LA-3930-MS

SUPPLEMENT

C.3

LOS ALAMOS SCIENTIFIC LABORATORY
of the
University of California

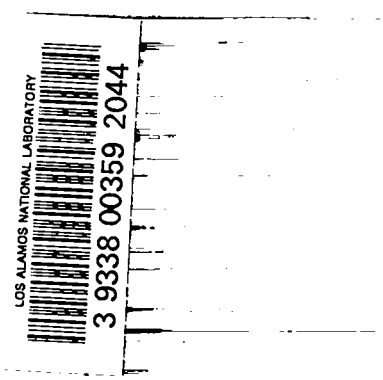
LOS ALAMOS • NEW MEXICO

Semiannual

Atomic Energy Commission

Computer Information Meeting

May 20-21, 1968



UNITED STATES
ATOMIC ENERGY COMMISSION
CONTRACT W-7405-ENG. 36

This report expresses the opinions of the author or authors and does not necessarily reflect the opinions or views of the Los Alamos Scientific Laboratory.

LA-3930-MS SUPPLEMENT
DISTRIBUTED TO ATTENDEES

LOS ALAMOS SCIENTIFIC LABORATORY
of the
University of California
LOS ALAMOS • NEW MEXICO

Report distributed: May 21, 1968

Semiannual
Atomic Energy Commission
Computer Information Meeting
May 20-21, 1968

APPLIED MATHEMATICS DIVISION

of the

ARGONNE NATIONAL LABORATORY

Summary of Computer Activities for the
AEC Computer Information Meeting

Los Alamos Scientific Laboratory

May 20-21, 1968

CONVERSATIONAL TIME SHARING

At the AEC Computer Information Meeting held at Brookhaven National Laboratory November 1 and 2 of 1967, we reported on the development plans for a fundamentally conversational time sharing system. In summary, input to the system is through a typewriter-type device, and computing requiring large segments of time and memory and not amenable to rapid interactions with the user will not be available in this mode of use. Plans call for the time sharing system to operate initially in a second IBM 360 priority partition of the Model 50 while the first priority partition is occupied by ASP. Time sharing will thus operate in the background to batch processing. Our studies of CPU usage by ASP assuming normally running jobs on the Model 75 support a prediction that at least 15 active users with at most 2-second response times are possible with a full ASP load.

Time sharing facilities to be available to the users may be described in five classes: 1) information management; 2) preprocessing; 3) batch job entry; 4) interpretive computing; 5) computer as a desk calculator.

The heart of the time sharing system is the time sharing monitor, which maintains the communication links with the user consoles, allocates the software and hardware resources to the user, and controls the time slicing. The first version of the monitor is complete, has been checked out, and is operational. More complete and efficient versions of the monitor are in progress. It is anticipated that a nearly final version of the sophisticated control program will be operational by mid-summer, 1968.

The first of several user languages is in advanced stages of checkout and is operational on a restricted basis. This language is noted by the mnemonic DISCOM and is described as a user resource in Applied Mathematics Technical Memorandum No. 149, dated November 17, 1967. ("Purpose and Scope of the DISCOM Interpretive System," available from the author, Charles J. Smith.) PL/1 and FORTRAN preprocessors are under construction.

The remaining facilities, including information management and remote job entry, are in various stages of design and component testing. The design concepts have been verified either through simulation or through operation of skeletal systems. The feasibility of the concepts has been proved and the operability of the major elements has been demonstrated.

COMPUTER UTILIZATION AND SYSTEMS DESIGN STUDIES

The Dynamic Status Recorder (DSR) system has been implemented by staff of the Applied Mathematics Division to serve as a tool for gathering data about the operation of a computing system. It is imbedded in the software monitor that controls the scheduling and peripheral input-output functions of the system. The output of DSR is a tape which contains elements representing the various events that have occurred. This tape allows us to simulate the operation of the monitor so that another program may analyze the functions of the system in detail at a more leisurely pace.

Such a simulator has been written. It is structured so that each event that may occur is processed by a separate subroutine. In addition a user's routine may also be executed for those events that describe the operation of a particular subsystem of the support processor monitor. The group of user's routines may tally events or produce averages, standard deviations, and other statistical measures of activity; these are formatted and printed at the end of the analysis.

An analysis was performed on two hours' worth of data to determine the amount of time that was available to another system residing in the other half of the support processor. The results showed that a total of 63.5 minutes was available to the second system. Out of that time, 57% of the intervals allotted were longer than 85.3 milliseconds; this leaves enough time to run a graphics system or a conversational computing system in the other half of the support processor.

The DSR work is reported thus far in three Applied Mathematics Division Technical Memoranda (available from their authors):

158 - "ASP Dynamic Status Recorder Operator's Guide," by A. F. Joseph and A. R. Hirsch.

159 - "Dynamic Status Specifications and Maintenance Manual," by A. R. Hirsch.

160 - "The ASP Dynamic Status Recorder Analysis Guide," by A. F. Joseph and R. Krupp.

A very low level of effort is being applied to the development of a similar monitor for the main processor. The level of effort reflects only a difficulty in allocating technical resources.

COMPUTER OPERATIONS

Since the November 1967 AEC Computer Information Meeting, computer operations for Argonne's two major computing systems may be described as follows: the CONTROL DATA 3600 continues to furnish a relatively stable and trouble-free flow of work, as it has since its major developmental work was completed. The IBM System/360 computing system, which was installed in its almost-final form in September 1967, is producing in excess of 100 hours per week of useful computations. The hardware is relatively reliable, but software still causes difficulties.

In February, the Applied Mathematics Division's 4K, 3-magnetic-tape 1401 was replaced by transfer from the Accounting Department of a 16K, 4-magnetic-tape 1401 system. This is used for routine peripheral processing.

REMOTE ACCESS DATA SYSTEM (RADS)

Initial software for RADS has been written and is now being debugged. The system is intended to provide means both for entering jobs into the batch stream of the IBM System/360 computer and for receiving output at remote locations, but may be expanded to include direct coupling to experimental devices for data collection.

RADS consists of (1) a message switching computer (MSC) interfaced by direct wiring to the IBM 360 Model 50, and (2) remote terminals interfaced to the MSC through telephone lines. The MSC will manage the data flow between the remote terminals and the Model 50. To the Model 50, the MSC will appear as additional card readers, line printers, and tape units. Each remote terminal will consist of a small computer, card reader, line printer, and, optionally, other devices such as plotters, laboratory devices, and paper tape equipment.

The Message Switching Computer, a Varian Data Machines 620I, is connected to the IBM 360 via an IBM 2701 data adapter unit. Job entry to the 360 system has been run using test programs from the MSC, and further testing of the data output capabilities are being completed. The first remote station, now under construction and evaluation, has a Varian 620I control computer, plus a card reader and a line printer.

Basic software for the remote station is now being debugged.

Not only have test programs between the MSC and the first remote station been run, but total system tests of remote entry of jobs from the remote

station to the 360 should start soon with initial operation of version I of the system by the end of the summer.

The initial array will consist of six stations at various remote locations. The first such station is scheduled for completion by the end of 1968.

Some divisions are requesting other devices at their locations: a card reader-punch combination, Calcomp plotter, magnetic tape, a faster printer. The engineering and programming for these devices will start after the basic system is in operation.

Other potential optional devices are a low-cost ink-jet serial printer and an incremental magnetic tape transport. These units would allow extension of the RADS facility to low-volume users.

PAULETTE EMULSION SCANNER

The goal of this project is to provide a computer-controlled scanner that will automatically count the tracks of nuclear particles on fine-grain emulsion plates exposed in a magnetic spectrograph. This scanner, utilizing a commercial image dissector as the scan tube, has been refined to the resolution limits of the image dissector.

The scanner as it now exists is considered operational for carefully prepared plates which are properly developed and on which the track densities are not too great. For plates of this type the scanner yields excellent quantitative data, and the scanner is being used for this type of production. For plates with high track densities, however, the scan tube resolution limits results to a more qualitative nature: the exact peak positions are given but the peak magnitudes are in error.

Initial evaluation of a new type of image dissector made by a different manufacturer indicates that the desired resolution might be achieved. In view

of this and the fact that the present tube is no longer in production, it is deemed advisable to proceed with development of a scan station utilizing the new tube. Fortunately, utilization of a new scan station will require no changes in the existing control logic, interface, and data collection programs. During this development period, the existing scanner will be used on a production basis.

MOSSBAUER-EFFECT DATA COLLECTION SYSTEM

This system, utilizing a small digital computer with interface, has been installed in the Physics Division. It was developed for the purpose of collecting data from four Mossbauer-effect experiments, of providing a display feature, and of outputting data onto paper tape. The system as it stands is performing very well and is considered fully operational in replacing up to four time-store-mode analyzers.

One function which periodically requires an analyzer is that of experimental apparatus calibration. The computer system as installed was not intended to perform this function, but subsequent discussions have brought to light the possibility of performing the calibration in addition to data collection. This possibility will be evaluated.

OPTO-ELECTRONIC INFORMATION STORAGE

The goal of this project is to store some 16 million binary digits of information on a plate of glass six inches square. Since the writing and photographic development of a single plate will take several hours, it is intended that this be a rapid access fixed store. Once the plate is prepared it is estimated that access to any group of 4096 bits can be accomplished in 2 microseconds, with all the bits being read out in parallel.

Most of the work done to date has been with regard to holographic preparation and in particular to the verification of theory. This will continue during the current initial stages of preparing for the construction of the electro-optic crystal array for read-out light beam deflection. At the same time holographic storage crystals that employ Bragg-Angle storage will be evaluated. The crystals could provide a storage medium much closer to conventional read-write times.

PATTERN RECOGNITION STUDIES

Current fingerprint identification work has centered around a program to extract ridge-endings from a fingerprint in the presence of a considerable amount of noise. These ridge-endings are to become a set of descriptors to be used in a fingerprint classification and identification system.

It is believed that the set of descriptors used for locating ridge-endings in the fingerprint work can also be used for other pattern recognition problems and possibly for graphical problems. These possibilities will be further explored.

BRAILLE READER SYSTEM

Interest in a portable Braille reader for the blind has recently developed. This unit would "read" a low-density magnetic tape containing the Braille translation of some written document such as a book, and would "write" what had been read onto a reusable plastic belt for presentation of the actual Braille characters to the user. There would be appropriate controls for speed variation, skipping, indexing, and backing up in the text.

The need for such a device arises from the systematic limitations of other approaches presently in use. Examples of these are the tremendous bulk of Braille books, and the inflexible speed and passive role of the user of voice recordings.

The two apparent areas of interest in this system are:

(1) The development of the actual reader.

(2) The development of a computer conversion system to translate text to Grade II Braille and to prepare compatible magnetic tapes.

There will be reader development costs but no large items. The translation system will most certainly require the use of a small computer and may require the purchase of a specific machine into which special hardware may be incorporated. Development time for the reader and the translator should be about one year each, not necessarily running concurrently.

COMPUTER-CONTROLLED FILM MEASURING SYSTEMS

A. POLLY

The POLLY I bubble chamber film measuring system, built in conjunction with the High Energy Physics Division as a prototype model, is now doing production film processing. Recent engineering and programming improvements are expected to result in an average measuring rate of 60 events per hour, with a peak rate of 80, as opposed to the original 30 events per hour and peak rate of 50. Measurement is considered to be as precise as that of conventional manual measuring tables. With the transfer of both POLLY I and the PDP 7 controlling computer to Building 362 in September 1967, the focus of responsibility for the project shifted to the High Energy Physics Division.

An improved production version, POLLY II, is nearing completion in the High Energy Physics Division. It features a more sophisticated operator's position with a true optical display capability, and utilizes a Scientific Data Systems Sigma 7 as the control computer. Applied Mathematics staff have continued to participate in developments and improvements.

B. ALICE

It is proposed that a general purpose film scanning device (ALICE) be built to take advantage of improvements in cathode ray tube technology and of the much faster control computers now available. Such a device would be more accurate, more versatile, and five to ten times as fast as the present CHLOE system. Its two major elements would be a commercial digital computer with magnetic tape and an Argonne-built cathode ray tube scanner.

CHLOE has proved useful in a wide variety of applications, as evidenced by a recent compilation of some 84 references to it. Potential improvements to CHLOE that could be implemented in ALICE, however, include:

1) Input facility for varied types of film material.
2) Increased resolution.
3) More accurate discrimination among shades of gray.
4) Rotational capability for input.
5) Color discrimination.
6) Flexibility in man-machine interaction.
7) Higher level programming language.
8) Several input stations, including perhaps a reflected light scanner.

It is planned to pay significant care to the light measuring capabilities of the equipment so that stable, reliable optical film density measurements can be made. Furthermore, some use would be made of analog deflection systems to allow for less accurate but very fast searching for information to be digitized.

NUMERICAL METHODS

A. Approximation of Functions

Over the past few years much work has gone into methods for generating approximations to functions, and into the generation of rational Chebyshev approximations for some of the special functions of mathematical physics. Recent effort has centered on Fermi-Dirac integrals, Fresnel integrals, exponential integrals, and rational Chebyshev approximations using linear equations.

Current work is concerned with approximations for the exponential integral Ei(x) and for the Coulomb phase shift, and with the proper treatment of the nearly degenerate case in rational approximation. The project is a continuing one.

B. Matrix Codes

The necessity for providing basic programs for matrix calculations on the IBM System 360 machines has initiated and continued development of a set of routines refined to make full use of recent research.

