QUARTERLY TECHNICAL PROGRESS REPORT

January, February, March 1974

UIUCDCS-QPR-74-1

Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801

TABLE OF CONTENTS

Page

I. CIRCUIT RESEARCH

Supported in part by Navy Contract #N000 14-67-A-0305-0024.

Summary

Garlan Huberts describes the detailed operation of NORMAN, a machine for analyzing the profile of a two-dimensional object independent of size and orientation. In his report on CAECOTRON--the device which converts a visual scene into an audible signal as an aid to the blind--Bernard Tse discusses the possible use of a controlling microprocessor. Jim Cutler's report in the area of molecular stochastics deals with the direct generation of stochastic sequences as measures of physical quantities. The fourth report is a first outline of ROBOGUARD. This is a new project with the objective of building a machine that can automatically keep watch over a storage area such as a warehouse and signal if something is missing.

M. Faiman

W. J. Poppelbaum
Principal Investigator

1.1 NORMAN (NORMalizing ANalyzer)

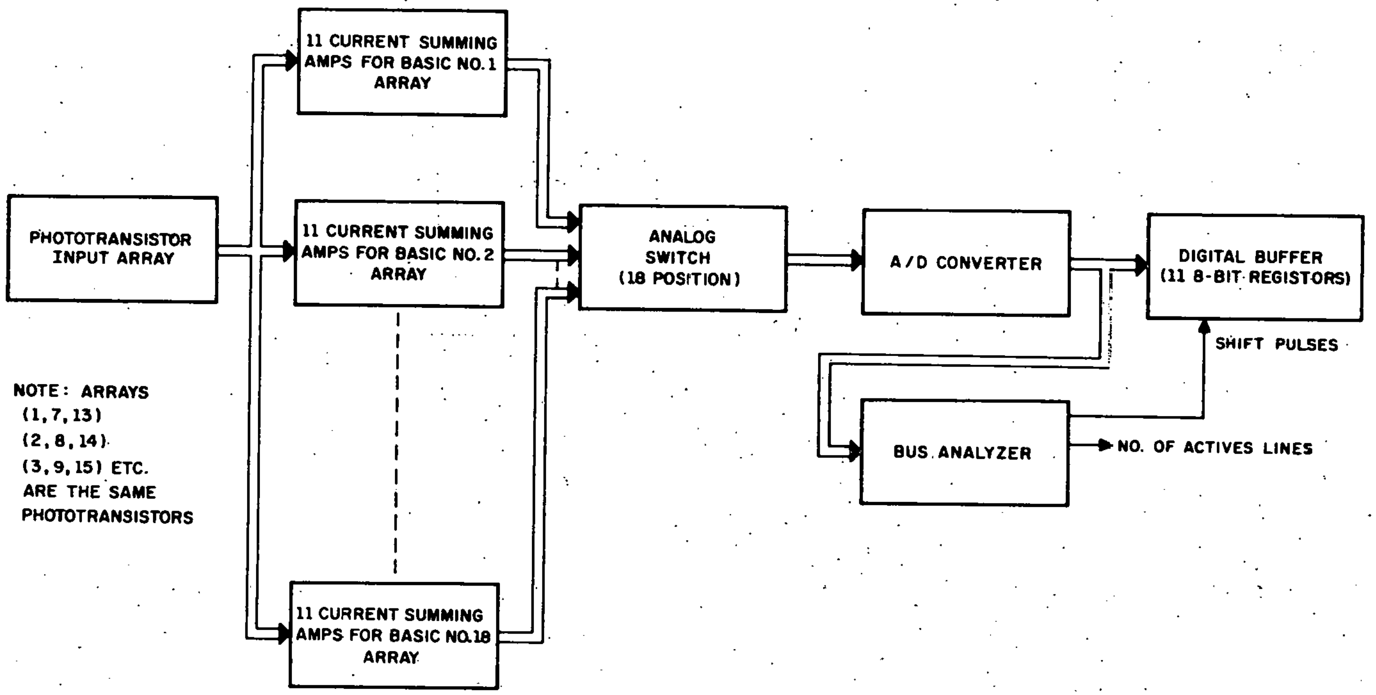
1.1.1 Machine Description

The purpose of NORMAN is to recognize two-dimensional figures under conditions of rotation, magnification, and translation. The figure to be recognized will be laid over a special array of phototransistors. As Figure 1 shows, current is then summed from the phototransistors to give analog voltages which are representative of the width of the input figure along certain "scan" lines of phototransistors (see earlier quarterly reports for more details on the input array geometry). Eleven of these analog voltages are then switched onto a bus at a time and digitized. This represents a certain angular view of the figure. The digitized numbers are stored in eleven four-bit registers.

