinear generalization of the familiar convolution integral.

The Volterra series also has the limitation that the degree of nonlinearity must be mild, as the representation otherwise requires an intractably large number of details for adequate modeling. The recently developed harmonic balance method avoids many of the time consuming mathematical approaches mentioned previously. This method is a hybrid time- and frequency-domain approach which allows all the advantages of a time-domain device model, combined with the strength of the steady-state frequency-domain technique, to be presented in the lumped and distributed circuit elements in which the device is embedded. The time-domain model can be completely general, thus bypassing complicated determination of coefficients by curve fitting over different bias levels.

How does the modern non-linear Program Work? [1]

For a fixed circuit topology (analysis case), the frequency domain is passed through only once; the admittance matrix of the linear subnetwork is computed and stored for subsequent use. In the timedomain path, the state-variable harmonics are first used to compute the corresponding time-domain waveforms. As mentioned earlier, these are fed to nonlinear device equipment to produce the time-domain device port voltages and currents. Voltage and current harmonics are then described by one- or two-dimensional fast Fourier transforms (FFTs) for the cases of single-tone and two-tone excitation, respectively. The voltage harmonics are used to generate "linear" current harmonics via the linear subnetwork admittance matrix. The two sets of current harmonics are finally compared to produce individual harmonic balance errors and a combined (global) harmonic balance error to be used in a convergence test.

In well-conditioned cases (e.g., FET circuits), a standard Newton-Raphson iteration may be used successfully as an update mechanism even though no starting-point information is available (i.e., if zero initial values are assumed for all harmonics). In such cases the harmonic balance errors are used via a simple perturbation mechanism to generate a Jacobian matrix. The latter is then inverted and applied to the error vector to generate the updated harmonic vectors. The algorithm is fast and accurate.

For circuits containing strongly nonlinear devices such as microwave diodes, a simple Newton iteration may sometimes fail to converge. To overcome this difficulty, Designer 5 incorporates a second iteration scheme based

on a variable metric algorithm (quasi-Newton iteration), which is slower although considerably more robust than the regular Newton method.

In ill-conditioned cases, the quasi-Newton iteration may be used to approach the required solution. After this has been done to a satisfactory extent, automatic switchover to Newton iteration takes place, so that the approach solution can quickly be refined to any desired accuracy.

When circuit optimization is requested, the algorithm flowchart is modified. Harmonic balance errors are computed in the same way, but now the variable circuit parameters are also updated and the linear subnetwork admittance is computed at each iteration. An objective function is defined as a combination of harmonic balance error and a contribution arising from the electrical specifications. Such an objective is then minimized by the variable metric algorithm until a minimum close enough to zero is reached. Circuit parameters and state-variable harmonics are updated simultaneously, thus avoiding the nesting of nonlinear analysis and circuit optimization loops. Designer 5 is a general tool using the harmonic balance method for microwave. The harmonic balance method is a generic mathematical approach and is used for the first time in a commercial CAD software. Today the program of interest is Ansys's Designer 5. These modern programs are written mathematically so well that stability problems in oscillators do not occur in amplifiers.

Harmonic Balance Analysis (HBA)

Harmonic balance analysis is performed using a spectrum of harmonically related frequencies, similar to what you would see by measuring signals on a spectrum analyzer. The fundamental frequencies are the frequencies whose integral combinations form the spectrum of harmonic frequency components used in the analysis. On a spectrum analyzer you may see a large number of signals, even if the input to your circuit is only one or two tones. The harmonic balance analysis must truncate the number of harmonically related signals so it can be analyzed on a computer.

Analysis parameters such as **No. of Harmonics** specify the truncation and the set of fundamental frequencies used in the analysis. The fundamental frequencies are typically not the lowest frequencies (except in the single-tone case) nor must they be the frequencies of the excitation sources. They simply define the base frequencies upon which the complete analysis spectrum is built.

A project for harmonic balance analysis must contain at least the following: A top-level circuit, at least one nonlinear active device, and a frequency specification (including the number of harmonics of interest). Designer 5 has five categories of harmonic balance analysis:

- Single-tone analysis (single RF signal)
- Two-tone intermodulation analysis (two RF signals)
- Two-tone mixer analysis (one RF signal and one LO signal)
- Three-tone intermodulation analysis (three RF signals)
- Three-tone mixer analysis (two RF signals and one LO signal).

Formation of the Harmonic Balance Equations

Harmonic balance analysis involves the periodic steady-state response of a fixed circuit given a pre-determined set of fundamental tones [4, 5]. The analysis is limited to periodic responses because the basis set chosen to represent the physical signals in the circuit are sinusoids, which are periodic. The Fourier series is used to represent these signals. In the single-tone case, a signal is given by:

$$x(t) = \sum_{k=-NH}^{NH} \left(X_k e^{jk\omega_0 t} \right) \quad (1)$$

where $X_k = X_{-k}^*$, w_o is the fundamental frequency and NH is the number of harmonics chosen to represent the signal.

In harmonic balance, the circuit is usually divided into two subcircuits connected by wires forming multiports. One subcircuit contains the linear components of the circuit and the other contains the nonlinear device models as shown in the figure. The linear subcircuit response is calculated in the frequency domain at each harmonic component ($k*w_o$) and is represented by a multiport Y matrix. This is the function performed by linear analysis.

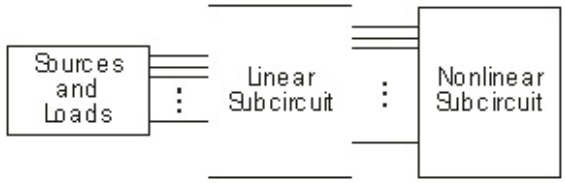


Figure 1: General block diagram showing the two subcircuits in the harmonic balance program.

Separation of the linear and nonlinear subcircuits

The nonlinear subcircuit contains the active devices whose models compute the voltages and currents at the intrinsic ports of the device (parasitic elements are linear and absorbed by the linear subnetwork). The port voltages (*v*) and currents (*i*) are analytic or numeric functions of the device state variables (*x*). Often the state variables represent physical voltages such as diode junction voltage or FET gate voltage, but are not restricted to physical quantities. The port voltages and currents are often functions of the time derivatives of the state variables (when a nonlinear capacitor is involved) and of time-delayed state variables (such as a time-delayed current source). Generally, the nonlinear device equations are of the form:

$$v(t) = \Phi \left[x(t), \frac{dx}{dt}, K, \frac{d^n x}{dt^n}, x(t-\tau) \right] \quad (2)$$

$$i(t) = \Phi \left[x(t), \frac{dx}{dt}, K, \frac{d^n x}{dt^n}, x(t-\tau) \right] \quad (3)$$

The device state variables, port voltages, and currents are transformed to the frequency domain using the discrete Fourier transforms as **X**, $\mathbf{V}_k(\mathbf{X})$ and $\mathbf{I}_k(\mathbf{X})$, respectively. Kirchhoff's current law is applied to the interface between the subcircuits at each harmonic frequency:

$$Y_k V_k(X) + I_k(X) + J_k = 0 \quad (4)$$

where $\mathbf{J}_k$ are the Norton equivalents of the applied generators. This constitutes the harmonic balance equations at each harmonic frequency. The object of the analysis is to find the set of state variables, **X**, to satisfy equation 4.

When the analysis begins, the state variables are typically set to zero and the left side of equation 4 is non-zero. We can write an error vector:

$$E_k(X) = Y_k V_k(X) + I_k(X) + J_k \quad (5)$$

whose Euclidean norm $\mathbf{E}^t\mathbf{E} = ||\mathbf{E}||$ is called the Harmonic Balance Error (HBE). If the HBE is reduced below a tolerance, we say that equation 4 is satisfied and a solution has been obtained.

Solving Methods

The process of solving the harmonic balance equations is an iterative one. An estimate of **X** is inserted into equation (5), **E** is calculated and if it is not below the tolerance then a new value of **X** must be determined and

tried. Each such loop is termed as an iteration. There have been several methods used in the past to determine new values of **X** and two that have proven to be the most general and efficient are discussed here.

The state variables, **X**, and harmonic balance residuals, **E** are complex valued. In practice these are decomposed into their real and imaginary parts so that the number of real unknowns in **X** is ND*(2*Nt+1) where ND is the total number of nonlinear device ports and Nt is the number of frequency components (=NH for single tone analysis). Now we can write **E**(**X**)=**0** as a Taylor series expansion truncated after the first derivative term:

$$E(X) = 0 \approx E\left(X^{(n)}\right) + J\left(X^{(n)}\right)\left(X - X^{(n)}\right) \quad (6)$$

where **J**, the Jacobian matrix, is the first derivative matrix of **E** with respect to **X** and superscript n indicates the current iteration. Solving for **X** and using this for the next trial:

$$X^{(n+1)} = X^{(n)} - J^{-1}\left(X^{(n)}\right)E\left(X^{(n)}\right) \quad (7)$$

This is the Newton-Raphson update method where the last right-hand term is the update. This method works in one iteration if the set of equations is linear, but will take an unknown number of iterations if nonlinear. Often the update is reduced by a factor called the Newton damping factor so the method takes smaller steps each iteration. Convergence to a solution is not guaranteed and the iterates may diverge if not controlled. Designer 5 uses enhanced versions of the Newton-Raphson method to improve convergence and speed.

Designer 5 uses an algorithm that dynamically changes the Newton damping factor during solving based on the rate of convergence. If the solver has trouble converging, the factor will be made smaller to improve convergence. If it has been reduced by more than a predetermined factor, the solver will stop and an error will be reported.

An important aspect to note is the size of the Jacobian. If **X** contains ND*(2*Nt+1) elements, then **J** contains this number squared. As a practical example, if ND=10 (5 FETs) and Nt=4, then there are 8100 entries in **J** which takes 63 kBytes. This is relatively small, but Nt becomes much larger when multi-tone excitation is considered.

Some of the controlling functions are made accessible through the **CTRL** block in the project (see the *Control Blocks* chapter). The HBE tolerance can be changed from its default by: **HBTOL** ***x*** where ***x*** is the tolerance per device port per frequency component.

The absolute harmonic balance error allowed is scaled by the number of device ports and number of frequency components so that large circuits with many frequency components meet HBE criteria similar to those of small circuits. The default for HBTOL is 1.0×10^{-6}. For the case of 2-tone intermodulation analysis and 3-tone analysis, the allowed harmonic balance is also scaled by the relative currents of the circuit. This reduces the allowed error (effectively reducing HBTOL) to provide better accuracy of the intermodulation products.

The number of allowed iterations before the program stops can be changed from its default value of 400 by: **MAXITER** *n,* where ***n*** is an integer.

Multi-tone Analysis

The discussion above was based on single-tone analysis for conceptual simplicity. Multi-tone analysis is simply an extension of single-tone [6,7,8]. In the single-tone case, a circuit is excited with an RF source and harmonics of that source are produced by the nonlinearities of the circuit. The set of harmonics, the frequency of excitation and DC are called the *spectrum* of the analysis. The single-tone spectrum is defined as: $S_1 = k*f_0$ where k=0, 1, ... NH and f_0 is the fundamental frequency. In multi-tone analysis the spectrum is modified to include the harmonic products of each fundamental tone. The harmonic products are just integer functions of the fundamental frequencies and indicate the allowed “bins” for power conversion within a circuit. The rest of the harmonic balance analysis is exactly the same.

The conversion between time-domain waveforms and Fourier coefficients is accomplished by the discrete Fourier transform in single-tone analysis. For each additional fundamental tone, a dimension is added in the transform. This allows efficient computation between domains, but becomes CPU-intensive when more than three-dimensions are encountered.

Local Oscillator Spectrum Initialization of Mixer Circuits

For mixer analysis cases where the primary interest is the conversion gain and the RF signal powers are small compared to the LO, the circuit can be analyzed using the LO signal only and the conversion gain is determined using small-signal (linear) frequency-conversion methods. This is performed using the Small-Signal Mixer Analysis option.

For cases where the RF signal power is not insignificant compared to the LO, a full mixer spectrum must be used. Compression of the conversion gain due to high RF power can then be analyzed. Here, the mixer problem can be divided into two parts to help speed the analysis. Firstly, the LO signal is analyzed using single-tone analysis; the RF signal is turned off. Single-tone analysis is usually very fast compared with a full two-tone analysis. Once the LO signal spectrum is found, the results are used to initialize the full mixer spectrum and the RF signal is turned back on. The full spectrum is then analyzed.

This method is most useful for three-tone mixer problems, due to the large number of spectral components used in the analysis. The primary use of the three-tone mixer analysis is to determine the intermodulation products of the IF products. This precludes the use of small-signal mixer analysis (since the intermodulation products cannot be determined using linear frequency conversion methods), but the RF signals are generally small compared to the LO. By solving the LO problem first, which is the primary nonlinear problem, and then introducing the RF signals, the analysis time can be reduced by a factor of about three. The actual time reduction depends on the circuit, the RF power levels, and the conversion gain.

Using the LO harmonic spectrum to initialize the full mixer spectrum is the default for three-tone analysis. The option is not the default for two-tone analysis, because significant time improvements have not been observed.

Number of Spectral Components and Reduced Spectrum Option

The number of spectral components considered in each type of analysis is related to the number of fundamental tones and the nonlinearity specified. The tables below list the number of spectral components for several nonlinearities considered in two-tone and three-tone analyses. The reduced spectrum option removes selected spectral components where significant harmonic power is not expected. The results of the analysis will not degrade at low power levels, but may yield different results for high power levels, depending on the circuit. Usually, the difference in results is negligible for practical cases.

The reduced spectrum option is especially useful for three-tone mixer analysis where the primary objective is to obtain the intermodulation intercept point with the IF. Low RF signal power levels are used and the analysis results are unaffected by the reduced spectrum option (the number of LO harmonics is not affected).

Number of Spectral Components (excluding DC) for Two-Tone and Three-Tone Intermodulation Analysis				
nonlinearity INTM m	**two-tone Full (default)**	**two-tone Reduced**	**three-tone Full (default)**	**three-tone Reduced**
3	12	8	31	21
4	20	12	64	31
5	30	22	115	79

6	42	30	188	115
7	56	44		209
8	72	56		
9	90	74		
10	110	90		

Number of Spectral Components (excluding DC) for Two-Tone Mixer analysis						
	#SB (M2) = 1		**#SB (M2) = 2**		**#SB (M2) = 3**	
#LO (M1)	**Full (default)**	**Reduced**	**Full (default)**	**Reduced**	**Full (default)**	**Reduced**
2	7	7	12	12	17	17
4	13	13	22	18	31	23
6	19	19	32	24	45	29
10	31	31	52	36	73	41
15	46	46	77	51	108	56
20	61	61	102	66	143	71
25	76	76	127	81	178	86
30	91	91	152	96	213	101

Note: #LO is the number of local oscillator harmonics; #SB is the number of RF sidebands

Number of Spectral Components (excluding DC) for Three-Tone Mixer analysis				
	INTM (M2) = 3		**INTM (M2) = 5**	
# LO (M1)	**Full**	**Reduced (default)**	**Full**	**Reduced (default)**
2	62	42	152	112
4	112	52	274	122
6	162	62		132
10	262	82		152
15		107		177
20		132		202
25		157		227
30		182		252

Notes: Entries that have been filled-in can be simulated

#LO is the number of local oscillator harmonics; INTM is the intermodulation order

The total number of spectral components grows very quickly with the level of nonlinearity and number and fundamental tones.

• Two-tone or three-tone intermodulation spectrum: The highest order group of spectral components, except those in the fundamental group (the intermodulation products), are ignored. In this case, the *n* in REDUCED*n* is ignored.

• Two-tone mixer analysis: All sidebands except the first sideband above the *n*th local oscillator harmonic will be ignored.

• Three-tone mixer analysis: All sideband groups at or below the *n*th local oscillator harmonic will be the same as the reduced two-tone intermodulation spectrum; all the sidebands above the *n*'th local oscillator harmonic will contain the two fundamental frequencies only.

Understanding the reduced spectrum is a little complicated. If the analysis is run with several reduced spectrum values and the spectrums are compared, then a better understanding of the spectral selections will be attained. Many studies were conducted and showed that the (default) reduced spectrum option for three-tone mixer intermodulation analysis affected analysis accuracy only slightly.

Sparse Jacobian Techniques

The Jacobian matrix, when properly arranged, can be treated as a sparse matrix by pre-setting some entries to zero [6]. The physical reason for doing this is that most of the power transfer takes place between the harmonic frequencies of the fundamentals and much less takes place between the other frequencies in the spectrum. We can therefore set these derivatives to zero within the Jacobian. When this criterion is not met, the band of non-zero entries is widened to include cross-harmonic terms.

Because the Jacobian structure is properly arranged, sparse matrix techniques are efficiently employed. General purpose sparse matrix solvers that analyze the sparsity structure are avoided and specialized solvers can be used that are much more efficient. Designer 5 automatically sets the bandwidth of the sparse tridiagonal matrix and dynamically alters it if the nonlinearity of the circuit is too great for the sparse assumptions. In this way the simulator achieves convergence using the minimal amount of computation time and memory that is possible for a given problem. For circuits with many devices under multi-tone operation, the CPU time may be decreased by a factor of 40.

A control parameter is made available to override the initial default sparsity parameter that controls the Jacobian bandwidth. The initial setting is 0 and can be changed by:

DIAG *n*

where ***n*** will be the initial sparsity parameter. Typical values range between 0 and 6. The sparsity parameter will still be dynamically altered during execution if needed. If ***n*** is greater than Nt/3 (Nt is the number of frequencies), then the program will use the full Jacobian. If only the full Jacobian is desired, then set n to a large number.

Using a sparse Jacobian does not affect the final values or accuracy of the results. It will only affect the convergence properties of the particular problem.

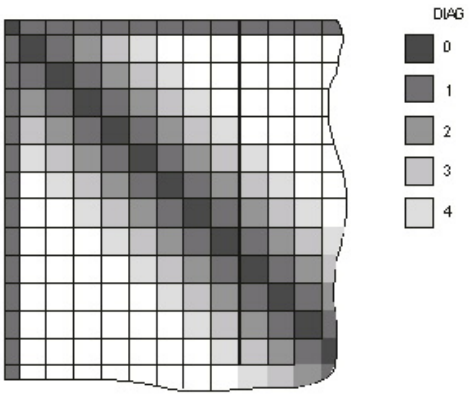


Figure 2: Sparse Jacobian block structure

Iterative Newton Method

One of the short comings of harmonic-balance methods is the large memory requirements when a circuit has many nonlinear devices and/or multi-tone analysis is needed. The Jacobian system matrix grows large and must be stored and factored. Sparse methods may not be enough to keep the problem within the memory bounds and acceptable computational resources of desktop computers. Designer 5 uses a technique that efficiently solves large systems of equations without direct factorization of the system matrix. In this way, there is no simplification or approximation made to the problem and the full accuracy of the conventional harmonic-balance method is completely maintained. The convergence and power-handling capabilities of conventional harmonic-balance analysis are also fully maintained. The method is completely automatic and does not require any user intervention. An internal software switch detects when the new method should be used and automatically invokes it.

A brief summary of the method and its advantages is given: Conventional harmonic-balance computes and stores the Jacobian matrix. The iterative solution of the harmonic-balance equations requires factorization of the Jacobian to obtain updates of the circuit voltages. As the number of nonlinear devices in the circuit increases and the number of spectral components used to analyze the circuit increases, the Jacobian matrix can become very large, requiring tens or hundreds of megabytes of storage and several minutes of CPU time to factor it. The calculation and factorization of the Jacobian typically occurs several times during a single harmonic-balance solution. The new method, based on an iterative approach known as *Krylov Subspace Methods*, avoids direct storage and factorization of the Jacobian. Rather, a series of matrix-vector operations replaces the full storage and factorization steps while retaining full numerical accuracy.

Observed speed-up factors depend on the number of nonlinear devices in the circuit and the number of spectral components used in the analysis as well as the convergence properties of the harmonic-balance algorithm. Speed improvements over conventional harmonic balance analysis from 2x to 10x for circuits consisting of a few transistors under two and three-tone excitation have routinely been observed. A circuit containing 20 FETs under three-tone analysis exhibited a speed improvement factor of 30x. Memory requirements have also been tremendously reduced. The 20 FET circuit originally required >200MB and now will analyze with 64MB. As the circuit becomes more “complex” the new methods provides better speed and memory improvements.

Designer 5 Outputs

During analysis, Designer 5 generates a number of output files that are used to store textual, graphical and initialization information. The files generated are: **myfile.aud**

The audit file contains textual information about the analysis. In its basic form it contains the final results of the network functions. Additional information can be requested by setting the verbosity flag in the control block as:

VERBOSE *n*

where $0 \leq \boldsymbol{n} \leq 4$. The higher the verbosity number is, the more output that is generated about the final and initial points at each sweep step.

Sweeping Frequency, Power and Voltage Sources

Each source in the design can be swept in amplitude. Also, the tones defined for the analysis can be swept. When more than one source or frequencies are swept, an ambiguity arises as to the order of precedence. The following rules apply in the cases of multiple sweeps:

1) When more than one source is swept and no frequencies are swept, then the sources sweep in unison. That is, each source is stepped at the same time. This is a one-dimensional sweep.

2) When more than one frequency is swept and no sources are swept, then the frequencies sweep in unison. This is also a one-dimensional sweep.

3) When frequencies and sources are both swept, the program performs a two-dimensional sweep where the sources are swept in the innermost loop. A matrix results where the source sweep is the most rapidly changing index. An exception to this case is during noise analysis, where the swept frequency deviation will be the innermost loop.

Generating Large-Signal S-Parameters

Since large-signal S-parameters are poorly defined, but widely used, we will show two methods of generating them. If your definition of large-signal S-parameters is different, you can redefine the example to suit your own needs. Here lies the ambiguity as to what one means by large-signal S-parameters. It will depend on the specific application and must be tailored in each case. Presented below is one interpretation:

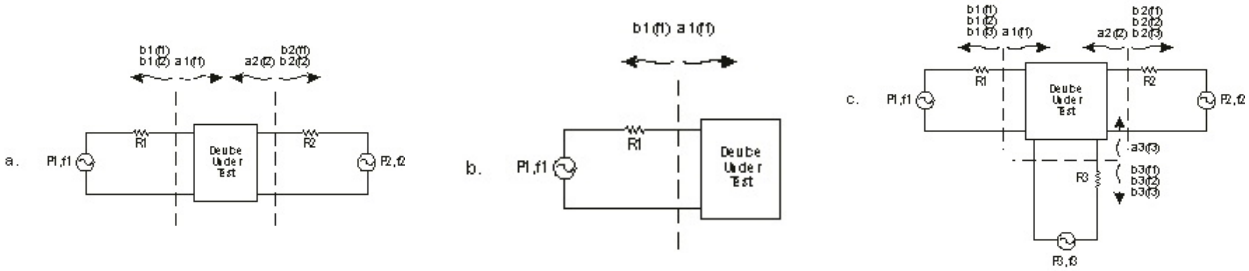


Figure 8: Schematic diagram: Generating large-signal *S*-parameters

The calculation of *S*-parameters in the large-signal regime is not as straightforward as it is in the linear, small-signal regime. The large-signal *S*-parameters are dependent on the power of the excitation sources at each external circuit port as well as the circuit bias and terminations. Guidelines will be given here on using Designer 5 to generate large-signal *S*-parameters, but the proper use of these S-parameters in circuit design is up to you.

Consider a two-port circuit whose large-signal *S*-parameters are desired. If we apply a source at port 1 with port 2 terminated, we could measure the reflected and transmitted waves, and conversely for a source applied to port 2 [10, 11]. However, this assumes that when the device under test is actually used, it will be terminated in the same impedance as it was tested. This is rarely the case. Typically the device is embedded in some matching network which presents a complex impedance to the device. Therefore, the operating regime of the device will change and its large-signal *S*-parameters will be altered.

We could then hypothesize that a source can be placed at each port and the traveling waves could be measured at each port. The problem here is that it is not possible to distinguish between the reflected wave at a port and the transmitted wave due to the source at the other port because the sources are the same frequency. If we perturb

the frequency of one of the sources, then the reflected and transmitted waves due to each source can be resolved. This, however, requires a two-tone analysis. The situation is illustrated in the diagram for a two-port device under test.

The difference in frequency between the two sources can be made small, on the order of 0.001%. This is recommended for circuits of large *Q*. Typically the difference used is about 0.1% because the S-parameters of the device under test do not change rapidly with frequency.

This example shows some of the inconsistencies associated with large-signal *S*-parameters. For example, what happens to the power that is converted to other harmonic products? These will depend on the bias point, harmonic terminations, etc. In practice, the powers measured include all harmonic powers incident on the detector, whereas in the calculations we can pick out the precise fundamental powers. Also, we chose incident power levels as 10dBm and 8dBm, but how do we know if these are correct until *after* the design is done? There are several approximations like these that are assumed to be small when using large-signal *S*-parameters in active circuit design. Nonetheless, these parameters persist in design and can be computed using Designer 5.

In some cases where it can be approximated that one or more ports will be conjugately matched so a source doesn't need to be present there, higher port parameters can also be computed using repetitive analyses.

Other so-called conversion parameters can be computed. For example, if a mixer conversion matrix is desired between the RF and IF frequencies, the corresponding transmission parameters can be computed using TG between the proper harmonic numbers. The reflection coefficients can be found by using RL at the source ports and source harmonics.

Algorithm for Single-Tone Y-parameters Evaluation of Nonlinear Systems [12]:

For single-tone bandpass analysis, it is assumed that a nonlinear element's measurements are obtained when both the source and load have a 50Ω termination and the input is a bandpass single-tone signal. The measurements obtained are directly related to the large signal S-parameters of the two-port nonlinear element and the power available at the input and output ports.

For a given single-tone input, we can refer to large signal S-parameters as the operating point of the nonlinear two-port element when operating independently and terminated in 50Ω. One would certainly expect this operating point to change when this nonlinear element is embedded in a nonlinear topology system (as in Figure 3) composed of linear and nonlinear components.

The new operating point (i.e., large-signal S-parameters) is determined for each nonlinear element using an iterative algorithm where the levels of the incident powers at both ports are interpolated iteratively until the algorithm converges to the actual operating point. This nonlinear frequency domain iterative algorithm accounts for all nonlinearities and inter-stage mismatches in the system.

For the multi-channel nonlinear topology in Figure 3, it is always assumed that parallel nonlinear channels connected to the same linear electrical subsystem are not coupled (i.e., non-interacting). This, in simple terms, implies that signals traveling in one nonlinear path do not spill over into the other nonlinear path (by virtue of the S-parameters describing the linear electrical subsystem).

This effectively implies that the signals in nonlinear channels are not coupled, and as a result, the impedances Zs_n, Zin_n, $Zout_n$, and Zl_n for the n^{th} nonlinear element ($1 \le n \le N$) in Figure 3 are well defined. With that important assumption, the iterative algorithm used for evaluating the operating point for each nonlinear element proceeds as follows (refer to Figure 3):

1. Assume an initial guess of $P1_n(0) = P2_n(0) = 0$for the first iteration (k=0) for the n^{th} nonlinear element ($1 \le n \le N$).

2. Calculate the power-dependent S-parameters for the n^{th} nonlinear element ($1 \le n \le N$) at the k^{th} iteration ($k \ge 0$). For the first iteration (k=0), this would basically yield the n^{th} nonlinear element small signal S-parameters since the initial estimate for the incident powers is zero.

3. Calculate the entire system's *Y* matrix, the impedances shown in the above figure $Zs_n(k)$, $Zin_n(k)$, $Zout_n(k)$, and $Zl_n(k)$, and the nodal voltages at the input and output ports of the n^{th} nonlinear element at the k^{th} iteration.