A fully adequate matrix package should include efficient and foolproof codes for various types of matrices (dense, sparse, triangular, band, symmetric or Hermitian, Hessenberg, etc.) in at least real and complex arithmetic and with the possibility of electing improved accuracy for 1) solving linear equations (with various possible relations between numbers of equations and unknowns), 2) calculating inverses, and 3) obtaining eigensystems (eigenvalues only or also eigenvectors).

The package of matrix codes being developed have compatible calling sequences and storage requirements to facilitate the interchange of subroutines, and hence methods, among programs involving a sequence of matrix computations. It is intended that the more fundamental routines be stored internally (as in the scientific subroutine package) for convenience and to encourage use of

accurate, fully tested algorithms.

The present effort is directed toward bringing Argonne's matrix subroutine library to the level of algorithms in the current literature, e.g., use of more-precise norms as error bounds, application of Rayleigh corrections in eigenvalue computations, and judicious use of multi-precision arithmetic.

C. NUMERICAL METHODS IN FUNCTIONAL ANALYSIS

The application of the theory of complex variables to the construction of numerical methods and algorithms is continuing.

Methods for finding zeros of analytic functions, for numerical differentiation, and for numerical quadrature based on complex function evaluation have been developed and found to compare favorably with corresponding classical methods and algorithms. It is hoped that the application of the same theory may lead to computational methods for integral and differential equations; an extended investigation along these lines is envisioned.

Other research projects have been in the areas of multidimensional quadrature and various special aspects of numerical quadrature.

THEOREM PROVING ON COMPUTERS

A theorem called the Maximal Model Theorem, which appears of interest in itself, has been proved, and a valuable application has been found. With the aid of this theorem a certain procedure based on the inference rule paramodulation has been shown to be a semidecision procedure for first-order predicate calculus with equality. A theorem-proving program based on this procedure is in the process of being developed, and abstracts covering these results will be published in the Journal of Symbolic Logic.

RESEARCH IN PROGRAMMING APPLICATIONS

A. Fluid Flow Problems

Work on convective flow continues, with a report on some comparison of the methods available to appear shortly (June 1968). The application of these methods has now reached a point at which convective flow through a two-dimensional heat exchanger with vertical and horizontal walls can be computed. The extension to two-dimensional systems having curved walls is in progress.

A paper describing a computing procedure which has been developed over the past four years to obtain steady supersonic gas flows in ducts has been submitted for publication, and a more detailed report will appear shortly.

B. Deformation of Fuel Plates

The safety of reactors using fuel plates of evolute cross section is the object of a study of the deformation of such plates by pressure and temperature stresses. The immediate object is the development of computing techniques using thin shell theory.

C. Biological Image Processing

Work on this project continues. The Biological and Medical Research Division is beginning a rather large project involving Chinese Hamsters, and an important part of this effort will be devoted to automated chromosome analysis. During the next year, the image analysis programs will be moved to the IBM 360/50-75 from the CONTROL DATA 3600.

D. Microscope Design

a) The systems design for a computer-controlled electron microscope will be largely completed in June, 1968, but construction will depend on the availability of funds. The microscope system could be considered for incorporation as part of the scanning machine in the planned film measuring system ALICE (see above).

b) A program is being written to calculate the magnetic field in a saturated microscope lens by a method involving Maxwell's equations in integral form. The field H (*not* B) is represented as the sum of a curl and gradient component and the integral equation is essentially derived from the condition that the divergence of B is zero. The (non-linear) integral equation is solved by a "source iteration" procedure similar to that used in reactor computations.

E. Perturbation Collapse in an Infinite Ocean

In the past year a computer program for solving the hydrodynamic equations of two-dimensional, incompressible flow has been written and thoroughly tested. The numerical method used is basically the same as the finite difference methods of Harlow and Fromm at Los Alamos. Interest has centered in the situation where a submerged body moves in a horizontal line through a liquid with a density gradient depending only on depth, leaving behind it a wake of mixed fluid which then collapses. The resulting disturbance can be approximately treated as a two-dimensional problem. Photographs of images produced directly by the computer on the screen of a cathode ray tube have proved of much value in the study.

During the next year the methods developed so far will be applied to more realistic problems such as: (a) collapse of a wake in fluids having a more realistic density gradient, including a thermocline; (b) collapse of a buoyant wake (paralleling experimental work done at Hydronautics, Inc.); (c) collapse of an incompletely mixed wake. A study of the long-time behavior of a collapsing perturbation by means of a numerical Fourier Transform solution of the linearized hydrodynamic equations has also been proposed.

It is also planned to study convective overturn due to surface cooling by applying the numerical methods developed so far to an ocean with a free surface undergoing a heat loss. There is interest in the onset and development of the downward convection which arises from the cooling of the ocean surface by conduction, radiation, and evaporation. If successful this approach should make it possible to observe numerically such effects as the development of the plunging sheets of cooled water reported by Spangenberg and Rowand (Physics of Fluids 4, page 743, 1961) and the development of the well-defined convection cells reported by other investigators.

If downward convection can be successfully treated by numerical solution of the Navier-Stokes equations it is planned to study the effect of the collapse of a perturbation in the underwater density gradient on the development of this phenomenon.

NOTE. This report (prepared by R. F. King) is to be regarded as an updating of selected portions only of the more complete account of the work of the Applied Mathematics Division which was distributed at the November 1-2, 1967 AEC Computer Information Meeting. While copies of the more complete report remain available, they may be obtained on request.

Wallace Givens, Director
Applied Mathematics Division
Argonne National Laboratory
Argonne, Illinois 60439

General Electric Company
KNOLLS ATOMIC POWER LABORATORY
Schenectady, New York

COMPUTER INSTALLATION REPORT

by

J. A. Beutler

May 16, 1968

COMPUTER INSTALLATION REPORT

Major activity at the Knolls Laboratory computing facility since the last progress report in September has revolved around the installation of the second CDC 6600 computer. The second machine is essentially a duplicate of the first but will be installed in a "loosely coupled" configuration, i.e., peripheral gear such as tapes, printers, card readers and microfilm will be shared via dual channel controllers. This arrangement provides better utilization of the peripheral hardware by making possible easy reassignment to the mainframe with the immediate need. The loosely coupled configuration also is attractive from a backup point of view particularly under situations of partial outage, e.g., if one central processor is down virtually all peripheral gear is available to the other machine thereby increasing its normal throughput. There is also the necessity of having common access to permanent files. This is accomplished via the dual channel controllers of the eight (logical) 808 disk units shared between the two machines thus avoiding the need for duplicate copies of large files or the execution of a job on only the machine where the pertinent files are located.

The installation of the second machine was complicated by the need to build an annex to the existing computer building. Unfortunately constructions was started after the first snowfall and sufficient delays were encountered so that the 6600 was ready for delivery three months before completion of the building was scheduled. The overall Laboratory computing load was well in excess of one 6600 so that the decision was made to accept delivery of the second 6600 minus the 808's and the dual channel controllers. Installation of this interim configuration was accomplished in December by closely packing the additional equipment in the area originally intended for one computer. Subsequently, during March the need for full capacity on the second machine caused us to ship and install the 808's in a temporary location formerly used as the CDC maintenance area. Finally, on April 1, the new annex was finished and equipment was moved - largely without interrupting regular operation. The new area is devoted to input-output and console operations and the older area contains the mainframes, disks and controllers. The result is a vastly improved setup for day-to-day operation although maintenance has to be done via headsets between the consoles and mainframes.

After enjoying the luxury of two full machines for two months, the original mainframe was shut down early in May for four to six weeks so that the necessary "hooks" for extended core storage (ECS) could be installed. It is anticipated that the first 500K words of ECS will be installed early in June with the second 500K arriving in November. The present plans for ECS call for access to one million words by both computers through a 6640 controller although consideration is also being given to dedicating 500K to each machine through separate controllers. The former configuration provides the most programming flexibility - the later provides the better operational backup. Reliability experience during the first few months is expected to determine which configuration is finally selected.

Upon completion of ECS installation and full availability of the dual 6600, the Philco 212 computer will at long last be retired. This computer passed its eighth birthday (actually, there have been a succession of upgrades over the years) on April 1. Even though the 212 outperforms most so called "third generation" systems and performs at a high level of reliability (typically 98% and better) it cannot compete on a cost-performance basis with the 6600-ECS combination and hence an era ends.

During the crowded interim operation and the moves KAPL made the switch from the SCOPE 2.0 to SCOPE 3.1 operating system on the 6600. Due to extensive pre-testing and specially written conversion aids by the systems group the transition was made with a minimum of interruption to the production work load. The availability of two machines simplified the conversion but the complexities of a mixed system operation will not soon be forgotten by our operations staff. The 3.1 operating system has resulted in an overall improvement in throughput of roughly 10%. Some further improvements in compute bound problems (typically 30%) is expected with the 2.4 version of FORTRAN which is now undergoing testing. With the advent of ECS, rather major systems and applications modification are anticipated in order to achieve further throughput improvements. KAPL expects to utilize the interim ECS operating system being written at Brookhaven and recently adopted by CDC to be the basis of a supported ECS software package.

During the past five months remote time-sharing utilizing a locally available service has been available at the Laboratory. This service will be replaced late in June when the CDC RESPOND system is installed on the 6600. As expected, the remote time-sharing service has significantly expedited methods development and short engineering problems and has consequently created a demand for additional terminals.

Over the past two years a Physics oriented language known as DATATRAN has been under development at KAPL and is now in regular production use. DATATRAN is an extension to FORTRAN providing rather extensive file manipulation capabilities and flexible commands for controlling the execution of modular programs. File manipulation includes the ability to reference files through a hierarchal naming structure and to create, merge, store and retrieve all or selected portions of data or program files. The KALL statement provides an ability to execute and pass data to and from a program or module in a sequence of FORTRAN statements within a DO loop and other FORTRAN logic. The primary applications of DATATRAN have been in the physics methods development area and in the processing and retrieval of cross-section data. An additional application has been in the reduction in the physical volume of input data and consequent reduction in input errors for large production runs of the NOVA reactor physics system. Currently DATATRAN has been implemented as a FORTRAN written preprocessor which translates DATATRAN statements into FORTRAN statements, subroutine calls, and sets up list structures required to pass file names during execution. Although some overhead penalty in computer time must be accepted when using DATATRAN, results to date indicate very significant reductions in manpower and elapsed time to complete methods development and cross-section evaluation work.

Computation Group
Staff Paper # 11
February 1968
C. Dickens and W.F. Miller

SUMMARY REPORT: SLAC COMPUTATION

July - December, 1967

I. SCC-SLAC Group

II. Computation Group

III. List of Publications and Reports

IV. Tables of Organization

I. SCC-SLAC Group

The principal activity during this period was the moulding of the IBM 360 Model 75 system into a batch production system. Late in this period there was also a major peripheral expansion installation, and software development for future systems continued. In the last three months of this period we have provided a limited multi-program capability.

A. Operations

During this period the operations staff gained experience with hardware acquisition and installation of an existing system and increased awareness of the development of procedures in the area of property control and hardware maintenance. The hardware acquisition included the second IBM 2301 storage drum,two IBM 2250 scopes, an IBM 2701 external world interface unit, and eight IBM 2260 display consoles plus a new printer with an alternate print train which will match the 2741 character set. This new equipment was installed on an existing system, while we continued production. It was installed by December and the Acceptance Testing continued through the end of the year.

B. Performance

The general schedule throughout this period consisted of a five-day two shift operation which has been in effect since August. On the weekend we scheduled one 8-hour shift of production work and provided available time during the graveyard shift. The following are tables of the allocation of time for the six months covered by this report. See also Figure 1.

	July	Aug.	Sept.(1)	Oct.	Nov.(2)	Dec.
Down Time	none*	10.0	40.0	3.0	47.0	12.0
Maintenance	none*	73.0	80.0	86.0	114.0	122.0
Systems	15.0	195.0	252.0	203.0	123.0	146.0
Users	48.0	196.0	165.0	225.0	246.0	291.0
Percentage of time available to user	76.2	41.3	30.7	43.5	46.4	50.9
Total Hours	63.0	474.0	537.0	517.0	530.0	571.0

--continued

* No time logged.

(1) In September we suffered the consequences of inexperience on the 360 System. Due to a lack of proper operations and maintenance procedures, the disk packs and drives mutually destroyed themselves.

(2) The month of November saw both a major installation of new peripherals and a 250 hour backlog of Engineering changes which comes under Maintenance. Also undertaken at this juncture was a major Software Systems change.

