A. HYBRID COMPUTING ELEMENTS


This paper describes an Integrating Analog-to-Digital Converter whose principle of operation is as follows. The output of an analog integrator is monitored by two comparators for positive and negative overflows with respect to a given level q. The comparators drive a bidirectional counter. After every overflow, a pulse of current is added to the integrator input; the polarity, magnitude, and duration of this pulse are calculated to offset the output of the integrator by the amount q.

The circuit concept lies somewhere between that of the "hybrid integrator" and that of the continuous ADC (analog-to-digital converter); both ideas have been discussed extensively. What is new, perhaps, is the authors' suggestion that the IADIC, by combining the operations of integration and AD conversion, is the AD counterpart of the MDAC (multiplying digital-to-analog converter) and should be as useful at the AD end of a hybrid system as the MDAC at the AD end. Three reasons are given to support this proposition: 1) by using a multiplicity of IADICs instead of a conventional ADC, analog sample-hold circuits are not required, 2) the multiplexer is eliminated, and 3) the ADC is not required.

These arguments are, at best, of questionable validity. The sample-hold amplifiers ahead of the conventional multiplexer-ADC are needed to assure that all analog values crossing the AD interface refer to the same point in time. This fact greatly simplifies the digital program. With a multiplicity of converters, either conventional or IADICs, sampling is still necessary but must be done digitally with dual-buffered outputs. Multiplexing is primarily a reflection of the digital computer architecture: even with multiple AD converters, the digitized information must be funneled through a limited number of I/O channels (usually just one). Multiplexing with IADICs will then be digital rather than analog in nature, and the cost differential is not large. Of course, it is entirely possible that digital computers that are essentially parallel in structure (e.g., SOLOMON, ILLIAC IV, parallel DDA) may be able to take advantage of a multiplicity of AD converters.

The last "advantage" claimed for the circuit—that it eliminates an ADC—is, in fact, its most serious drawback. The authors did indeed eliminate the DA feedback normally found in continuous as well as conventional ADCs, but the lack of this feedback must be compensated for by providing a very high integrator gain, which (as the authors point out) presents unfortunate hardware problems. Without the high gain feature, the digital value in the bidirectional counter could (and normally would) be in error, since the output of the analog integrator would then constitute a significant portion of the value of the integral. In fact, with low integrator gain, the circuit approaches the "hybrid integrator" concept previously mentioned and is subject to the restrictions and special mode of operation appropriate to that device.

The scaling scheme proposed for the IADIC is unrealistic, except perhaps for special-purpose, fixed-configuration computers. This is due in part to the use of a synchronous pulse generator whose period must be adjusted as part of the scaling process. The reset pulses should preferably be generated asynchronously, in response to the comparator outputs. However, even this modification will not ease materially the scaling difficulties that are inherent in this type of device.

The suggestion that the lack of reset and hold modes for the IADIC is a desirable attribute cannot be taken seriously by anyone familiar with general-purpose analog computation.

The presentation is clear and to the point.

OMRI SERLIN
Control Data Israel
Tel Aviv, Israel


This paper describes a number of computing elements, each combining AD or DA conversion with one or more arithmetic operations or function generation. The elements, termed PHENOs, are claimed to have bandwidth-accuracy products equal to or better than the best linear analog computing components. The authors suggest that PHENOs may be put to good use in two areas. First, in general-purpose hybrid computation, where they represent a logical extension of the MDAC (multiplying digital-to-analog converter) in improving the AD interface. Second, in applications requiring miniaturization, the ability to function in an adverse environment, and the ease of interface with DDA (digital differential analyzer) components, as in airborne and shipboard devices. Both claims appear to be well-founded and merit further investigation by hardware experts in these two fields.

The authors briefly describe how an MDAC, alone or in the feedback path of an ADC (analog-to-digital converter) may be used to obtain analog products or quotients of an analog and a digital variable, as well as digital quotients of two analog variables.

By far the most interesting discussion concerns new designs for an ADC and a function generator.

The PHENO ADC combines the principles of continuous conversion and sub-ranging. The parallel comparison section of the conventional subrange ADC is replaced by a continuous converter using a 10-bit bidirectional counter operating at 6 MHz and 0.1-percent accuracy (±10-volt inputs). The most significant 4 bits of the converted quantity are generated by a "subrange selecting" DAC constructed of...
much slower logic and operating at 0.01-percent accuracy. The ADC incorporates a “look ahead” feature which automatically prepares the next subrange before it is actually required. In addition, the subranges are overlapping. These two features prevent the instabilities which are normally encountered when the input oscillates about a subrange boundary. Furthermore, an automatic “step size” selection feature adjusts the length of the bidirectional counter in inverse proportion to the rate of change of the input signal, resulting in a constant bandwidth-accuracy product. While the cost of these features is by no means negligible, the resulting improvement in performance is considerable. The PHENO ADC also provides, at no added cost, the analog equivalent of the low-order 10 bits, and a “change” pulse indicating a transition between subranges. Both are useful in hybrid function generation. It should be noted that the ability of the PHENO ADC to “fend for itself” in the presence of an oscillating input removes one of the major objections to the use of analog breakpoint identification in hybrid function generation; namely, the danger of overloading the interrupt facilities of the digital computer.