4. Recalculate the incident powers at each port with ($1 \le n \le N$) and ($k \ge 0$) according to

$$P1_n(k+1) = \frac{\left|\frac{v1_n(k)[zin_n(k) + Zs_n(k)]}{Zin_n(k)}\right|^2}{4Re\{Zs_n(k)\}}$$

$P1_n(k+1)$ = The power available at the input port of the n^{th} nonlinear element at the k^{th} iteration when the output port is terminated in $Zl_n(k)$.

and

$$P2_n(k+1) = \frac{\left|\frac{v2_n(k)[Zout_n(k) + Zl_n(k)]}{Zout_n(k)}\right|^2}{4Re\{Zl_n(k)\}}$$

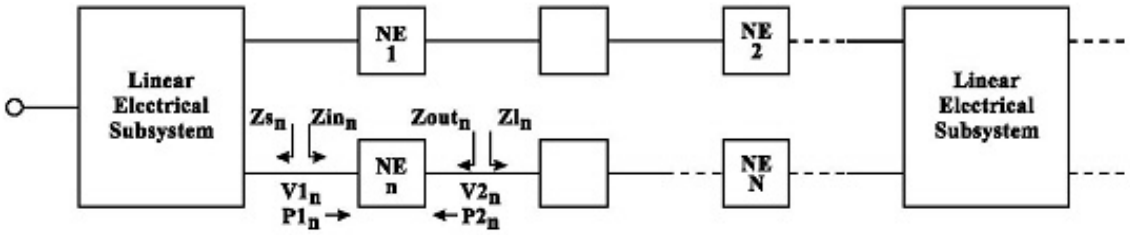


Figure 3: General non-linear electrical topology containing N non-linear elements (NE= nonlinear elements)

$P2_n(k+1)$ = The power available at the output port of the n^{th} nonlinear element at the k^{th} iteration when the input port is terminated in $Zs_n(k)$.

5. Form the error function

$$Error(k) = \frac{\sum_{n=1}^{N}\sqrt{[P1_n(k+1) - P1_n(k)]^2 + [P2_n(k+1) - P2_n(k)]^2}}{\sum_{n=1}^{N}\sqrt{[P1_n(k)]^2 + [P2_n(k)]^2}}$$

The algorithm is assumed convergent if the condition $Error(k) \le N \times 10^{-6}$ is met. If this convergence condition is not satisfied, steps 2-5 above are repeated until the algorithm converges. If the algorithm fails to converge after 30 iterations, an error in analysis message will be displayed.

Linear Electrical Discrete Time Simulation

During signal analysis, all electrical components and sub-designs are converted to time domain behavioral models (i.e., unidirectional models). This time domain model is extracted from the corresponding frequency response evaluated during the course of the simulation.

The extracted time domain behavioral models will typically contain the transient, steady state, and noise responses of the electrical sub-design (all in accordance with the frequency-domain signal and noise response of the electrical sub-design as well as impedance mismatches within the sub-design). Therefore, the processing of modulated signals through electrical components/sub-designs will include the transient, steady state, group delay, and noise effects.

There are two techniques available for discrete time simulation of linear electrical components and sub-designs: Convolution and Impulse Invariance.

Setting Discrete Time Simulation Control Parameters

For the discrete time simulation of a mixed-mode communications system, the user must carefully select the following control parameters:

1. $t_s = 1/f_s$= Simulation time step (which may assume different values at different points in the system)
2. *MIN_BW* = Minimum bandwidth of an electrical component or sub-design in the complete mixed mode system.
3. *MAX_RATIO* = Ratio of a local maximum to the global maximum in the impulse response.

Step 1: Choosing the Simulation Time Step

Choosing the simulation time step t_s is not a direct process in discrete time signal analysis. For a typical wireless communications system, the user always begins by setting the bit rate for binary data sources (or sampling rate if a source happens to be a waveform source).

Binary source components in Designer 5 (e.g., BSRC) have a parameter that determines their output bit rate. In a typical baseband modulation process, binary bits (at a user-defined bit rate) are mapped unto information symbols (to yield a given symbol rate). Each symbol is then represented by a user-selected number of samples (typically by upsampling or repeating each symbol) to finally yield a desired sampling rate f_s and the corresponding simulation time step of $t_s = 1/f_s$. These samples are then filtered to yield the discrete baseband modulation waveform $S(nt_s)$ described above.

When choosing the simulation time step $t_s$caution should be exercised to preserve the Nyquist criterion for the signal $S(t)$. In other words, the user must ensure that the discrete signal $S(nt_s)$ (at any point in the system) has at least a Nyquist sampling rate or higher, where the Nyquist sampling rate is equal to twice the bandwidth of the continuous signal $S(t)$.

At the same time, the simulation time step must be chosen in accordance with the Nyquist criterion for system bandwidth. In other words, if the bandwidth of a filter or electrical sub-design is *BW*, then, $f_s \geq 2BW$. Good results may be obtained for values of$f_s \geq 5BW$.

In conclusion, the simulation time step t_s must be chosen in accordance with the expected (baseband or bandpass) signal $S(t)$ and (baseband or bandpass) system bandwidths. For most practical applications, the signal and system bandwidths are of the same order, but in general, t_s at any point in the system must be chosen in accordance with the larger of the signal bandwidth and system bandwidth at that point.

In some applications, the user may be interested in generating direct waveforms without having to convert binary information to symbols and symbols to samples. The Designer 5 system has a good number of waveform sources that can generate a variety of periodic and transient waveforms. In addition, arbitrary waveforms may be imported for the discrete time system analysis from MATLAB, WinIQSim and other system simulators by means of an external waveform file.

All waveform source components in the Designer 5 system have a parameter that determines the desired sampling rate f_s, which in turn will set the desired simulation time step.

As an example, consider the second order (type 1) **PLL** project shown below in Figure 4. Note that the sample rate parameter for the complex constant source (CCONST) is set to 50 kHz (well beyond the Nyquist rate of the electrical sub-design or loop filter of the Phase Locked Loop). This sampling rate implies a time step of 20µs. After analyzing the project, the PLL time domain response shown below in Figure 5 will be displayed.

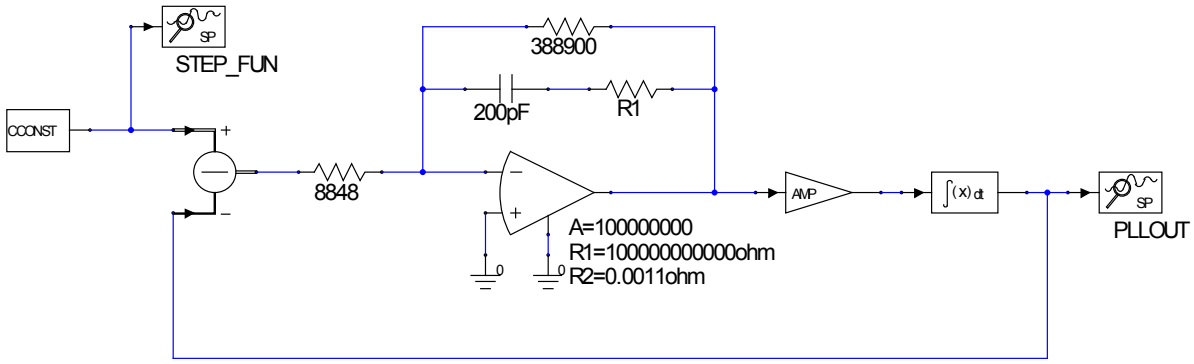


Figure 4: Example of PLL project to show the time domain response

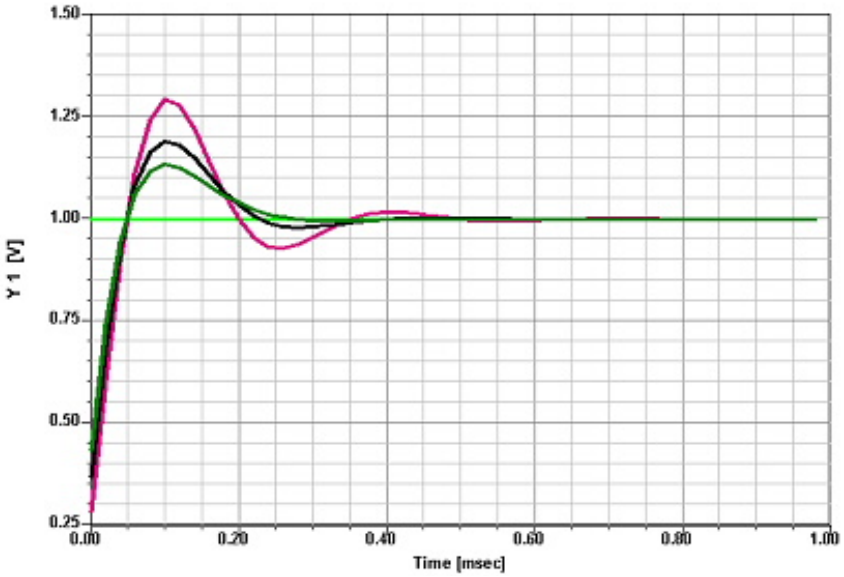


Figure 5: PLL time domain response for 3 different R1 values in the loop filter. Simulation sampling rate $f_s = 50KHz$ (values of R1 used are R1=74800, R1=174800, R1=274800)

If the sample rate of the CCONST source is readjusted to 20 KHz and the project is reanalyzed, the resulting change in time domain response for the PLL can be seen below in Figure 6 in terms of the “smoothness” of the output curves. There is a warning message displayed for this step.

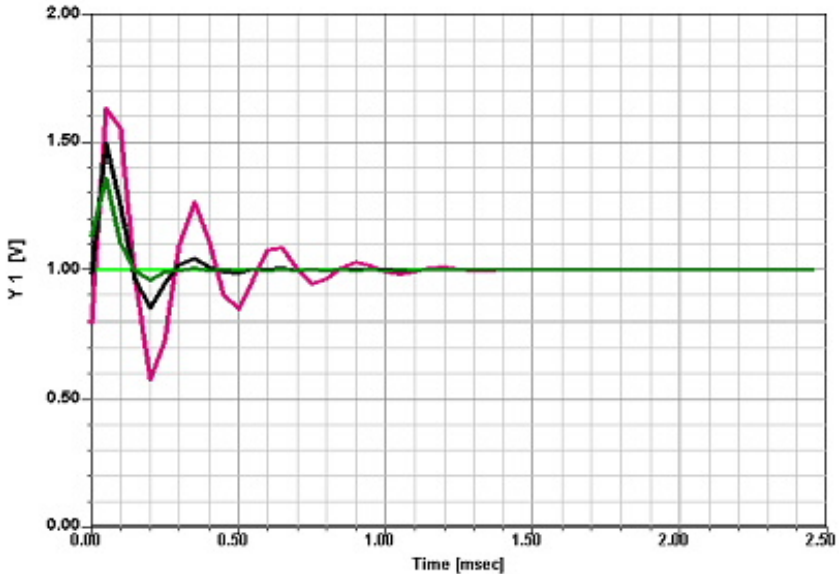


Figure 6: PLL time domain response for 3 different R1 values in the loop filter. Simulation sampling rate $f_s = 20KHz$ (values of R1 used are R1=74800, R1=174800, R1=274800)

As can be seen from the two figures (figure 5 and figure 6), choosing a higher sampling rate results in more accurate simulation results.

Step 2: Choosing the Minimum Bandwidth Control Variable

Another important control parameter for analyzing mixed systems is *MIN_BW* which is the minimum bandwidth of an electrical component or sub-design. It is up to the user to define this minimum bandwidth as the point where the frequency domain response is down by 3dB, 10dB or more. The simulation speed and accuracy of mixed mode systems largely depends on selecting this control parameter. The parameter *MIN_BW* may be specified through the Discrete Time Domain System solution setup. To specify a reasonable value for *MIN_BW*, it may be worthwhile to evaluate the frequency response (i.e., $S_{21}(f)$) of electrical sub-designs using the single tone frequency domain sweep analysis discussed in the Frequency Domain Analysis topic. This will help identify the *MIN_BW* in the system more appropriately.

The number of frequency points used to evaluate the frequency response of an electrical component/sub-design (during discrete time analysis) is given by the next power of 2 greater than or equal to *K*, where

K = {Minimum power of 2$\geq \frac{10f_S}{MIN_BW}$}

and the frequency step is given by

$df = \frac{f_s}{K} \leq \frac{MIN\_BW}{10}$ = Sweep resolution used for warning message in the discrete time domain analysis dialog, where f_s is the sampling rate of the input signal(s) to the electrical component/sub-design**.**

Step 3: Choosing the Maximum Ratio Control Variable

Once the frequency response of a linear electrical RF sub-design is obtained, the impulse response is extracted from the frequency response by means of the inverse FFT operation and the pre-defined variable MAX_RATIO (0 < MAX_RATIO ≤ 1).

MAX_RATIO is a critical parameter used to determine the actual length of the impulse response used for discrete time simulation. Since the inverse FFT of the frequency response yields an impulse response of length $K/2$ samples, not all of these samples will be used in the actual discrete time domain simulation. Starting at the last impulse response sample at time $Kt_s/2$ and moving towards $t = 0$ (see Figure 7), the pre-defined variable MAX_RATIO is used to truncate the impulse response at the point where the ratio of the impulse response local maximum to the global maximum is ≥ MAX_RATIO. Thus, specifying a smaller MAX_RATIO value will typically result in longer impulse responses (and longer simulation time) but more accuracy.

The default value for MAX_RATIO is $1e^{-10}$, which should produce the most accurate results. At the expense of accuracy, the user may specify larger values for MAX_RATIO if a shorter simulation time is desired.

A graphical illustration of the effects MAX_RATIO has on determining the length of the impulse response is shown below.

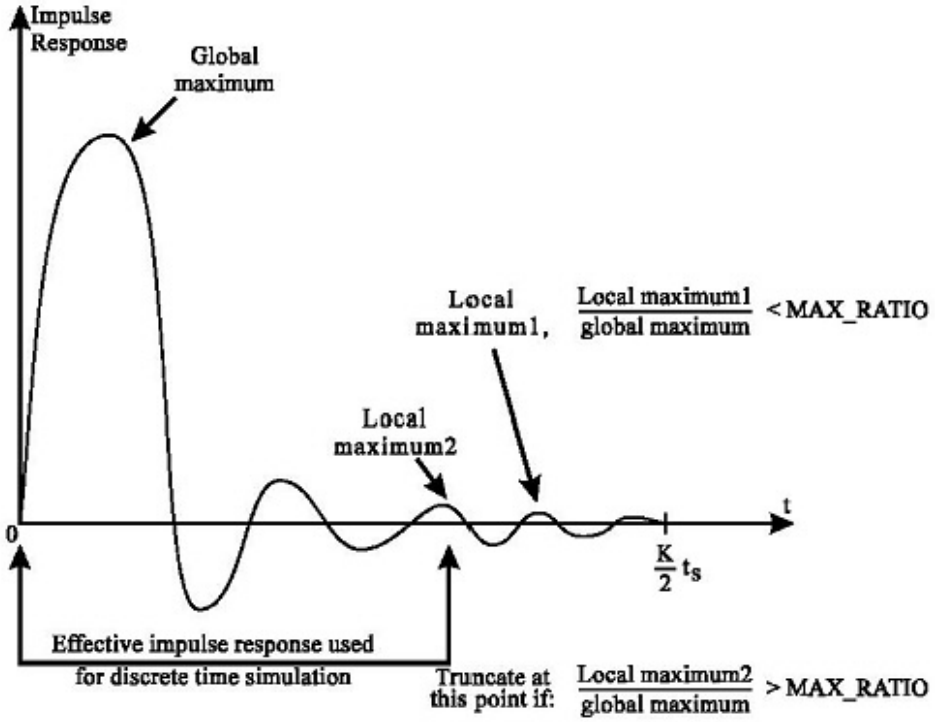


Figure 7: Graphical illustration of the significance of the maximum ratio control parameter.

Consider the PLL example discussed above. The responses shown in Figures 8 and 9 below reflect two different values for the MAX_RATIO control parameter, namely 0.001 in Figure 8 and 0.1 in Figure 9, with a fixed sampling rate of 50 kHz. Clearly, the impact of choosing a larger MAX_RATIO (Figure 9) can be seen. This is due to the fact that a larger MAX_RATIO resulted in a premature truncation of the impulse response of the PLL electrical loop filter, and consequently, the simulation results are inaccurate. In general, a longer discrete time impulse response (smaller MAX_RATIO) requires a longer simulation time, but will tend to yield more accurate results.

Note If starting at $Kt_s/2$ and moving towards $t = 0$, the search process fails to detect a local maximum/global maximum ratio that is less than MAX_RATIO, a warning message at the end of the simulation will be displayed.

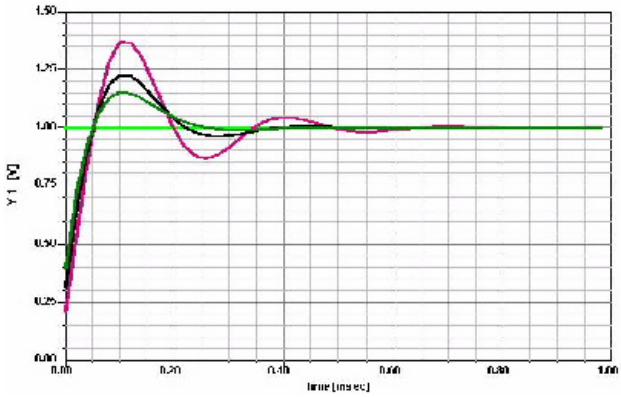


Figure 8: PLL response from MAX_RATIO = 0.001

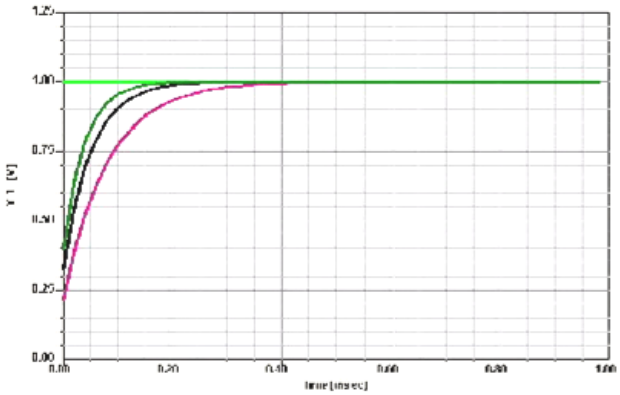
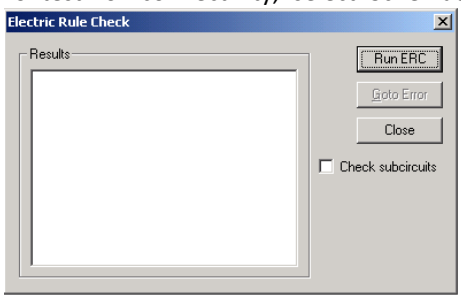


Figure 9: PLL response from MAX_RATIO = 0.1

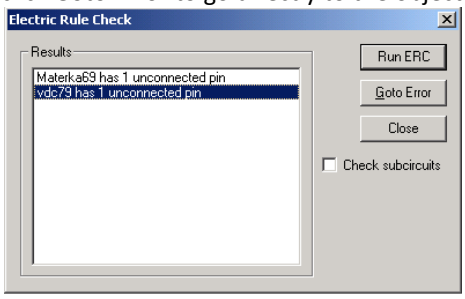
Checking Connectivity

The Electric Rule Check (ERC) feature checks the circuit for valid connectivity. ERC automatically conducts rule checking for ports, connections, and components of the active schematic.

1. To test for connectivity, select Schematic > Electric Rule Check, which opens the following dialog:



2. Select Check subcircuits to run the electric rule check on subcircuits of the active schematic display.
3. Click Run ERC to begin the error check.
4. If an error is displayed in the Results window double-click the error message or select the message and click Goto Error to go directly to the object in the Schematic Editor that caused the error.



Possible causes of Electric Rule Check error messages include:

• Unconnected Pins — A component, or port, with a pin that is not connected to anything else.

• Overlapping Components — A component that completely overlaps another such that the two components appear as one component. (Often caused by accidentally clicking twice when placing a component.)

• Nets with Multiple Output Pins — A net which has more than one output pin connected to it. (This is rare since most component pins are labeled as input and output.)

If You Encounter Convergence Difficulties

The nonlinear solver have been greatly improved, but you may still have convergence problems with some circuits, particularly, highly nonlinear circuits with bipolar transistors or circuits with high drive levels may pose a problem. For such circuits, the following hints are suggested to enable finding a solution:

1) Check the circuit connections. Improper node connections and/or missing units on parameters are the most common causes for convergence problems and messages that indicate "Singular Jacobian." This commonly happens when the active devices are not biased properly or the signal path is not connected. Use the **Show Bias Point** option on the **Analysis** dialog to check for proper bias.

2) Check that the bias sources are properly connected. If constant current sources are used, make sure the current flow is in the desired direction.

3) Add losses in the circuit. When initial designs are simulated it is common to use ideal elements that don't have losses (e.g., transmission lines using characteristic impedance and electrical length only). This may pose problems in the analysis of the linear subcircuit at DC or when computing the Jacobian for nonlinear analysis.

4) Approach the solution point incrementally. By sweeping the source voltage or power toward the desired level, the circuit is driven gradually into the region where convergence is difficult to obtain. During a source sweep, the results of the previous step are used for the initial iterate of the subsequent step, the starting point is closer to the vicinity of the desired solution than a "cold" start from zero initial values. Designer 5 also employs automatic step reduction on power sweeps, whereby the step size is halved if convergence was not obtained on the previous step.

5) A similar solution to the above is to start the analysis from the previous solution. The *.VAR file should be backed up, the solution options should start from a previous solution, and the DC initialization should be disabled. This method can also be useful when manually tuning the circuit to achieve a desired response (if it is a single-point analysis).

6) If insufficient sampling points are used to represent the time-domain waveforms, there will be significant aliasing errors in the FFT.

References

[1]. G. Vendelin, A. Pavio, U.L. Rohde, *Microwave circuit design using the Linear and Nonlinear Techniques*, Wiley, New York, 2005

[2]. U.L. Rohde and D. Newkirk, *RF/Microwave circuit design for Wireless Applications*, Wiley, New York, 2000.

[3]. U.L. Rohde, A.K. Poddar, and G. Boeck, *The Design of Modern Microwave Oscillators for Wireless Applications: Theory and Optimization*, Wiley, New York, 2005.

[4]. V. Rizzoli, A. Lipparini, and Ernesto Marazzi, "A General-Purpose Program for Nonlinear Microwave Circuit Design," *IEEE Trans. Microwave Theory Tech.*, vol. 31, no. 9, pp. 762-770, September 1983.

[5]. V. Rizzoli, A. Lipparini, A. Costanzo, F. Mastri, C. Cecchetti, A. Neri, and D. Masotti, "State-of-the-Art Harmonic-Balance Simulation of Forced Nonlinear Microwave Circuits by the Piecewise Technique," *IEEE Trans. Microwave Theory Tech.*, vol. 40, no. 1, pp. 12-28, January 1992.

[6]. V. Rizzoli, C. Cecchetti, A. Lipparini, " A General-Purpose Program for the Analysis of Nonlinear Microwave Circuits Under Multitone Excitation by Multidimensional Fourier Transform," *17th European Microwave Conf.*, pp. 635-640, September 1987.

[7]. V. Rizzoli and A. Neri, "State-of-the-Art and Present Trends in Nonlinear Microwave CAD Techniques," IEEE Trans. Microwave Theory Tech., vol. 36, no. 2, pp. 343-365, February 1988.

[8]. V. Rizzoli, C. Cecchetti, A. Lipparini, and F. Mastri, "General-Purpose Harmonic-Balance Analysis of Nonlinear Microwave Circuits Under Multitone Excitation," *IEEE Trans. Microwave Theory Tech.*, vol. 36, no. 12, pp. 1650-1660, December 1988.

[9]. V. Rizzoli, F. Mastri, F. Sgallari, V. Frontini, “The Exploitation of Sparse-Matrix Techniques in Conjunction with the Piecewise Harmonic-Balance Method for Nonlinear Microwave Circuit Analysis,” *1990 MTT-S Int. Microwave Symp. Digest*, pp. 1295-1298, June 1990.

[10]. V. Rizzoli, A. Lipparini, and F. Mastri, “Computation of Large-Signal S-Parameters by Harmonic-Balance Techniques,” *Electron. Lett.*, vol. 24, pp. 329-330, Mar. 1988.

[11]. V. Rizzoli, F. Mastri, and F. Sgallari, and G. Spaletta, “Harmonic-Balance Simulation of Strongly Nonlinear Very Large-Size Microwave Circuits by Inexact Newton Methods,” *IEEE MTT-S*, pp. 1357-1360, 1996.

[12]. Ansys Designer 5 manual.

[13]. Andrei Vladimirescu, *The Spice Book,* Wiley NY, 1994

Using Simulation at RF

By Dr. Ulrich Rohde, N1UL

Introduction

While RF is ill-defined, RF effects start already at about 100 kHz. This was first noticed while building high Q inductors for receivers. At this time Litz wire was invented. Here braided copper wire was covered with cotton and these were braided again. So self-resonance effects were largely avoided.

The simulator, having a graphic input (schematic entry) and graphic output (rectangular or polar > Smith Chart) solves the DC/RF current calculation in the circuit based on the Z or Y matrix (Series or parallel circuits parts) based on the voltages applied. For passive components there are practically no distortion products and the calculations are trivial. As frequencies get higher, the physical dimensions will get closer to the wavelength and the RF values of the components change drastically. Even below these frequencies the passive elements will show the effects of parasitic elements such as lead inductors and stray capacitances.

To complicate matters active elements such as diodes and transistors force the designer to more complex simulators. Simulators should provide information about all practical circuits, from a low number of transistors to millions of them in ICs. Typical parameters are gain, input/output impedance or matching (S-parameters, noise figure and stability). Of course DC information is also available. For this purpose SPICE was developed. There are many dialects of SPICE available. From Cadence, SPECTRE is probably the most powerful on the market.

SPICE, while doing DC, frequency and time domain simulations, has some problems. The time domain calculation uses very complex mathematics. The algorithmic research carried out converged to the use of the Newton-Raphson solution on nonlinear equations. More detail is available from [11]. These methods are not always stable. All kind of experiments with the program settings may be necessary to get a conversion of the process; also SPICE has problems with very high-Q circuits. The noise analysis, if not based on the noise correlation matrix approach, is also not correct as the feedback capacitance Im(Y12) begins to play a key role. All modern SPICE programs are at least based on SPICE3 from Berkley.

The real powerful simulators however are based on the harmonic balance method where passive component circuit parts a calculated in the frequency domain and the nonlinear (active) components in the time domain.

This partly stems from the fact that the transient and time domain analysis is proportionally to the values of the components. Therefore a 10nF capacitor requires more computation time as a 1nF capacitor. This is not the case in the frequency domain.

Generally, SPICE finds a solution to most circuit problems. However, because of the nonlinearity of the circuit equations and a few imperfections in the analytical device models a solution is not always guaranteed when the circuit and its specification are otherwise correct.

In the majority of the cases when a solution failure occurs it is due to a circuit problem, either its specification or its inoperability. A convergence problem can be categorized as either failure to compute a DC operating point or

abortion of the transient analysis because of the reduction for the time step below a certain limit without finding a solution.

Failure to find a solution can occur at the level of the linear equation, the Newton-Raphson iteration, or the numerical integration. Rather than present the convergence issues based on the algorithm causing the problem, it has been deemed beneficial to describe the causes for failure from a user's perspective.

Specific procedures can be followed when SPICE fails to find a DC solution of the circuit. The prescribed remedies include redefinition of analysis options, use of built-in convergence-enhancing algorithms, and DC operating point solution with a different analysis.

Time-domain analysis can provide an inaccurate solution or fail because of a number of reasons related either to the integration method and associated time-step control or the iterative solution of nonlinear equations. Knowledge of the specifics of different types of electronic circuits can assist the user in finding an accurate solution by specifying appropriate analysis modes, options, tolerances, and suitable model parameters. Thus, oscillators require certain initializations not necessary for amplifiers, and bipolar circuits may need different convergence tolerances than do MOS circuits. Also only the hybrid programs, based on SPICE type approach and harmonic balance and its dialects offer linear and non-linear optimization!