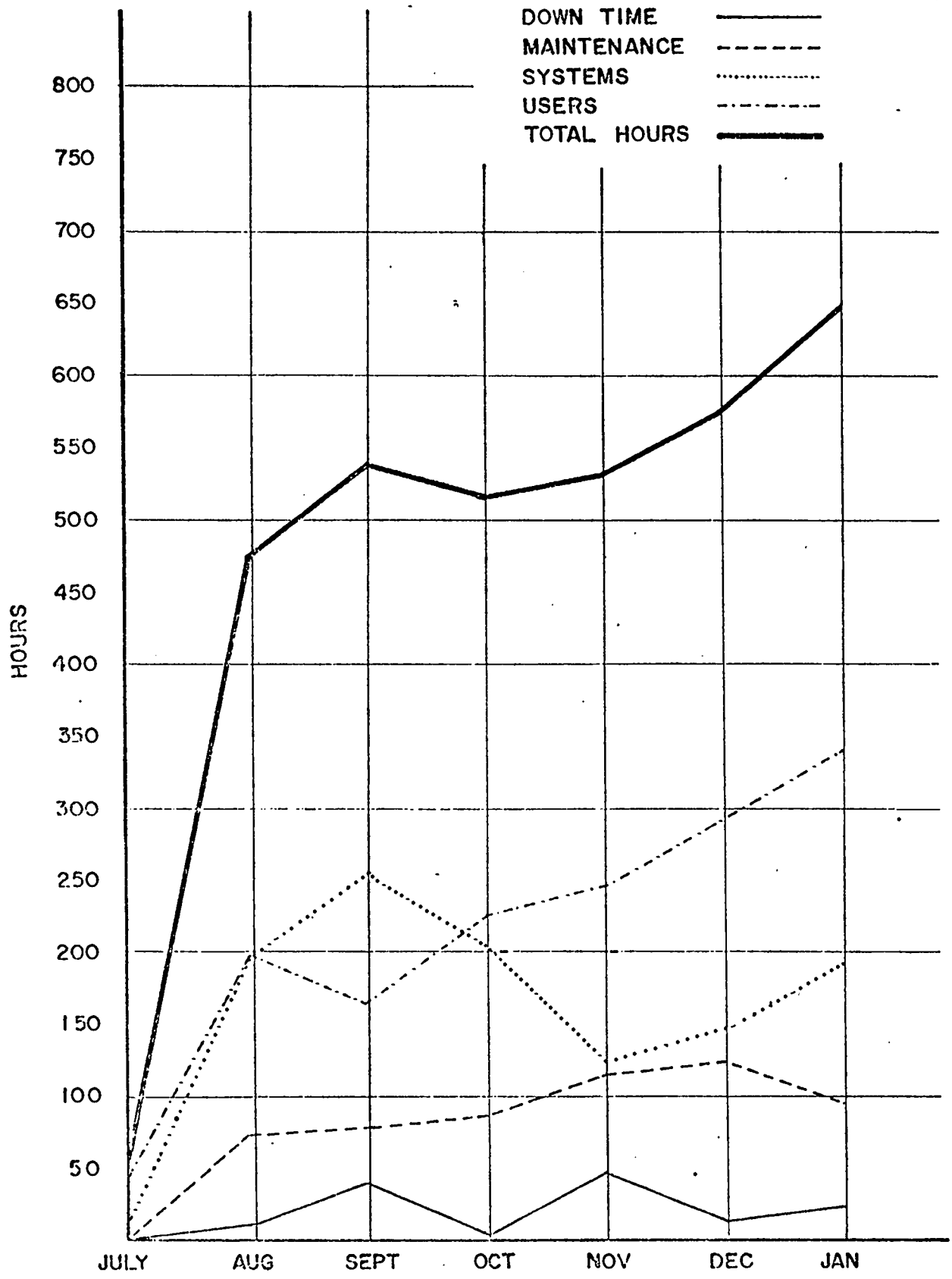
During this period on the third shift, we averaged 3.5 hours. Most of the jobs run during the graveyard were production type runs with longer average run times. In November, we found it necessary to schedule an additional Saturday shift for P.M. The Engineering Changes for the 75 turned out to be more numerous than we had initially anticipated. Preventive maintenance thus consisted of three hours each day during Monday through Friday and 16 hours on Saturday.

C. Software Development

Improvements to the scheduled HASP OS 360 system continued, and experimentation with future systems began in December. In order to improve job control in the operations of the 75, we proceeded to an external sequence number whereby the operations staff provided the sequence number of the job and HASP subsequently recognized this for accounting purposes. The many errors in OS and HASP were resolved throughout this period. A move from Release 11 to Release 13 helped in this respect, and the average number of systems failures went from approximately six per day to one and one-half. During December the standard MVT system, which is the first official system for the 91, became available in a pre-released version. Many new routines to facilitate the systems work were developed or borrowed and instituted in our facility.

D. Other Projects

Development work proceeded in several areas. In particular, in the area of TORTOS, the principal effort was continued work in the area of the DSM text editor which is now at a working stage and is undergoing debugging. The first user documentation was written late in the year. The 1800 project which consisted of a standard IBM link between the Model 75 and the 1800 failed. Essentially, the reason for this was that the channel was extended to the 1800 and as a



USE OF SLAC IBM 360/75

919A3

FIGURE 1

consequence in going nearly 2000 feet, data rates were impossibly slow and 75 maintenance had to be effected all the way to the 1800. An alternate scheme was initiated during this period and consisted of a generalization of van der Lans' scheme for connecting the Hummingbird scanners to the Model 75.

E. User Services

Soon after production began on the Model 75, a need was recognized for a group to act as a liaison between the general user community and the system programming group. This group, User Services, was created and provides for user education as well as organizing the courses which the systems people take. A PL/1 course was given to the user community during this period. In order to facilitate this liaison a log of user complaints and difficulties is kept and a summary is made which enables the systems group to concentrate their efforts where they are most needed. User Services also handles the documentation which includes the Newsletter and the User Handbook. The User Handbook is not yet available and indeed was delayed simply because, as a consequence of dealing with the users, User Services found that a handbook would be relatively useless unless it included a whole spectrum of simple examples.

II. SLAC Computation Group

A. Program Library and Applications

1. Program Library

The major effort of the reporting period has been the program libraries. There are now about forty programs in the SCC Program Library and sixty in the SLAC Program Library. There are still important routines missing from the former library which must be written soon. There will be a slow conversion of the routines to PL/1 as they are requested. New additions to the library include: Fortran error-trace routine; random-number generator; macro instruction and I/O routine for simple line-image output from Assembler Language programs.

The multiple-precision arithmetic package was released for testing; on the basis of results set by four different users, the division and comparison routines were rewritten and the entire package will be submitted to SHARE soon. Research continues in the area of numerical integration with

emphasis on the automatic generation of quadrature rules. New rules derived from Tchebycheff systems of functions are being explored.

A number of matters relating the activities of the SHARE Fortran Project were discussed by correspondence with the Project Manager, as a result of discussions during the August SHARE meeting. There is little hope of further hardware improvements on the IBM System/360, but maybe the next generation will be better.

A paper on logical arithmetic was rewritten and resubmitted to the Communications of the ACM; an algorithm by W. Gantschi was refereed for the Algorithm section of the Communications.

Work continues on the System/360 Assembler Language lecture notes.

2. TRANSPORT

A production version of TRANSPORT was put on the disk as a load module in early July when the 360/75 became available for SLAC use. The linkage editor time is eliminated, and the user needs only a small "call deck" in front of his data in order to run TRANSPORT.

Another version of TRANSPORT was written so that the program can be run in the 100K-byte partition which is allotted to the 2250. The program does not solve the second-order equations, but does allow fitting; it has about 53 overlay segments in three regions. The I/O for the scope is currently being written.

PROFILE was reprogrammed to run from the 2250. This version is most useful because very few people have the instant insight into six-dimensional space needed to know the proper range of input parameters. One can now zero in on this range in a very few minutes by watching the change in the display. The use of this program reduced the elapsed time needed to design one particular spectrometer from several weeks to a few hours.

3. SUMX

An on-line data retrieval project was begun in September, 1967, in collaboration with the IBM Scientific Center, Palo Alto. The objective of this effort is to let physicists create, manipulate and display abstracts of large quantities of elementary particle event data that is produced by the various physics data-reduction codes (e.g., TVGP). The first step is to

produce an on-line version of the existing 'batch' mode program CERN-SUMX. The first version of this program is in demonstrable form at the IBM Scientific Center. It will now be transferred to the SLAC computer. The beam switchyard computer system continues to require some attention. Most of this effort goes into helping the RAD programmers to learn the system.

B. Pattern Recognition

Several modifications have been made to the Hummingbird I film digitizer so that it can be used as a "matrix gray-scale" scanner. This means simply that the digitizing is in the form of a rectangular matrix array of octal digits, each representing the approximate optical density of a small spot on the film. The communication between this scanner and the main computer at SLAC remains essentially identical to that for Hummingbird I; a second threshold-setting command has been added to allow finer gray-scale quantizing even though only eight levels are distinguished in any one scan. As a result the scanner can in principle determine optical density to an accuracy of one part in 32 by performing repeated scans with different upper and lower threshold settings. The scanner can be operated in a "sparse" or "sense" mode, the latter allowing a normal 35 mm film frame to be digitized onto a matrix approximately (1000 x 1000).

Programs are being designed and implemented to allow user programs in both PL/I and FORTRAN IV access to this scanner. In addition a system of PL/I programs is being written so that scanning may be accomplished under control of lightpen interrupts from the 2250 display scope. This will enable a user to "see" the digitizing, adjust scanner command variables, rescan, etc., until satisfied at which point he can output a complete frame onto tape or disk for later processing. It is felt that this truly on-line mode of scanner operation will be extremely useful, especially for picture processing research. When fully implemented the system will be used to test the efficacy of several picture transforms that may be applied to smooth bubble chamber photographs.

The short paper, "Preparing Film for Automatic Gray-Scale Digitization" (GSG 49), suggests certain precautions to be taken when preparing film images for subsequent automatic or partially automatic processing.

C. Graphics and Control

1. Graphics Packages

The basic input-output subroutines for the IBM 2250 scope for use in FORTRAN and PL/I programs have been modified to make more efficient use of a partitioned system. In addition, subroutines for writing to the 2250 buffer are now in the library.

Two IBM-supplied graphic packages are now being tested. CGTM 25 describes a subset of GPAK with examples. The IBM Graphic Subroutine Package for FORTRAN which is now available provides symbolic references to graphic entities and use of multiple 2250's.

2. Graphic Processing and Interactive Program Development and Testing.

Neither convention batch processing nor typewriter-terminal-oriented time-sharing systems have, to date, provided completely satisfactory facilities for highly-interactive program development and testing. With the advent of practical page-size display stations, a third approach is clearly indicated, but no supported software have been available for this purpose. A pilot study has been initiated for a system that could, running a conventional multi-programming environment (e.g., OS/360 MFT), support this function on a variety of suitable terminals including the IBM 2250 and the proposed SLAC graphic interpretation station. A comparison will be made of this pilot system and the specifications and/or performance of similarly-oriented systems being developed at MIT and at IBM as information on these systems becomes available.

3. Graphic Interpretation Facility

Extensive time and effort has gone into considering various approaches of acquiring hardware for the graphic interpretation facility. The effort will lead to the selection of various components of the graphic interpretation facility in the near future. An assembler for a very probable computer selection is being implemented to execute on the 360 to facilitate software development for the GIF. This effort is about 60 per cent complete.

4. 9300 Disk Monitor

Work is continuing on the disk monitor for the 9300, and the monitor will hopefully be in operation sometime in February, 1968. Since the last

report, the loaders,which handle external references and have an overlay capability combinable with interrupts, and the FORTRAN input-output editor have been coded (about 4K more words). Source decks for the FORTRAN compiler and Meta-symbol assembler have been obtained from SDS. The Spectre on-line loading and coreload modification idea has been dropped. Remaining steps are: Sift rest of FORTRAN support library; modify FORTRAN and Metasymbol for inclusion; code disk storage allocator and library handler; code a control card interpreter; debug the system; provide documentation.

5. On-Line Film Measuring Project

Richard Ericksen from the Conventional Data Analysis group joined the project on 1 July. Until 1 September responsibility was shared by the Computation Group and R. Ericksen on the following efforts:

a. On-line measuring table checkout routines.
b. Incorporating two additional measuring tables into the system.
c. Providing modifications to BUCAPS to facilitate the measuring of spark chamber film.
d. General debugging and system improvement.

Since 1 September, R. Ericksen has assumed full responsibility for the system and the Computation Group assumed a consulting service role only.

D. Hardware Developments

1. Hummingbird

The Hummingbird I filmreader is specifically designed for the accurate digitization of the spark and bubble chamber film. On this film the information is binary in nature, i.e., only the presence or absence of dark areas on the otherwise transparent film is detected and their center coordinates sent to the 360/75.

In order to evaluate other scanning algorithms the Hummingbird I has been modified to also operate in a point-by-point scanning mode. This has been done by taking light samples at regular intervals along a scanline. Their intensity is digitized in eight levels, lying between two program specified levels. Up to 1024 points along a scanline may be sampled in this way while up to 1024 scanlines may be scanned. The addressing of

rectangles on the film occurs in the same manner as in the "normal" mode of operation. An accurate simulation of a point scanner has thus been accomplished.

TICK, a high-resolution timing station on the 360/75* has been partially implemented. This station is controlled via the direct control feature of the 75. A static register contains day, month, year, and shift information, while a 32-bit counter, incrementing every 10 μs, holds the time of day. This scaler may be preset or read-out on a byte-by-byte basis with the data on the direct-out and direct-in lines respectively. The register and byte addresses are on the signal bus-out lines while the interrupts generated when the scaler goes through zero and at predetermined intervals (pacer) are transmitted over the signal bus-in lines. Software to access and control this station is now being incorporated into the system.

2. Spiral Reader

The computer control requirements for the spiral reader project have been organized and specified. Both the hardware interface and the software requirements have been established to the point where final designs on the hardware interface are being completed. Fabrication on several units has begun. One of the main requirements in the design of a spiral reader is that all major I/O to and from the spiral reader must be completely controlled by the control program. A control program of such a nature would undoubtedly involve multi-processing several I/O devices concurrently. A preliminary design of such a control program is being made.

The PDP-9 purchased for the use of controlling the spiral reader has very limited peripheral equipment. That is, the system purchased contains the 8K 18-bit word PDP-9 computer, one teletype and keyboard, one paper-tape reader-punch, and one Ampex 9-channel magnetic tape recorder without the controller. In view of the amount of programming and dubugging that have to go into such a system, and the limited peripheral equipment, it was decided that an assembler for the PDP-9 on the IBM 360 was definitely needed.