The bus analyzer then takes over and does two things. First, it shifts each four bit number up from one register to the next until the top register no longer contains the number zero. The normal case for an input is shown in Figure 3 where an input figure, a triangle, is placed somewhere on the array. The scan lines not covered by the triangle have the number "zero" in their respective registers. This shifting action has the effect, to the rest of the machine, of moving the input figure up to the top edge of the array. This action removes positional dependence for recognition of figures.

Secondly, the bus analyzer starts with scan line eleven and counts down until it hits the first non-zero line. The counter then contains the number of scan lines covered by the input figure (termed "active" lines). This number is used to determine the size of the figure.

Referring back to Figure 2, a time division distributor then samples two adjacent registers of the buffer and these numbers are converted into


PHOTOTRANSISTOR INPUT ARRAY
11 CURRENT SUMMING AMPS FOR BASIC NO. 1 ARRAY
11 CURRENT SUMMING AMPS FOR BASIC NO. 2 ARRAY
11 CURRENT SUMMING AMPS FOR BASIC NO. 18 ARRAY
ANALOG SWITCH (18 POSITION)
A/D CONVERTER
DIGITAL BUFFER (11 8-BIT REGISTORS)
SHIFT PULSES
BUS ANALYZER
NO. OF ACTIVES LINES

NOTE: ARRAYS
(1,7,13)
(2,8,14)
(3,9,15) ETC.
ARE THE SAME