1. LIMITS OF LOW-FREQUENCY SIMULATION

1.1 Upper limit of conventional SPICE simulators

The basic SPICE simulator has ideal elements and some transmission line applications. As we approach frequencies where the lumped elements turn into distributed elements and special connecting elements become necessary, the use of the standard elements ends. It also has the mathematical limitations mentioned above. Temperature effects are allowed, meaning that the influence of temperature on the components is taken into account.

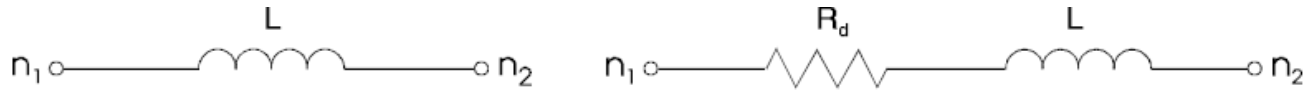
Here is a list of some standard passive devices and its description for temperature and frequency, which low frequency simulators don't have:

IND – Inductor [5]

Inductor has the following options:

- Ideal Inductor
- Toroidal Inductor
- Air Core Inductor

Ideal Inductor



keyword	description	unit	default
L	Inductance		
Rs	Series resistance		
Q	Quality factor, constant resistance with *Q* specified		
Q1	Quality factor, resistance proportional to square root of frequency (skin-effect model)		
Q2	Quality factor, constant *Q* model (resistance inversely proportional to frequency)		
F	Frequency at which quality factor (**Q**, **Q1** or **Q2**) has assigned value		
TEMP	Local temperature used for noise calculations	K	298
TC1	Temperature coefficient /degree		0.0
TC2	Temperature coefficient /degree2		0.0
M	Multiplication factor		1.0
DTEMP	Temperature difference	degree	0.0

Notes

1. The model is described by the following:
Q =w L/Rd
where $w = 2\pi f$
f = operating frequency

2. If **Rs** is specified, Rd = **Rs**.

3. If neither **Rs**, **Q**, **Q1**, nor **Q2** is specified, the inductor is assumed to be ideal; that is, Rd = 0.

4. Let $\mathbf{Q}_{REF}$, $\mathbf{Q1}_{REF}$ and $\mathbf{Q2}_{REF}$ represent the numeric values of **Q**, **Q1** and **Q2**; that is, the actual numbers that replace the symbol *x2*. Let *F*REF represent the numeric value of **F**. The frequency dependence of the three *Q* models is given by

$$\mathbf{Q}: Q(f) = \mathbf{Q}_{REF} f / \mathbf{F}_{REF}$$

$$\mathbf{Q1}: Q(f) = \mathbf{Q1}_{REF} * \sqrt{\frac{f}{\mathbf{F}_{REF}}}$$

$$\mathbf{Q2}: Q(f) = \mathbf{Q2}_{REF}$$

5. If an ideal inductor is referenced by the KMUI element, this inductor must be labeled.

Please see the reference for the KMUI element. The Label name can be entered through the element properties dialog.

6. If **TC1**, **TC2**, **M**, or **DTEMP** is defined, the inductance *L* represented by the IND element is given by

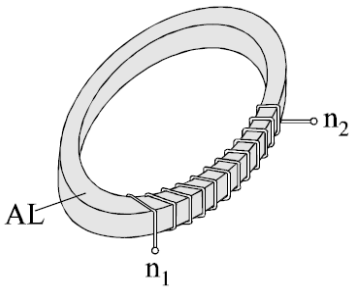
L = **L** * (1.0 + **TC1** * **DTEMP** + **TC2** * **DTEMP**2) / **M**

Toroidal Inductor

The Toroidal Inductor has the following options

- Ideal Model
- Physical Model

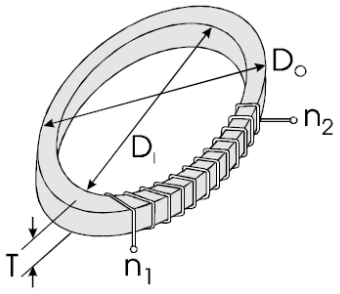
Ideal Inductor



keyword	description	unit	default
N	Number of winding	turns	
AL	Inductance index	Henrys/turn	

Note
L = **N**2***AL**

PHYSICAL MODEL



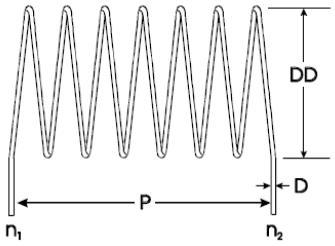
keyword	description	unit	default
DO	Outer diameter of core	meter	
DI	Inner diameter of core	meter	
N	Number of winding	turns	
T	Thickness of the core	meter	
MU	Relative permeability of the core		
RB	Conductor resistivity	micro-ohm*cm	0.0
D	Diameter of the wire	meter	0.0

Air Core Inductor

The Air Core Inductor has the following options:

- Physical Model with wire diameter
- Physical Model with wire gauge

PHYSICAL MODEL WIRE DIAMETER [6]

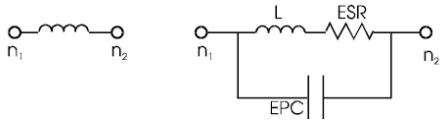


keyword	description	unit	default
DD	Diameter of the core	meter	
N	Number of winding	turns	
P	Physical length		
D	Diameter of the wire	meter	
RB	Conductor resistivity	micro-ohm*cm	

PHYSICAL MODEL WITH WIRE GAUGE

keyword	description	unit	default
DD	Diameter of the core	meter	
N	Number of winding	turns	
P	Physical length	meter	
AWG	Gauge of the wire		
RB	*Conductor resistivity*	*micro-ohm*cm*	

INDQ – Chip Inductor 2



Keyword	description	unit	default
L	Inductance	Henry	0.0
FRES	Self-resonance frequency	Hz	0.0

Q	Quality factor	0.0	
FQ	Frequency at which **Q** is given	Hz	0.0
TC	Temperature coefficient	PPM	0.0
TEMP	Element temperature	K	298
MOD	Keyword to indicate the name of a chip inductor in a related device library		

Notes

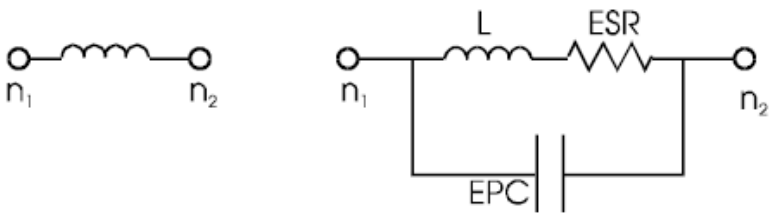
1. ESR is the equivalent series resistance and can be calculated from **Q** as ESR = 1/(*w***L**×**Q**).

2. EPC is the equivalent parallel capacitance and can be calculated from the self-resonance frequency, **FRES**.

3. **TC** is defined in PPM (parts per million). For example, an inductor has a nominal value, *L*0 (keyword **L**), at 298 K. At temperature **TEMP**, the value *L* of the inductor is calculated as

L = **L**×(1+(**TEMP**- 298)×**TC**×1.0e-6).

INDR – Chip Inductor 3



Keyword	**Description**	**unit**	**default**
L	Inductance	Henry	0.0
FRES	Self-resonance frequency	Hz	0.0
ESR	Equivalent series resistance	Ohm	0.0
FESR	Frequency at which ESR is given	Hz	0.0
TC	Temperature coefficient	PPM	0.0
TEMP	Element temperature	K	298
MOD	Keyword to indicate the name of a chip inductor in a related device library		

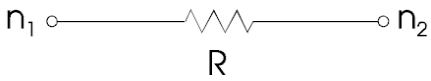
Notes

1. EPC is the equivalent parallel capacitance and can be calculated from the self-resonance frequency, **FRES**.

2. **TC** is defined in PPM (parts per million). For example, an inductor has a nominal value, *L*0 (keyword **L**) at 298 K. At temperature **TEMP**, the value of the inductor is calculated as

L = **L**×(1+(**TEMP**- 298)×**TC**×1.0e-6).

RES – Resistor



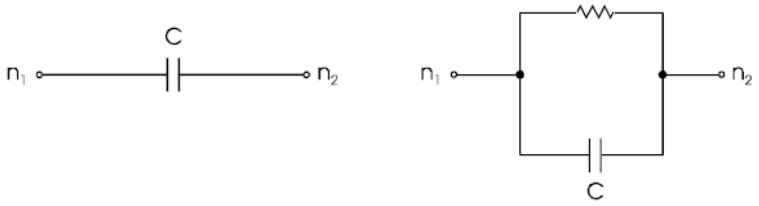
keyword	description	unit	default
R	Resistance	ohm	
TEMP	Local temperature used for noise calculations	K	298
TC1	Temperature coefficient	/degree	0.0
TC2	Temperature coefficient	/degree2	0.0
M	Multiplication factor	1.0	
DTEMP	Temperature difference	degree	0.0

Notes

1. The default value for **TEMP** is obtained from the value assigned to the global ambient temperature parameter *T*ambient (default, 298 K).

2. If **TC1**, **TC2**, **M** or **DTEMP** is defined, the resistance *R* represented by the RES element is given by $R = \mathbf{R} * (1.0 + \mathbf{TC1} * \mathbf{DTEMP} + \mathbf{TC2} * \mathbf{DTEMP}2) / \mathbf{M}$

CAP – Capacitor



keyword	description	unit	default
C	Capacitance	farad	
Q	Quality factor, constant conductance model		-
Q1	Quality factor, conductance is proportional to (1/**F**)**EXP**. By default, the conductance is proportional to the square root of frequency.		-
Q2	Quality factor, constant-*Q* model (conductance proportional to frequency)		-
F	Frequency at which the quality factor (**Q**, **Q1** or **Q2**) has the assigned value	Hz	
EXP	Quality factor dependence exponent for **Q1**. The default is □0.5; i.e., the conductance is proportional to the square root of frequency.		0.5
TEMP	Local temperature used for noise calculations	K	298
TC1	Temperature coefficient /degree		0.0
TC2	Temperature coefficient /degree2		0.0
M	Multiplication factor		1.0
DTEMP	Temperature difference	degree	0.0

Notes

1. The model is described by the following: $Q(f) = wC / Gd$ where $w = 2xf$ and f is the operating frequency

2. $Q(f)$ can be defined as **Q**, **Q1**, or **Q2** respectively. If neither **Q**, **Q1** nor **Q2** is specified, then the capacitor is assumed ideal; that is, $Gd = 0$.

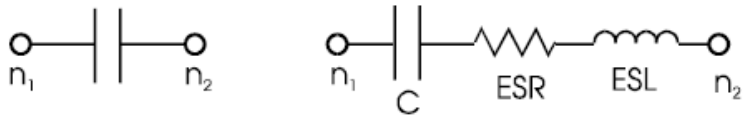
3. Let **Q**REF, **Q1**REF and **Q2**REF represent the numeric values of **Q**, **Q1** and **Q2**; that is, the actual number that replaces the symbol. Let **F**REF represent the numeric value of **F**. The frequency dependence of the three *Q* models is given by **Q**: $Q(f) = \mathbf{Q}\text{REF} * f/\mathbf{F}\text{REF}$

$$\mathbf{Q1}: Q(f) = \mathbf{Q1}\text{REF} * (\mathbf{F}\text{REF}/f)^{\mathbf{EXP}}$$
$$\mathbf{Q2}: Q(f) = \mathbf{Q2}\text{REF}$$

4. **EXP** is used in conjunction with **Q1** to define frequency dependence. This parameter defaults to a value of - 0.5, yielding a conductance value that is proportional to the square root of the frequency. The value of **EXP** must be in the range: $-6.0 < \mathbf{EXP} < +6.0$

5. If **TC1**, **TC2**, **M**, or **DTEMP** is defined, the capacitance *C* represented by the CAP element is given by $C = \mathbf{C} * (1.0 + \mathbf{TC1} * \mathbf{DTEMP} + \mathbf{TC2} * \mathbf{DTEMP}2) * \mathbf{M}$

CAPR – Chip Capacitor 3



keyword	description	unit	default
C	Capacitance	Farad	0.0
FRES	Self-resonant frequency	Hz	0.0
ESR	Effective Series Resistor	Ohm	0.0
FESR	Frequency at which ESR has assigned a value	Hz	0.0
TC	Temperature Coefficient	PPM	0.0
TEMP	Element temperature	K	298
MOD	Keyword to indicate the name of chip capacitor in a related device library		

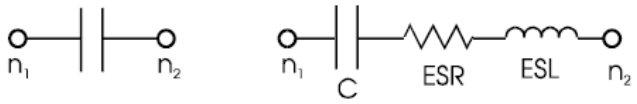
Notes

1. ESR is Effective Series Resistance.

2. ESL is Effective Series Inductance and can be calculated from the Self-Resonant Frequency FRES

3. TC is defined as PPM (Parts per million). For example, a capacitor has nominal value Co, at 298K, at temperature TEMP, the value of the capacitor is calculated as C = Co*(1+(TEMP-298)*TC*1.0e-6)

CAPQ – Chip Capacitor 2



Keyword	description	unit	default
C	Capacitance	Farad	0.0
FRES	Self-resonant frequency	Hz	0.0
Q	Quality Factor		0.0
FQ	Frequency at which ESR has assigned a value	Hz	0.0
TC	Temperature Coefficient	PPM	0.0
TEMP	Element temperature	K	298
MOD	Keyword to indicate the name of chip capacitor in a related device library		

Notes

1. ESR is Effective Series Resistance and can be calculated from

$$Q\left(ESR = \frac{1}{\omega C * Q}\right)$$

2. ESL is Effective Series Inductance and can be calculated from the Self-Resonant Frequency, FRES

3. TC is defined as PPM (Parts per million). For example, a capacitor has nominal value Co, at 298K, at temperature TEMP, the value of the c

---

To demonstrate the need of complex elements, here is a practical example of a 6 to 18GHz amplifier simulated with lumped elements.

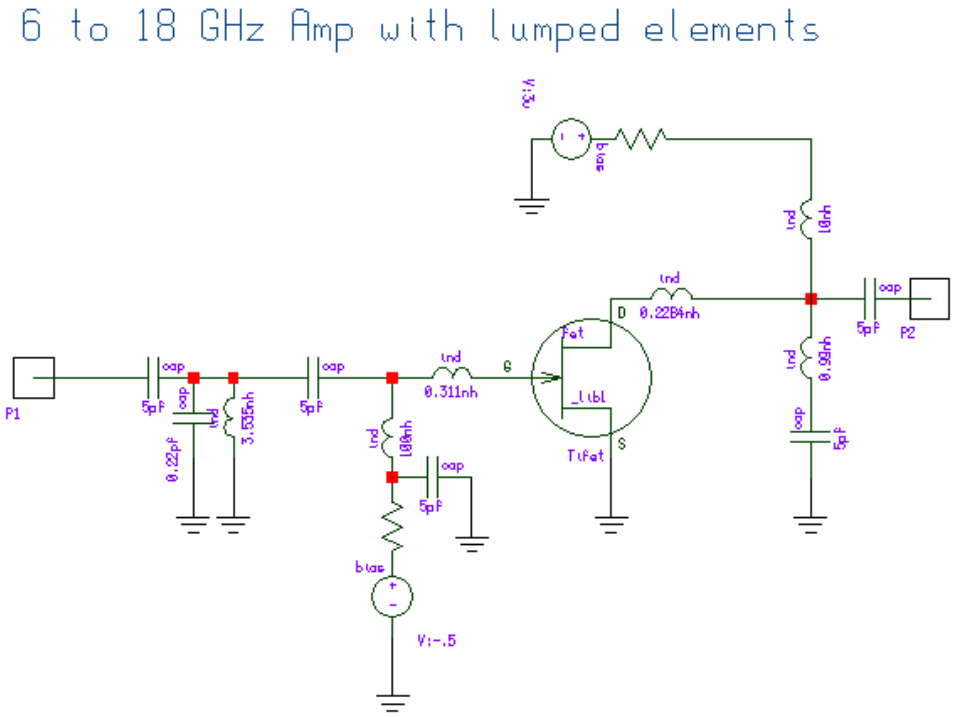


Figure 1 (a): Schematic of 6 to 18GHz Amplifier using Lumped elements

The chosen values were optimized to give a perfect response. To do this of course is silly as the components, based on their parasitic, in reality most definitely have different values.

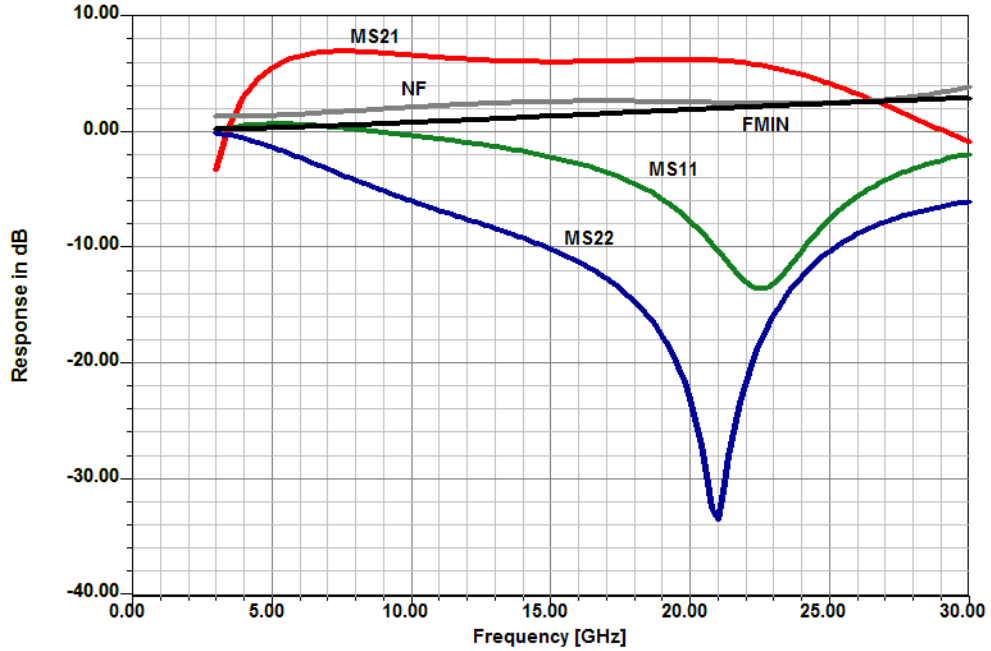


Figure 1 (b): Simulated plot showing the input matching (S11), output matching (S22), Gain (S21) and the F_{MIN} and NF in dB over the frequency range.

Fig 1 (a) shows the circuit using discrete components to simulate a single stage amplifier. The simulator needs to be able to handle a GaAs model such as the Materka model. And where do the real time domain values come from? However this is not an accurate modeling since the microwave components and junctions elements are missing. The results are too perfect. The minimum noise figure Fmin and the actual spot noise figure NF are very low and very close at 25GHz

Now we do the correct simulation and the things change drastically.

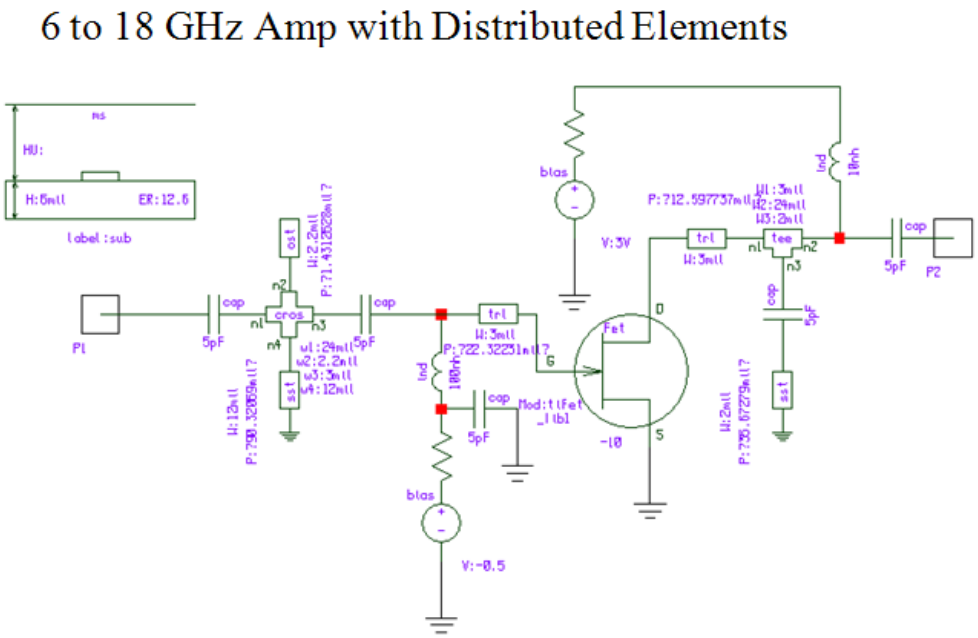


Figure 2 (a): Schematic of 6 to 18GHz Amplifier using Lumped elements

1.2 Frequency range above which RF simulators should be used

Figure 2 shows essentially the same circuit but with accurate modeling, and follows the layout reality. This type of circuit goes far beyond any simple SPICE program.

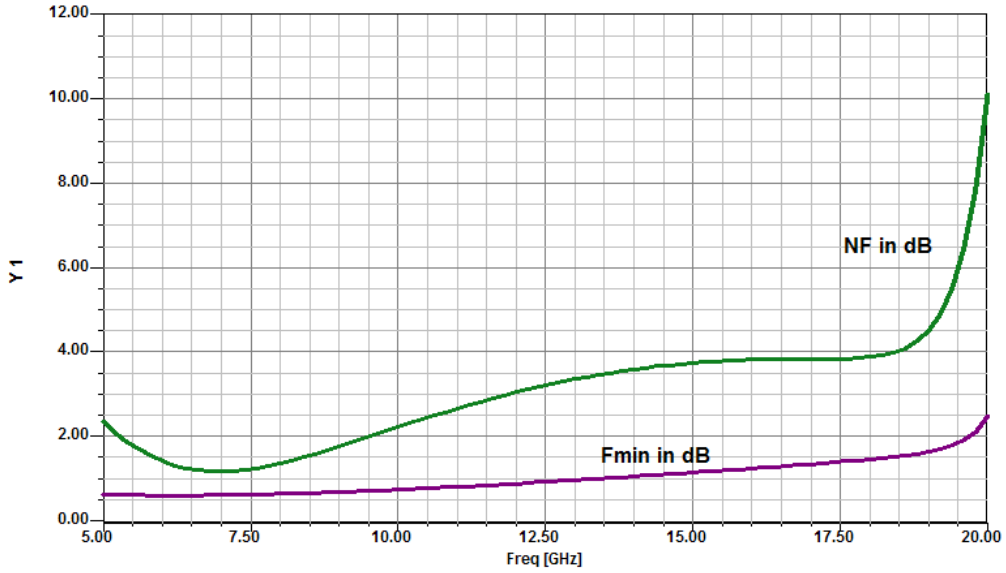


Figure 2 (b): Simulated plot showing the F_{MIN} and NF in dB over the frequency range for figure 2(a).

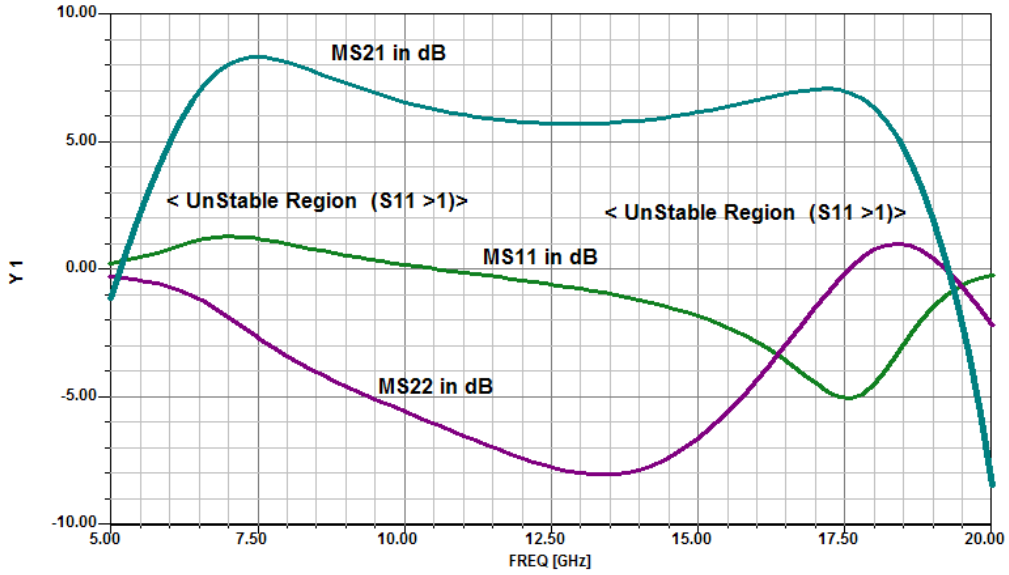


Figure 2 (c): Simulated plot showing the input matching (S11), output matching (S22) and Gain(S21) in dB over the frequency range for Figure 2(a).

1.3 Symptoms of low-frequency simulator breakdown

Non-linear programs are also used to predict the nonlinear performance of analog circuits. The following is the mathematical introduction:

Amplitude Linearity Issues and Figures of Merit. A network's amplitude nonlinearity can be characterized by the expansion:

$$y = k_1 f(x) + k_2 [f(x)]^2 + k_3 [f(x)]^3 + \text{higher - order terms} \qquad (1)$$

where y represents the output, the coefficients k_n represent complex quantities whose values can be determined by an analysis of the output waveforms, and $f(x)$ represents the input. Even though all practical networks exhibit amplitude nonlinearity, we can (and often do) refer to many networks as "linear." We say this of networks that are *sufficiently amplitude-linear for our purposes*--for example, weakly nonlinear networks in which small-signal operation is assumed even though the signal levels involved are sufficient to cause slight distortion. For many practical purposes, the first three terms of (1) adequately describe such a network's nonlinearity:

$$y = k_1 f(x) + k_2 [f(x)]^2 + k_3 [f(x)]^3 \qquad (2)$$

In adopting this simplification, we assume also that the nonlinearity is frequency-independent--that is, that the network has sufficient bandwidth to allow all of the products predicted by (1) to appear at its output terminals unperturbed [7].

When multiple signals are present in a network, even weak nonlinearity can result in profound consequences. To illustrate this, we'll let $f(x)$ consist of two sinusoidal signals:

$$f(x) = A_1 \cos \omega_1 t + A_2 \cos \omega_2 t \qquad (3)$$

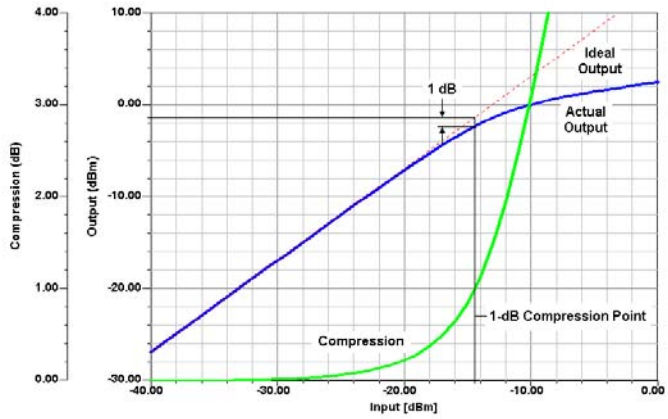


Figure 3: The power level at which a network's power output is down 1 dB relative to that of its ideally linear equivalent is a figure of merit known as the 1-dB compression point (P−1dB). The 1-dB compression point can be expressed relative to input power (P−1dB,in) or output power (P−1dB,out). For the amplifier simulated here, P−1dB, in ≈ −14.5 dBm and P−1dB,out ≈ −1.3 dBm.