* GSG 36 - Dickens and van der Lans

For three months an assembler for the PDP-9 on the IBM 360 was written and debugged. This assembler has been thoroughly checked and it is ready for immediate use. This assembler accepts input from cards, produces an assembly listing, and generates a binary output tape. It also produces an alphabetic list of symbol cross references, and recognizes literals. A complete report on this assembler will be available soon.

E. Miscellaneous Projects

1. Formal Logic and Inferential Analysis

(Note: "Inferential analysis" is a term coined by Hao Wang to denote that activity which bears a similar relation to formal logic as does numerical analysis to real analysis. It is preferred to "automatic theorem proving," and "artificial intelligence" because the former is too narrow and the latter much too broad a term. It is not yet in common use, however, even among the specialists.)

Current and recent work (in collaboration with L. Wos of Argonne National Laboratory) is directed toward (1) development of efficient machine-oriented proof algorithms for first-order predicate calculus with equality; (2) formal demonstration that these algorithms have the required semi-decision properties; (3) investigations of axiom systems for certain mathematical domains, both from a formal standpoint and with respect to computational efficiency. Inferential analysis is one of three areas thought to have substantial bearing on the problems of question answering, decision making, and related aspects of artificial intelligence. Sometimes it is termed "logocentric" to contrast it with psychocentric and linguistic approaches.

A paper "The Concept of Demodulation in Theorem Proving" (co-authored with L. Wos, D. Carson, and L. Shalla) appeared in JACM 14 pp. 698-709 (October, 1967). A paper on dependence of equality axioms in group theory has been drafted and is being circulated privately for comment prior to possible submission for publication. Preliminary plans are being made for a new generation of theorem-proving programs on the System/360 as successors to the highly successful PG1-PG5 family on the CDC 3600.

2. Spark Chamber Data Logging

A special system was provided for the PDP-8 tailored to the specific I/O and interrupt handling needs in utilizing the computer for on-line data logging and monitoring functions of Group E's spark chamber experiments.

This system provides the linkage to programs that are provided by the experimenter and provides subroutines to reduce his programming effort for I/O and interrupt handling.

3. Small Computer Acquisition Committee

This committee was active in assisting experimental group G in the evaluation of small computers for a wire chamber application. Primarily, this activity has been on an informal basis and acts in an advisory capacity only.

III. List of Publications and Reports

Graphic Study Group Memos -- 1 July through 31 December, 1967

GSG # 47	B. Russell, Revised 1800 Configuration, August 1967
48	A. Leino and J. Brown, Some Miscellaneous Remarks on Large 2250 Displays, August 1967
49	C. Zahn, Preparing Film for Automatic Grey-Scale Digitization, September 1967
50	J. George, Picture Generation Based on the Picture Calculus, October 1967
51	W.F. Miller and Alan C. Shaw, Search Procedures in the Picture Calculus, October 1967
52	E. Sandewall, Use of Ambiguity Logic in the Picture Description Language, December 1967
53	S. Torok, Notes on an Automatic System for Bubble Chamber Film Processing, December 1967
300	C. Dickens and J. van der Lans, Graphic Interpretation Facility, August, 1967

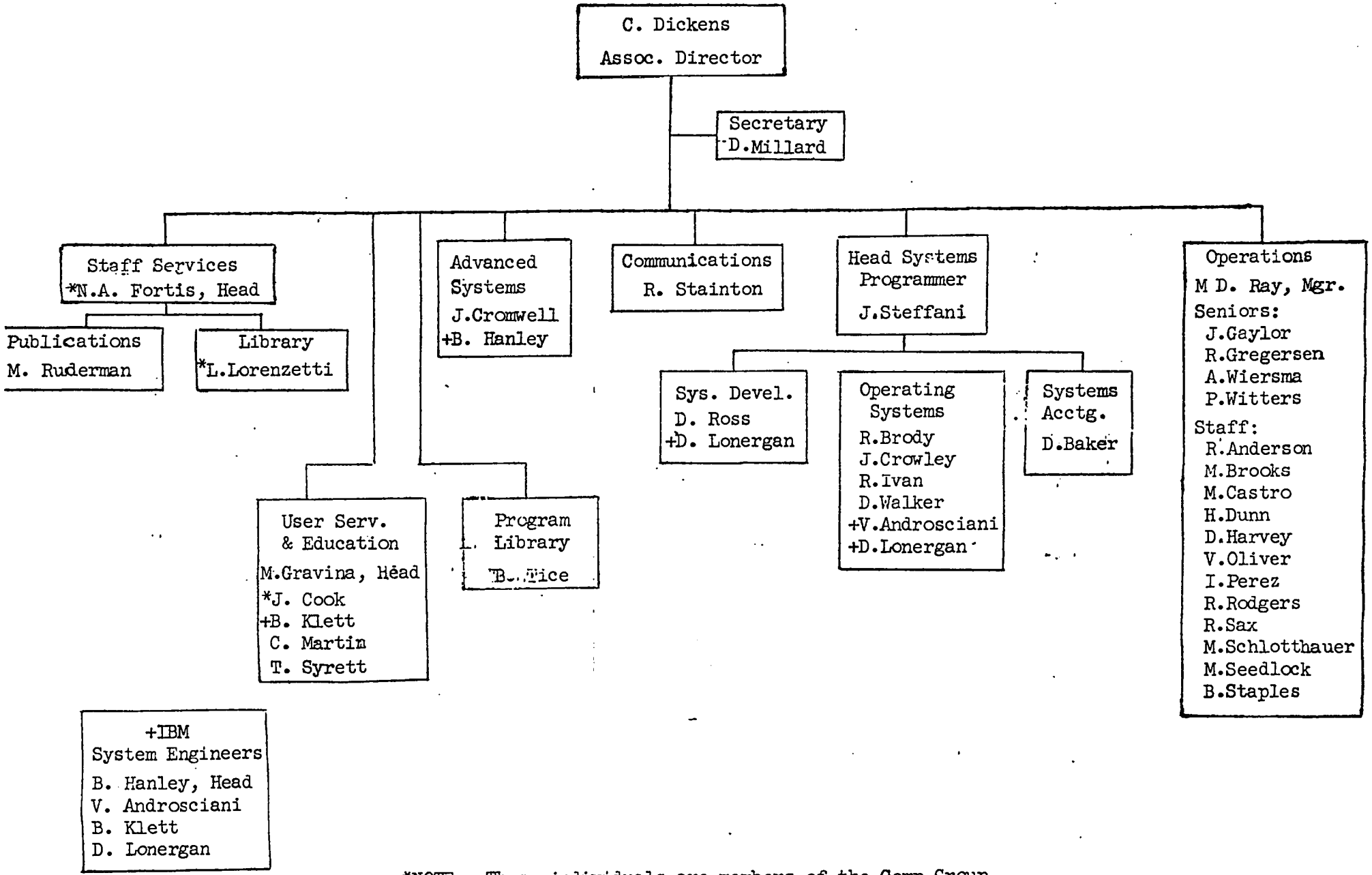
SLAC Publications -- 1 July through 31 December

SLAC-PUB-358	W.F. Miller and Alan C. Shaw, A Picture Calculus, October 1967

Computation Group Technical Memos -- 1 July through 31 December

CGTM # 17	J. Ehrman, System/360 Assembler Language Programming, August
18	J. Ehrman, Multiple-Precision Floating Point Arithmetic Package for System/360, August
19	J. Welsch, A Study of Arithmetic Type Conversion on the IBM System/360, July
20	J. Welsch, A Study of the Round Operation on the IBM System/360, August
21	J. Welsch, Study of Swap Instructions for the IBM System/360, July
22	H.J. Mortell, A Comparison of Code Densities on the 7090 and S/360 Computers, August
23	Ira Pohl, Translation from B5500 Extended Algol to OS/360 Algol, September
24	G.S. Watson, Goodness-of-Fit of Curves, August
25	M.A. Fisherkeller, A Subset of GPAK, a Graphics Package for the 2250 , October
26	V. Lesser, VRL-1 Computer, October
27	E. Mueller, PDP-8 I/O and Interrupt Programming System for Spark Chamber Data Logging Programs, October
28	C. Fridh, An Attempt to Put Parts of Roberts' "Machine Perception of 3-Dimensional Solids" into framework of Picture Calculus , October
29	R. Braden, W.F. Miller, The SLAC Central Computer, Oct.
30	J. Ehrman, Algorithms for Performing Multiplication and Division with "Logical" Operands , November
31	Yves Noyelle, Implementation on PDP-1 of Subset of Picture Calculus, November
32	W.E. Riddle, Syntax of an Interactive Language, Nov.
33	J. George, The SPIRES Scope Demonstration System, Nov.
34	J. George, SARPSIS: Syntax Analyzer, Recognizer, Parser and Semantic Interpretation System, Nov.
101	D. Gries, Compiler Implementation System, November
102	D. Gries, Compiler Implementation System, November

SLAC Facility, Stanford Computation Center



*NOTE: These individuals are members of the Comp Group

30 December 1967

30 December 1967

C O M P U T A T I O N G R O U P

Group Leader: W.F. Miller
Staff: G.A. Robinson, Jr.
Staff: Robert T. Braden
Executive Assistant: N. Fortis
Secretary: C. West
Secretary: K. Maddern
Group Librarian: L. Lorenzetti

ENGINEERING

Head: J. van der Lans

LIBRARY, APPLICATIONS AND USER SERVICES

Head: J. Ehrman

J. Cook
D. Gries
S. Howry
B. Kear
F. Rothacker

GRAPHICS AND CONTROL

Head: Mary Anne Fisherkeller

R. Beach
A. Gromme

PATTERN RECOGNITION AND PICTURE PROCESSING

Head: W.F. Miller (acting)

M. Hu
E. Mueller
C. Zahn

VISITORS, STUDENT RESEARCH ASSISTANTS, AND PART TIME

D. Adams
I. Carlbom
C. Crane
J. George
J. Halperin
F. Hansen
L. Hoffman
J. Levy
L. Morris
I. Pohl
J. Ryder
R. Russell
E. Satterthwaite
A. Shaw
L. Smith
J. Welsch
B. Weyers
W. Wilner

SYSTEMS ANALYSIS AND DEVELOPMENT

Head: Robert T. Braden

COMPUTING TECHNOLOGY CENTER
UNION CARBIDE CORPORATION - NUCLEAR DIVISION
OAK RIDGE, TENNESSEE

Summary of Recent Computing Activities for the
AEC Computer Information Meeting
Los Alamos Scientific Laboratory
Los Alamos, New Mexico, May 20-21, 1968

The Computing Technology Center is currently operating as principle computer systems two IBM 7090's and an elaborate intercoupled IBM 360/65/50. It is worthy of note that although the 360/65 was only installed late in November and started into production use in an intercoupled manner since the first of the year, it is already averaging only 15% idle time based on 24 hour per day, 7 day per week availability. At the same time, the increased computing capacity at CTC and ORNL on the 360's has not resulted in any significant reduction in 7090 load, which is currently leaving only 12% idle time.

During the preceding six-month period the intercoupled system was installed and achieved production status, currently averaging over 400 jobs per day. Software system support has been developed so the prime-shift work is now processed with remote demand inquiry from IBM 1050's and 2740's, batch background, and I/O spooling proceeding concurrently. Development work is being continued on IBM 2260 stations. Also during the period an IBM 360/20 was installed at the Division of Technical Information Extension, and programs to allow batch remote job entry from this equipment into the intercoupled system at CTC are currently undergoing final checkout. A diagram of the present CTC configuration is supplied in the attachment.

The five manual remote terminals identified in the last semi-annual report are all in operation, the only change being that the one at A.E.C.-D.T.I.E.

has been temporarily moved to the office of the President of the Nuclear Division of Union Carbide Corp. for an evaluation as a management tool. An additional five terminals are now on order for installation over a several month period starting the end of June.

In the area of peripheral support to the IBM 7090's, our leased RCA 70/25 system which was under study for possible expansion appeared unable to fully meet the expanded requirements. Impending availability of a surplus IBM 360/30 has resulted in a recommendation to release the 70/25 and acquire the necessary support equipment to go with the 360/30. This will enable immediate release of the oldest of the three 1401's, and eventual enlargement of the 30 configuration would allow release of the next oldest 1401. Studies are currently underway for replacement for the substantial EAM equipment at CTC and is expected to result in a request for approval of perhaps two modest 360/20's or their equivalent.

The most persistent hardware troubles encountered during the last six months have been intermittent power kick-offs on the 360/50 and general complaints on the data cells. At the time this report was prepared IBM was working diligently on both these problems. The 7090's continue to perform well, with the principal source of trouble being mag tape troubles as usual. Increasing age is the underlying cause of the 1401 and EAM equipment complaints.

Planning has been initiated on the next expansion step at CTC in large computing equipment. At the present time a major obstacle is the absence of any further machine room space in the existing building. Attempts to obtain funds for a major building expansion have so far proven fruitless.

DEPARTMENT OF ENGINEERING

UNIVERSITY OF CALIFORNIA
LOS ANGELES

PROGRESS REPORT

for

Atomic Energy Commission
Computer Information Meeting
Los Alamos, New Mexico

May 20, 21, 1968

GE/ck
April 26, 1968

MODELS OF COMPUTATIONS AND SYSTEMS

D. Bovet has completed design of an a priori memory allocation algorithm which uses as input information an acyclic graph corresponding to a program. The algorithm, designed for multiprocessor systems, seeks to obtain an efficient packing of Space-Taking Entities (program segments) by automatically recognizing those STE's whose periods of activity are non-overlapping and those which are mutually exclusive. In FORTRAN-like memory allocators such recognition is left to programmers to carry out explicitly. A second feature of the proposed allocator is the a priori detection of main memory overflows with the consequent application of a transferring algorithm. The transferring algorithm inserts transfer instructions into the program seeking to remove the overflow condition by temporarily transferring to secondary memory STE's whose period of main memory activity does not overlap with the period of time during which they must reside in secondary memory. Input information required for execution of the memory allocation algorithm is: an acyclic graph; a given assignment and sequencing of a computational graph on the P processors of a computer system and; the capacity of the main memory. Output information produced by the algorithm consists of: a base address in main memory for each STE associated with the graph and; automatically inserted transfer instructions in those cases where memory overflow can be detected and corrected. The memory allocation algorithm has been programmed in FORTRAN IV and is in the final phase of debugging on our Sigma 7 computer. A simple example graph has been used to test the program. Evaluation of the algorithm will be made as soon as inputs are derivable from E. Russell's FORTRAN analysis program.

