June 2007

Proposal for development of a D-OA-based first-level electronic circuits course

Motivation

In the many years since transistors and integrated circuits were introduced, first-level course textbooks have essentially reached an asymptotic form in which many dozens of competing books are almost the same. Along with this evolution, the texts have become encyclopedic with more and more dependence on computer solutions.

Whether a consequence or not, hiring companies complain that new analog graduates “don’t know anything and can’t do anything.” At the same time the graduates themselves are less interested in analog because the electronics world is “going digital,” and indeed some schools have actually dropped their basic linear circuits courses.

On the contrary, there is still a need for analog designers, albeit less than before. Gordon Moore, upon retiring from Intel (basically a digital company!), said “One area I think gets neglected [in schools] is linear circuitry. That seems to be more of an art form than an engineering science. As we move to higher and higher frequencies, everything becomes an analog circuit and relatively few people can design those.”

In other words, digital designers are recognizing that even in a digital world there is something between zero and one, and that is the world of analog.

Dependence on circuit simulation without physical understanding of the intuitive, physical sources of circuit behavior is a mistake, and can waste hours of fruitless trial and error churning.

What’s “wrong” with the conventional approach?

Conventional analysis solves a system of simultaneous loop or node equations for transfer functions as ratios of sums of products of the circuit elements. These are “high entropy expressions” that give no insight into how the element values affect the results.

In learning the conventional analysis method, most of us unconsciously adopt the following strategies:

1. Put everything into the model and simplify later.
2. Postpone approximation as long as possible and don’t even dare to make an approximation unless you can justify it on the spot.
3. The “answer” is acceptable in whatever form it emerges from the algebra.
4. The more work you do, the more valuable the result.
5. Every problem is a brand-new problem, and requires a brand-new strategy to solve it.

This approach is a recipe for failure. An “answer” can only be achieved for very simple circuits; in more complicated “real-life” circuits, the algebra goes into paralysis after only a few lines. Resorting to a computer does nothing to restore the physical insight lost in algebra, and ultimately engenders the “falling off a cliff” syndrome experienced by many analog designers in their first job.

An alternative approach: Design-Oriented Analysis (D-OA)

The initial conditions for tackling a design problem are that there are never nearly enough equations to solve for the number of unknowns, and consequently the design problem cannot be solved in a strictly mathematical sense. What is needed is a Divide & Conquer approach, based on physical insight, to replace the missing equations by inequalities, assumptions, approximations, and tradeoffs.

The D-OA approach deviates from the conventional from the very first step. Instead of solving the loop or node equations *simultaneously*, they are solved *sequentially* by successive circuit reductions.

There are many different sequences (different "algorithms") in which these steps can be taken, each of which leads to a different "low entropy result" in which the terms are ordered, or grouped, so that additional insight is gained into the relative importance of the various contributions to the result. A low entropy result permits more than one useful piece of information to be extracted from one equation, thus making one equation "work harder." You can choose a circuit reduction algorithm that produces a desired low entropy result, and such "steering" of the analysis is your most powerful tool of D-OA.

There are specific strategies to implement the D-OA approach:

1. Put only enough into the model to get the answers you need.

2. Make all the approximations you can, as soon as you can, justified or not. Plow through the problem leaving a wake behind you of assumptions and approximations. You can't lose by trying!

3. Figure out in advance as many of the quantities as you can that you want to have in the answer, and put them into the statement of the problem as soon as possible – even into the circuit model.

4. The less work you do, the more valuable the result. *You* control the algebra. You *make* the algebra come out in low entropy form by applying strategic mental energy before and during the math.

5. Every problem is not unique. There are problem solving strategies and techniques that apply to almost all engineering problems.

The D-OA approach is a recipe for success. It fends off algebraic paralysis. Approximations are not "bad things;" on the contrary, assumptions and approximations are the key to arriving at low entropy answers that can be worked backwards for design. The D-OA paradigm bestows a sense of empowerment: the algebra is malleable, you have choices. The math becomes the designer's slave, rather than his master.

The basic D-OA philosophy is that Design is the Reverse of Analysis: you start with the "answer," the Specification, and you have to work backwards to the circuit configuration and element values to obtain the desired performance.

It is proposed that a new first-level electronic circuits textbook be developed, based on the D-OA approach. The D-OA paradigm is not new, having been developed by Caltech professor R. David Middlebrook over many years. In his second-level circuits course, he developed a design-oriented approach in which the design objective is kept in view from the start and maintained throughout the process. Some of these techniques resulted from his consulting experience "putting out fires" caused by nonoptimal designs that usually were supported by little or no analysis.

His first paper on the subject, "Low-Entropy Expressions: the Key to Design-Oriented Analysis," was presented at the IEEE Frontiers in Education Conference in 1991, and followed the next month by an EETimes article, "Analog design needs a change in perspective."

Since his Caltech course was "after-the-fact," he began doing a short course for design engineers in industry, titled "Structured Analog Design," which has been well appreciated by more than a thousand attendees throughout North America and Europe. The most frequent comment was "Why didn't I learn this in school?"