We'll assume that ω_1 and ω_2 are close enough so that the coefficients k_i can be considered equal for both signals. We'll also assume for simplicity that all of the k_i are real. If equation (2) describes the network's response to an input $f(x)$, the response will be

$$
\begin{aligned}
y &= k_1(A_1\cos\omega_1 t + A_2\cos\omega_2 t) + k_2(A_1\cos\omega_1 t + A_2\cos\omega_2 t)^2 + k_3(A_1\cos\omega_1 t + A_2\cos\omega_2 t)^3 \\
&= k_1(A_1\cos\omega_1 t + A_2\cos\omega_2 t) \\
&+ k_2\left[A_1^2\frac{1+\cos 2\omega_1 t}{2} + A_2^2\frac{1+\cos 2\omega_2 t}{2} + A_1A_2\frac{\cos(\omega_1+\omega_2)t + \cos(\omega_1-\omega_2)t}{2}\right] \\
&+ k_3\left\{\begin{array}{l}\left[A_1^3\left(\frac{\cos\omega_1 t}{2} + \frac{\cos\omega_1 t}{4} + \frac{\cos 3\omega_1 t}{4}\right) + A_2^3\left(\frac{3\cos\omega_2 t}{4} + \frac{\cos 3\omega_2 t}{4}\right)\right] \\ + A_1^2A_2\left[\frac{3}{2}\cos\omega_2 t + \frac{3}{4}\cos(2\omega_1-\omega_2)t + \frac{3}{4}\cos(2\omega_1-\omega_2)t\right] \\ + A_2^2A_1\left[\frac{3}{2}\cos\omega_1 t + \frac{3}{4}\cos(2\omega_2+\omega_1)t + \frac{3}{4}\cos(2\omega_2-\omega_1)t\right]\end{array}\right\}
\end{aligned}
\qquad (4)
$$

The k_1 term of equation (4) represents the results of amplitude-linear behavior. No new frequency components have appeared; the two sine waves have merely been "rescaled" by k_1.

The second- and third-order terms of equation (4) represent the effects of harmonic distortion and intermodulation distortion. Second-order effects include second-harmonic distortion (the production of new signals at $2\omega_1$ and $2\omega_2$) and IMD (the production of new signals at $\omega_1+\omega_2$ and $\omega_1-\omega_2$). Third-order effects include gain compression, third-harmonic distortion (the production of new signals at $3\omega_1$ and $3\omega_2$), and IMD (the production of new signals at $2\omega_1\pm\omega_2$ and $2\omega_2\pm\omega_1$).

Gain Compression. Gain compression occurs when a network cannot increase its output amplitude in linear proportion to an amplitude increase at its input; gain *saturation* occurs when a network's output amplitude stops increasing (in practice, it may actually decrease) with increases in input amplitude. We can deduce from equation (4) that the amplitude of the $\cos\omega_1 t$ signal has become

$$A_1' = k_1 A_1 + k_3\left(\tfrac{3}{4} A_1^3 + \tfrac{3}{2} A_1 A_2^2\right) \tag{5}$$

Because k_3 will normally be negative, a large signal $A_2 \cos \omega_2 t$ can effectively mask a smaller signal $A_1 \cos \omega_1 t$ by reducing the network's gain. This third-order effect, known as *blocking* or *desensitization* when it occurs in a receiver, is a special case of gain compression. The presence of additional signals results a greater reduction in gain; the gain reduction for each signal is a function of the relative levels of all signals present. A receiver's blocking behavior may be characterized in terms of the level of off-channel signal necessary to reduce the strength of an in-passband signal by a specified value, typically 1 dB; alternatively, the decibel ratio of the off-channel signal's power to the receiver's noise-floor power may be cited as *blocking dynamic range*. Desensitization may be also characterized in terms of the off-channel-signal power necessary to degrade a system's SNR by a specified value.

Multiple signals need not be present for gain compression to occur. If only one signal is present, the ratio of gain with distortion to the network's idealized (linear) gain is

$$A_1' = \frac{k_1 + k_3\left(\tfrac{3}{4} A_1^2\right)}{k_1} \tag{6}$$

This is referred to as the *single-tone gain-compression factor*. Figure 3 shows how the k_3 term causes a network's gain to deviate from the ideal. The point at which a network's power gain is down 1 dB from the ideal for a single signal is a figure of merit known as the *1-dB compression point* (P_{-1dB}). Many networks (including many receiving and low-level transmitting circuits, such as low-noise amplifiers, mixers and IF amplifiers) are usually operated under small-signal conditions--at levels sufficiently below P_{-1dB} to maintain high linearity. As we'll see, however, some networks (including power amplifiers for wireless systems) may be operated under large-signal conditions – near or in compression – to achieve optimum efficiency at some specified level of linearity.

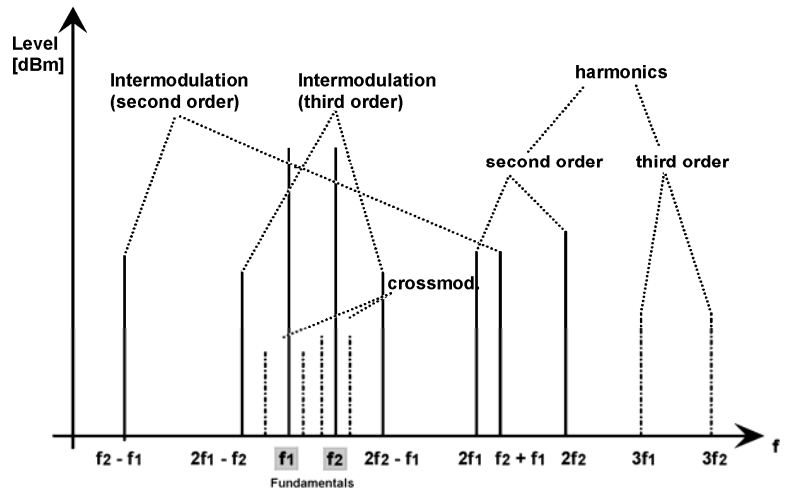


Figure 4: Relationships between fundamental and spurious signals, including harmonics and products of intermodulation.

Intermodulation. The new signals produced through intermodulation distortion (IMD) can profoundly affect the performance even of systems operated far below gain compression (Figure 4). IMD products of significant power can appear at frequencies remote from, in and/or near the system passband, resulting in demodulation errors (in reception) and interference to other communications (in transmission). Where an IMD product appears relative to the passband depends on the passband width and center frequency, the frequencies of the signals present at the system input, and the order of the nonlinearity involved. These factors also determine the strength of an IMD product relative to the desired signal.

Second-order IMD (IM_2) results, for an input consisting of two signals ω_1 and ω_2, in the production of new signals at $\omega_1 + \omega_2$ and $\omega_1 - \omega_2$; third-order IMD (IM_3) results, for an input consisting of two signals ω_1 and ω_2, in the production of new signals at $2\omega_1 \pm \omega_2$ and $2\omega_2 \pm \omega_1$.

Under small-signal conditions--that is, at levels well below compression--the power of an IM_2 product varies by 2 dB, and the power of an IM_3 product varies by 3 dB, per decibel change in input power level. This allows us to derive a network figure of merit, the *intermodulation intercept point* (*IP*), for a given IM order by extrapolating a network's linear and IM responses to their point of intersection (Figure 5) – the point at which their powers would be equal if compression did not occur. Because of the system noise and/or intermodulation distortion products, there is a minimum discernible signal (MDS) that limits the dynamic range at the lower end. Theoretically, Figure 5 should show a noise floor or IMD-spur floor for a given input signal that represents a lower limit below which signals cannot be detected. The intercept point for a given IM order *n* can be expressed, and should always be characterized, relative to input power ($IP_{n,in}$) or output power ($IP_{n,out}$); the IP_{in} and IP_{out} values differ by the network's linear gain. For equal-level test tones, $IP_{n,in}$ can be determined by:

$$IP_{n,in} = \frac{nP_A - P_{IM_n}}{n-1} \tag{7}$$

where *n* is the order, P_A is the input power (of one tone), P_{IMn} is the power of the IM product, and *IP* is the intercept point. The intercept point for cascaded networks can be determined from

$$IP_{2,in} = \frac{1}{\left(\frac{1}{\sqrt{IP1}} + \frac{G}{\sqrt{IP2}}\right)^2} \tag{8}$$

for IP_2 and from

$$IP_{3,in} = \frac{1}{\frac{1}{IP1} + \frac{G}{IP2}} \tag{9}$$

for IP_3. In both equations, *IP*1 is the input intercept of Stage 1 in watts, *IP*2 is the input intercept of Stage 2 in watts, and *G* is the gain of Stage 1 (as a numerical ratio, *not* in decibels). Both equations assume the worst-case condition, in which the distortion products of both stages add in-phase.

The ratio of the signal power to the IM-product power, the *distortion ratio*, can be expressed as:

$$R_{dn} = (n-1)\left[IP_{n(in)} - P_{(in)}\right] \tag{10}$$

where n is the order, R_{dn} is the distortion ratio, $IP_{n(in)}$ is the input intercept point, and $P_{(in)}$ is the input power of one tone.

Discussions of IMD have traditionally downplayed the importance of IM_2 because the incidental distributed filtering contributed by the tuned circuitry once common in radio communication systems was usually enough to render out-of-passband IM_2 products caused by in-passband signals, and in-passband IM_2 products caused by out-of-passband signals, vanishingly weak compared to fundamental and IM_3 signals. In broadband systems that operate at bandwidths of an octave or more, however, in-passband signals may produce significantly strong in-passband IM_2 and second-harmonic products. In such applications, balanced circuit structures (such as push-pull amplifiers and balanced mixers) can be used to minimize IM_2 and other even-order nonlinear products.

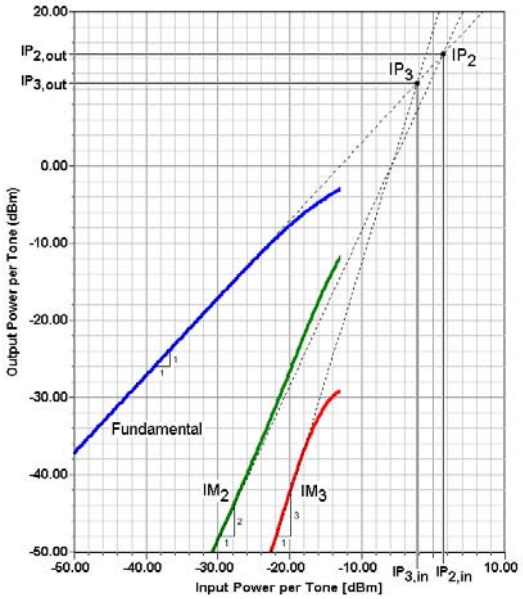


Figure 5: The level at which the power of one of a network's IM products equals that of the network's linear output is a figure of merit known as the intermodulation intercept point (IP). The intercept point for a given IM order n can be expressed, and should always be characterized, relative to input power (IPn, in) or output power (IPn, out); the IPin and IPout values differ by the network's linear gain. For the amplifier simulated here, IP2,in ≈ 1.5 dBm, IP2,out ≈ 14.5 dBm, IP3,in ≈ –2.3 dBm and IP3,out ≈ 10.7 dBm. Each curve depicts the power in one tone of the response evaluated.

As with IM_2, which IM_3 products are important depends on the spacing of the signals involved and the relative width of the system passband. If ω_1 and ω_2 are of approximately the same frequency, the additive products $2\omega_1 + \omega_2$ and $2\omega_2 + \omega_1$ will be outside the passband of a narrowband system. The subtractive products $2\omega_1 - \omega_2$ and $2\omega_2 - \omega_1$, however, will likely appear near or within the system passband. The IM_3 performance of any network subjected to multiple signals is therefore of critical importance, and an array of IM_3-related, sometimes application-specific, figures of merit has evolved as a result.

Dynamic Range. As we have seen, thermal noise sets the lower limit of the power span over which a network can operate. Distortion--that is, degradation by distortion of the signal's ability to convey information--sets the upper limit of a network's power span. Because the power level at which distortion becomes intolerable varies with signal type and application, a generic definition has evolved: The upper limit of a network's power span is the level at which the power of one IM product of a specified order is equal in power to the network's noise floor. The ratio of the noise-floor power to the upper-limit signal power is referred to as the network's *dynamic range (DR)*, often more

carefully characterized as *two-tone IMD dynamic range*, which, when evaluated with equal-power test tones, is a figure of merit commonly used to characterize receivers. The MDS relative to the input, as already defined, is

$$\mathrm{MDS_{in}} = \mathrm{kTB} + 3\,\mathrm{dB} + \mathrm{NF}$$

When $IP_{(n)in}$ and MDS are known, IMD DR can be determined from:

$$\mathrm{DR_n} = \frac{(\mathrm{n}-1)[\mathrm{IP_{n(in)}} - \mathrm{MDS_{in}}]}{\mathrm{n}} \qquad (11)$$

where *DR* is the dynamic range in decibels, *n* is the order, $IP_{(in)}$ is the input intercept power in dBm, and MDS is the minimum detectable signal power in dBm. The so-called *spurious-free dynamic range* (*SFDR* or DR_{SF}) is calculated from

$$\mathrm{DR_{SF}} = \frac{2}{3}\left(\mathrm{IP_3} - 174\,\mathrm{dBm} + \mathrm{NF} + 3\,\mathrm{dB}\right)$$

This equation allows us to determine how to measure the spurious-free dynamic range. This is done by applying the two-tone signals (in the case of IP_3) and increasing the two signals to the point where the signal-to-noise ratio deteriorates by 3 dB or, if the measurement is done relative to MDS, where the noise floor rises by 3 dB. The factor 2/3 is derived from the fact that the levels of IM_3 outputs increase 3 dB for 1 dB of input increase. This definition of dynamic range now is referenced to a noise figure rather than a minimum level in dBm, and is therefore independent of bandwidth. (By choosing smaller bandwidths [1 kHz instead of 10 kHz], a dynamic range measurement can be made to look better. Basing the specification on noise figure directly avoids this problem.)

1.4 An important test example:

Modern signals are multitone signals and according to international standards sensitive circuits such as CATV amplifiers must be specified in this area. Figure 6 shows the circuit diagram of such a circuit in this case a distribute amplifier.

Its frequency range is from 3GHz to 21GHz. The figure 6 shows how the simulation is organized and since all modern SPICE programs use schematic entries other simulators may do it different but essentially the same way.

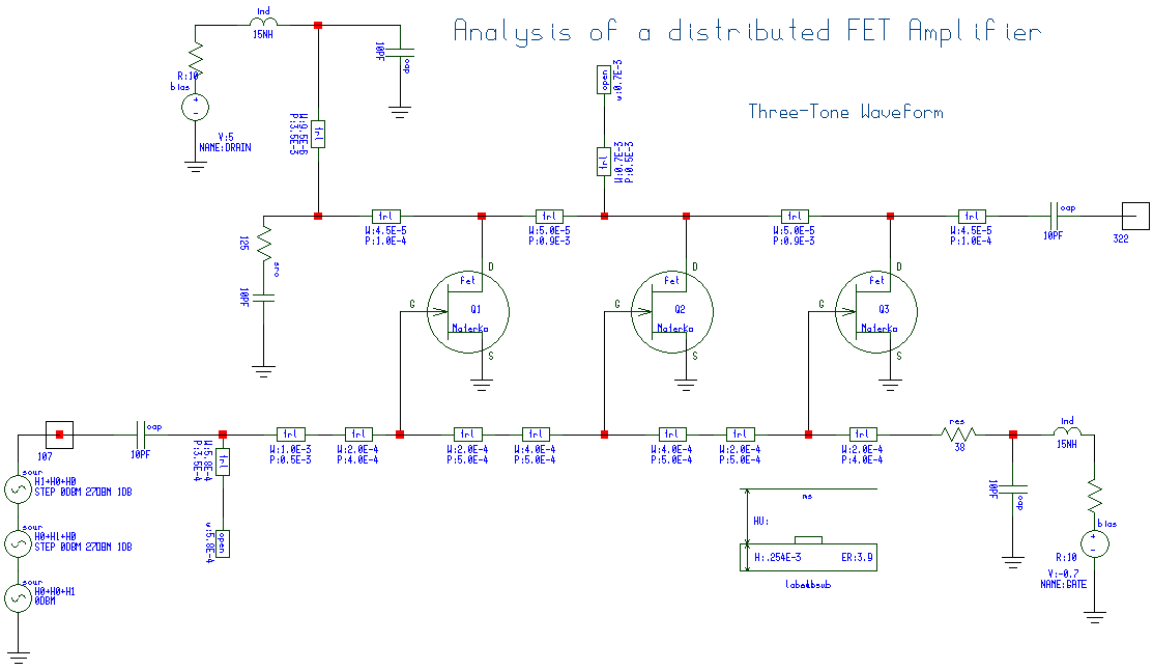


Figure 6: Circuit diagram of the distributed amplifier for 3 to 21GHz

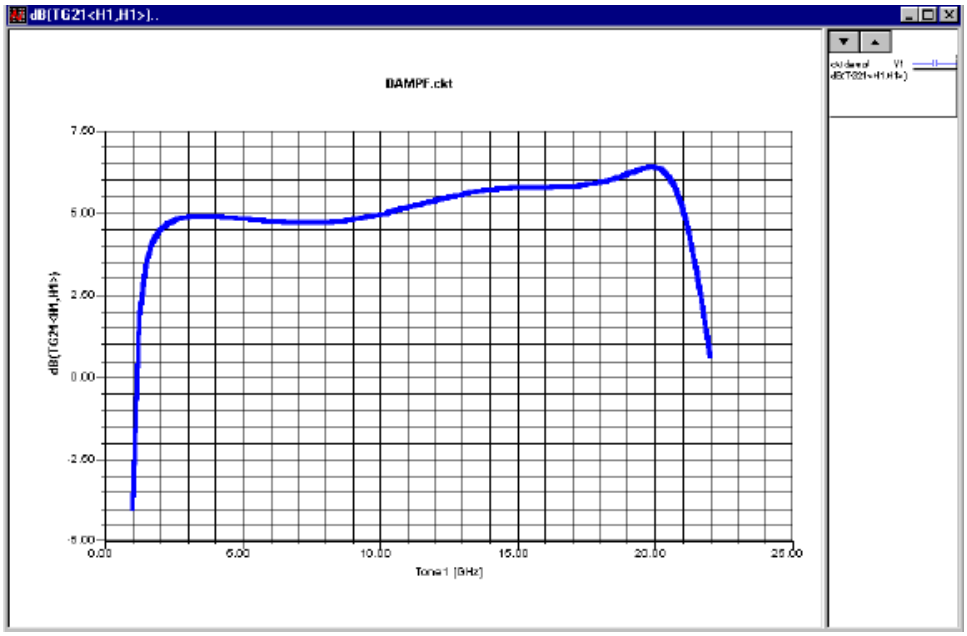


Figure 7: Output gain of the amplifier as predicted over the operating sweep range.

Figure 7 shows the resulting predicted gain plot for the amplifier in the Figure 6. Non-linear circuits respond differently and figure 8 shows some predicted and measured data. [10]

Triple-Beat Distortion and Cross-Modulation.

P_{-1dB} is a single-tone figure of merit; blocking, intercept point and dynamic range evaluate two-tone behavior. For networks that must handle AM and composite (AM and angle modulation) signals very linearly, such as television transmitters and cable TV distribution systems, a three-tone figure of merit called *triple-beat distortion*

has gained acceptance. Signals at ω_1 and ω_2 (closely spaced) and ω_3 (positioned far away from ω_1 and ω_2) are applied to the network under test, at levels, frequencies and spacings that vary with the application. One triple-beat distortion figure of merit is the ratio, expressed in decibels, of the IM product at $\omega_3 + (\omega_2 - \omega_1)$ to one of the network's linear outputs at a specified output level. Alternatively, the triple-beat figure of merit may express the network output level at which a specified triple-beat ratio occurs.

Triple-beat distortion is the mechanism underlying cross-modulation, a form of intermodulation in which one or more AM signals present in a network amplitude-modulate all signals present in the network [8]. Figures 8a and 8b graph the results of gain compression, two-tone intermodulation, cross-modulation and triple-beat testing on a wideband (5 to 1000 MHz) amplifier.

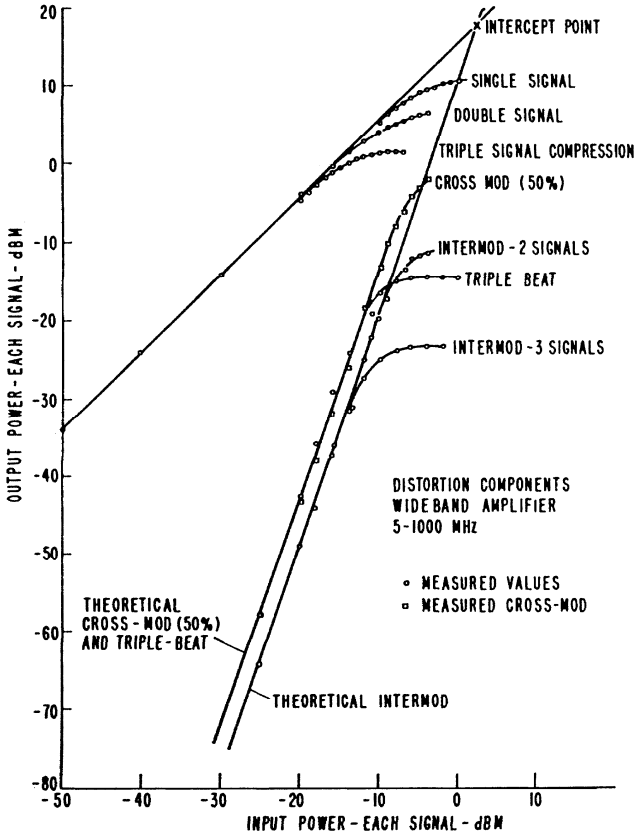


Figure 8a: Measured distortion components in a wideband (5 to 1000 MHz) amplifier. Figure 8b shows a magnified view of the gain-compression region [9].

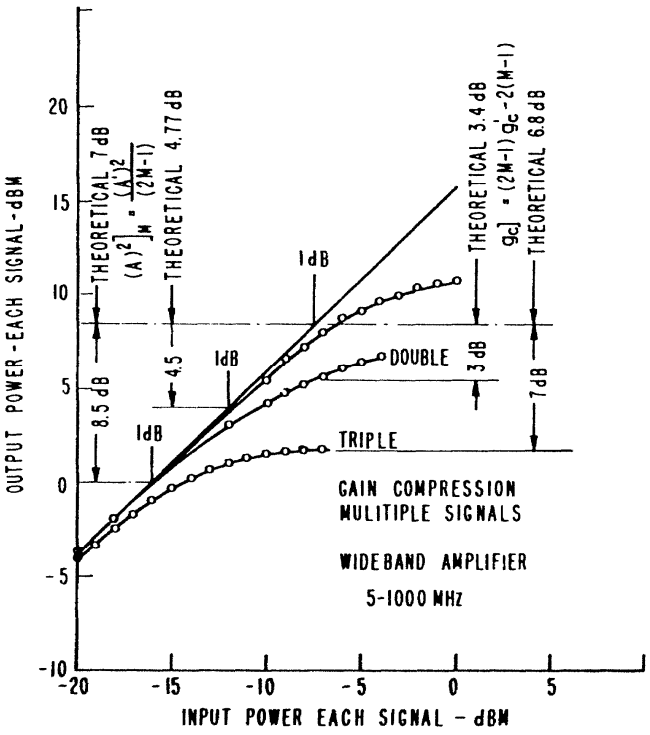


Figure 8b: Measured multiple-signal gain compression of the 5- to 1000-MHz amplifier [9].

How to determine that a conventional simulator tool is unsuitable? One type of breakdown is the question of convergence, the simulator will give an error advising the numerical problem, but mostly not giving a solution. The next problem is the missing components as shown above. Or the question about noise figure of amplifiers or phase noise of an oscillator cannot be answered by a SPICE program accurately.

To satisfy for our queries we have taken the following examples. The first one is a 4GHz amplifier designed using CMOS technology. One transistor is used for biasing and the other two forms a cascade for stability shown in Figure 9(a). The inductor in the source of the FET3 makes close matching of a low noise figure and a good S11 (power) matching possible. This is an application case for simulator testing, but cannot build like this practically.

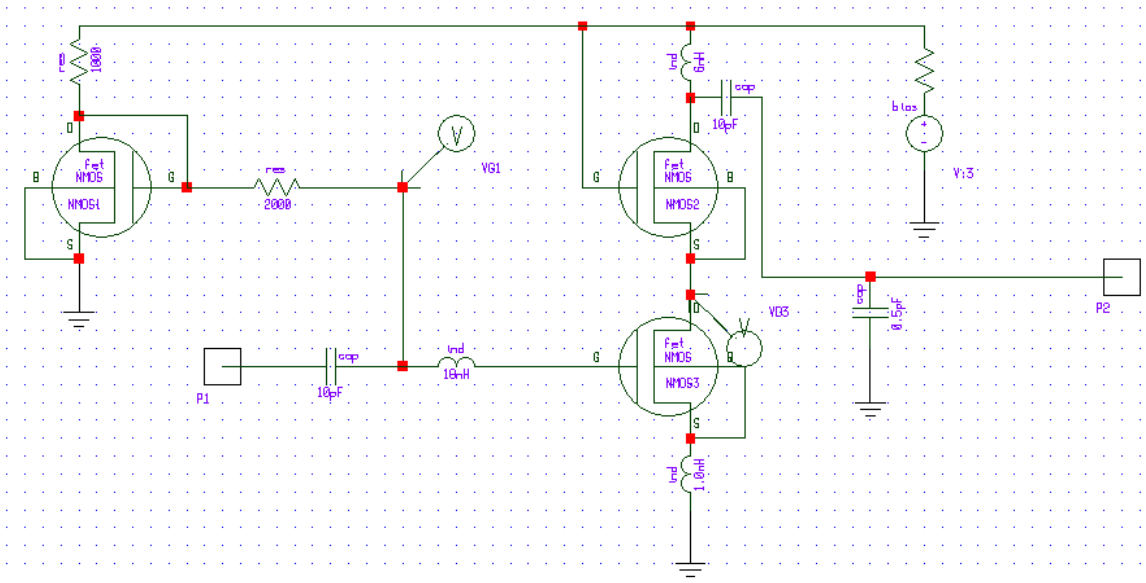


Figure 9 (a): Schematic of 4GHz Amplifier using CMOS technology

The 8dB gain and the roughly 1dB noise figure makes for a good performance. Key is that F_{MIN} and NF are close together at the resonant frequency. The correct noise figure calculation is beyond SPICE and only a few SPICE dialects can calculate all S-parameters.