J. Baer has continued work on the a priori model of computations. Two new algorithms calculating the mean path length associated with allocated directed acyclic bilogic (d.a.b.) graphs have been implemented. Studies of mean path length measures for array processors (e.g., Illiac IV and pipeline) have been done and it has been observed that the model should have constraints

introduced at an early stage to reflect strictly global control of elementary processors. A new single pass a priori scheduling algorithm has been programmed and run with a number of different urgency criteria (FIFO, local, structural and global) on sample graphs. Algorithms to determine lower and upper bounds on the number of processors to obtain maximum parallelism have been devised. At the same time techniques based upon actual measurement of program parameters have been developed to find bottlenecks, if any, preventing effective use of parallel processing. All programs can run on the Sigma 7 and 360 systems.

E. Russell has continued work on analysis of FORTRAN programs and transformation of sequential graphs to other parallel equivalents. The transformation program is written in PL/I and accepts input from a Meta5 syntax analyzer. The Meta5 program to analyze syntax has been made operational on both the /360 and Sigma 7 in a form which reduced compilation time by a factor of almost 4. A preprocessor for the /360 version has been written to make FORTRAN programs format-free and thus amenable to Meta5 stream-oriented input.

L. Kleinrock has continued to obtain analytic results from models of time-shared processors as reported in the several listed papers.

DESIGN AUTOMATION

This period has seen extensive development of programming tools for our design automation system, definition of library structure, implementation of new algorithms for transforming Boolean equations into a form suitable for integrated circuit inventory and implementation of algorithms for assignment of modules under packaging constraints.

R. Mandell developed a directed graph model of computer configurations which is particularly susceptible to manipulation by the meta compiler programming language. J. Kunz implemented Mandell's specification of a translator of a flexible format language permitting full use of PL/I in the design automation system. Mandell also carried forward his development allowing interspersal of PL/I and Meta5 statements. R. Mandell and R. Gardner completed, debugged and refined extensions to Meta5 permitting the language to manipulate a graph structure. This covered most of the new Meta5 verbs described in Internal

Memorandum #56 plus new verbs. In particular .SUCCESSOR and .PREDECESSOR allow the programmer easy access to lists of successor and predecessor nodes attached to every node of a given graph. Reports by Mandell and Gardner are in final phases of preparation. Discussion about programming languages for design automation are also included in the recently published Proceedings of the 1967 Share-ACM Design Automation Workshop.

M. Marin has completed a work on algorithms for synthesis of logical nets using a fixed inventory of integrated circuit modules. A FORTRAN IV program for single output circuits works in three modes: computation of total number of existing solutions of a given system of Boolean equations (presently 9 variables) and generation of the truth tables; computation of solutions of a system of Boolean equations belonging to a given inventory of functions; computations of those solutions in the second mode with lowest weight factor (according to a given criterion). An algorithm has also been developed for multiple output circuits obtaining maximum sharing of modules at each logical level. A paper will be presented at the Fifth Annual Design Automation Workshop July 15-18, 1968. Further, a simulator of the Svoboda Boolean Analyzer was developed with capacity for computing solutions for an expression containing 100 Boolean terms of up to 22 variables. This is the capability set as a goal for the hardware system Boolean Analyzer.

K. Gostelow completed development of programs for component placement. This subsystem accepts coded representations of a logic diagram, hardware specification of circuits to be used, circuit cards, panels, gates and other packaging levels. Execution of the program produces a table giving assignment of each logic element to an appropriate unit of packaging level. The programs have up to now been tested only on elementary structures.

H. Potash is continuing his work on algorithms for automatic generation of the control states of a computer derived from a program description.

M. Melkanoff and M. Holly developed an on-line computer system simulation program on the 360/75 using a 2250 graphic console. M. Hadjioannu is now seeking to modify this system and increase performance.

SYSTEM EXPERIMENTATION

Our program on the instrumentation of computer systems (ARPA SP-184) has had a distinct influence on variable structure system studies aside from the value of measurements and the existence of the SDS SIGMA 7 to act as the fixed general purpose part of the system.

Basic hardware permitting first measurements is due for completion during the summer of 1968. The instrumentation computer sensory system is designed to achieve a maximum transfer rate of from 12 to 16 megawords per second even though such information rates can only be accepted by the measuring system in very short bursts. Variable structure hardware is usually connected to the Sigma 7 through its memory ports. B. Bussell has noted that the sensory system processors may themselves have a variable structure form. In that case any object computer may be treated as the "fixed" general purpose computer and the instrumentation system may act at the "variable" structure system under program control. File transmission system and message transmission system links between the object and instrumentation systems provide all the control and communication needed to satisy our model of variable structure systems. The existence of the Sigma 7 as the general purpose computer is our system and Bussell's approach leave us much more independent of operational changes in other facilities.

Hardware packaging to permit assembly of integrated circuits on standard size boards has been completed. One special circuit board has been designed and fabricated to serve as an interface between systems. It can be customized to match the impedance of other lines between 30 ohms and 160 ohms. It can have its voltage level and swing adjusted. It can have the rise and fall times adjusted when interfacing from high to low speed systems. The general board consists of a carrier board holding a 4 x 6 array of 14 pin-in-line packages. A removeable two-sided signal harness is used to interconnect the integrated circuits. A card cage has back-panel pins on a standard grid for automatic wirewrapping. A special transmission line tape cable permits interconnection between card cage assemblies or between our system and external systems. Signal harnesses and prototype boards are fabricated in our etched circuit facility. Production lots are sent to outside vendors.

Boards for our Graphics Station are in fabrication. M. Wingfield projects connection to the Sigma 7 this summer. Following that event our RAND Tablet and Lincoln Wand are to be tied in through the Graphics Station control along with our flexible double keyboard.

A. Avizienis and F. C. Tung have completed an investigation of complex combinational arithmetic nets using an arithmetic building unit employing signed-digit arithmetic. Combinational arithmetic nets for evaluation of a number of different mathematical functions were studied. A technical report is in process.

INSTRUMENTATION

The ARPA supported project on instrumentation of computer systmes is scheduled to show first measurements by the end of the summer, 1968. This implies completion of the basic sensory system hardware and completion of a functional Sigma 7 instrumentation software system. A progress report as of March 1968 is attached as Appendix B.

GENERAL

A list of pertinent publications since the November 1967 Information Meeting is attached along with a list of MS and Ph.D. degrees completed. G. Estrin and A. Svoboda received the honor of being made Fellows of IEEE as of January 1, 1968.

INTERNAL MEMORANDA

Cerf, V. "Instrumentation Software Design--Progress Report", Internal Memorandum # 67, March, 1968.

Gostelow, K. "The Component Division and Placement Module of the Design Automation System", Internal Memorandum # 65, April, 1968.

Marin, M. "A Fortran IV Program for Solving Boolean Equations", Internal Memorandum # 64, Dec., 1967.

Soha, Z. "Specification for Preparation of a Numerical Tape Control for SLO-SYN Driller", Internal Memorandum # 68, March, 1968.

Svoboda, A. "Decimal Adder With Signed Digit Arithmetic", Internal Memorandum # 69, April, 1968.

Wingfield, M. "Specifications for the Graphics Station-Graphics Input Interface", Internal Memorandum # 66, March, 1968.

PUBLICATIONS

Avizienis, A. "An Experimental Self-Repairing Computer," to appear in the Proceedings of the International Federation for Information Processing Congress '68, Edinburgh, Scotland, August 1968.

Avizienis, A. "Application of Concurrent Diagnosis and Replacement in a Self-Repairing Computer," (D.A. Rennels and J.A. Rohr), Digest of the IEEE International Convention, New York, N.Y., March 18-21, 1968.

Avizienis, A. "Design of Fault-Tolerant Computers," AFIPS Conference Proceedings, Vol, 31, Fall Joint Computer Conference 1967, pp. 733-743.

Avizienis, A. "Self-Repairing Computers," Proceedings of the IEEE, to appear in late 1968 (an invited paper).

Baer, J. and Bovet, D. "Compilation of Arithmetic Expressions for Parallel Computation," to appear in the Proceedings of the International Federation for Information Processing Congress '68, Edinburgh, Scotland, August 1968.

Kleinrock, L. "Applications of the Mathematical Theory of Sequential Sampling to Gamma Scanning in Nuclear Medicine," (with A.R. Stubberud, S. Friedland, H. Katzenstein), Journal of Mathematical Biosciences, accepted for publication.

Kleinrock, L. "Certain Analytic Results for Time-Shared Processors," to appear in the Proceedings of the International Federation for Information Processing Congress '68, Edinburgh, Scotland, August 1968.

Kleinrock, L. "Computer Scheduling Methods and their Countermeasures," (with E.G. Coffman) Proceedings of the Spring Joint Computer Conference, to be held April 30-May 2, 1968, Atlantic City, N.J.

PUBLICATIONS (Continued)

Kleinrock, L. "Distribution of Attained Service in Time-Shared Systems," Journal of Computers and Systems Science, Vol. 3, pp. 287-298, October, 1967.

Kleinrock, L. "Some Recent Results for Time-Shared Processors," Proceedings of the International Conference on System Sciences, University of Hawaii, January 29-31, 1968.

Kleinrock, L. "Some Results on the Design of Communication Nets," Proceedings of the 1968 IEEE International Conference on Communications, to be held June 12-14, 1968.

Kleinrock, L. "Time-Sharing Systems--Analytical Mehtods," Proceedings of the Symposium on Critical Factors in Data Management/1968, UCLA March 20-22, 1968.

Leondes, C. "Artificial Intelligence in Control," (with J.M. Mendel) to appear in Survey of Cybernetics, 1968.

Leondes, C. "Computational Results for Extensions in Quasilinearization Techniques for Optimal Control,"(with L.G. Paine), to appear in Journal of Optimization Theory and Applications, Vol. 2, No. 6, Nov. 1968.

Leondes, C. "Extensions in Quasilinearization Techniques for Optimal Control," (with L.C. Paine), Journal of Optimization Theory and Applications, Vol. 2, No. 5, Sept. 1968.

Mandell, R. "Is There a "Best" Programming Language for Design Automation?," Proceedings SHARE-ACM Design Automation Workshop, 1967.

Martin, D. "Boolean Matrix for the Detection of Simple Precedence Grammars," to appear in Communications of the ACM.

Svoboda, A. "Boolean Analyzer," to appear in the Proceedings of the International Federation for Information Processing Congress '68, Edinburgh, Scotland, August 1968.

M.S. DEGREES

Brinsley, James R. "Design for Stochastic Resource Constrained Activity Networks, March 1968.

Cheng, Yu Ping "Design of the Hardware and Software Interface Between an IBM Selectric Typewriter and SDS Sigma 7 Computer," Dec. 1967.

Fletcher, Alvin J. "A Neighboring Optimal Controller," March 1968.

Garton, Wendell P. "Selection of Experiments for a Manned Space Station: A Resource Allocation Problem," March 1968.

Gostelow, Kim P. "A System for the Computer Aided Selection of Electronic Modules," March 1968.

M.S. DEGREES (Continued)

Gran, Paul "Design of a Storage Tube Display Processor," March 1968.

Gunn, Michael J. "Sensitivity Analysis of the Ballistic Missile Strategies," March 1968.

Li, Mu T. "Optimizaiton of Discrete-time Stochastic Control System With Constrained Observation and Control Schemes," Dec. 1967.

Maughmer, Robert W. "An Investigation of Analysis Methods for Frequency-Stability of Non-linear Oscillators," Dec. 1967.

Rice, Robert F. "Data Compression/Capsule Mass Spectrometer," March 1968.

Rogers, Michael C. "N-Dimensional Graphical Optimization Under Constraints," March 1968.

Shapiro, Allen L. "A Concept for Real Time PCM Computer Processing," March 1968.

Taylor, James R. "On the Signal Processing Eclipsing, Ranging, Clutter Spectra, and Detection Probability of Airborne Pulse Doppler Radars," Dec. 1967.

Ph.D. DEGREES

Bovet, D. "Memory Allocation in Computer Systems," June 1968.

Holly, Michael A. "An On-Line Graphical System of Digital Design," March 1968.

Marin, M. "Investigation of the Field of Problems for the Boolean Analyzer," June, 1968.

Tung, C. "A Combinational Arithmetic Function Generation System," June 1968.

Turnblade, Richard C. "The Method of Steering Parameters--An Application of the Theory of Bounded Functions to the Synthesis of Non-linear Control," Dec. 1967.

Appendix A

INTERNAL MEMORANDUM #62

Program Deck Set-up for Using the UCLA META5 System, /360 Implementation

by

R. I. Gardner

Digital Technology Research
Department of Engineering
University of California
October, 1967

The UCLA version of the META5 system is available to all users of the Campus Computing Network /360 and other groups with copies of this META5 system. The following pages describe the data sets in which the META5 system resides and show sample card deck set-ups for typical META5 jobs.

