A panel discussion

The impact of new technology on the analog hybrid art–I

G. A. BEKEY, Chairman
University of Southern California
Los Angeles, California

Present day hybrid systems are characterized by increasingly sophisticated software requirements. The first attempts at creation of useful analog-digital computer systems were faced with a multitude of hardware problems associated with communication between discrete and sequential machines on the one hand and continuous and parallel machines on the other. Now, however, since the hardware marriage has been successfully consumated, a multitude of software problems remain. This panel will concentrate on the most important of these problems. Position papers by each of the four panelists are presented below.

Hybrid executive and problem control software

by E. HARTSFIELD
TRW Systems
Redondo Beach, California

Hybrid executive and problem control software provides features quite analogous to those provided in Monitor systems associated with stacked job oriented data processing centers. The major difference is that a hybrid executive routine stresses *execution* time utility as opposed to the compilation and assembly time utility emphasis of stacked-job Monitors. If one accepts the proposition that hybrid simulation is an extension of analog simulation, then the need for a software system that facilitates communication between the user and his program is brought clearly into focus.

Unlike a data-center mode of operation, where the user submits his job to an operations desk and does not receive results (good or bad) for several hours, a hybrid user operates on the same program for several hours at a time. In addition, there are a number of operations common to a vast majority of hybrid problems that lend themselves to standardization through an executive routine. This would include such operations as data input and output, bilateral problem control (i.e., the sequencing of the analog computer through modes from either the digital computer or the analog computer), and problem debugging procedures.

Hybrid executive software is aimed at solving these problems. There is a wide diversity of executive routine philosophies and implementations, reflecting, perhaps, the fact that no two hybrid installations are the same either in a systems sense or in terms of the modes operandi of the facility.

This presentation reviews the general state-of-the-art in hybrid executive software and some of the major considerations one faces when initiating the development of software of this type.

Diagnostic software for operation and maintenance of hybrid computers

by R. E. LORD
Comcor/Astrodata
Anaheim, California

Diagnostic software is based upon the premise that programmers occasionally make mistakes and that sometimes hardware behaves less than ideally. The major problems in producing good diagnostics are: (1) how does one find the errors and (2) once they have been found, what does one do about them. In the case of maintenance software, the problems are primarily in finding the errors. Once found, the course of action is fairly clear: report it. In the case of operational diagnostics, the course is not quite so clear. For those operations that occur during a time critical phase, one must examine whether there is time available for complete diagnostics. or these cases, it is often better to build diagnostics into hardware than it is to rely on software. During non-critical phases of a problem, complete diagnostics can usually be performed and errors readily detected. The problems here are in determining what action to take. In general, batch processing takes the view that any error serious enough to diagnose for, is serious enough to cause an abort. The hybrid problem however, generally involves a high degree of man-machine interaction and hence, upon diagnosing an error, one should determine if the user can fix or quickly program around the error. If this is the case, then a complete report of the error should be made. Only as a last resort should an abort be initiated.

A large multi-console system for hybrid computations: software and operation

by C. K. BEDIENT
Lockheed Missiles and Space Company
Sunnyvale, California

In the first quarter of 1966, Lockheed Missiles and Space Company completed negotiation for a new hybrid computer system. This system contained four CI 5000 analog computers and two intracoms manufactured by Comcor, Inc. It also contained a 6400 digital computer system manufactured by Control Data Corporation. Astrodata, the parent company of Comcor, was given complete system responsibility including software development. This discussion traces this software development and the resulting operating system.

To implement this software development, Lockheed, Control Data, and Astrodata were each to supply programming support. To direct the programming effort, Astrodata retained Dr. Ralph Dames of Spectrodata.

The first part of the discussion reviews the features of the 6400 computer and the Chippewa operating system. It reviews the modifications required to this system to give it the features required to do hybrid computation. Tracing the design of the linkage equipment it is shown how features of the hardware and software are used to advantage to produce the final integrated software hardware package. This includes a description of interrupt processing, pattern input-output, and the effects of multiproblem usage.

The second part of the discussion shows the organization of a simple problem and demonstrates its operation on the system.

Other features of the system are briefly outlined, including:

1. Fortran Library
2. Display System
   a. Job Control
   b. Variable Display
   c. Source Modification
   d. Input/Output
   e. Preventive Maintenance
   f. Engineering Aids
   g. Hybrid Utility
   h. Datafile Display
3. Preventive Maintenance
4. Automatic Problem Verification

Simulation languages and the analog hybrid field

by JON C. STRAUSS
Carnegie Institute of Technology
Pittsburgh, Pennsylvania

Languages to facilitate the representation and simulation of continuous dynamic systems on digital computers are currently related to analog/hybrid computation mainly in the sense that they both are concerned with the same problem class (with the possible exception of real time simulation). There is, however, some indication that current work will lead to general hybrid programming languages. This brief position paper describes some of the current work in simulation languages, its relation to previous work, and its potential relation to the analog/hybrid field.

The early simulation languages were very much influenced by the outlook and objectives of their analog computer oriented designers; i.e., the work was motivated by the need for accurate dynamic check solutions. As digital computers become faster and less expensive, these analog oriented simulation languages were increasingly employed for stand alone digital simulation. The justification was generally based on the greater accuracy, accessibility, and result reproducibility of the digital computer coupled with a much smaller relative setup cost and time for one shot simulations.

Due primarily to the orientation of their designers, most of the early simulation languages provided for continuous system simulation by simulating the operational device characteristics of the analog computer; system simulations were programmed by, in effect, describing an analog computer wiring diagram and pot setting sheets to the computer. It was not until very recently that simulation language designers began to take advantage of the fact that the simulations are being performed on a digital computer with capabilities that are just not present on the typical analog. The most recent and certainly the most carefully designed simulation language is CSSL (the Continuous System Simulation Language); it is the product of several years work of the SCI Simulation Software Committee. CSSL, although not presently implemented in its entirety, serves as an excellent basis for discussion of simulation languages, their relation to, and their ultimate effect on, the analog/hybrid simulation field.

CSSL was designed primarily as a digital simulation language but care was taken to provide for future upwards compatible expansion to a complete hybrid programming language. The main contribution of CSSL lies not in its detailed syntactical structure but rather in the underlying ideas that it presents. These include:

1. The explicit recognition of the different requirements of simulation in general; namely: model representation, programmed experimental control, interactive control/communication, and problem oriented operators to describe their problems.

2. The need for a flexible model representation scheme so that investigation in different fields can use the same language system with different sets of problem oriented operators to describe their problems.

3. The concept of programmable structure that permits and, in fact, encourages use of the same simulation system by users, and for problems, varying greatly in sophistication.

The question at hand is how does all this relate to the analog/hybrid field. The following observations are pertinent if somewhat controversial:

1. Simulation Languages and all digital simulation will continue to find greater application areas now considered to be the exclusive province of analog and hybrid computers. This trend will be accelerated by the development of graphic consoles and time shared programming techniques that will allow the user to achieve the same or perhaps even greater degree of communication with his problem that he now enjoys with analog computation.

2. As evidenced by the design features of CSSL, development of HSL, a dialect of CSSL, by EAI, and recent indications in the literature (e.g., R.T. Dames, Simulation, March 1967), simulation languages will find increased application as hybrid programming languages; i.e., they will be used to program the digital portion of a hybrid problem.

3. If the interest in, and need for, hybrid computation continues to increase, the current unpleasant task of getting a problem on the computer will be automated through development of hybrid computer analogs of APACHE. Such work could be readily mechanized in the general framework provided by CSSL.