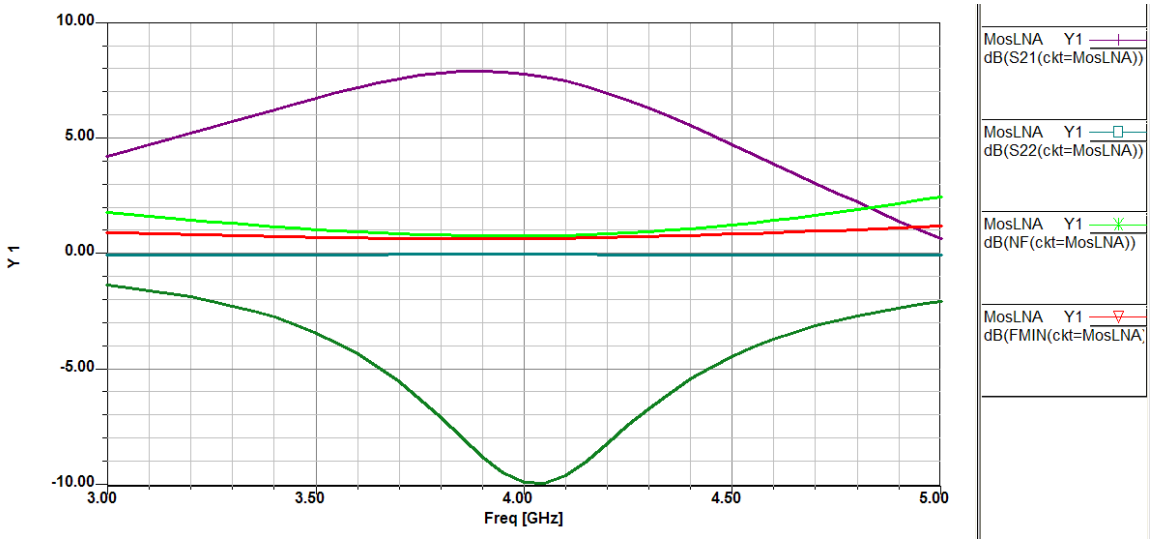


Figure 9 (b): Simulated plot showing the input matching (S11), output matching (S22), Gain(S21) and the F_{MIN} and NF in dB over the frequency range for the amplifier in figure 9(a).

The design of oscillators in SPICE does not give a reliable output frequency, and some of the latest SPICE programs resort to some approximation calculation. These are also more expensive then harmonic balance programs. The following oscillator is a good example designed for 1296MHz shown in figure 10(a), uses distributed printed elements and the user wants to know outpour power, harmonics and phase noise .

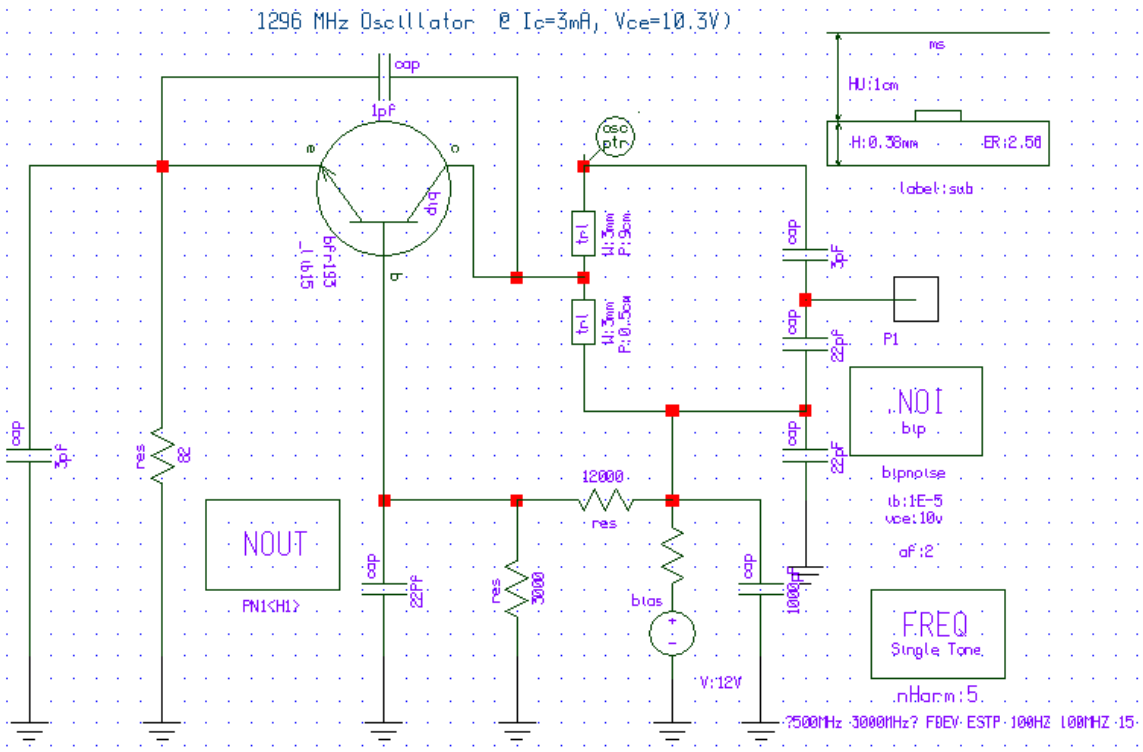


Figure 10 (a): Schematic of 1296MHz oscillator

The oscillator is a standard grounded base arrangement using transmission lines as resonators. The next two figures plotted in simulation tool show the predicted output power and harmonic content as well as the phase noise.

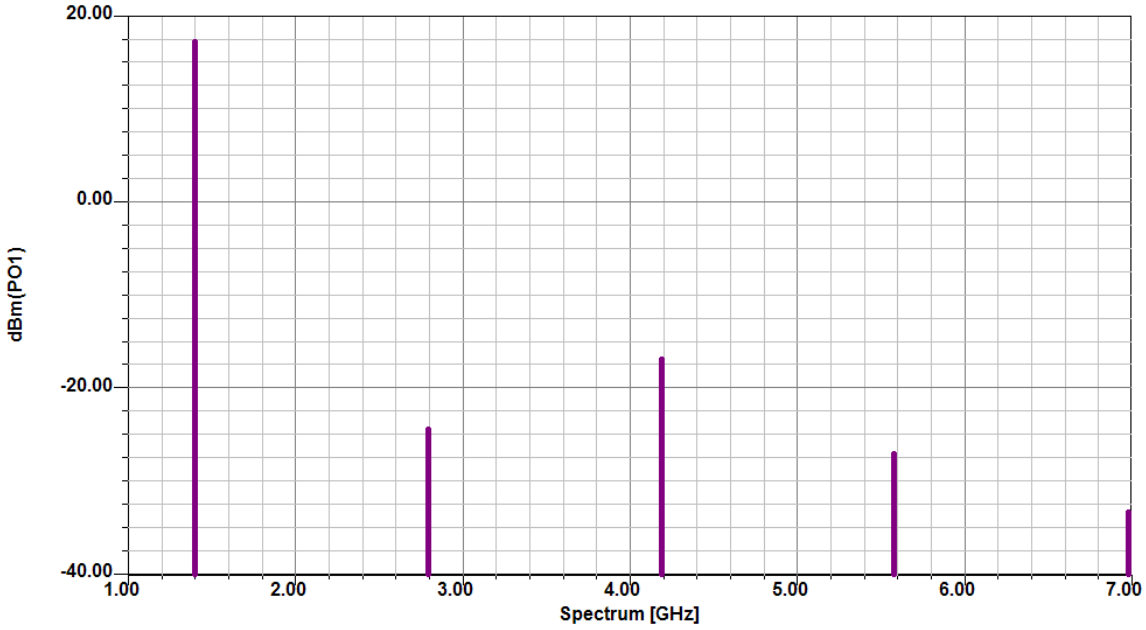


Figure 10 (b): Simulated plot showing the harmonic content and the predicted output power for the 1296 MHz oscillator

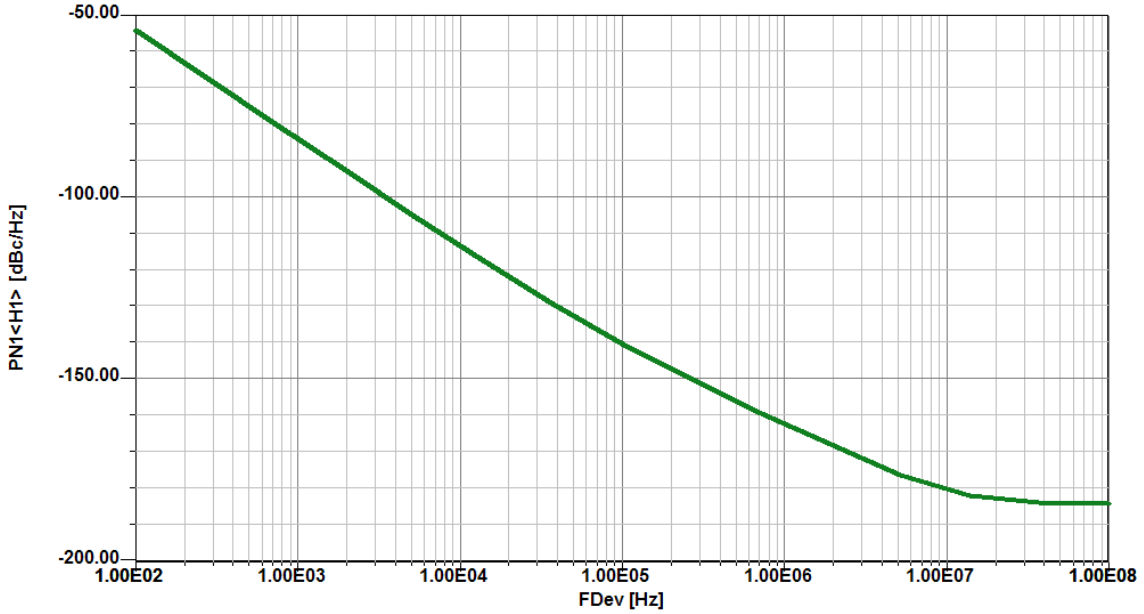


Figure 10 (c): Simulated plot showing the predicted phase noise for schematic in figure 10(a)

The following time domain analysis using the enhanced technique is a good example showing the progress. A microwave oscillator is keyed on and off and a transient analysis is performed. This is shown in the next figure.

When using the standard SPICE based on SPICE3, the initial calculation shows a wrong response after one iteration, see the following figure. It takes about 80 pulses (80th period of the pulsed drain voltage) until the simulation follows the Krylov-subspace based HB and gives a correct answer. (See Appendix 1 – Krylov-subspace) The speed improvement is 11 times faster and the required memory is about 10%

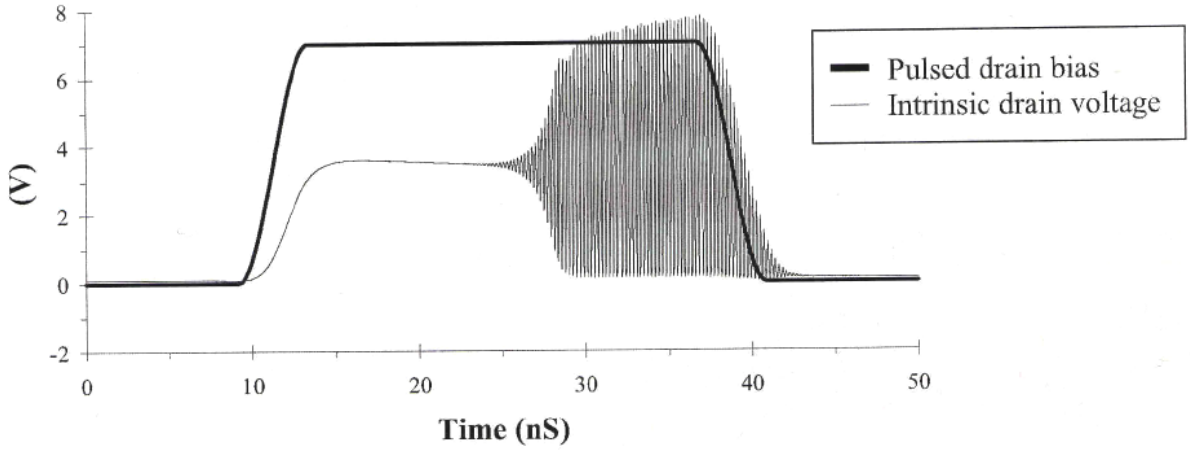


Figure 11 (a): Krylov-subspace based HB result

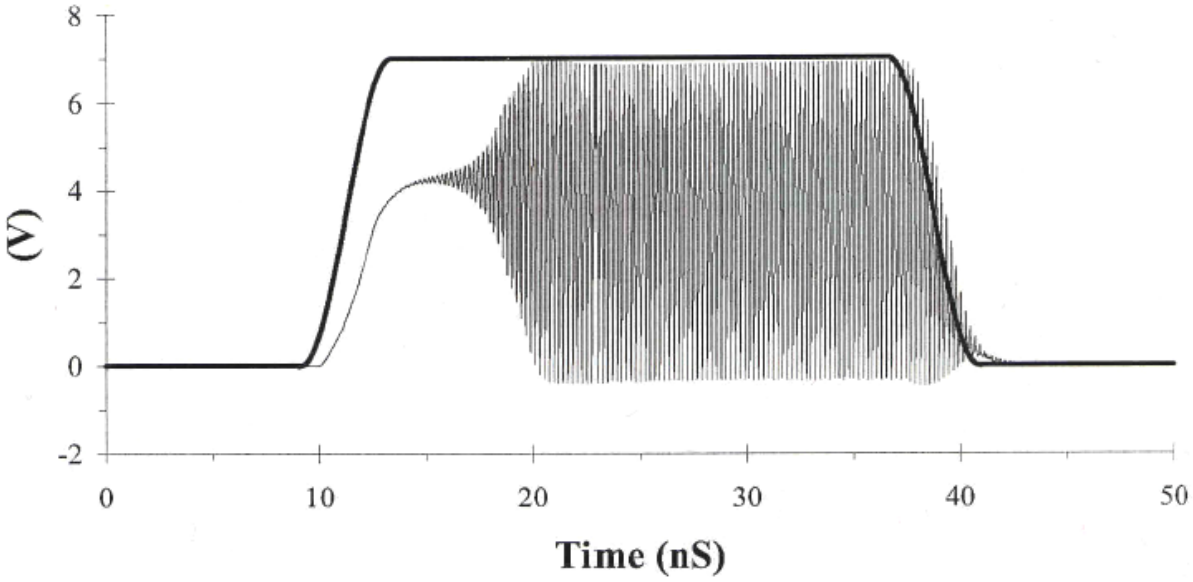


Figure 11 (b): Standard SPICE result

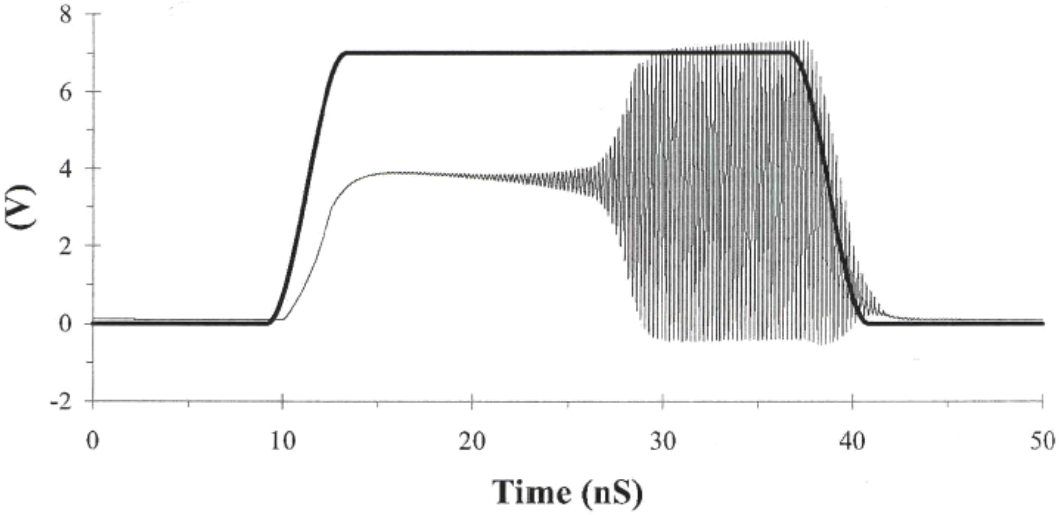


Figure 11 (c): 80 iterations later the result of the standard SPICE provides the same result as the Krylov-subspace solution.

The next picture, Figure 12, shows the transient performance relative to V_{be} as oscillation occurs. At the time t=0 the base emitter is the standard 600mV. After a short time oscillation tries to start and there is a DC shift, based on the biasing network impedance and capacitance. At the steady state value, there is the RF riding on a reduced DC voltage which is responsible of the amplitude stabilization. This is a nice application for RF analysis. Using quartz crystals, this can be a high as a few seconds based on the Q of 1 million or more.

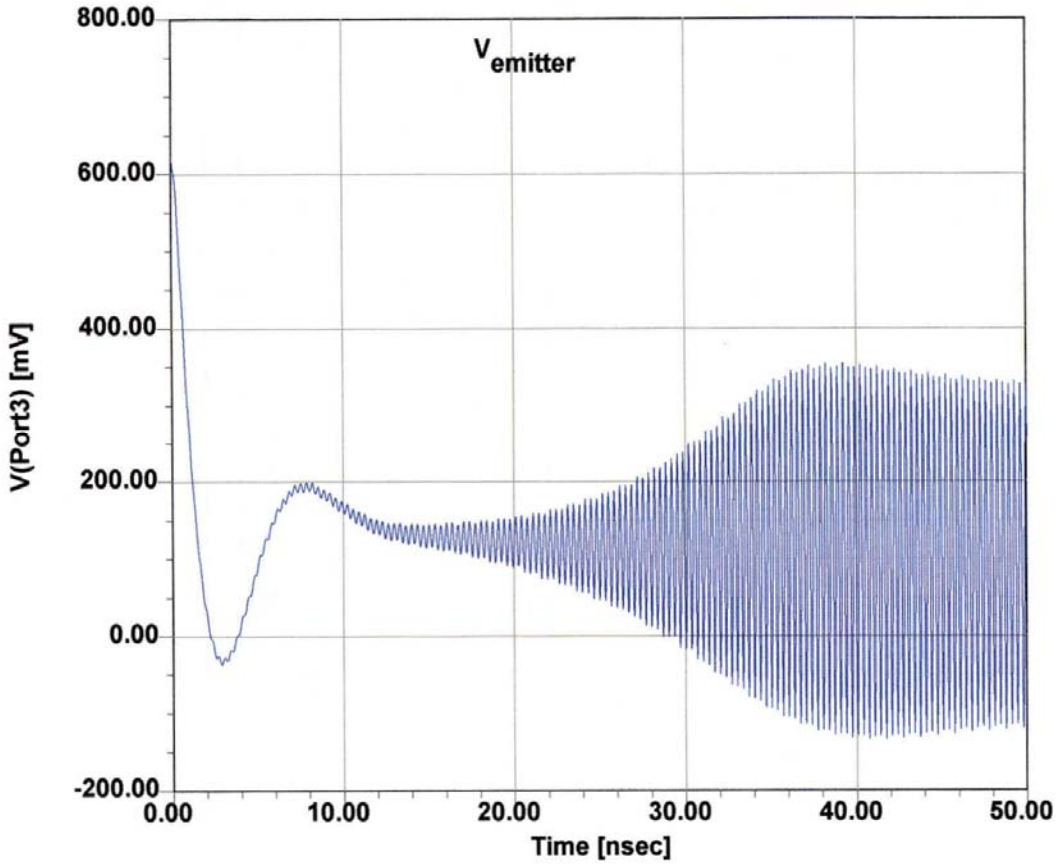


Figure 12: Transient response relative to V_{be}

2. RF SIMULATION TOOLS AND TECHNOLOGY

2.1 RF simulator technology overview

RF simulators fall in the categories of SPICE, HB programs and EM (electromagnetic) programs The EM simulators are more exotic programs. Two types are common, the 2 (2.5)D and the full 3D versions. They are use to analyze planar circuits, including Vias (ground connections) or wrap arounds, top to ground plane side connections. The 3D simulator helps with transitions or other crosstalk or resonant conditions. We will not address these here as they go far beyond the SPICE concept.

The modern HB programs have found better solutions for both handling a very large number of transistors, up to 1 million and even more and handle the math solutions now much more efficient. Memory management and solving non-linear equations for transient analysis are some of the key factors. HB analysis performs steady-state analysis of periodically excited circuits. The circuit to be analyzed is split into linear and nonlinear subcircuits. The linear subcircuit is calculated in the frequency domain. Features of this aspect of the HB process include:

- Use of distributed models in the spectral domain
- Matrix formulation that can enable reduction of internal nodes
- Major speed advantage
- Straightforward intermodulation and mixer analysis

The nonlinear subcircuit is calculated in the time domain. Features of this aspect of the HB process include:

- Nonlinear models derived directly from device physics
- Intuitive, easy and logical circuit representation

Figure 13 diagrams this approach for a MESFET amplifier. Figure 14 charts a general-purpose nonlinear design algorithm that includes optimization. Modern analysis tools that must provide accurate phase-noise calculation should be based on the principle of harmonic balance. In Section 8, Application 2, Figure 46 shows a BJT microwave oscillator entered into the schematic-capture module of a commercially available HB simulator (Ansoft Serenade 8.0); Figure 47 shows this oscillator's simulated phase noise. By the way, HB analysis can also handle mixers.

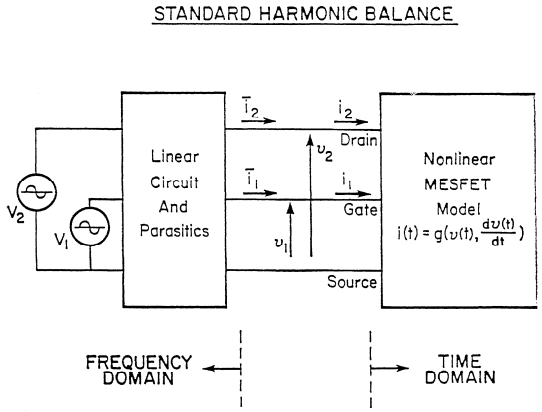


Figure 13: MESFET circuit partitioned into linear and nonlinear subcircuits for harmonic-balance analysis. Applied gate and drain voltages, and relevant terminal voltages and currents, are indicated.

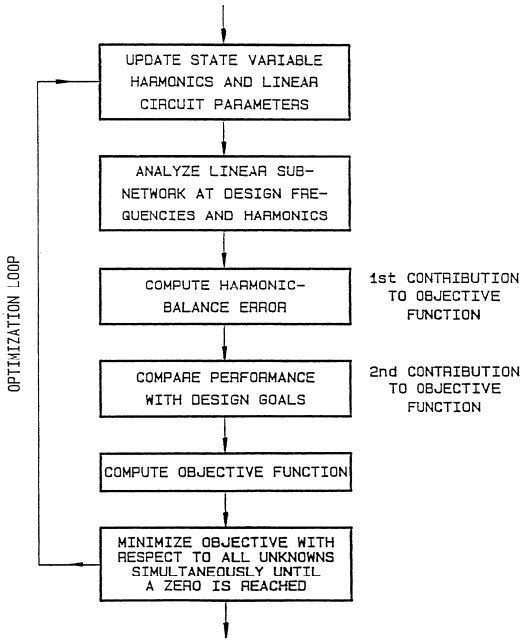


Figure 14: Flowchart of a general-purpose harmonic-balance design algorithm that includes optimization.

Transient analysis in microwave oscillators includes problems of primary importance such as oscillation buildup at bias turn-on and frequency settling in VCOs. Until now this class of problems has been tackled by two families of algorithms, i.e., either rigorous but computationally inefficient time-domain methods, or fast but approximate envelope-oriented techniques. It has been shown that an excellent trade-off retaining the advantages of both the above without significant shortcomings may be achieved by Krylov-subspace based inexact-Newton harmonic-balance (INHB) analysis.

3. EXAMPLES OF RF SIMULATION TOOLS, PROFESSIONAL AND STUDENT/HOBBYIST

PSPICE: This popular version of SPICE, available from Orcad (now Cadence) runs under the PC and Macintosh platforms. An evaluation version, which can handle small circuits with up to 10 transistors, is freely available. For a full fledged version or for more information, please contact Orcad. AIM-SPICE is a pc-version of SPICE with a revised user interface, simulation control, and with extra models. A student version can be downloaded. A complete list of all SPICE offerings (and software downloads for a wide range of platforms).

This student version, from my experience, is the best on the market. There are a number of PC-based SPICE programs in the $1000 region but they are more for switching power supplies and logic circuit optimization than RF. Here are two important programs:

www.intusoft.com/demos.htm

www.spectrum-soft.com/index.shtm

HSPICE:

- RF and High Speed Simulation
- Best RF Simulator for PLL and VCO applications
- Most Accurate RF Simulator
- Fastest RF Simulator
- High Capacity RF Simulator, 10000+ transistors with both Harmonic Balance and Shooting Newton algorithms
- Comprehensive solution simulates low noise amplifiers, power amplifiers, filters, AGC circuits, oscillators, mixers, multipliers, modulators, demodulators, and VCOs.

Agilent, AWR and Ansys offer very modern CAD tools, mixed mode, and they combine the concept of SPICE and the advanced technologies. The question remains of where to obtain RF models?

4. SPECIFIC AREAS OF CONCERN

4.1 Noise

Here we need to look at the noisy two-port description and the application of the noise correlation matrix, which only the harmonic balanced based or hybrid (SPICE) programs have.

Noisy Two-Port Description: Based on the convention by Rothe and Dalke [12], any linear two-port can be in the form shown in Figure 15(a, b, c). This general case of a noisy two-port can be redrawn showing noise sources at the input and at the output. Figure 15(b) shows this in admittance form and Figure 15(c) in impedance form. The internal noise sources are assumed to produce very small currents and voltages, and we assume that linear two-port equations are valid. The internal noise contributions have been expressed by using external noise sources:

$$I_1 = y_{11}V_1 + y_{12}V_2 + I_{K1}$$
$$I_2 = y_{21}V_1 + y_{22}V_2 + I_{K2} \qquad (12)$$

$$V_1 = z_{11}I_1 + z_{12}I_2 + V_{L1}$$
$$V_2 = z_{21}I_1 + z_{22}I_2 + V_{L2} \qquad (13)$$

where the external noise sources are I_{K1}, I_{K2}, V_{L1}, and V_{L2}.

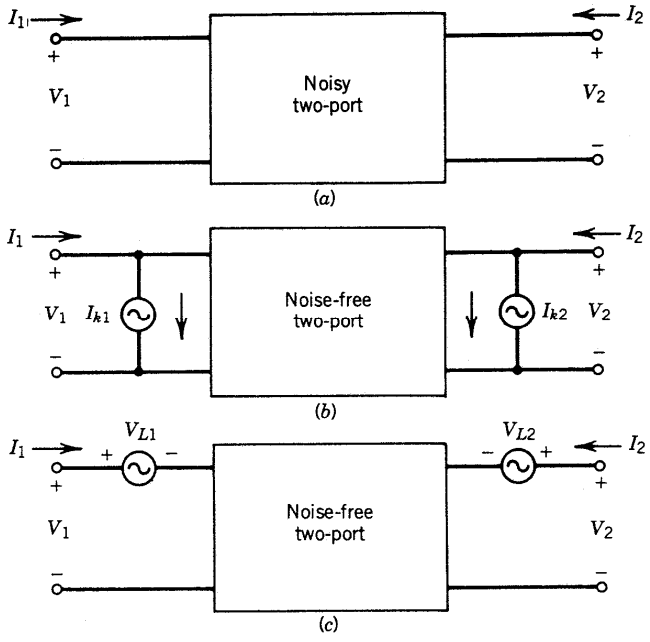


Figure 15: Noise linear two-ports: (a) general form; (b) admittance form; (c) impedance form.

Since we want to describe our noisy circuit in terms of the noise figure, the *ABCD*-matrix description will be more convenient since it refers both noise sources to the input of the two-port [13]. This representation is given below (note the change in direction of I_2):

$$V_1 = AV_2 + BI_2 + V_A$$
$$I_1 = CV_2 + DI_2 + I_A \tag{14}$$

where V_A and I_A are the external noise sources.