Two professors who had taken the Structured Analog Design short course, Don Peter and Art Witulski, pressed Prof. Middlebrook to develop a first-level course so that students *could* learn the methods of D-OA from the get-go, rather than having to do it "remedially."

Prof. Middlebrook thereupon set up an Analog Workshop, inviting professors to consider the questions:

1. Can the D-OA paradigm, successful at a second-level and working engineer level, be equally successful if adopted at a first-level active circuits course?

2. If so, a different perspective must be employed. How can this be done?

The outcome was unanimously positive, and a second workshop took place a year later. Twenty-seven people participated in Analog Workshop I and/or II, including the three people who have themselves conducted Middlebrook's Structured Analog Design Short Course. To include the "user's" point of view, some engineers from industry were invited to participate in AWII, and their companies helped to support the workshop.

Several AW participants, including Prof. Middlebrook, committed themselves to producing material incorporating the D-OA techniques as supplements to various well-known texts.

The Proposed New Course, incorporating Computer-Assisted D-OA

The product of the previous work was D-OA supplements to existing textbooks. Profs. Middlebrook, Don Peter, and Art Witulski, the instigators of the Analog Workshops, have done this and found that the incompatibility between the conventional and D-OA approaches makes it difficult to pursue both at the same time. They conclude that a new D-OA-based textbook should be developed.

The first step will be to determine the material to be covered, starting with time domain and frequency domain properties of basic circuit blocks and culminating in the specialized applications to high frequency analog amplifiers (including configurations made practically possible by IC technology, such as current mirrors, cascodes, and diffamps), power electronics (switching converters), and digital electronics (gates, etc.). It's interesting that most conventional texts do not discuss power electronics, perhaps because such applications were primarily developed in industry rather than in academia.

In any case, a more suitable classification for the future might be to describe the building blocks in terms of the properties of passive and active RLC element combinations, leading to the specialized applications in terms of signal processing and power processing objectives. In signal processing, subdivided into analog and digital (and mixed-signal) we want to preserve the information content while avoiding inductance, because inductors do not fit on to an IC; in power processing, we want to preserve the power content while avoiding resistance and therefore linear operation of active devices, because resistance is lossy.

The proposed D-OA-based first-level course should be the foundation for all electrical/electronics engineering curricula, because all engineers should be familiar with the language of low entropy expressions, even if they didn't derive them themselves.

A member of a design review committee should insist that a presenting design engineer should show his results in low entropy forms, so that the committee can determine whether the results are reasonable and can contribute meaningfully to the discussions, instead of just saying "Well, it looks as though it's coming along all right; carry on!"

The same applies to managers, reliability, system integration, and procurement engineers.

The second step, which actually will be concurrent with the first, will be to develop "Computer-Assisted D-OA." This is *not* a conventional circuit simulator that merely does simultaneous solution of the loop or node equations without contributing any physical insight, whether the result is numeric or symbolic.

Instead, the purpose of Computer-Assisted D-OA is to implement the sequential circuit reduction steps that the User would normally do by hand analysis. For example, the User would enter a symbolic gain expression as the ratio of polynomials in complex frequency s, and type the command "Extract the constant term from the numerator." The program would write the constant term in front, and divide it into the remaining terms of the polynomial. Then, the same for the denominator. Then, the User would type the command, "Define the ratio of the numerator and denominator constant terms as a reference gain."

Next, the User would type the command "Normalize the s terms to $p=sCR$," where C and R are specified in terms of existing circuit elements. The program would replace s by p, and would multiply the corresponding coefficient by the appropriate power of CR, thus making each coefficient dimensionless. The User could then instruct the program to make each coefficient a modified low entropy version of his choice.

In a following sequence, the User could instruct the program to insert predetermined tentative numerical values for the circuit elements into all or selected coefficients of the polynomials. The program would *not* immediately compute the “answer” for the value of the coefficient; it would repeat the low entropy format for the coefficient with each element symbol replaced by its numerical value. The User could then decide whether any terms are negligible, and if so instruct the program to drop those terms but to retain the others in symbolic form. This implements one of the valuable techniques of D-OA, namely, “Using numerical values to justify *analytic* approximations.” Of course if, as part of design iterations, the User changes numerical values, the validity of the approximation needs to be checked, but *you can’t lose by trying*.

An extension of this technique is “Finding approximate analytic roots of an arbitrary degree polynomial,” in which the program would use numerical values to check certain inequalities that permit analytic factorization of the polynomial.

For all the commands, the User retains complete control, and the program simply does what it is told to do.

Later versions of Computer-Assisted D-OA could incorporate a schematic capture front end that could do the sequential reductions directly on the circuit diagram. For example, the User could type the command “Make a Thevenin equivalent of the circuit to the left of Port AB,” and the program would redraw the circuit with the Thevenin elements labeled with low entropy combinations of the original elements.

The overall objective of Computer-Assisted D-OA is to do as much of the writing, algebraic manipulation, and numerical computation as possible. This allows the User to work much faster. Besides, the computer doesn’t make misteaks!

The Bottom Line

As Gordon Moore said, “[Analog design is] more of an art form than an engineering science.” This will continue to be true, since there are never nearly enough equations to solve for the number of unknowns. However, D-OA can greatly increase the contribution of engineering science to the design objective.