PHOTOTRANSISTORS

Figure 1. NORMAN Preprocessor

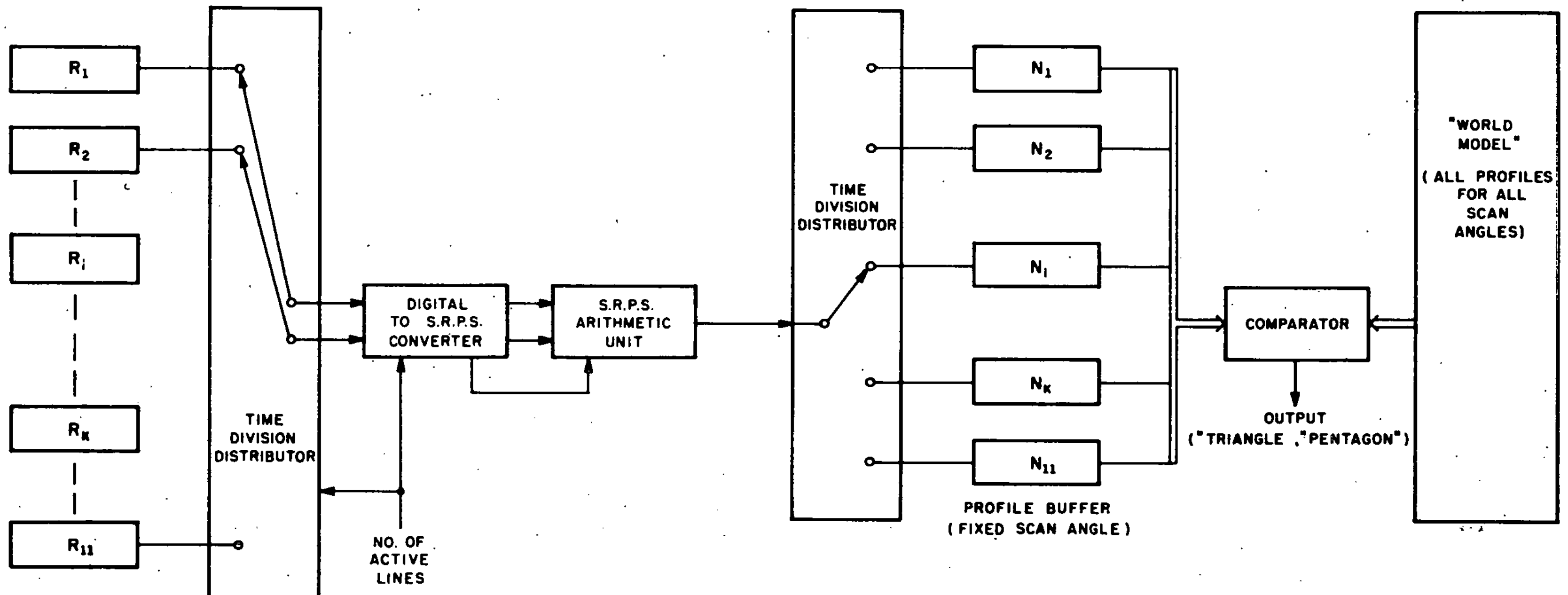
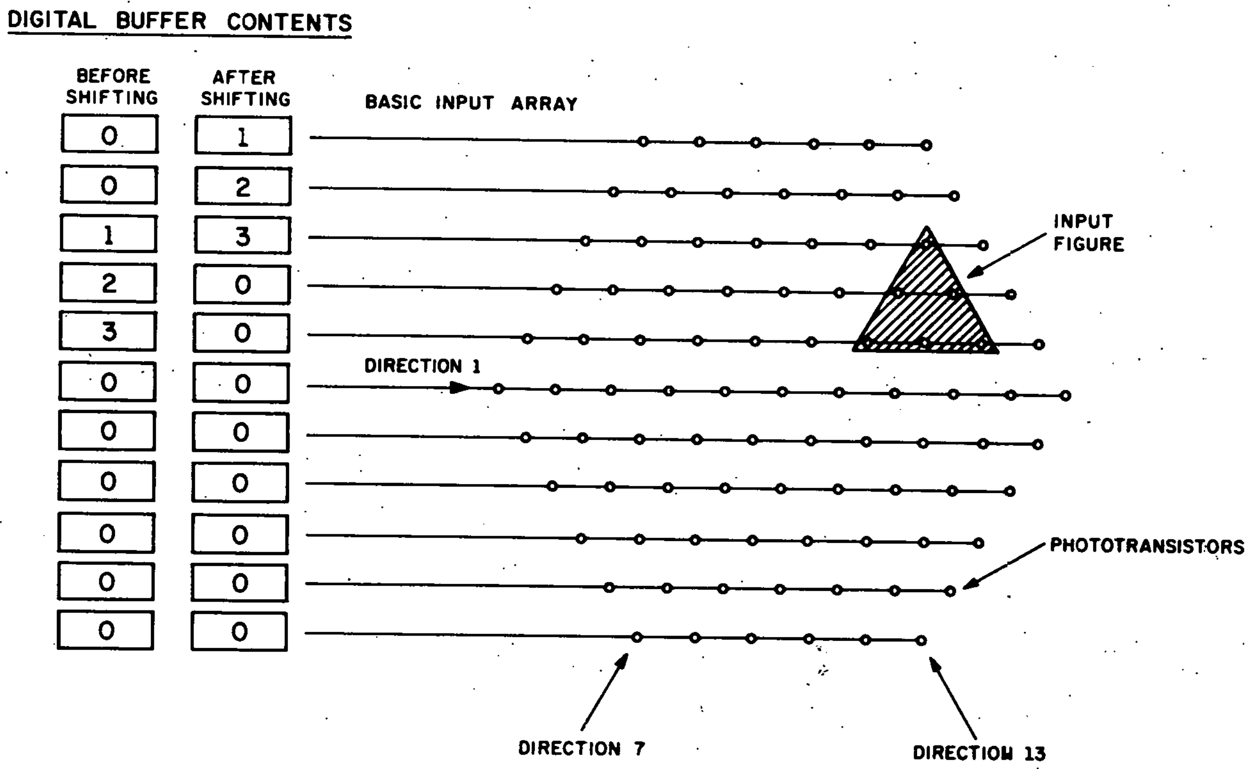

R_1
R_2
R_i
R_K
R_{11}
TIME DIVISION DISTRIBUTOR
DIGITAL TO S.R.P.S. CONVERTER
S.R.P.S. ARITHMETIC UNIT
NO. OF ACTIVE LINES
TIME DIVISION DISTRIBUTOR
N_1
N_2
N_i
N_K
N_{11}
PROFILE BUFFER (FIXED SCAN ANGLE)
COMPARATOR
OUTPUT ("TRIANGLE ,"PENTAGON")
"WORLD MODEL" (ALL PROFILES FOR ALL SCAN ANGLES)

Figure 2. NORMAN Normalizer and World Model Store



BEFORE SHIFTING	AFTER SHIFTING
0	1
0	2
1	3
2	0
3	0
0	0
0	0
0	0
0	0
0	0
0	0

Figure 3. Bus Analyzer Shifting Action

two synchronous random pulse sequences (SRPS's). Also converted to an SRPS is the number of active lines information from the bus analyzer.

Size independence is gained by sampling over the number of active lines during a certain fixed period of time. In other words, the number of active lines information controls the switching rate of the first time division distributor, because a larger figure requires a higher switching rate than a smaller figure, having more active lines to cover in exactly the same amount of time. Two registers are sampled at once, because the SRPS arithmetic unit can be made to switch gradually from one to the other. More information on how this can be done will be given in a later quarterly report when the arithmetic board is actually designed.