It is important to remember that all of these matrix representations are interrelated. For example, the noise noises for the *ABCD*-matrix description can be obtained from the *z*-matrix representation shown in (13). This transformation is

$$V_A = -\frac{I_{K2}}{y_{21}} = V_{L1} - \frac{V_{L2} z_{11}}{z_{21}} \tag{15}$$

$$I_A = I_{K1} - \frac{I_{K2} y_{11}}{y_{21}} = -\frac{V_{L2}}{z_{21}} \tag{16}$$

The *ABCD* representation is particularly useful based on the fact that it allows us to define a noise temperature for the two-port referenced to input. The two-port itself (shown in Figure 16) is assumed to be noise-free.

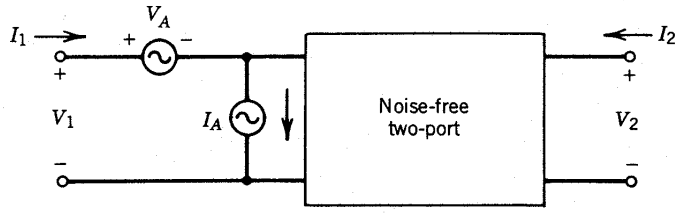


Figure 16: Chain-matrix form of linear noisy two-ports.

In the past, *z* and *y* parameters have been used, but in microwave applications it has become common to use *S*-parameter definitions. This is shown in Figure 17. The previous equations can be rewritten in their new form using *S* parameters:

$$\begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} + \begin{bmatrix} b_{n1} \\ b_{n2} \end{bmatrix} \tag{17}$$

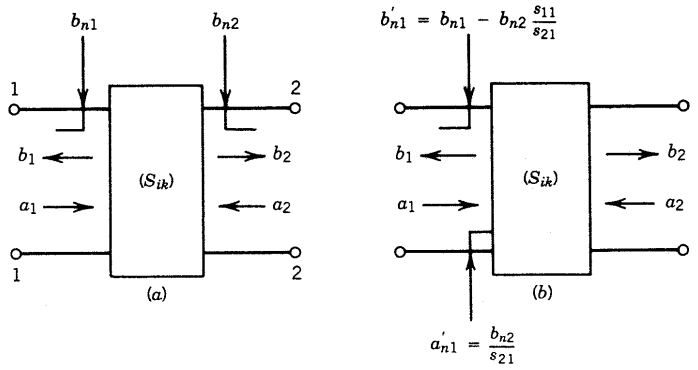


Figure 17: *S*-parameter form of linear noisy two-ports.

There are different physical origins for the various sources of noise. Typically, thermal noise is generated by resistances and loss in the circuit or transistor, whereas shot noise is generated by current flowing through semiconductor junctions and vacuum tubes. Since these many sources of noise are represented by only two noise sources at the device input, the two equivalent noise sources are often a complicated combination of the circuit internal noise sources. Often, some fraction of V_A and I_A is related to the same noise source. This means that V_A and I_A are not independent in general. Before we can use V_A and I_A to calculate the noise figure of the two-port, we must calculate the correlation between the V_A and I_A shown in Figure 16.

The noise source V_A represents all the device noise referred to the input when the generator impedance is zero; that is, when the input is short-circuited. The noise source I_A represents all the device noise referred to the input when the generator admittance is zero; that is, the input in open-circuited.

The correlation of these two noise sources considerably complicates analysis. By defining a correlation admittance, we can simplify the mathematics and get some physical intuition for the relationship between noise figure and generator admittance. Since some fraction of I_A will be correlated with V_A, we split I_A into correlated and uncorrelated parts as follows:

$$I_A = I_n + I_u \tag{18}$$

I_u is the part of I_A uncorrelated with V_A. Since I_n is correlated with V_A, we can say that I_n is proportional to V_A and the constant of proportionality is the correlation admittance.

$$I_n = Y_{\text{cor}} V_A \tag{19}$$

This leads us to

$$I_A = Y_{\text{cor}} V_A + I_u \tag{20}$$

The following derivation of noise figure will use the correlation admittance. Y_{cor} is not a physical component located somewhere in the circuit. Y_{cor} is a complex number derived by correlating the random variables I_A and V_A. To calculate Y_{cor}, we multiply each side of (20) by V_A^* and average the result. This gives

$$\overline{V_A^* I_A} = Y_{cor}\overline{V_A^2} \tag{21}$$

where the I_u term averaged to zero since it was uncorrelated with V_A. The correlation admittance is thus given by

$$Y_{cor} = \frac{\overline{V_A^* I_A}}{\overline{V_A^2}} \tag{22}$$

Often, people use the term "correlation coefficient." This normalized quantity is defined as

$$c = \frac{\overline{V_A^* I_A}}{\sqrt{\overline{V_A^2}\,\overline{I_A^2}}} = Y_{cor}\sqrt{\frac{\overline{V_A^2}}{\overline{I_A^2}}} \tag{23}$$

Note that the dual of this admittance description is the impedance description. Thus the impedance representation has the same equations as above with Y replaced by Z, I replaced by V and V replaced by I.

V_A and I_A represent internal noise sources in the form of a voltage source acting in series with the input voltage and a source of current flowing in parallel with the input current. This representation conveniently leads to the four noise parameters needed to describe the noise performance of the two-port. Again using the Nyquist formula, the open-circuit voltage of a resistor at the temperature T is

$$\overline{V_A^2} = 4kTRB \tag{24}$$

This voltage is a mean-square fluctuation (or spectral density). It is the method used to calculate the noise identity. We could also define a noise equivalent resistance for a noise voltage as

$$R_n = \frac{\overline{V_A^2}}{4kTB} \tag{25}$$

The resistor R_n is not a physical resistor but can be used to simulate different portions of the noise equivalent circuit.

In a similar manner a mean-square current fluctuation can be represented in terms of an equivalent noise conductance G_n, which is defined by

$$G_n = \frac{\overline{I_A^2}}{4kTB} \tag{26}$$

and

$$G_u = \frac{\overline{I_u^2}}{4kTB} \tag{27}$$

for the case of the uncorrelated noise component. The input generator to the two-port has a similar contribution.

$$G_G = \frac{I_G^2}{4kTB} \tag{28}$$

with Y_G being the generator admittance and G_G being the real part. With the definition of F above, we can write

$$F = 1 + \left| \frac{I_A + Y_G V_A}{I_G} \right|^2 \tag{29}$$

The use of the voltage V_A and the current I_A has allowed us to combine all the effects of the internal noise sources. We can use the previously defined (22) correlation admittance, $Y_{cor} = G_{cor} + jB_{cor}$, to simplify (29). First, we determine the total noise current:

$$\overline{I_A^2} = 4kT(|Y_{cor}|^2 R_n + G_u)B \tag{30}$$

where R_n and G_u are defined in (25) and (26). The noise factor can now be determined.

$$F = 1 + \frac{G_u}{G_g} + \frac{R_n}{G_g}\left[(G_G + G_{cor})^2 + (B_G + B_{cor})^2\right] \tag{31}$$

$$F = 1 + \frac{R_u}{R_g} + \frac{G_n}{R_g}\left[(R_G + R_{cor})^2 + (X_G + X_{cor})^2\right] \tag{32}$$

The noise factor is a function of various elements, and the optimum impedance for the best noise figure can be determined by minimizing F with respect to generator reactance and resistance. This gives

$$R_{0n} = \sqrt{\frac{R_u}{G_n} + R_{\text{cor}}^2} \tag{33}$$

$$X_{0n} = -X_{\text{cor}} \tag{34}$$

and

$$F_{\min} = 1 + 2G_n R_{\text{cor}} + 2\sqrt{R_u G_n + (G_n R_{\text{cor}})^2} \tag{35}$$

(To distinguish between optimum noise and optimum power, we have introduced the convention 0*n* instead of the more familiar abbreviation *opt*.) At this point we see that the optimum condition for minimum noise figure is not a conjugate power match at the input port. We can explain this by recognizing that the noise source V_A and I_A represent all the two-port noise, not just the thermal noise of the input port. We should observe that the optimum generator susceptance, $-X_{\text{corr}}$, will minimize the noise contribution of the two noise generators.

In rearranging for conversion to *S* parameters, we write

$$F = F_{\min} + \frac{g_n}{R_G} \left| Z_G - Z_{0n} \right|^2 \tag{36}$$

$$F = F_{\min} + \frac{R_n}{G_G} \left| Y_G - Y_{0n} \right|^2 \tag{37}$$

From the definition of the reflection coefficient,

$$\Gamma_G = \frac{Y_0 - Y_G}{Y_0 + Y_G} \tag{38}$$

and with

$$g_G = \frac{G_G}{Y_0} \tag{39}$$

$$r_n = \frac{R_n}{Z_0} \tag{40}$$

the normalized equivalent noise resistance

$$F = F_{\min} + \frac{4r_n \left| \Gamma_G - \Gamma_{0n} \right|^2}{g_G \left(1 - \left| \Gamma \right|^2\right) \left| 1 + \Gamma_{0n} \right|^2} \tag{41}$$

$$r_n = (F_{50} - F_{\min}) \frac{\left| 1 + \Gamma_{0n} \right|^2}{4 \left| \Gamma_{0n} \right|^2} \tag{42}$$

$$\Gamma_{0n} = \frac{Z_{0n} - Z_0}{Z_{0n} + Z_0} \tag{43}$$

The noise performance of any linear two-port can now be determined if the values of the four noise parameters, F_{min}, $r_n = R_n/50$, and Γ_{0n} are known. Figure 18 shows the noise factor of a high-frequency transistor as a function of B_g for G_g = constant and as a function of G_g for $B_g = B_{opt}$.

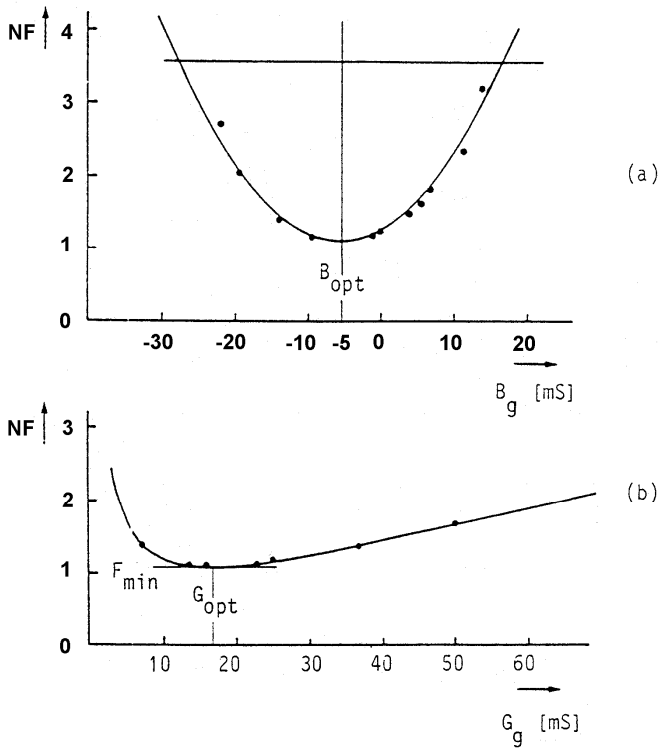


Figure 18: Noise factor in high-frequency BJTs for f = 600 MHz: (a) as a function of B_g for G_g = constant; (b), as a function of G_g for $B_g = B_{opt}$.

4.2 Noise figure of cascaded networks.

In a system with many circuits connected in cascade (Figure 19), we must consider the contributions of the various circuits. In considering the equivalent noise resistor R_n in series with the input circuit.

$$F = \frac{R_G + R_n}{R_G} \tag{44}$$

$$F = 1 + \frac{R_n}{R_G} \tag{45}$$

The excess noise added by the circuit is R_n/R_G.

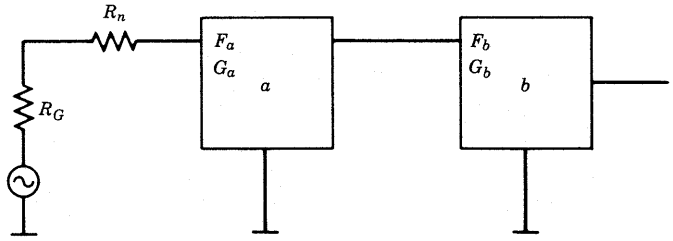


Figure 19: Cascaded noisy two-ports with the noise figures F_a and F_b and the gain figure G_a and G_b.

In considering two cascaded circuits *a* and *b*, by definition the available noise at the output of *b* is

$$N_{ab} = F_{ab} G_{ab} kTB \tag{46}$$

with *B* the equivalent noise bandwidth in which the noise is measured. The total available gain *G* is the product of the individual available gains, so

$$N_{ab} = F_{ab} G_a G_b kTB \tag{47}$$

The available noise from network *a* at the output of network *b* is

$$N_{a/b} = N_a G_b = F_a G_a G_b kTB \tag{48}$$

The available noise added by network *b* (its excess noise) is

$$N_{b/b} = (F_b - 1) G_b kTB \tag{49}$$

The total available noise N_{ab} is the sum of the available noise contributed by the two networks:

$$\begin{aligned} N_{ab} &= N_{a/b} + N_{b/b} = F_a G_a G_b kTB + (F_b - 1) G_b kTB \\ &= \left(F_a + \frac{F_b - 1}{G_a} \right) G_a G_b kTB \end{aligned} \tag{50}$$

$$F_{ab} = F_a + \frac{F_b - 1}{G_a} \tag{51}$$

For any number of circuits, this can be extended to be

$$F = F_1 + \frac{F_2 - 1}{G_1} + \frac{F_3 - 1}{G_1 G_2} + \frac{F_4 - 1}{G_1 G_2 G_3} + \cdots \tag{52}$$

When considering a long chain of cascaded amplifiers, there will be a minimum noise figure achievable for this chain. This is a figure of merit and was proposed by Haus and Adler [6]. It is calculated by rearranging (52).

$$(F_{\text{tot}})_{\text{min}} = (F_{\text{min}} - 1) + \frac{F_{\text{min}} - 1}{G_A} + \frac{F_{\text{min}} - 1}{G_A^2} + \cdots + 1 \tag{53}$$

where F_{min} is the minimum noise figure for each stage and G_A is the available power gain of the identical stages. Using

$$\frac{1}{1-X} = 1 + X + X^2 + \cdots \tag{54}$$

we find a quantity ($F_{tot} - 1$), which is defined as the noise measure M. The minimum noise measure

$$(F_{tot})_{min} - 1 = \frac{F_{min} - 1}{1 - 1/G_A} = M_{min} \tag{55}$$

refers to the noise of an infinite chain of optimally tuned, low-noise stages, so it represents a lower limit on the noise of an amplifier.

The minimum noise measure M_{min} is an invariant parameter and is not affected by feedback. It is somewhat similar to a gain-bandwidth product, in its use as a system invariant. The minimum noise measure is achieved when the amplifier is tuned for the available power gain and $\Gamma_G = \Gamma_{0n}$, given by (43).

4.3 Noise correlation in linear two-ports using correlation matrices

Noise correlation matrices form a general technique for calculating noise in *n*-port networks. Haus and Adler have described the theory behind this technique [14]. In 1976, Hillbrand and Russer published equations and transformations that aid in supplying this method to two-port CAD [9].

This method is useful because it forms a base from which we can rigorously calculate the noise of linear two-ports combined in arbitrary ways. For many representations, the method of combining the noise parameters is as simple as that for combining the circuit element matrices. In addition, noise correlation matrices can be used to calculate the noise in linear frequency conversion circuits. The following is an introduction to this subject.

Linear, noisy two-ports can be modeled as a noise-free two-port with two additional noise sources. These noise sources must be chosen so that they add directly to the resulting vector of the representation, as shown in (56) and (57) and Figure 17.

$$\begin{bmatrix} I_1 \\ I_2 \end{bmatrix} = \begin{bmatrix} y_{11} & y_{12} \\ y_{21} & y_{22} \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} + \begin{bmatrix} i_1 \\ i_2 \end{bmatrix} \tag{56}$$

$$\begin{bmatrix} V_1 \\ V_2 \end{bmatrix} = \begin{bmatrix} z_{11} & z_{12} \\ z_{21} & z_{22} \end{bmatrix} \begin{bmatrix} I_1 \\ I_2 \end{bmatrix} + \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \tag{57}$$

where the *i* and *v* vectors indicate noise sources for the *y* and *z* representations, respectively. This two-port example can be extended to *n*-ports in a straightforward, obvious way.

Since the noise vector for any representation is a random variable, it is much more convenient to work with the noise correlation matrix. The correlation matrix gives us deterministic numbers to calculate with. The correlation matrix is formed by taking the mean value of the outer product of the noise vector. This is equivalent to multiplying the noise vector by its adjoint (complex conjugate transpose) and averaging the result:

$$\langle \bar{i}\bar{i}^{+} \rangle = \begin{bmatrix} i_1 \\ i_2 \end{bmatrix} \begin{bmatrix} i_1^* & i_2^* \end{bmatrix} = \begin{bmatrix} \langle i_1 i_1^* \rangle & \langle i_1 i_2^* \rangle \\ \langle i_1^* i_2 \rangle & \langle i_2 i_2^* \rangle \end{bmatrix} = [C_y] \tag{58}$$

where the angular brackets denote the average value.

Note that the diagonal terms are the "power" spectrum of each noise source and the off-diagonal terms are complex conjugates of each other and represent the cross "power" spectrums of the noise sources. "Power" is used because these magnitude-squared quantities are proportional to power.

To use these correlation matrices in circuit analysis, we must know how to combine them and how to convert them between various representations. An example using *y* matrices will illustrate the method for combining two-ports and their correlation matrices. Given two matrices *y* and *y'*, when we parallel them we have the same port voltages, and the terminal currents add (Figure 20).

$$\begin{aligned} I_1 &= y_{11}V_1 + y_{12}V_2 + y'_{11}V_1 + y'_{12}V_2 + i_1 + i'_1 \\ I_2 &= y_{21}V_1 + y_{22}V_2 + y'_{21}V_1 + y'_{22}V_2 + i_2 + i'_2 \end{aligned} \tag{59}$$

or

$$\begin{bmatrix} I_1 \\ I_2 \end{bmatrix} = \begin{bmatrix} y_{11} + y'_{11} & y_{12} + y'_{12} \\ y_{21} + y'_{21} & y_{22} + y'_{22} \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} + \begin{bmatrix} i_1 + i'_1 \\ i_2 + i'_2 \end{bmatrix} \tag{60}$$

Here we can see that the noise current vectors add just as the *y* parameters add. Converting the new noise vector to a correlation matrix yields

$$\langle \bar{i}_{\text{new}} \bar{i}^{+}_{\text{new}} \rangle = \left\langle \begin{bmatrix} i_1 + i'_1 \\ i_2 + i'_2 \end{bmatrix} \begin{bmatrix} i_1^* + i_1'^* & i_2 i_2'^* \end{bmatrix} \right\rangle \tag{61}$$

$$= \begin{bmatrix} \langle i_1 i_1^* \rangle + \langle i'_1 i'^*_1 \rangle & \langle i_1 i_2^* \rangle + \langle i'_1 i'^*_2 \rangle \\ \langle i_2 i_1^* \rangle + \langle i'_2 i'^*_1 \rangle & \langle i_2 i_2^* \rangle + \langle i'_2 i'^*_2 \rangle \end{bmatrix} \tag{62}$$

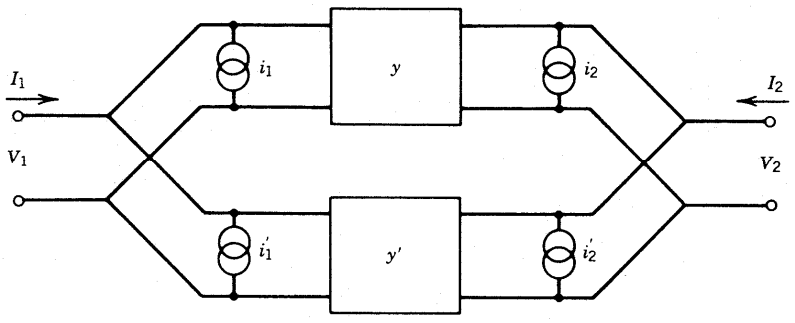


Figure 20: Parallel combination of two-ports using *y* parameters.

The noise sources from different two-ports must be uncorrelated, so there are no cross products of different two-ports. By inspection, (60) is just the addition of the correlation matrices for the individual two-ports, so

$$[C_{y\text{new}}] = [C_y] + [C_y'] \tag{63}$$

The same holds true for *g*, *h*, and *z* parameters, but *ABCD* parameters have the more complicated form shown below. If

$$[A_{\text{new}}] = [A][A'] \tag{64}$$

then

$$[C_{A\text{new}}] = [C_A] + [A]\,[C_{A'}]\,[A]^{+} \tag{65}$$

The transformation of one representation to another is best illustrated by an example. Let us transform the correlation matrix for a *Y* representation to a *Z* representation. Starting with

$$\begin{bmatrix} I_1 \\ I_2 \end{bmatrix} = [Y] \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} + \begin{bmatrix} i_1 \\ i_2 \end{bmatrix} \tag{66}$$

we can move the noise vector to the left side and invert *y*:

$$\begin{bmatrix} V_1 \\ V_2 \end{bmatrix} = [Y^{-1}] \begin{bmatrix} I_1 - i_1 \\ I_2 - i_2 \end{bmatrix} = [Y^{-1}] \begin{bmatrix} I_1 \\ I_2 \end{bmatrix} + [Y^{-1}] \begin{bmatrix} -i_1 \\ -i_2 \end{bmatrix} \tag{67}$$

Since $(Y)^{-1} = (Z)$, we have

$$\begin{bmatrix} V_1 \\ V_2 \end{bmatrix} = [Z] \begin{bmatrix} I_1 \\ I_2 \end{bmatrix} + [Z] \begin{bmatrix} -i_1 \\ -i_2 \end{bmatrix} \tag{68}$$

so

$$= [Z] \begin{bmatrix} -i_1 \\ -i_2 \end{bmatrix} = [T_{yz}] \begin{bmatrix} -i_1 \\ -i_2 \end{bmatrix} \tag{69}$$

where the signs of i_1 and i_2 are superfluous since they will cancel when the correlation matrix is formed. Here the transformation of the *Y* noise current vector to the *Z* noise voltage vector is done simply by multiplying by (*Z*). Other transformations are shown in Table 1.

Table 1: Noise Matrix $T_{\alpha\beta}$ Transformation

		Original Form (α Form)		
		Y	*Z*	*A*
Resulting Form (β Form)	*Y*	$\begin{matrix}1 & 0\\ 0 & 1\end{matrix}$	$\begin{matrix}y_{11} & y_{12}\\ y_{21} & y_{22}\end{matrix}$	$\begin{matrix}-y_{11} & 1\\ -y_{21} & 0\end{matrix}$
	Z	$\begin{matrix}z_{11} & z_{12}\\ z_{21} & z_{22}\end{matrix}$	$\begin{matrix}1 & 0\\ 0 & 1\end{matrix}$	$\begin{matrix}1 & -z_{11}\\ 0 & -z_{21}\end{matrix}$
	A	$\begin{matrix}0 & A_{12}\\ 1 & A_{22}\end{matrix}$	$\begin{matrix}1 & -A_{11}\\ 0 & -A_{21}\end{matrix}$	$\begin{matrix}1 & 0\\ 0 & 1\end{matrix}$

To form the noise correlation matrix, we gain from the mean of the outer product:

$$\langle vv^+ \rangle = \begin{bmatrix} \langle v_1 v_1^* \rangle & \langle v_1 v_2^* \rangle \\ \langle v_1^* v_2 \rangle & \langle v_2 v_2^* \rangle \end{bmatrix} = [Z] \left\langle \begin{bmatrix} i_1 \\ i_2 \end{bmatrix} [i_1^* \quad i_2^*] \right\rangle [Z]^+ \tag{70}$$

or

$$[C_z] = [Z]\,[C_y]\,[Z]^+ \tag{71}$$

where

$$v^+ = [i_1^* \quad i_2^*]\,[Z]^+ \tag{72}$$

This is called a congruence transformation. The key to all of these derivations is the construction of a correlation matrix from the noise vector, as shown in (60). These correlation matrices may easily be derived from the circuit matrices of passive circuits with only thermal noise sources. For example,

$$[C_z] = 2kT\Delta f\,\mathrm{Re}([Z]) \quad \text{and} \tag{73}$$

$$[C_y] = 2kT\Delta f\,\mathrm{Re}([Y]) \tag{74}$$

The $2kT$ factor comes from the double-sided spectrum of thermal noise. The correlation matrix from the *ABCD* matrix may be related to the noise figure, as shown by Hillbrand and Russer [9]. We have

$$F = 1 + \frac{\overline{Y}[C_a]\,\overline{Y}^+}{2kT\,\mathrm{Re}(Y_G)} \tag{75}$$

where

$$\bar{Y} = \begin{bmatrix} Y_G \\ 1 \end{bmatrix}$$

Expressing the noise factor in terms of the correlation matrix, here is a complete formula:

$$F = 1 + \frac{C_{22}^{A}(f) + 2\ \mathrm{Re}\{Y_g(f)\ C_{12}^{A}(f)\} + |Y_g(f)|^2\ C_{11}^{A}(f)}{2\ k\ T_0\ \mathrm{Re}\{Y_g(f)\}} \tag{76}$$

Once we transform this in the *Y* parameter form, we obtain the following equation:

$$F(f) = F_{\min}(f) + \frac{R_n(f)\,|\,Y_{opt}(f) - Y_g(f)\,|^2}{\mathrm{Re}\{Y_g(f)\}} \tag{77}$$

It should be noted that all these values are frequency-dependent as expressed in this equation. The *ABCD* correlation matrix can be written in terms of the noise-figure parameters as (double-sided spectrum)

$$[C_a] = 2kT \begin{bmatrix} R_n & \dfrac{F_0 - 1}{2} - R_n Y_{0n}^* \\ \dfrac{F_0 - 1}{2} - R_n Y_{0n} & R_n |\,Y_{0n}\,|^2 \end{bmatrix} \tag{78}$$

The noise correlation matrix method forms an easy and rigorous technique for handling noise in networks. This technique allows us to calculate the total noise for complicated networks by combining the noise matrices of subcircuits. It should be remembered that although noise correlation matrices apply to *n*-port networks, noise-figure calculations apply only to pairs of ports. The parameters of the C_a matrix can be used to give the noise parameters:

$$Y_{0n} = \sqrt{\frac{C_{ii*}}{C_{uu*}} - \left[\mathrm{Im}\left(\frac{C_{ui*}}{C_{uu*}}\right)\right]^2} + j\ \mathrm{Im}\left(\frac{C_{ui*}}{C_{uu*}}\right) \tag{79}$$

$$F_0 = 1 + \frac{C_{ui*} + C_{uu*} Y_{0n}^*}{kT} \tag{80}$$

$$R_n = C_{uu*} \tag{81}$$

5. TRANSMISSION LINE MODELS AND OTHER MICROWAVE CONNECTING ELEMENTS

In order to decouple the transistor from the biasing, we use RF chokes and dc decoupling (bypass) capacitors. This is the technique we have already used in the previous examples, so one might ask the question, "What's new?" As frequency increases, these inductors are either not manufacturable or have such a low *Q* that their use becomes questionable. This is the point where one may introduce the so-called distributed elements.

Figure 21 shows the general RF amplifier circuit but resorting to distributed rather than lumped elements. The elements we are introducing now are part of any good, up-to-date CAD tool.