A. The META5 System

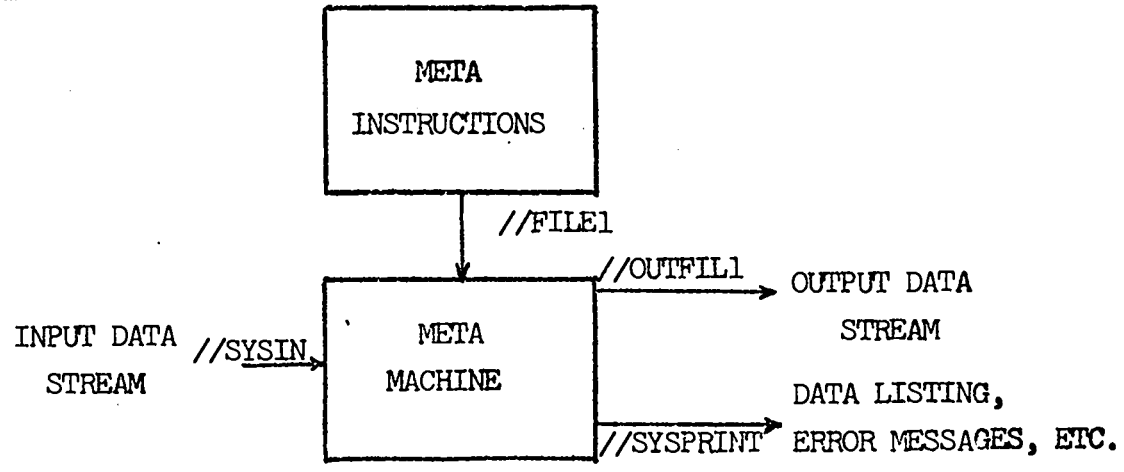


Fig. 1

There are two distinct components of the META5 system, the META MACHINE and the META INSTRUCTIONS, or interpretive code (not to be confused with META5 source statements). META5 is an interpretive system, where the meta machine interpretively executes the program of meta instructions. Section B below describes how to produce a program of meta instructions using the META5 system as a compiler.

The program comprising the meta machine is an executable load module in a partitioned data set with DS name MER007.META5.USER(GØ), which is catalogued. The DD name of the meta machine's input data set (if any) is SYSIN. The DD name of the meta machine's output data file is ØUTFIL1. The DD name of the file containing the program of meta instructions is FILE1. The DD name of the file producing error messages, etc. is SYSPRINT. Refer to Fig. 1.

A sample deck set-up for using the META5 system to read unput data, operate on this data as directed by a program of meta instructions, and to output data is shown below:

```
1.  //JØBNAME JØB
2.  //JØBLIB DD DSNAME=MER007.META5.USER,DISP=(ØLD,PASS)
3.  //STEP1 EXEC PGM=GØ
4.  //SYSPRINT DD SYSØUT=A
5.  //FILE1 DD ... ( The Data Set Containing the Program of Meta Instructions)
6.  //SYSIN DD ... (Your Input Data Set)
7.  //OUTFIL1 DD ... (Your Output Data Set)
```

The JCL statement on line 2 concatenates the private library containing the meta machine to the system library. The execute statement at line 3 passes control to the meta machine. The three JCL statements at lines 5, 6, and 7 define the data sets for the program of meta instructions, the input data set, and the output data set, respectively.

B. Using the META5 System as a Compiler

The META5 system can be used as a compiler to produce programs of meta instructions (interpretive code). These programs of meta instructions can be interpretively executed by the meta machine in a later step.

There is an established program of meta instructions available which enables the META5 system to operate as a compiler for the META5 LANGUAGE[1]. This program of meta instructions is retained as a sequential data set with DS name MER007.CØDE.USER, which is catalogued.

---

1. The specifications of the META5 language are contained in: SDC Tech. Memo TM-2396/000/02, Systems Development Corporation, 2500 Colorado Ave, Santa Monica Calif. A memo describing the features unique to the UCLA version of META5 may be obtained by writing Belle Mosst, 3732 Boelter Hall, UCLA.

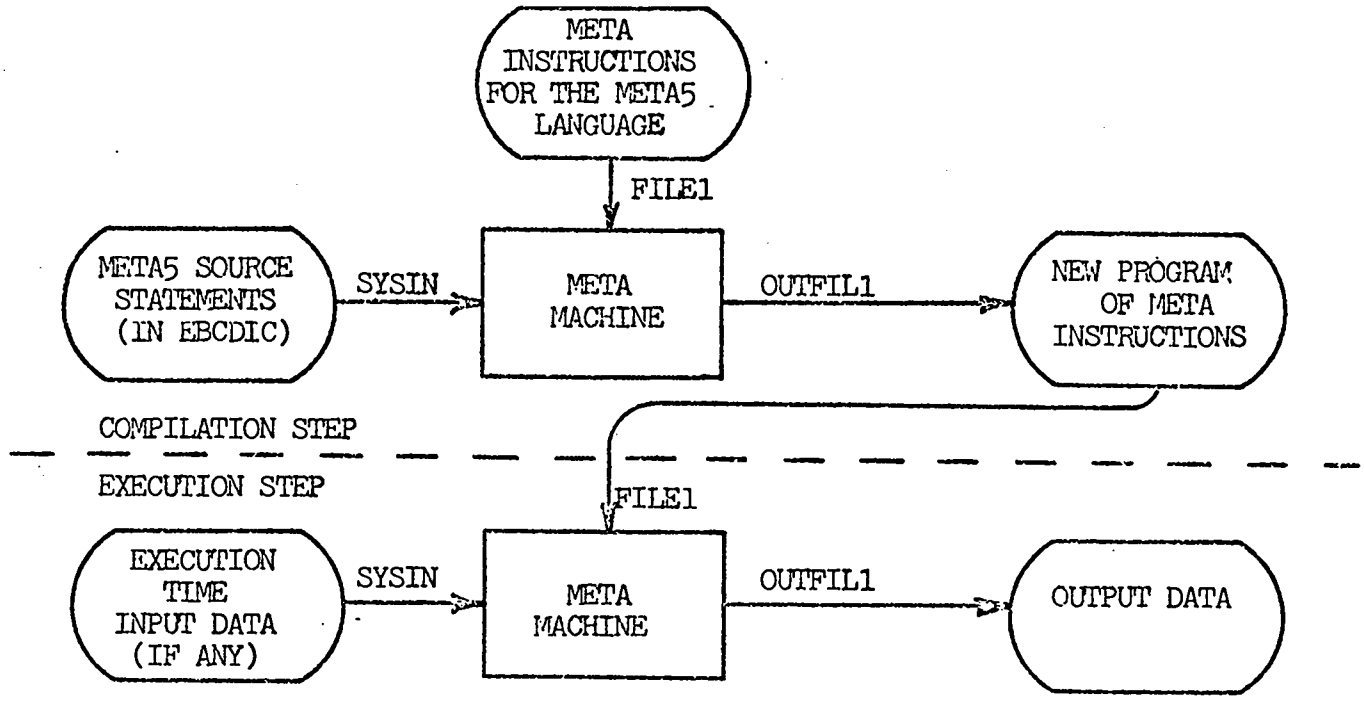


Fig. 2

A sample deck set-up for using the META5 system to first compile a META5 source program into a new program of meta instructions and then to use this new program is shown below:

```
1.  //JØBNAME JØB
2.  //JØBLIB DD DSNAME=MER007.META5.USER,DISP=(ØLD,PASS)
3.  //STEP1 EXEC PGM=GØ
4.  //SYSPRINT DD SYSØUT=A
5.  //FILE1 DD DSNAME=MER007.CØDE.USER,DISP=ØLD
6.  //ØUTFIL1 DD UNIT=SYSDA,DISP=(NEW,PASS),
                DCB=(RECFM=FB,LRECL=80,BLKSIZE=3600),
                SPACE=(TRK,(5,1))
7.  //SYSIN DD *
```

META5 SOURCE STATEMENTS:

```
.META5...
   .
   .
   .
.END
```

```
8.  /*
9.  //STEP2 EXEC PGM=GØ
10. //SYSPRINT DD SYSØUT=A
11. //FILE1 DD DSNAME=*.STEP1.ØUTFIL1,DISP=ØLD
12. //SYSIN (The Execution Time Input Data Set)
13. //ØUTFIL1 (The Execution Time Output Data Set)
```

The program flow shown in Fig. 2 corresponds to the above deck set-up. The first step of the job (at line 3) performs the compilation of the META5 source program defined by data set SYSIN at line 7, producing a new program of meta instructions (intermediate code) on the data set ØUTFIL1, defined at line 6. This new program of meta instructions is used in the second step: line 11 supplies the output of step 1 to the program of meta instructions used in step 2. It should be noted that the sequence of steps for using the meta machine is not limited to just compiling a source program and then running it. For example, there can be three steps: 1. Write a new compiler in the META5 LANGUAGE and pass it through the META5 compiler, producing a program of meta instructions representing the new compiler; 2. Using this new program of meta instructions and the meta machine compile a program written in the source language of this new compiler into some executable code; 3. Execute this code, producing the desired results. There is no limit to the sequence of steps that can be run in this manner. Compiling a compiler as in step 1 above is generally called meta compiling.

C. Utility Programs in the META5 System

1. BCD to EBCDIC Translator

A program that will translate data coded in BCD to EBCDIC is available as a component of the META5 system. This program is most often used to translate META5 source programs and data written in BCD to EBCDIC, the code accepted by the meta machine.

The translator is an executable load module residing in a partitioned data set with DSNAME=MER007.TRANSL(TRANSL), which is catalogued. The following sample deck set-up shows how this program is used to translate a META5 source program prior to a compilation:

```
1.  //JØBNAME JØB
2.  //JØBLIB DD DSNAME=MER007.META5.USER,DISP=(ØLD,PASS)
3.  //        DD DSNAME=MER007.TRANSL,DISP=ØLD
4.  //STEP1 EXEC PGM=TRANSL
5.  //SYSPRINT DD SYSØUT=A
6.  //SYSØUT DD UNIT=SYSDA,DISP=(NEW,PASS),
    //          DCB=(RECFM=F,LRECL=80,BLKSIZE=80),
    //          SPACE=(80,(150,50))
7.  //SYSIN DD *
                .META5
                   .      META5 Source Program Coded in BCD
                   .
                   .
                .END
8.  /*
9.  //STEP2 EXEC PGM=GØ
10. //SYSPRINT DD SYSØUT=A
11. //FILE1 DD DSNAME=MER007.CØDE.USER,DISP=ØLD
12. //SYSIN DD DSNAME=*.STEP1.SYSOUT,DISP=OLD
13. //ØUTFIL1 DD UNIT=SYSDA, . . .
.
.
.
```

The JCL statement at line 3 concatenates the data set containing the translator to the system library. Step 1 at line 4 passes control to the translator. The compilation step, executed at line 9, gets its input data from the output data set of the translator step, as shown at line 12.

2. Intermediate Code Dumper

Most users will not be directly concerned with the program of meta instructions, but will assume that the meta compiler does produce it and that the meta machine can interpret it successfully. However, when it is suspected that the meta machine is operating erroneously, a listing of the intermediate code can be used to trace through the program. There is a program in the UCLA META5 system that reformats a program of meta instructions, producing a

readable listing. This reformatting program is an executable load module residing in a partitioned data set with DSNAME=MER007.ASMBLR(ASMBLR), which is catalogued. The following sample deck set-up shows the JCL necessary to use this program:

1. //JØBNAME JØB
2. //JØBLIB DD DSNAME=MER007.ASMBLR,DISP=ØLD
3. //STEP1 EXEC PGM=ASMBLR
4. //SYSPRINT DD SYSØUT=A
5. //FILE1 DD (The data set containing the program of meta instructions to be listed)

The DD name of this program's input file is FILE1. The DD name of the output file on which the listing is produced is SYSØUT.

Appendix B

INTERNAL MEMORANDUM #67

Instrumentation Software Design

Progress Report

by

Vint Cerf

Digital Technology Research Group
Department of Engineering, UCLA
March 4, 1968

Instrumentation Software Design

Progress Report

Section 1

Our main goal at the present time is to isolate and identify the various software functions that will have to be performed both during instrumentation and during the analysis of acquired data. Our main concern is to identify those functions necessary to acquire data. It is clear that this specification must take precedence over the analysis of data because if we haven't acquired any data we can't very well analyze it. In the material that follows the following terminology applies: Instrumentation System Software means both software to acquire data and software to analyze data. The portion of the software that acquires data will be called Data Acquisition Software. The portion of the software that analyzes the data will be called Data Analysis Software.

Section 2

The Data Acquisition Software package breaks naturally into number of logical functions. When one considers the problem of gathering data from a computer system one naturally asks a series of questions; for example: What data do I want to acquire? In what format do I want to acquire it? What will I do with the data once I've got it? Will I write it out to tape? Will I keep it in memory? Will I require any counters? Will I be doing timing experiments?

Which portions of the object computer software will I want to look at? What portion of the data that is available do I really want to acquire? What kind of filtering of the data is necessary? Such questions will help us design the data acqusition software because they force the development of logical modules which define and filter source data. In our proposal to ARPA dated September, 1967, Appendix A describes a portion of the support software for data acquisition. Although the title of Appendix A is Sigma-7 Instrumentation System Monitor, the material described in Appendix A really only covers the service substrate for the monitor. The real functions for constructing and monitoring an experiment were not described but, in what follows, I hope to give a better specification of the full monitor system.

Section 3

The data acquisition system must perform the following tasks:

A. It must support or facilitate the description of an experiment by a user. The software associated with this support will be called Experiment Definition Software.