The function generator (termed FGDAC for “function generating DAC”) is based on the well-known fact that under certain circumstances, the digital representation of a function argument contains the correct lower breakpoint number in the most significant bits and the correct interpolating ratio $(x-x_i)/\Delta x$ in the least significant portion. The authors use potentiometers and an FET switching matrix driven by the breakpoint portion of the argument value to obtain the function value $f_i$ at the lower breakpoint, as well as the difference $\Delta f=f_{i+1}-f_i$. These quantities are fed into a summing MDAC whose digital input is the interpolating ratio. The output of this MDAC is the interpolated function value

$$f_i + \Delta f \frac{(x-x_i)}{\Delta x}.$$

By placing such a FGDAC in the feedback path of an ADC the authors also obtain an inverse-function generator.

The paper is slightly marred by the authors’ attempt to defend the misconception that a multiplicity of ADCs should eliminate the need for sampling and multiplexing of analog signals in a hybrid system. Actually, as pointed out by this reviewer in another review,1 the sample-hold and multiplexing operations are functions of the characteristics of hybrid simulation and the I/O structure of the digital computer. The authors are correct, however, in implying that a multiplicity of continuous ADCs permits a natural interface to a DDA (digital differential analyzer).

Curiously, while the authors take pains to define the common term “DDA,” they neglect to interpret the acronym of their own creation, PHENO.

Readers familiar with hybrid computing, AD conversion, and function generation will find the paper well worth the effort.

Omri Serlin  
Control Data Israel  
Tel Aviv, Israel

---

1 See foregoing review of “The IADIC: a hybrid computing element.”

**B. HYBRID SIMULATION**


This is a somewhat discursive but informative article on the hybrid Apollo docking simulation. The authors do not identify their objective in writing the paper until the very last sentence of the concluding section, where they state: “It is hoped that some part of the description of the problems, solutions, and experiences will be useful to others who are now involved, or who may become involved, in simulations similar to the one described.” With that as an objective, much of the detailed material on problem characteristics and specifications of the docking test device could have been condensed into an introductory or parenthetical paragraph. The discussion of why a hybrid computer was selected in preference to an all-analog or all-digital system, and of the way the problem was distributed between the analog and digital portions of the hybrid system, also adds little to the paper or to previously published material on these subjects.

However, the authors have considerable working experience with hybrid systems, and have a great deal to tell us regarding problems and solutions. In large measure, therefore, they do accomplish their objective. In particular, they discuss the importance of a careful documentation procedure for recording modifications in the simulation, and of defining and implementing a setup and checkout philosophy for the overall hybrid configuration. Each of these is the type of mundane matter that is often overlooked, and yet can be critical to the success of a project. Automation of setup, checkout, and maintenance procedures, utilizing the resources of the digital computer, undoubtedly will receive increased attention as the digital portions of hybrid systems continue to grow in size, complexity, and cost. For as this happens, time lost in such activities becomes an increasingly significant economic factor. Fortunately, however, the feasibility of automating complex sequences of operations also increases as the digital computer grows. The authors review the evolution of their procedures and make some interesting comments on what they would now do if the Apollo simulation were just being started. They would develop, for example, a problem-oriented interpreter for the pot setting and static checkout calculations, the objective being to permit rapid modification of the analog program with corresponding setup program changes. This approach would also allow them to do a static check for each configuration. The interpreter would operate under the main simulation control program and would be able to communicate with the rest of the simulation by means of a symbol table; it would also be able to handle multiple analog computers. The authors also note the need for a fully automatic scaling program.

Other problems discussed include those relating to the mechanics of operating a simulation facility when the computer and simulator are housed in widely separated buildings, the matter of selecting suitable numerical integration methods, and the ever-present problem of providing adequate grounding and noise rejection between the several elements of the simulation facility.

Hopefully, future articles by these authors will reflect their experience with these proposals, and again contribute to our understanding of practical problems.

Martin Y. Silberberg  
IBM Corporation  
San Jose, Calif.

---


This paper deals with the problem of selecting the best approximation to an arbitrary function, using a given number of straight line segments to form the approximation. This is, of course, the basic problem of programming functions for generation by analog function generators. The author has formalized a noteworthy technique for optimal selection of breakpoints, and coupled his technique with a slope-selecting procedure based on least-squares optimization.

The subject matter is especially pertinent today with the increasing application of the card-memory type of diode function generator and its attendant need of digital autoprogramming support, as well as the formalized programming of hybrid multivariable function generation.

The author outlines a scheme for optimal slope selection which is limited to approximations having equally spaced break points. This technique, presented in 1963 by K. B. Norkin, selects an optimal set of slope increments and is based on a least-squares optimization of slope selection.