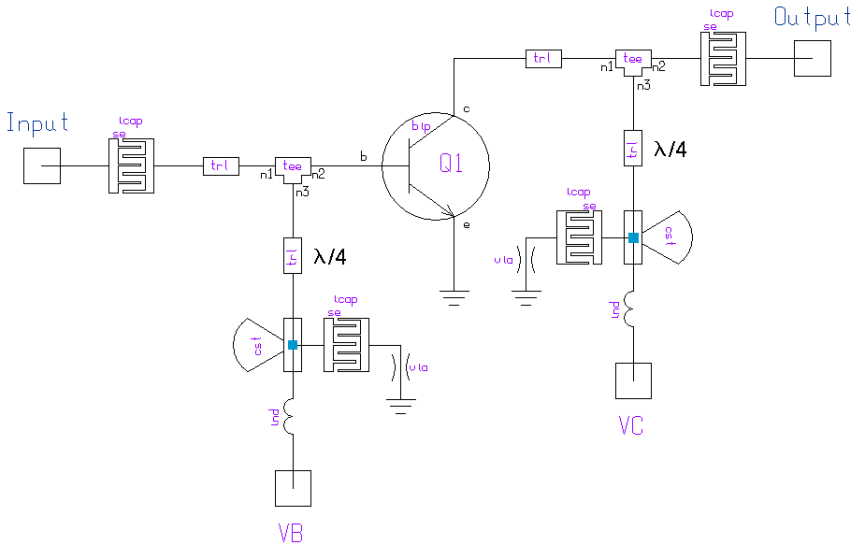


Figure 21: Simple BJT RF amplifier with distributed elements.

- Transmission Line

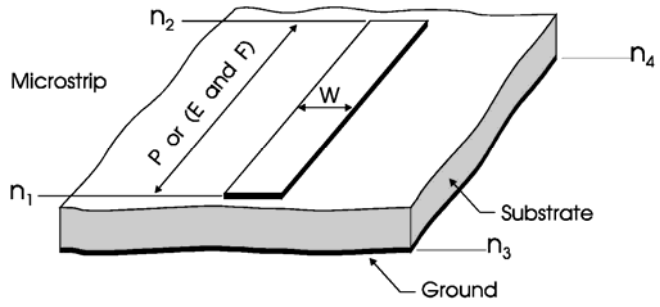


Figure 22: Transmission line in microstrip.

Any printed connection between two points on a circuit board is a transmission line (Figure 22). Its characteristic electrical impedance is a function of the square root of the dielectric constant (ε_r), the width, metallization, thickness and height above substrate of the line, and the loss tangent of the substrate. Since lines frequently have to be laid out in the form of curved connections or have a bend in their direction, we have to add

elements capable of describing the high-frequency consequences of such connections. Figures 23 and 24 show mitered and radial bend elements that perform this function.

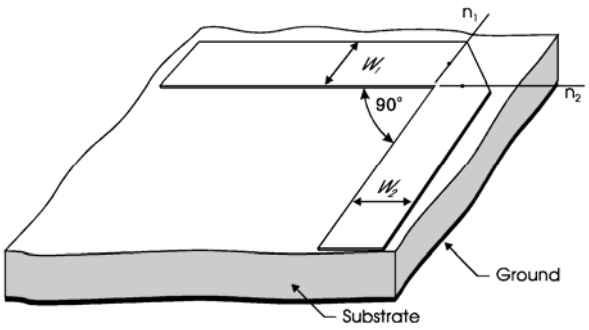


Figure 23: Mitered bend.

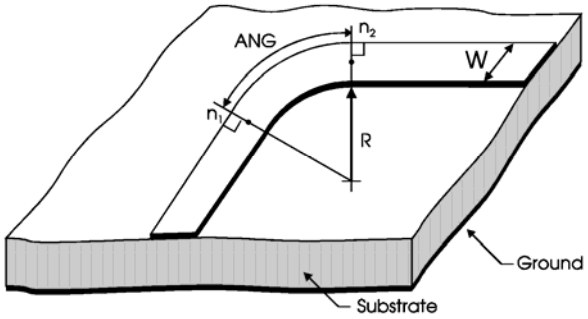


Figure 24: Radial bend.

- T, Cross, and Y junction

By the time a point like a collector or base, or its FET equivalent, spreads out into connecting with other elements, we need additional modeling capability to describe T connections, crossings, and Y junctions. Figures 25, 26, and 27 show the way in which these connections need to be modeled.

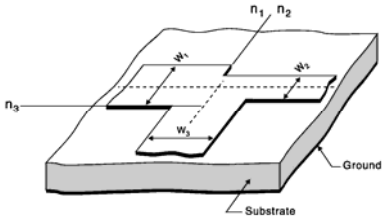


Figure 25: T junction.

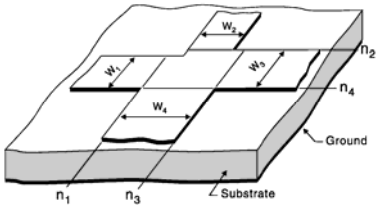


Figure 26: Cross.

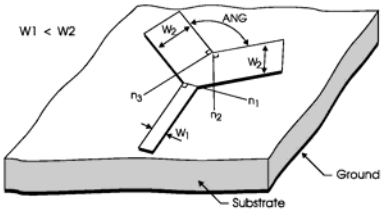


Figure 27: Y junction.

If the need exists the standard inductances must be replaced with a transmission line whose length is λ/8 at the operating frequency. At higher frequencies, these transmission lines, however, then go into λ/4 resonant mode and later become capacitive. This type of design makes it fairly narrowband. A way around this is the use of printed inductors, as shown in Figures 28 and 29.

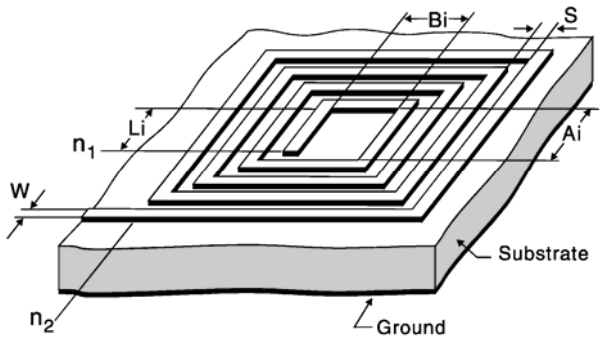


Figure 28: Rectangular inductor.

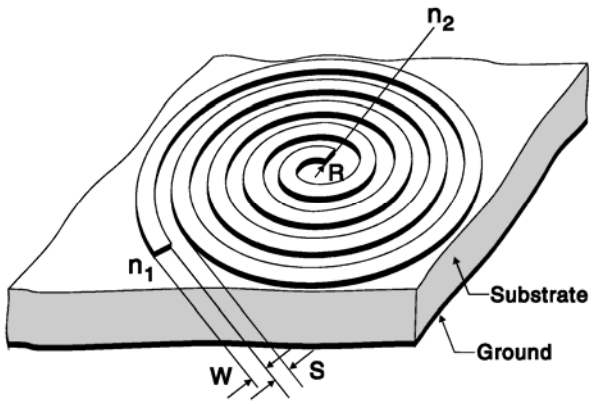


Figure 29: Spiral inductor.

These inductors have a self-resonant frequency similar to the transmission line mentioned above, but the safety margin is significantly higher.

Talking about printed inductors, a logical extension of this is the printed transmission-line-based transformer as shown in Figure 30. One can consider this as two interlaced rectangular inductors, and based on the substrate material, they are useful over a wide frequency range. Besides being used as a transformer, they can also be used to transit from unbalanced to balanced transmission provided that the difference in length from a connection point of view does not cause any problems (this is a layout issue).

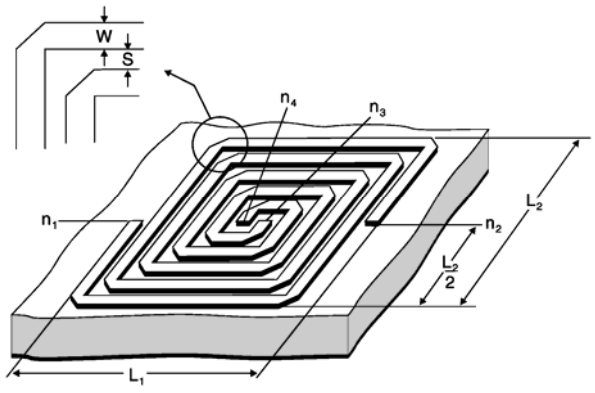


Figure 30: Transformer in microstrip.

A popular form of combining stages is the so-called Lange coupler (Figure 31) invented by the German Julius Lange. It is one of the major contributions in wideband applications.

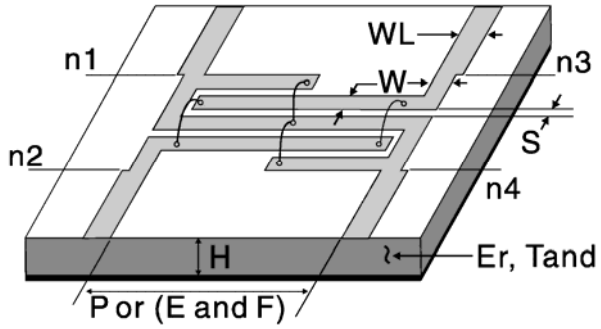


Figure 31: Four-strip Lange coupler.

The useful application of the Lange coupler probably starts at 4GHz. It consists of parallel transmission lines with the appropriate connections as shown. Lange couplers are typically built with four, six, and eight fingers. The Ansoft Serenade product has a Lange coupler synthesis program that can be used to gain more insight into this coupler's application. We assume that other modern software has similar capabilities.

Where meander-type of inductors are necessary a neat way to implement and simulate them is to use the multiple coupled line element (Figure 32) of the Serenade product, which both fast and accurately calculates the behavior of the meander, including self-resonances and losses. We made use of this arrangement in our previous examples.

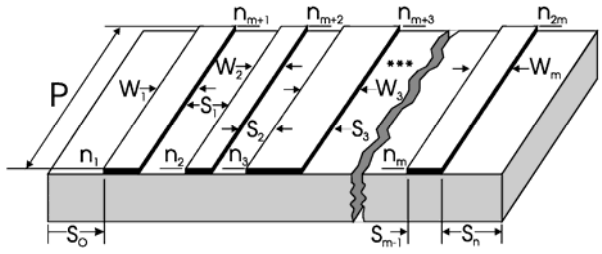


Figure 32: Multiple coupled lines element in microstrip.

- Interdigital Capacitors

The issue of tolerances of small capacitors already has been raised. The interdigital capacitor can be made on printed circuit board material as well as gallium arsenide, and if its dimensions are continuous with the transmission-line width does not cause any abrupt changes in the impedance. This type of capacitor permits to obtain very small values. By the way, an alternative to this is the use of transmission lines being 3/8λ. We have learned above that a transmission line below its resonant frequency is inductive, goes into resonance, and then becomes capacitive. Again, bandwidth is also an issue. An interdigital capacitor consists of a number of parallel fingers as shown in Figure 33, and its capacitance can be varied by adjusting the number of fingers and their spacing. The advantage of the interdigital capacitor compared to discrete components is its low variation in value.

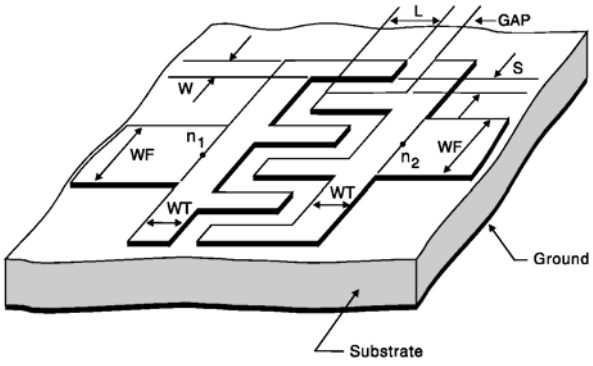


Figure 33: Interdigital capacitor.

- Radial Stubs

The radial stub (Figure 34) is not much different from a λ/4 resonator, but its bandwidth is much greater than a simple transmission line. This is another way to ground the "cold" side of a transmission line or part of a circuit that needs to be grounded for RF. Of course, the interdigital capacitor comes in a version that is a combination of a capacitor and a via hole.

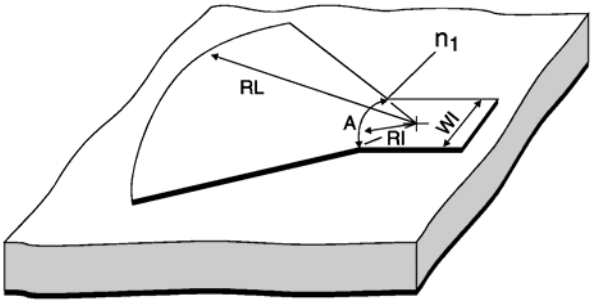


Figure 34: Radial stub.

- Via Holes

The "cold" end of the transmission line, being either considered an inductor or capacitor needs to be connected to a sufficiently large copper backplane. One very efficient way to do this, especially if there is not enough copper left on the top of the board, is the use of via holes (Figure 35). One could theoretically generate a via hole with a rivet, but most manufacturing processes don't allow this; the normal solution is to use plated-through holes left open. In the PC boards, via holes are typically cylindrical; on substrates like GaAs, they may be conical.

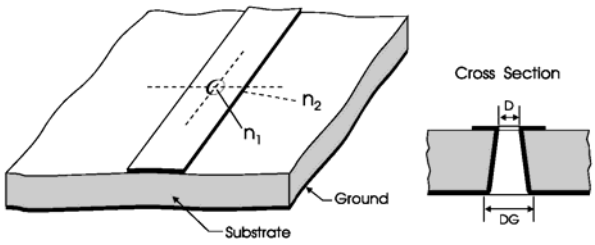


Figure 35: Via hole.

- Correction Elements

Although the behavior of actual circuits proceeds regardless of our ability to measure and describe it, we do not enjoy this luxury in simulating circuit behavior in software. In a simulator, effects that are insufficiently described will be inaccurately simulated. For instance, segments of high-impedance transmission line (120Ω, for example) are frequently used for dc feeds and RF chokes. In predicting the effect of a transition from 120-Ω line to 50-Ω line, a simulator must be alerted to the discontinuity so it can do the necessary mathematical corrections to account for the impedance jump. To do this, a specific circuit element, the STEP (Figure 36), must be inserted between the 120-Ω and 50-Ω line elements in the simulation circuit file. In addition to substrate data, we characterize a STEP by providing the widths of its input and output lines. The element itself has no physical length.

A similar correction is necessary if a transmission line is used as a resonator or just "left open" at one end. Such a transmission line tends to radiate, and because of its high-impedance properties reacts differently as far as its electrical length is concerned. A zero-length one-port element, the OPEN (Figure 37), must be added to such a line for mathematically correct calculation.

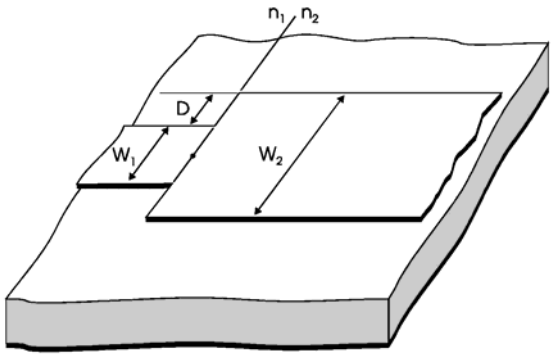


Figure 36: The STEP element tells the simulator to calculate the effects of joining transmission lines of differing characteristics.

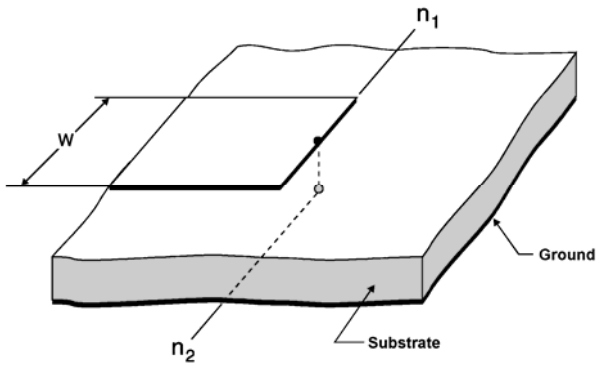


Figure 37: The OPEN element tells the simulator to calculate the effects of leaving the end of a transmission line unconnected.

We end this excursion into distributed elements here. Interested readers may want to obtain a CD containing the element library of their favorite CAD tools that combine capabilities in the microwave area with a full set of distributed circuit elements. It is most important to keep in mind that as frequency increases, we rapidly move into the area where we must consider all these distributed elements to achieve accurate simulations--even if doing so makes simulation a painful and time-consuming effort.

Finally, anyone who adventures in this area *must* obtain a foundry manual from the company that will build the integrated circuit or hybrid under design. There are basically two foundries, one applicable for MOS technology and one for GaAs. In the case of the bipolar transistor, the foundry service is not yet well-established. While we were able to interest one German company and one US company in generating a custom bipolar IC within the activities of this book, the only "open" foundry we know of for bipolar (actually HBT) technology is the one by TRW. As of

1999, this seems to be the leading company for BJT-related products, and many currently available ICs have been developed with this foundry service.

We mention the issue of foundries here again because each foundry has its own proprietary approach to modeling discontinuities. The availability of a foundry service somewhat eases the requirement that a designer be fully up to speed on the nuances of discontinuities, because a foundry's designer service will help customers account for all relevant parasitics or discontinuities in their designs. In addition, there are tables of *S* parameters provided by the vendor for standard cells of either capacitors, resistors, or inductors. The designer may then be forced to adjust the circuit so that it will work with a particular inductance value or value of another component within the resolution of the table that describes these elements. Information on the active part such as diodes and transistors is provided below and the details can be found in ref [10]

6. ACTIVE DEVICE MODELS

6.1 Typical SPICE parameters and sources

This Web site makes available a large number of needed time domain (SPICE) parameters:
homepages.which.net/~paul.hills/Circuits/Spice/ModelIndex.html

Since we are about to evaluate bipolar microwave transistors, junction FETs, MOSFETs (model level 3), and GaAs FETs, here is a list of typical parameters for the devices we have used. These parameters can be obtained by a suitable program with the appropriate measurements. The BSIM model for MOSFETs, applicable for sub-micron technology transistors, requires an enormous level of parameter extraction and has not fully been validated for the LDMOS-type transistors currently favored for RF and microwave applications.

The meaning and significance of the various parameters is best explored in a book on SPICE or semiconductor physics [15,16,17].

Some Popular Devices

Table 2: BFR193W BJT

```
IS=2.738e-16      BF=125          NF=0.95341
VAF=24            IKF=0.26949     ISE=1.0627e-14
NE=1.935          BR=14.267       NR=1.4289
VAR=3.8742        IKR=0.037925    ISC=3.7409e-17
NC=0.94371        RB=15           IRB=0.00091763
RBM=1.8368        RE1=0.76534     RC2=0.11938
CJE=1.1824e-15    VJE=0.70276     MJE=0.48654
TF=1.8828e-11     XTF=0.69477     VTF=0.8
ITF=0.00096893    PTF=0           CJC=9.3503e-13
VJC=1.1828        MJC=0.30002     XCJC=0.053563
TR=1.0037e-09     VJS=0.75        MJS=0
XTB=0             EG=1.11         XTI=3
FCC=0.72063       LB=0.57e-9      LC=0.00e-9
LE=0.43e-9        CBCP=0.101E-12  CCEP=0.175E-12
CBEP=0.061e-12    VCMX=10V
```

Table 3: 2SK125 JFET

```
IDSS=  .5250E-01  VP0 = -.3111E+01  GAMA= -.1867E-01  E   =  .1520E+01
KE  = -.3856E-03  SL  =  .2818E-01  KG  = -.2398E+00  T   =  .0000E+00
SS  =  .7448E-04  IG0 =  .2000E-14  AFAG=  .3846E+02  IB0 =  .1000E-04
AFAB=  .3800E+02  VBC =  .3000E+02  R10 =  .1711E+02  KR  =  .0000E+00
C10 =  .6609E-11  K1  =  .1675E+01  C1S =  .6818E-33  CF0 =  .7261E-11
KF  =  .1156E+01  RG  =  .5000E+00  RD  =  .1542E+01  RS  =  .1333E+01
LG  =  .6098E-09  LD  =  .5159E-08  LS  =  .1482E-08  CDS =  .4813E-16
CGE =  .1590E-11  CDE =  .3394E-26  CGSP=  .8282E-13  CDSP=  .4832E-12
ZGT =  .5000E+02  LGT =  .4712E-01  ZDT =  .5000E+02  LDT =  .3998E-01
CGDP=  .3653E-12  ZST =  .5000E+02  LST =  .1495E-01  CGDE=  .3831E-12
CGSB=  .3120E-13  CDSB=  .5896E-12  VDMX=10
```

Note for junction FETs: The currently implemented model for junction FET is too primitive for serious RF applications. We have therefore taken the approach (liberty) to use the Materka parameter extraction approach for silicon junction FETs. This has resulted in unparalleled high quality parameters; in particular, the knee voltage behavior has significantly improved, as well as the overall frequency response.

Table 4: GaAs MESFET

```
IDSS=  .1077E+00  VP0 = -.1800E+01  GAMA= -.5741E-01  E   =  .1290E+01
KE  = -.1155E-01  SL  =  .1652E+00  KG  = -.1782E+00  T   =  .0000E+00
SS  = -.1208E-02  IG0 =  .2130E-11  AFAG=  .2740E+02  IB0 =  .5680E-09
AFAB=  .1826E+01  VBC =  .9000E+01  R10 =  .8382E+01  KR  =  .6359E+00
C10 =  .5964E-12  K1  =  .1296E+01  C1S =  .0000E+00  CF0 =  .6110E-13
KF  =  .9775E+00  RG  =  .1996E+01  RD  =  .1296E+01  RS  =  .1234E+01
CDS =  .7852E-13  CDSD=  .1000E-07  RDSD=  .1581E+03  CGE =  .1609E-12
CDE =  .8674E-13  VDMX=8
```

Table 5: 1 μm × 750 μm Level 3 LDMOS FET

```
CBD = 0.863E-12   CGD0 = 166E-12    CGS0 = 246E-12    GAMA = 0.211
IS = 6.53E-16     KAPA = 0.809      MJ = 0.536        NSUB = 1E15
PB = 0.71         PBSW = 0.71       PHI = 0.579       RD = 39
RS = 0.1          THET = 0.588      TOX = 4E-8        U0 = 835
VMAX = 3.38E5     VT0 = 2.78        XQC = 0.41
```

7. NOISE MODELING

Diode Noise Model.

The noise model for the diodes (Figure 38) consists of two contributions: the shot noise and the flicker noise. The shot noise is computed automatically and does not require any parameters. The flicker noise can be specified in two ways:

1. Using the enhanced SPICE noise model by specifying KF, AF, and FCP in the model\ parameter list (this option is usually sufficient for most applications).

2. Using bias-dependent flicker noise coefficients (specifying KF and AF at multiple bias points).

Diode Noise Model Keywords

keyword	description	unit	default
ID	Required bias current for the data point	Ampere	
KF	Flicker noise coefficient		0.0
AF	Bias exponent of the flicker noise model		1.0
FCP	Frequency exponent of the flicker noise model		1.0
FC	Flicker noise corner frequency	Hz	

The noise generators in the diode noise model are the series parasitic resistance, R_S, and the intrinsic junction. The figure below illustrates the intrinsic junction noise generator. Let Δf be the bandwidth (usually normalized to 1 Hz). The intrinsic noise generator has a mean-square value of:

$$\left\langle i_{Dn}^2 \right\rangle = 2qI_D\,\Delta f + \mathrm{KF}\frac{I_D^{\mathrm{AF}}}{f^{\mathrm{FCP}}}\Delta f \qquad (82)$$

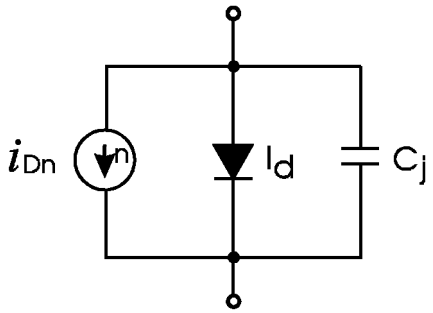


Figure 38: Equivalent noise circuit for a diode chip.

Notes on the Diode Noise Model:

1. Shot noise is always present unless the SN parameter is set to zero. Turning noise off is useful for comparing the total circuit noise that is generated by the nonlinear devices and that generated by the linear circuit components.

2. If the value of KF is specified as zero, then the flicker noise will not be contributed by the device and only shot noise is considered in the intrinsic model.

3. The corner frequency noise model uses the system noise floor to internally compute the flicker noise coefficient, KF. The system noise floor is computed by the program using the diode parameters and *kT*.

4. This noise model of course considers the actual operating temperature, which must be supplied to the model.

BJT Noise Model.

The noise model for the Gummel-Poon BJT model consists of two contributions: shot noise and the flicker noise. The shot noise is computed automatically and does not require any parameters. The flicker noise can be specified in two ways:

1. Using the enhanced SPICE noise model by specifying KF, AF, and FCP in the model parameter list (this option is usually sufficient for most applications).

2. Using bias-dependent flicker noise coefficients (specifying KF and AF at multiple bias points).

Option 1: Specifying the Bias-Independent Flicker Noise Coefficient. This option involves the straightforward specification of KF, AF, and FCP that are constant with bias, as in the SPICE noise model. Notes on Option 1:

1. Shot noise is always present unless it is turned off. Turning noise off is useful for comparing the total circuit noise that is generated by the nonlinear devices and that generated by the linear circuit components.

2. If the value of KF is specified as zero, flicker noise will not be contributed by the device and only shot noise is considered in the intrinsic model.

Option 2: Specifying The Bias-Dependent Flicker Noise Coefficient or Flicker Corner Frequency

Option 2 allows a bias-dependent flicker noise coefficient (that is, KF and AF vary with the bias point).

BJT Noise Model Keywords

keyword	description	unit	default
IB	**Required** base bias current for the data point	ampere	
VCE	**Required** collector-emitter voltage for the data point	volt	
VBS	Base-substrate voltage required for LPNP type when four nodes are used.	volt	
VCS	Collector-substrate voltage required for NPN or PNP type when four nodes are used.	volt-	
KF	Flicker noise coefficient		0.0
AF	Bias exponent of the flicker noise model		1.0
FCP	Frequency exponent of the flicker noise model		1.0
FC	Flicker noise corner frequency	Hz	

Notes on the BJT Noise Model:

1. KF, AF, and FC can be specified as bias dependent. If only one set of noise data is specified, the corresponding bias point is not meaningful because all parameters are considered constant over all bias values. However, the bias point is needed for the program to identify the data as bipolar noise data.

2. The corner frequency noise model option uses the system noise floor to compute the flicker noise coefficient, KF. The system noise floor is computed by the program using the transistor parameters and *kT*.