B. It must obtain the appropriate interrupt handler programs to satisfy all possible interrupts that might occur during the experiment. Because of the wide variety of possible interrupt conditions, all interrupt software will not be entirely present in core during the experiment. Rather, at experiment definition time, just enough interrupt software will be obtained from some library to support the interrupts that might occur up to the time that the experiment has actually begun. At that time additional interrupt handling packages will be brought

in from the library to support interrupts that might occur during actual instrumentation. Essentially we seek to minimize core residence wherever possible but to maximize rapid response to important interrupts as they occur.

C. While an experiment is actually running, that is, while we are actively gathering data from the object computer system, we will make use of the graphic station as an on-line display system to monitor the activities of our experiment. The idea here is not to present elaborate displays or elaborate analyses of the data acquired, but rather to keep us informed of the status of the experiment. The bulk of the Instrumentation Computer's time will probably be spent in the answering of interrupts and in the transmission of data to various secondary devices. Hence, we can't very well spend a large amount of time supporting display software for the Deck 340. The complexity of our displays depends entirely on the amount of time available to the Sigma-7 for producing them. The Instrumentation System divide its time between background and a foreground. The background will provide such services as monitoring displays and massaging or manipulating of acquired data preparatory to putting that data out on a secondary device. The foreground obviously will be activated by interrupts that occur as the experiment proceeds. The implication here is that the system will have a time-shared background. This clearly places certain demands on I/O scheduling and the allocation of core storage; these problems will be discussed in detail later.

D. Finally, the data acquisition software must support sufficient I/O functions to preserve acquired data on some secondary memory device. We have available two tape drives and one RAD on which to save such data. At present our main thought is to preserve such acquired data on a continuous tape. It is possible however, that the design of an experiment will require that we use the disc rather than tape for the storage of data, the reason being that the disc is randomly accessible. We also note that the disc is faster than tape. Considering that the tape drive has a data rate of 60 KB it is conceivable that data will come to us rapidly enough that we will buffer the data onto the disc and then from disc to tape. Phase 0 and Phase 1 experiments do not appear to require this kind of buffering; of course, it is always possible to design an experiment in such a way that these data rates will not be encountered.

E. Essentially the previous four functions comprise all of the requirements for the data acquisition system, namely the support of an experiment definition, the selection of the appropriate interrupt handlers, the preparation for experimentation monitoring by the graphic station and finally preparation for preservation of acquired data on secondary memory devices.

Section 4

There are essentially two sources of data available to us, the first and most obvious is the actual hardware of the object computer system. The second is

the program being executed by the hardware. Ultimately, data from both of these sources must come through the Instrumentation Computer Sensory System. Setting up these sources of information involves modification of Object Computer hardware and software. In the first case, we must decide where we want to put hardware probes on the object computer system. To do this successfully we must answer two questions, (a) Will this hardware provide us with interesting and important information and,(b) Will the attachment of such probes cause any distortion or damage to the object computer? The same sort of questions have to be answered when considering software sources of information. We cannot arbitrarily modify the Object Computer's software system in order to gather data. A poorly placed module of artifact may cause the operating system to fail completely or to produce data so greatly distorted from its ordinary appearance that the data we gather will be insignificant or at least will not apply to a true analysis of Object Computer activities. We can consider then that the selection of hardware sources and software sources of data are the first steps in the definition of an experiment. Having made such decisions our source of information then becomes the Instrumentation Computer Sensory System. Virtually all data coming from the Object Computer System passes through the sensory system. The sensory system has the capability for data selection and translation, or manipulation. A major task of experiment definition becomes the description of those data sources which should be allowed entry into the sensory system. Once such data has been recognized and acquired by the sensory system, it is passed along to the processor nodes. Virtually all data that passes through the data receiving units is made available to all the processing nodes. It becomes necessary then to identify which sources of data are to be operated on by a particular processing node. Each source of information from the

object computer system has associated with it an 8 bit source information byte. This facilitates the identification of those sources of data that are associated with a given processor node. Each processor node has an 8 bit mask with which it can determine whether a source of data is of interest to it or not. Clearly one of the functions of the experiment definition will be to set-up the 8 bit mask in each of the processor nodes in order to describe or identify those data sources associated with that processor node. The connection of certain sources of data to a processor node insure us that data will be processed as we desire. However we still have the problem of determining where the data should go once it has be manipulated by the processor node. If the processor node requires access to the Sigma-7 memory it will have a base register telling it where such access is to be made. Part of the job of the experiment definition package will be to allocate storage required by each of the processor nodes. Appendix A of our September 1967 proposal includes a very brief description of a module called the core storage monitor. This service package will be made available to the experiment definition processor so that storage when required can be allocated. At present it is assumed that such allocation will be on an a priori basis, that is, storage will not be allocated dynamically during the course of an experiment. Of course an experiment may have several phases, and it is possible to allocate storage in between phases as required, but such allocation in general will not occur while data is being acquired. Because data analysis cannot occur during experimentation, most of the data acquired will be saved on the secondary devices. At first glance this seems to be a fairly simple process, data which is written into memory by the processor nodes in the sensory system will be simply written out to tape. However, a great many decisions are

involved in deciding what to write on the tape, at what time it should be written, how frequently it should be written and in what format it should be written. The same considerations apply if we write data to disk or to any other secondary storage device. Suppose for example, that our experiment involves counting of events; if we wish only to know the number of occurrences of an event then we are not concerned about frequency. Then we will not have to write data absorbed in the memory to tape continuously but rather at the end of an experiment all the counters and associated descriptive information would be written on secondary storage. In contrast to this some experiments will require that we record great blocks of data and a clock telling when these blocks were recorded. Decisions such as these concerning preservation of data will depend heavily on what type of off-line processing will occur after the raw data has been acquired. The experiment definition processor then must have considerable facility in allowing for the description of data blocks and the description of recording requirements. Not all of the data acquired by the sensory system will pass unnoticed by the Instrumentation Computer Monitor. Clearly at least two control signals must be specified (a) a signal that tell us to begin absorbing and acquiring data and (b) a signal which tells us to stop acquiring data. Such signals may take many different forms for example, the encounter of a particular control card by the Object Computer operating system may cause us to begin acquiring data. Similarly, the encounter of another control card may cause us to terminate the acquisition of data. Error conditions encountered in either the Object Computer system or the Instrumentation Computer system may also cause termination of data acquisition. It is even possible that a series of events may cause experimentation to begin or to terminate. In order to facilitate experiment definition some sort of language must be constructed which allows

an experimenter to describe his experiment on some high level. At first of course our experiments will be defined solely in terms of control instructions to the sensory system. Recognition of certain events by the sensory system will cause interrupts to the Sigma-7 CPU. We can design interrupt handlers specifically for a given experiment and indeed this approach offers the most rapid means of beginning our experiments. However, such an approach does not provide much facility for changing experiments; it's a highly painful thing to rewrite experiment definition instructions on an assembly language level and at such a level errors are much more likely to occur. Hence, I propose the use of Meta 5 to construct the compiler, to translate an experiment definition language or an experiment program into the appropriate instructions to setup the sensory system, acquire appropriate interrupt handlers, provide for I/O operations, and provide for on-line monitoring of the experiment on the graphics station. The experiment definition language and its associated compiler are described more fully in the next section.

Section 5

The Experiment Definition Language

The Experiment Definition Language must have the following capabilities: We must be able to statically assign data receiving units to processor nodes. We will also have to define which subset of the data receiving units may be active during the course of the experiment. This enables us to setup a library of instructions which maybe needed during the experiment to activate or deactivate data receiving units in the sensory system. I choose to call this portion of the language the Structural Assignment Section. At the same

time data receiving units can be assigned priority within the data selection unit in anticipation of interference or a conflict of request for data buss service. To allow for true generality in experiment definition, we may have a special structural section which describes the physical organization of probes into the object computer system. Essentially what is needed is a description of the source information bytes associated with each object computer interface unit. Although the probe structure will probably not change very often, as we move from one computer system to another or as our areas of instrumentation interest vary, such structure will change and the experiment definition program should allow for such changes with a minimum of software difficulty. There are several levels of activation of sensory system devices. Each data receiving unit can be activated or deactivated, an entire data selection unit can be activated or deactivated and with suitable filtering in the processor node, data from each object computer interface unit can either be accepted or ignored. In order for the experiment programmer to make decisions about the structural assignment of data receiving units and processor nodes, some sort of information proof sheet will have to be available. Such an information sheet might include the configuration of object computer interface units and their associated data receiving units and the description of the processor nodes which are available. There would of course be symbolic names in the experiment definition language to refer to these devices. Similarly, a list of synchronization signals should be available to the experiment programmer. Modifications of the object computer operating system will produce such signals. Among others, we expect to have signals known as emits coming out of the object computer system. These will not be available or cannot be brought to the attention of the experiment programmer, but synchronization signals will be available and

such a list should be presented to the programmer so that he can decide under what conditions to terminate his experiment or under what conditions to activate a different phase of his experiment. What I have in mind here is something similar to the PL/1 "on condition" coding convention. Associated with each sync interrupt or with each unique error condition, or error interrupt, the experiment definition language will have an on-condition statement. With the use of conditional statements a programmer will be able to specify actions to be taken by the instrumentation system upon the occurrence of some event. Clearly, this facility is needed to start and stop the experiment and to handle a multi-phase experiment. Such conditional statements will probably be translated into a sort of on-condition information vector which will be present at all times during experimentation. Upon the occurrence of certain interrupts or upon the receipt of certain synchronization signals, this condition vector could be interrogated and the appropriate action taken. The sort of actions taken first may be rather gross in nature for instance, START, STOP, PROCEED TO NEXT PHASE, or BOMB OUT. More sophisticated actions might be to change the monitor display when certain events occur. Since the programmer may not think about every error condition that might occur, the system will have to have some standard action to be taken in the event of error or interrupt conditions if they are not specified by the programmer. The problem of assigning storage to each processor node and the problem of assigning output buffers for preservation of absorbed data can be solved by the use of a single function called a Data Block Declaration. The structure of the data block declaration is similar to that used by COBOL or by PL/1. Each data block has associated with it a certain processor node. Each processor node has associated a certain set of sequential memory cells. Hence we see

that the data block has several levels, each level corresponding to a processor node. At any given time a single processor node will be associated with only one data block. Of course if there are many phases to the experiment the processor node may be assigned to several different data blocks but never at the same time. Perhaps in the future we will be able to arrange an experiment in which a processor node is associated with several different data blocks depending on certain events or conditions that have occurred. For example, in the memory utilization experiment a processor node will operate on two different data block buffers, however it only operates on one at a time. This kind of data block switching can be implemented through the use of an on-condition or an on-clock condition which causes the data block assignment for that processor node to be changed dynamically. It's worth noting here that since the sensory system has direct access to all the Sigma-7 memory, that we have no direct means of protecting our memory from arbitrary clobbering by the sensory system. It maybe necessary to perform some kind of background checking, that is to say, a sort of scan of the memory being operated on by the sensory system processor nodes in order to detect overflow or arbitrary clobbering. I haven't given too much thought to this yet, but the problem certainly exists. Activation of data output could be performed through the use of the on-conditional statements. We might employ a sort of block structure for these conditionals so that a block of control could be executed whenever a certain condition occurs. For example, we might say that on receiving the START signal we would begin to produce output on the tape, we would write a label, we would display something on the graphic station, and then we would write one of the buffers perhaps associated with a couple of the processor nodes. We might also indicate whether the buffer was to be written with a clock

record telling us when the record was written. We could also indicate whether the buffer was to be written in a continuous mode, that is after each interrupt is received that we would begin writing the buffer again. We could also indicate the possibility of double buffering, this would require a slightly more complicated conditional statement in which we indicated that a buffer was to be written and then switched. Clearly, this requires the dynamic activation of the instructions to the sensory system so that a certain processor node's base register pointing to Sigma's memory will be switched to point to an active buffer while the inactive buffer is being written out to tape. Although there are great deficiencies in the Meta-symbol assembly language it does have the capability of handling procedure type statements. I think that our approach should be to construct a series of macro statements which will allow us to perform the kind of control described before. If this turns out to be too much of a problem to program directly, then we can use Meta 5 and some higher level language which will produce the appropriate macro calls. Then a series of macro source statements could be run through the Meta-symbol assembler, producing code which would be used directly to perform experimentation. Meta-symbol does in fact provide for the inclusion of system procedures. These procedures are called in by a SYSTEM directive at the time that the meta-assembly occurs, so that we could have on the library a collection of system procedures associated specifically with experiment definition and these will be called in at the appropriate time. The output from the Meta-symbol processor is what is known as SDS Standard Object Language. It's clear that since the program that results from the meta-symbol assembly is not all entirely executable, the loader that we use to read in this program will have to be somewhat specialized. For example the portions of the program which turn into the interrupt information

vector will have to be read-in in some absolute form. Other portions comprise blocks of instructions to be executed for the control of the sensory system and others will comprise calls to various I/O routines. I have not yet worked out all of the details involved in loading in the Meta-symbol output but it's clear that we will have a number of interesting decisions to make about loading this data in. We will have an ENABLE instruction which allows us to enable a data receiving unit to send out either a single piece of information and then terminate or send out information continuously. We will also have an EXTERNAL function which allows us to describe a phase or a process as external and hence allow the experiment programmer to write in assembly language, some control programs which cannot provide for in the experiment language compiled by Meta 5. These external assembly language programs will then be assembled along with source that is put out from the Meta 5 experiment compiler and the entire process will be loaded in and executed.