3. This noise model of course considers the actual operating temperature, which must be supplied to the model.

Figure 33 shows the BJT noise model. Let Δf be the bandwidth (usually normalized to a 1-Hz bandwidth). The noise generators introduced in the intrinsic device are shown below, and have mean-square values of:

$$\left\langle i_{bn}^2 \right\rangle = 2qI_B\Delta f + \mathrm{KF}\frac{I_B^{AF}}{f^{FCP}}\Delta f \tag{83}$$

$$\left\langle i_{cn}^2 \right\rangle = 2qI_C\Delta f \tag{84}$$

$$\left\langle i_{R_{bb}}^2 \right\rangle = \frac{4kT}{R_{bb}}\Delta f \tag{85}$$

$$\left\langle i_{R_{e1}}^2 \right\rangle = \frac{4kT}{R_{e1}}\Delta f \tag{86}$$

$$\left\langle i_{R_{c2}}^2 \right\rangle = \frac{4kT}{R_{c2}}\Delta f \tag{87}$$

$$I_B = \frac{I_{bf}}{BF} + I_{le} \tag{88}$$

$$I_C = I_{cf} - I_{cr} \tag{89}$$

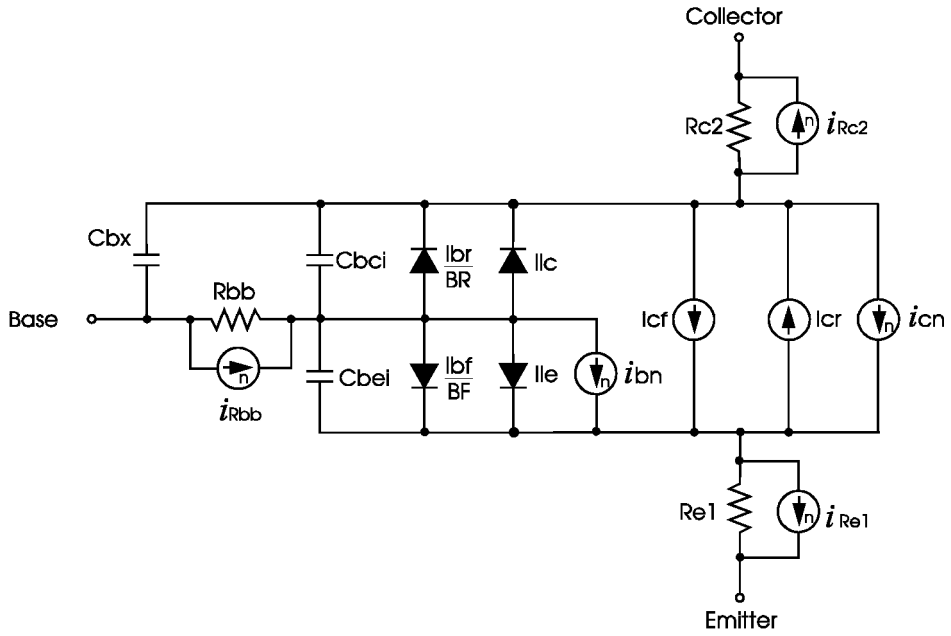


Figure 39: BJT noise model (not showing extrinsic parasitics). Current sources with *n* are noise sources.

JFET and MESFET Noise Model.

The noise model for the FETs consists of two contributions: the shot noise and the flicker noise. There are two options to specify noise in the FET model:

1. Using the enhanced SPICE noise model by specifying KF, AF, and FCP in the model parameter list to determine the flicker noise (this option is usually sufficient for most applications). The shot noise will be automatically computed using the SPICE equation.

2. Using bias-dependent flicker noise coefficients through a reference in the DATA block (specifying KF and AF at multiple bias points) and specifying the four noise parameters (Fmin, MGopt, PGopt, and Rn) at multiple bias points.

Option 1: Specifying the Enhanced SPICE Noise Model. Option 1 is the straightforward specification of KF**,** AF, and FCP that are constant with bias, as in the SPICE noise model.

The drain noise model has the form:

$$\left\langle \left| I_{dn} \right|^2 \right\rangle = 4K_B T \frac{2g_m}{3} \Delta f + \mathrm{KF} \frac{\left| I_D \right|^{\mathrm{AF}}}{f^{\mathrm{FCP}}} \Delta f \tag{90}$$

where the shot noise is derived from g_m and the flicker noise is proportional to KF and the drain channel current, I_D, and inversely proportional to frequency. The AF and FCP parameters tailor the flicker noise dependence on bias and frequency, respectively.

Notes on Option 1:

1. Shot noise is always present unless it is turned off. Turning noise off is useful for comparing the total circuit noise that is generated by the nonlinear devices and that generated by the linear circuit components.

2. If the value of KF is specified as zero, then flicker noise will not be contributed by the device and only shot noise is considered in the intrinsic model.

(Option 2) Specifying The Bias-Dependent Flicker Noise Coefficient or Flicker Corner Frequency. Option 2 allows the specification of the complex bias-dependent nature of the shot noise and flicker noise. At high frequencies, the equivalent noise sources are correlated (the SPICE noise model does not account for this correlation). The complete evaluation of the shot noise sources can be determined from the four noise parameters. Since these are functions of bias, they can be specified over the (V_{GS}, V_{DS}) bias plane. Additionally, a bias-dependent flicker noise coefficient (that is, KF and AF vary with current) can be specified.

The MESFET noise model uses the four measured noise data (F_{min}, Γ_{opt}, and R_n) at one frequency and multiple arbitrary bias points. The program uses this data and the FET model parameters to de-embed the noise data to an intrinsic noise model. The intrinsic model is accurate at all frequencies, and therefore can predict the noise performance at all frequencies given data at just one frequency point. Built-in bias-dependent characteristics are used if multi-bias noise data is not provided.

FET Noise Model Keywords

keyword	description	unit	default
FN	Noise data measurement frequency	Hz	1.0 GHz
VGS	**Required** gate-source voltage for the data point	volt	
VDS	**Required** drain-source voltage for the data point	volt	
FMIN	**Required** minimum noise figure in dB at FN		
MGO	**Required** magnitude of optimum noise reflectioncoefficient at FN		
PGO	**Required** phase of optimum noise reflection coefficient at FN		
RN	**Required** normalized noise resistance at FN		
KF	Flicker noise coefficient		0.0
AF	Bias exponent of the flicker noise model		1.0
FCP	Frequency exponent of the flicker noise model		1.0
FC	Flicker noise corner frequency	Hz	

Notes on the FET noise model:

1. The corner frequency noise model option uses the system noise floor to compute the flicker noise coefficient, KF. The system noise floor is computed by the program using the transistor parameters and *kT*.

2. This noise model of course considers the actual operating temperature, which must be supplied to the model.

Noise in a MESFET is produced by sources intrinsic to the device. The same approach, but with different flicker corner frequencies, is highly applicable to JFETs and MOSFETs. For more detail as to simulation, see the Element library book for the active device portion of Ansoft's Serenade Design Environment product. The equivalent noisy circuit of an intrinsic FET is represented in Figure 40.

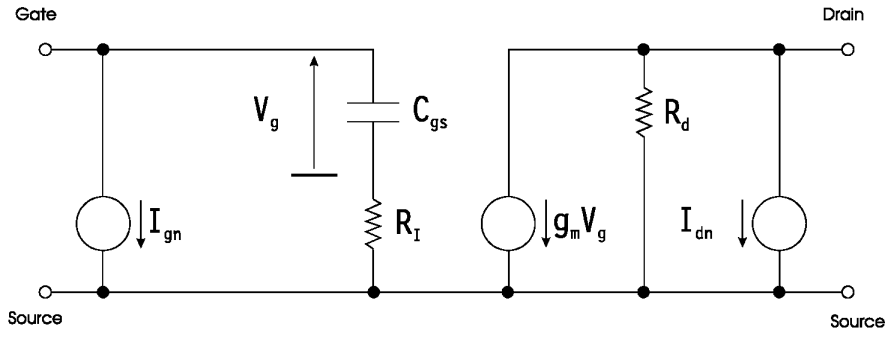


Figure 40: Equivalent noise circuit of an intrinsic FET device.

The intrinsic FET is internally represented as a noiseless nonlinear two-port with one equivalent noise current connected across the gate-source terminal, and one across the drain-source terminal. The correlations of the gate and drain noise current sources are:

$$\left\langle \left| I_{gn} \right|^2 \right\rangle = 4K_B T \Delta f \frac{\omega^2 C_{gs}^2}{g_m} R \tag{91}$$

$$\left\langle \left| I_{gn} \right|^2 \right\rangle = 4K_B T \Delta f g_m P \tag{92}$$

$$\left\langle I_{gn} I_{dn}^* \right\rangle = 4K_B T \Delta f j \omega C_{gs} \sqrt{PR}\, C \tag{93}$$

The correlation matrix of the noise current sources is

$$C_{dc}(\omega) = \frac{2}{\pi} K_B T d\omega \begin{bmatrix} \frac{\omega^2 C_{gs}^2}{g_m} R & -j\omega C_{gs} \sqrt{PR}\, C \\ j\omega C_{gs} \sqrt{PR}\, C & g_m P \end{bmatrix} \tag{94}$$

The gate and drain noise parameters R and P and the correlation coefficient C are related to the physical noise sources acting in the channel and are functions of the device structure and bias noise parameters. By defining measured noise parameters, F_{min}, R_n and Γ_{opt}, and using a noise-de-embedding procedure, the parameters R, P, and C and the intrinsic noise correlation matrix of a FET device as functions of device bias are determined by the program.

In addition to the noise sources shown above, the flicker (1/f) noise can also be modeled by means of a noise current source connected in parallel with the intrinsic drain port. The flicker noise component in a narrow band, Δf, is expressed in the form

$$\left\langle \left| I_f \right|^2 \right\rangle = \mathrm{Q} \Delta f \frac{\left| I_D \right|^{\mathrm{AF}}}{f^{\mathrm{FCP}}} \tag{95}$$

where I_D is the instantaneous value of the channel current, and Q, AF, and FCP are empirical parameters. In most practical cases, AF and FCP are directly obtained from measurements (typically, AF = 2 and FCP = 1), while Q is not. In Ansoft's Serenade Design Environment, Q is either provided directly using KF or is computed by providing the flicker corner frequency (FC). FC is the frequency at which the flicker noise equals the shot/diffusion noise. The corner frequency is defined by the equation

$$\mathrm{Q} \frac{\left| I_D \right|^{\mathrm{AF}}}{f_{\mathrm{C}}^{\mathrm{FCP}}} = g_m P \tag{96}$$

Given the corner frequency FC and the measurement bias point V_{gs} and V_{ds}, the program automatically computes I_D, g_m, and P, and finally Q.

More information on FET noise modeling can be found in [18, 19, 20, 21, 22, 23 and 24].

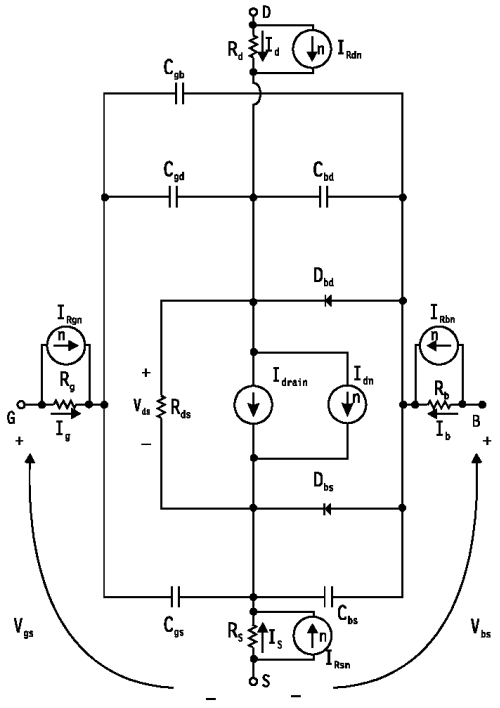


Figure 41: Equivalent noise circuit of an intrinsic MOSFET device.

MOSFET Noise Model.

The MOSFET noise model (Figure 41) consists of two contributions: the shot noise and the flicker noise. The shot noise is computed automatically and does not require any parameters. It can be turned off by specifying SN = 0. The flicker noise can be specified in two ways:

1. Using the enhanced SPICE noise model by specifying KF, AF, and FCP in the model parameter list (this option is usually sufficient for most applications).

2. Using bias-dependent flicker noise coefficients through a reference (specifying KF and AF at multiple bias points).

(Option 1) Specifying the Enhanced SPICE Noise Model

Option 1 is the straightforward specification of KF, AF, and FCP that are constant with bias, as in the SPICE noise model (the flicker noise is considered bias dependent.

Notes on the MOSFET Noise Model:

1. Shot noise is always present unless the SN parameter is set to zero. Turning noise off is useful for comparing the total circuit noise that is generated by the nonlinear devices and that generated by the linear circuit components.

2. If the value of KF is specified as zero, then the flicker noise will not be contributed by the device and only shot noise is considered in the intrinsic model.

(Option 2) Specifying The Bias-Dependent Flicker Noise Coefficient or Flicker Corner Frequency.

Option 2 allows a bias-dependent flicker noise coefficient (that is, KF and AF varies with drain current). The MOSFET noise model data is given and referenced by a model parameter.

Noise Model Keywords

keyword	**description**	**unit**	**default**
VGS	***Required*** gate-source bias for the data point	volt	
VDS	***Required*** drain-source for the data point	volt	
VBS	***Required*** drain-bulk for the data point	volt	
KF	Flicker noise coefficient		1.0E−13
AF	Bias exponent of the flicker noise model		2.0
FCP	Frequency exponent of the flicker noise model		1.0
FC	Flicker noise corner frequency	Hz	

Notes on the MOSFET Noise Model:

1. KF, AF, and FC can be specified as bias dependent. If only one set of noise data is specified, the corresponding bias point is not meaningful because all parameters are considered constant over all bias values. However, the bias point is needed for the program to identify the data as MOSFET noise data.

2. The corner frequency noise model option uses the system noise floor to compute the flicker noise coefficient, KF. The system noise floor is computed by the program using the transistor parameters and kT.

3. This noise model of course considers the actual operating temperature, which must be supplied to the model.

Let Δf be the bandwidth (normalized to 1 Hz). The noise generators introduced in the intrinsic device are shown below, and have mean-square values of:

$$\left\langle i_{dn}^2 \right\rangle = \frac{8kTg_m}{3}\Delta f + \mathrm{KF}\ \frac{I_D^{\mathrm{AF}}}{f^{\mathrm{FCP}}}\Delta f$$

$$\left\langle i_{Rgn}^2 \right\rangle = 4\frac{kT}{R_g}\Delta f$$

$$\left\langle i_{Rdn}^2 \right\rangle = 4\frac{kT}{R_d}\Delta f$$

$$\left\langle i_{Rsn}^2 \right\rangle = 4\frac{kT}{R_s}\Delta f$$

$$\left\langle i_{Rbn}^2 \right\rangle = 4\frac{kT}{R_b}\Delta f \qquad (97)$$

We include this MOSFET noise model (used for quite awhile) for completeness. At the moment, we do not know which MOSFET noise model the industry will settle on in the future.

Modern CAD tools, such as Ansoft's Serenade product, use these models allowing to generate quite accurate noise data based on a good linear equivalent model. Internally it uses the noise-correlation-matrix method (first introduced by Russer).

8. APPLICATIONS AND EXAMPLES

Application 1

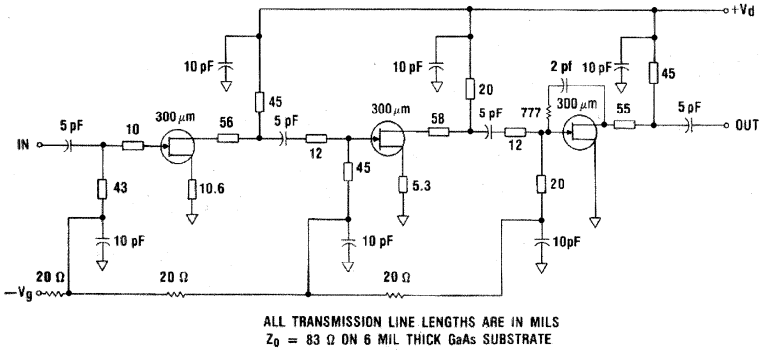


Figure 42: Schematic of the X-band GaAs monolithic low-noise amplifier (Texas Instruments EG8021).

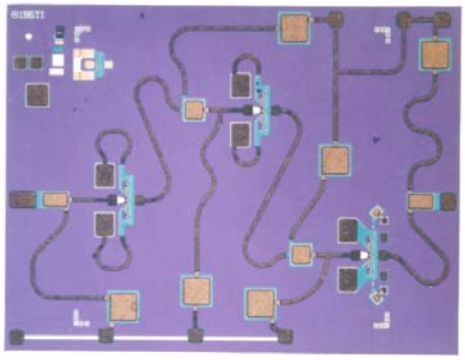


Figure 43: Photograph of the EG8021 monolithic amplifier chip. The area pictured is 0.09 inch by 0.12 inch in size.

Schematic of the X-band GaAs monolithic low-noise amplifier (Texas Instruments EG8021) (Figure 42) and its layout (Figure 43) are shown as example to rigorous modeling. The round transmission lines were chosen to reduce radiation and are beyond the analysis capacity of known simulators. It took a set of new models to be developed to accomplish this. For reasons of linearity, all three stages operate in Class A. This circuit was developed with some of the early devices and two test simulations were done. The first was the linear equivalent circuit of the transistors (total linear analysis) and then the non-linear time domain parameters were used, to validate their accuracy. Both approaches gave really good and similar answers. Compared to the measurements, the linear FET model gave a slightly closer answer, indicating that the non-linear model was not (correctly) optimized. This is a general phenomenon with non-linear models

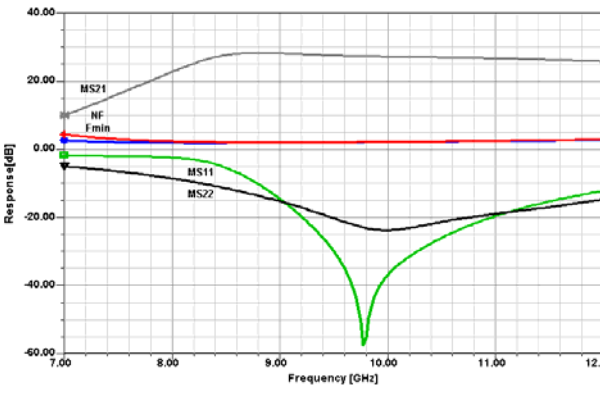


Figure 44: Simulated F_{min}, noise figure, S_{11}, S_{21}, and S_{22} responses for the three-stage GaAsFET amplifier using the TI linear FET model. The values at 10 GHz are: F_{min}, 2.20 dB; NF, 2.21 dB; S_{11}, −36.8 dB; S_{21} 27.2 dB; and S_{22}, −23.7 dB.

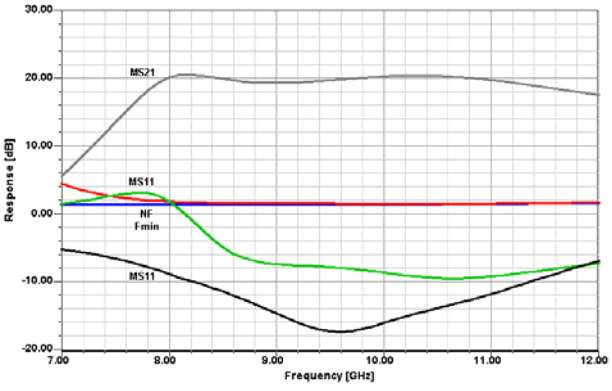


Figure 45: Simulated F_{min}, noise figure, S_{11}, S_{21}, and S_{22} responses for the three-stage GaAsFET amplifier using the nonlinear FET model. The values at 10 GHz are: F_{min}, 1.53 dB; NF, 1.65 dB; S_{11}, −8.5 dB; S_{21} 20.3 dB; and S_{22}, −15.9 dB.

Application 2

In cooperation with Motorola, we also analyzed an 800-MHz VCO. In this case, we also did the parameter extraction for the Motorola transistor. Figure 5-46 shows the circuit, a Colpitts oscillator that uses RF feedback in the form of a 15-Ω resistor and a capacitive voltage divider consisting of 1 pF between the BJT's base and the feedback resistor, and 1 pF between the feedback resistor and common. Also, the tuned circuit is loosely coupled to this part of the transistor circuit. Figure 5-47 shows a comparison between predicted and measured phase noise for this oscillator.

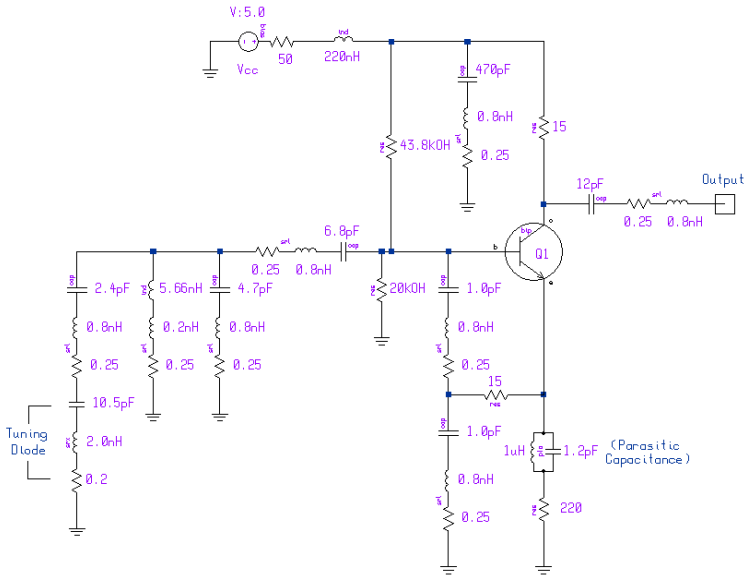


Figure 46: Colpitts oscillator for 800MHz with lumped elements modeled by their real values.

Figure 46 shows a Colpitts oscillator that uses RF negative feedback between the emitter and capacitive voltage divider. To be realistic, we have also used real components rather than ideal ones. The suppliers for the capacitors and inductors provide some typical values for the parasitics. The major changes are 0.8nH and 0.25Ω in series with the capacitors. The same thing applies for the main inductance, which has a parasitic connection inductance of 0.2nH in series with a 0.25Ω resistance. These types of parasitics are valid for a fairly large range of components assembled in surface-mount applications. Most engineers model the circuit only by assuming lossy devices, and not adding these important parasitics. One of the side-effects we have noticed is that the output power is more realistic and, needless to say, the simulated phase noise agrees quite well with measured data. This circuit can also serve as an example for modeling amplifiers and mixers using surface-mount components.

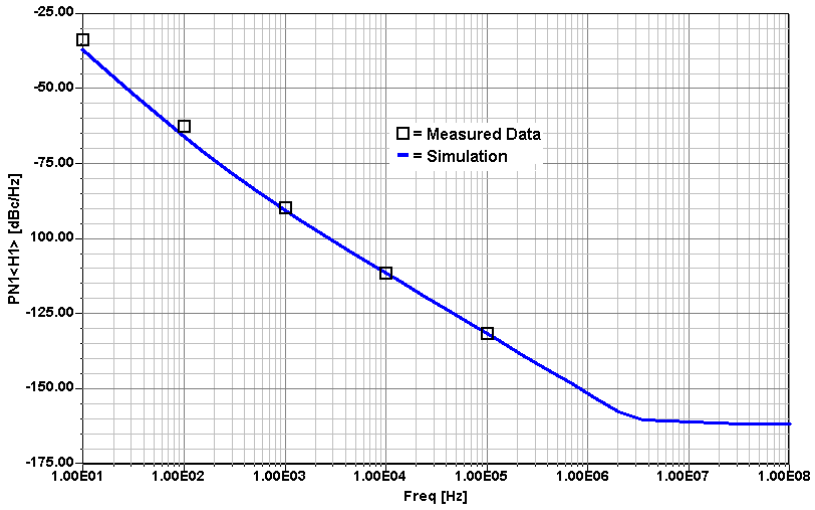


Figure 47: Comparison between predicted and measured phase noise for the oscillator shown in Figure 46.

Bibliography and References

1. bwrc.eecs.berkeley.edu/Classes/icbook/SPICE/
2. Transient Analysis of Microwave Oscillators by Krylov-Subspace Harmonic Balance
3. www2.dac.com/App_Content/files/2009_Best_Paper_23.2.pdf
4. en.wikipedia.org/wiki/Krylov_subspace
5. Carl T. A. Johnk, *Engineering Electromagnetic Fields and Waves*, New York: John Wiley & Sons, Inc. New York, NY, 1975, Chapter 5.
6. Robert Schetgen, ed., *The ARRL Handbook for Radio Amateurs*, 1995 edition, Newington: American Radio Relay League, 1995, chapters 6 and 24.
7. Jack Smith, *Modern Communication Circuits*, second ed. (New York: McGraw-Hill, 1997, ISBN 0-07-059283-7), pp. 93-99.
8. S. M. Perlow, "Third-Order Distortion in Amplifiers and Mixers," *RCA Review*, June 1976, pp. 234-265.
9. Edward C. Jordan, ed., *Reference Data for Engineers*, 7th Edition (Indianapolis: Howard W. Sams & Co, Inc, 1985, ISBN 0-672-21563-2), pp. 12-33 to 12-35.
10. U.L. Rohde and D. Newkirk, *RF/Microwave circuit design for wireless applications*, Wiley, NY 2000
11. Ulrich L. Rohde, Anthony M. Pavio, and Robert A. Pucel, "Accurate Noise Simulation of Microwave Amplifiers Using CAD," *Microwave Journal*, Vol. 31, No. 12, pp 130-141, December 1988.]
12. H. Rothe and W. Dhalke, "Theory of Noisy Fourpoles," *Proceedings of the IRE*, Vol. 44, June 1956, pp. 811-818.
13. "Representation of Noise in Linear Twoports," *Proceedings of the IRE*, Vol. 48, January 1960, pp. 69-74.
14. H. A. Haus and R. B. Adler, *Circuit Theory of Linear Noisy Networks*, Cambridge, Mass., MIT Press, 1959, and New York, Wiley, 1959.
15. Chris Bowick, *RF Circuit Design* (Indianapolis: Howard F. Sams & Co, 1982).
16. Adel S. Sedra and Kenneth C. Smith, *Microelectronic Circuits* (Chicago: Saunders College Publishing, 1991).
17. Paul R. Gray and Robert G. Meyer, *Analysis and Design of Analog Integrated Circuits*, second ed. (New York: John Wiley & Sons, Inc., 1984).
18. S. H. Lee, A. K. Wong, M. G. Wong, *A Current-Combiner Circuit for Better Mixer Conversion Gain*, Philips Semiconductors applications document.
19. F. M. Fano, "Theoretical Limitations in the Broadband Matching Arbitrary Impedances," J. Franklin Institute, Vol. 249, 1950, pp 57-85 and 139-155.
20. Ravender Goyal, ed., *High-Frequency Analog Integrated-Circuit Design* (New York: John Wiley & Sons, 1995, ISBN 0-471-53043--3), Chapter 6, Wideband Amplifiers.
21. Stefan Georgi and Ulrich L. Rohde, "A Novel Approach to Voltage Controlled Tuned Filters Using CAD Validation," *Microwave Engineering Europe*, October 1997, pp. 35-42.
22. *Analog/Interface ICs Device Data*, Vol. II, Q4/96, DL128, Rev 6, Motorola, Inc., MC13158 Wideband FM IF Subsystem specifications, pp 8-308-8-329.

23. E. J. Wilkinson, "An *N*-Way Power Divider," *IRE Trans. Microwave Theory Tech.*, Vol. MTT-8, January 1960, p 116.
24. Motorola Application Note AN-282A, *Systemizing RF Power Amplifier Design*.

Appendix 1 - Krylov-subspace

In linear algebra, the order-*r* **Krylov subspace** generated by an *n*-by-*n* matrix *A* and a vector *b* of dimension *n* is the linear subspace spanned by the images of *b* under the first *r* powers of *A* (starting from $A^0 = I$), that is,

$$\mathcal{K}_r(A, b) = \operatorname{span}\{b, Ab, A^2b, \ldots, A^{r-1}b\}.$$

It is named after Russian applied mathematician and naval engineer Alexei Krylov, who published a paper on this issue in 1931.

Modern iterative methods for finding one (or a few) eigenvalues of large sparse matrices or solving large systems of linear equations avoid matrix-matrix operations, but rather multiply vectors by the matrix and work with the resulting vectors. Starting with a vector, *b*, one computes *Ab*, then one multiplies that vector by *A* to find A^2b and so on. All algorithms that work this way are referred to as Krylov subspace methods; they are among the most successful methods currently available in numerical linear algebra.

Because the vectors tend very quickly to become almost linearly dependent, methods relying on Krylov subspace frequently involve some orthogonalization scheme, such as Lanczos iteration for Hermitian matrices or Arnoldi iteration for more general matrices.

The best known Krylov subspace methods are the Arnoldi, Lanczos, Conjugate gradient, GMRES (generalized minimum residual), BiCGSTAB (biconjugate gradient stabilized), QMR (quasi minimal residual), TFQMR (transpose-free QMR), and MINRES (minimal residual) methods.