Section 6

In the preceding discussion we have made the assumption that only the operating system has been modified to produce synchronization or emit signals to the sensory system. Since our Phase 0 operation involves the emission of data from user FORTRAN programs we must also provide for the collection of emit signals from the background of the operating system. Essentially emit signals coming from FORTRAN programs in execution on the object computer will be handled in the same fashion that data coming from the operating system was handled. Namely, the emits are associated with the central processing unit data receiving unit in the sensory system and this data will be associated with some specific processor node. At first it did not seem likely that we would need to produce

SYNCS in the user FORTRAN program however, it is not unreasonable to only instrument certain sections of the program. This can most easily be done if our pre-processor will insert synchronization messages to be put out by the executing FORTRAN program. For example, in general if the entire program is to be instrumented,that is, if we will produce emits from the entire program then a SYNC will be put out at the beginning of execution and a SYNC will be put out upon exit from the program. These SYNCS will provide us with a means of control for turning on and turning off data acquisition in the sensory system. Once again our instrumentation system proof sheet will be of great value. Standard synchronization messages put out by the object computer software system will be listed on the proof sheet and appropriate on-conditional statements can be used to describe actions to be taken on receipt of such SYNCS. Similarly, the experiment programmer can define actions to be taken on SYNCS received from user programs executing in the object computer's background.

Section 7

This section is basically a series of notes on topics which I haven't considered deeply but which must be considered. Number one is the problem of error handling. All kinds of errors can occur during experimentation. There may be transmission errors between the sensory system and the memory, or transmission failures between the object computer and the sensory system, failures of storage of data on the secondary tapes, or synchronization failures between the object computer system and the instrumentation system. We will need some kind of technique for handling such errors. The on-conditional statements supply part of the technique but there will have to be others of more standard form that are always present in the

instrumentation system. Number two, we'll have to modify the control card interpreter which presently exists in the basic control monitor in order to call in the various processors associated with experimentation. For example we'll need an ! EXPDEF for experimentation definition control card and perhaps some other types of control cards like ! GRAPHIC for the graphics assembler. We have another interesting problem associated with the file transmission system. Since this accomplishes memory to memory transfer and we are dealing with a 36 bit object computer and using the 32 bit instrumentation computer, there will have to be some sort of standardized translation function. A portion of the translation function can be handled by the software but other portions will have to be handled directly by the hardware. For example in a 36 bit machine we may take 18 bits of data and put this into a 32 bit word in the Sigma-7's memory. Obviously, we will want to design the translation process in such a way that when we move to other object computers our translation process will not be difficult to provide for.

Section 8

This is a description of progress to date on current programming efforts. Meta-5 has reached the stage of user implementation; that is, we have a system that user's can use. However on the Sigma-7 not all primitive instructions have been implemented, this is also the case in the 360 although most of the primitive instructions defined in the Meta-5 specifications have been implemented on the 360 system. We have prepared ourselves to make the necessary changes in the Meta-5 on the Sigma-7 to get the entire instruction set available certainly within 2 weeks. Additional documentation is now being written for the Sigma-7 implementation of Meta-5, this will include flowcharts and detailed descriptions of

the Meta machine and interpreter and also a rewrite of the specifications of Meta 5 and some sort of user manual. I'd like to draw your attention to the experiments being done by Ed. Russell and by Leon Presser. They have been measuring certain activities that take place in the Meta machine during execution of the Meta 5 meta-compiler or of other programs. These measurements have helped us to isolate certain bottlenecks in the implementation of the Meta machine and in the implementation of the interpreter and in fact they have been highly useful in guiding our rewriting of certain portions of these systems. These certainly could be considered a sort of Phase 0 measurement system or introspective example of measurement on the Sigma-7 and on the 360. The FORTRAN preprocessor which Chuck Kline has been working on has not gotten too much further do to the lack of complete instruction set from Meta 5. However this should be remedied very shortly, and I expect his complete processor to be completed sometime by the end of February. File management has reached a completion stage. It cannot go much further into the debugging phase until we complete the changes necessary in the basic control monitor to support I/O scheduling required by file management. I propose to utilize Bob Goldman to work on this basic control monitor change. Yu Ping Cheng, who has since left us, did most of the ground work for this change to the basic control monitor, but we were held up largely by lack of good source copy of the basic control monitor. This will be remedied on Tuesday when Hal Wyman from SDS will bring out all of the appropriate source copies of the basic control monitor. The experiment definition specifications have been left dormant for the last month or two largely as a result of the many commitments that I have made to my time. However since we will be able to work simultaneously on the implementation via Meta 5 and the implementation via Meta symbol we should be able

to catch up or at least no slip more than a month or two on the completion on the Experiment Definition Processor. Chuck Kline will do the work on the implementation of Meta-5 compiler for the experiment definition language, and I plan to take responsibility for the Meta symbol implementation. Peggy Schneider will began work shortly on the specifications for the overlay loader, the process requester and the core storage monitor. This can proceed in parallel with the implementation of changes on the basic control monitor to support the new file management system. I'd like to note here that the File Management System that we provided for the instrumentation system is not compatible with the File Management provided either by Read 75 or by the Batch monitor, these are all different and there will be some problem about protecting information on the RAD as we switch from Batch monitor to instrumentation and back. However when we get the large RAD presumeably third quarter this year we can reserve portions of the RAD for either batch or for the instrumentation system thereby avoiding the problem of clobbering data inadvertently. We may still see some difficulty in communicating data between the two systems but I am confident that we can work out these problems when they arise. John Podolsky has pretty well laid out the sensory system interrupt specifications and is also going to do specifications for the graphics station interrupts, these will probably not include service interrupts for the Lincoln Wand. If it's possible he also plans to do some programming although I am convinced that it is more important for him to complete specifications of the entire interrupt system than it is for him to do any of the programming, considering that his time with us is so short. According to the instrumentation development schedule we should be into the sensory system interrupt implementation stage, this is not the case but we are slipped back roughly a month, so that our implementation will not

start until the end of this month and the interrupt specifications will be completed at the end of this month rather than at the end of November as specified. Barry Coggan has attacked the graphic display assembler with considerable energy and I'm sure that we will see some sort of a product within the month. He has already finished most of his specifications and has actually begun test programs written in Meta symbol. As you know we do have a difficulty using Meta symbol since it does not have character manipulation capability. Since the language does have procedure capability and we obviously need that for the assembler, we will windup making a two pass graphics assembler. We will assemble as much as we can using procedure calls in Meta symbol and we will assemble executable instructions to provide the character manipulation necessary and execute this. The result will be an absolute program that will be output either to tape or left in core which can be executed by the graphic station. This is an additional advantage since we could also produce certain executable codes capable of modifying the graphics program in some well defined way. This would be useful if we were to compile graphics programs that were going to put out display information during instrumentation. For example if we produce a bar chart of some sort during instrumentation, and it becomes useful or necessary to change the bar chart dynamically, such a control program associated with the display instructions could be utilized with relative ease to make changes in our monitor display. There is no question that there will be insufficient time during instrumentation to bring in the graphics assembler and to reassemble graphics programs for the purpose of dynamically changing monitor displays. To assist Barry in constructing the graphics assembler, we have commissioned the writing of a display simulator. Stefano Crespi-Reghezzi

and Shahla Zahedi have completed a large part of the graphic simulator. It has been broken up into two sections. The first section will accept absolute input in the form of graphics display instructions and proceed to disassemble these into a psuedo assembly code, at the same time it will execute the instructions as far as is possible and produce a protocol describing the status of the graphics display station as each instruction is executed. We cannot very well simulate light pen interrupts and tracking and so forth but it is possible in the protocol at least to indicate that light pen tracking is taking place or that the light beam is intensified or that an edge violation has occurred or some other problem has arisen. At the same time that this graphics protocol is being produced we will produce pseudo-code probably in the form of control instructions for a CALCOMP plotter which simulates the graph that would have been produced on our DEC-340. This psuedo-code could be used either to drive a CALCOMP system directly from tape or could be intrepreted by some other program which then produces on the line printer a simulated hard copy output of the contents of the graphics station display. This may have uses even after the graphic station has been activated, since we have no means of producing hard output, except by taking photographs of the face of the graphics display. Software control functions for the Lincoln Wand have been discussed in considerable detail with Ted Makranczy and next week we will formalize these control instructions and try to have some sort of internal memo produced by the end of the week. In anticipation of further loading of the Sigma's 7 computer time and in particular the reduction of time available for computing while our hardware is being attached and debugged, I propose to hire a computer operator to begin March 1 to run closed shop operations in the morning presumably from 8 until noon or perhaps from 8 to 10 depending on the job load. This has two advantages, in

the first place it will formalize record keeping in the computer center so that we will have a better idea of how the computer's time is parcelled out among it's various tasks. Furthermore it will assist us in improving job through-put. There are great many users of the Sigma-7 system whose work is not directly associated with the instrumentation project. Because of the convenience of the Sigma-7 it would be nice if we could continue to supply service to these people, however it cannot be done on a open-shop basis since it's highly time consuming for a queue of people to come in, mount tapes and so forth, and wait around until their turn comes up. A more formalized close-shop operation may allow us to service these people and still continue with our software development effort. If this proves not to be the case of course, we will have to cut down on our service capabilities to people using the system for projects other than instrumentation.

Section 9

Hardware difficulties with the peripheral units of the Sigma-7's system. Ever since June when the computer was first delivered to us we have had considerable difficulty with the 400 card per minute reader. In that period of time a great many portions of that reader have been replaced or repaired, for example just last Surday the entire drive motor had to be replaced. I was informed earlier this week by Gordon Norris that SDS has approve our application for a 1500 card per minute reader and this reader will be delivered within the next three to four weeks. I have also noted difficulties with the two tape drives. In particular they seem to be incompatible or marginally compatible with foreign tapes produced on other computer systems. In particular tapes produced by the Model 75 at the northern campus nodedo not seem to be very readable on the Sigma-7. Skew has been checked several times by Si Korn our SDS customer

engineer. His claim is that skew problems do not seem to exist, therefore we must seek some other reason for this incompatibility. It seems surprising that nearly all 360 tapes that have been produced for use on the Sigma-7 have caused difficulty. Of course it is possible that the tapes used at the CCN are old and are only marginally readable anyway. However should we switch to faster tape drives such sensitivity is likely to increase, therefore we must be prepared for incompatibility with other systems, this is essentially not tolerable. There is even some evidence that one tape drive cannot read tapes produced by the other on the Sigma-7. This has been investigated and the skews have been adjusted on both drives. We have also had some difficulty with tape breakage on these tape drives, this seems largely to be a design problem for example, if the tape is not accurately positioned over the vacuum column at the time that the load button is pressed the supply reel spins tape even though it doesn't go into the column. Shortly afterwards, of course, the tape breaks as it stretches against the read head of the unit. Although this has happened only nine or ten times, even that sort of frequency is not very acceptable. The implication is of course, that back-up tapes must be kept of any important master files. The Sigma-7 line printer had been giving us considerable difficulty in the past, it was apparently arbitrarily skipping pages from time to time. This seems to have been solved by replacing the carriage control tape and replacing the read brushes on the carriage control tape reader. Another problem with ribbon positioning has been solved as well. After the addition of the keyboard automatic send/receive device we discovered that several control programs for that device were not written quite properly. In particular the batch monitor when accessing the keyboard output function also geneated a read signal due to an improper device address in the start I/O instruction.

We will have to go through the basic control monitor I/O handlers and make suitable corrections so that the automatic send/receive device is properly handled. SDS has not officially announced any fix for this problem.

Section 10

Visitors to the Instrumentation Project

Dr. Lawrence McNamee hosted two gentlemen from Siemens Aktienqesellschaft in Germany and asked me to describe to them the nature of our project. Their cards will be attached to this memo. I promised to keep them informed about our progress as they seemed highly interested in the project. Of course, this seems to be true of all visitors to the Sigma-7 center. No papers were published this month. Although I have been nominated for the Vice-Chairmanship of the SDS User's Group, the results of the election will not be known until March. I have also been scheduled to present a technical progress reprot concerning the SNUPER system to the Arrowhead chapter of the ACM. This presentation will be made on the 12th of March.

Section 11

By next week I will have completed at least the first draft of the Experiment Definition Processor and its associated language. This will be split into three parts. One part describing the Meta symbol procedure functions, another part describing the Meta-5 high level compiler language and the third part describing the actual execution of the experiment definition. A major effort will be made to complete the required primitive verbs for Meta-5 in order to

get Charles Kline's preprocessor going so that we will have information from the Phase 0 instrumentation to present to ARPA on March 13. I propose to construct FORTRAN program's whose flowcharts and loop structure are well defined for the purpose of testing both the preprocessor and the data acquisition system. Because the transfer path is not yet hooked up to the sensory system, it's just as useful for us to perform experiments on preprocessed FORTRAN programs which will run in the Sigma-7, since the test of the preprocessor is equally valid on both computers. Unless there is some major objection to this approach, the Phase 0 data that we present to ARPA will be data obtained directly from the Sigma-7 rather than from the 7094.

Section 12

It would be highly useful for us to hire a programmer who is experienced with the 7094 IBSYS system, or at least has assembly language exposure to that system. Although we originally intended to employ Steve Wolf in this capacity, his time is now almost solely devoted to the support of the Model 75 with the campus computer center. As was suggested earlier, perhpas one of the programmers who answered the ad for ITTE's programming request might be available for our use. This is certainly worh looking into.

Section 13

Larry Mallach, who worked for us last summer, will be available from June 1 to November this year. Norm Hawkins, currently with IBM, is planning to return to UCLA in September. Both of these people are experienced programmers and would be an asset to our project. I have held off making formal requests for hiring them until we know the outcome of our ARPA presentation next month.