The output of the arithmetic unit is a single SRPS signal which varies almost linearly in time with respect to the profile of input figure. The second time division distributor switches this signal during the same fixed period of time into eleven counters which serve as the profile buffer. Thus, at the end of the time period the profile buffer contains the profile of the input figure which is both position and size independent. By comparing this profile at a certain scan angle to all of the profiles at all the scan angles of all the figures stored in a memory, recognition of the input figure is also possible under rotation.

1.1.2 Progress during the Quarter

During this latest quarter a cabinet was obtained to house the project and after soldering the phototransistors into the input array board, the board was mounted in the cabinet. Also mounted were four sixty watt light bulbs above the phototransistors and two exchange cards below it. The exchange cards accept flat cables from the phototransistor board and have outputs in the form of wire-wrap pins. The system was then checked out at the exchange card level, the criterion being that each phototransistor must saturate with a 16k ohm load from the light of the bulbs. Only five phototransistors out of the 379 were found to be bad.

The analog to digital converter board, which was designed last time, was built during this quarter. Also designed and built was a board called the analyzer board, which contains both the digital buffer for the A/D converter and the bus analyzer. Since the switching points of the first time division distributor must be very accurately controlled, a hard-wired read-only-memory was designed and built to give accurate switching pulses depending on the number of active lines. The fourth board designed and built was the digital SRPS converter board, which uses feedback shift register techniques to generate the SRPS's. These four boards were just completed and still have to be checked out.

Finally, design work has just begun on the memory boards, which will hold the normalized profiles of the different input figures to be used in the comparisons.

Garlan Huberts

1.2 CAECOTRON

CAECOTRON is an instrument designed for the blind using two television cameras to extract both light and depth information for the user. During the past quarter, the use of a microprocessor to determine depth information was investigated. The main drawback of microprocessors in general--the slow speed--is not a problem in this application because of the much slower transfer of information between the machine and the user.

The Intel 8008 microprocessor in particular was considered. This microprocessor possesses the necessary instruction codes for the application. Although a 16-bit microprocessor would be preferable since CAECOTRON operates on a television line that has 16 discrete cells, the Intel 8008 has the advantage that it is readily available.

In the next quarter, a microprocessor chip together with the accompanying memories will be purchased.

B. Tse

1.3 Molecular Stochastics (Project No. 61)

1.3.1 Project Summary

Molecular Stochastics is a project investigating the feasibility of producing stochastic sequences directly from physical measurements such as temperature, velocity of a liquid, etc.

1.3.2 Project Status

In this past quarter, it has been shown that it is possible to produce a stochastic sequence that is a function of temperature. Thermal noise of a resistor is used as a transducer. This noise is amplified and compared with a voltage level which results in a stochastic sequence. The range and resolution of the temperature is a function of the voltage level with certain restrictions; i.e. this voltage level may not be equal to the mean of the thermal noise.

1.3.3 Future Work

A velocity transducer is presently being constructed and tests will be conducted in the next quarter. The initial attempt will use generated vortices produced by an obstacle in the flow of the liquid. These vortices will be counted by a microphone.

Producing a stochastic sequence that depends upon light intensity is also being investigated. Since photons arrive at a surface according to a

Poisson process, a stochastic sequence may be produced if a transducer can sense photons.

Jim Cutler

1.4 ROBOGUARD

Roboguard is a robot-like machine which is to be capable of moving about a warehouse, observing what is stored there, and sounding an alarm if anything is amiss.

Roboguard will use some type of transducer to obtain the two-dimensional view of a room. The two prominent techniques which are being investigated are radar and sonar. Radar would provide faster responses with better accuracy than sonar, but sonar can be implemented much more economically.

Edward Pott

2. HARDWARE SYSTEMS RESEARCH

Supported in part by Navy Contract #N000 14-67-A-0305-0024.

Summary

COLFTAR has a new crystal cooling and monitoring system, a new electron gun, and will soon incorporate dynamic focussing: details by Stan Kopec. Lucien Facchin's report describes the display configuration for the digital flight aids project. Dev Bose has completed the design of the final module for FROG and construction is under way. INCOM, the interface computer for recognizing hand drawn symbols, has been fully simulated by Mohamed El-Sonni. Finally, Less Daley reports on minor changes to THESPIAC and the start of construction.

M. Faiman

W. J. Poppelbaum
Principal Investigator

2.1 COLFTAR (Project No. 12)

2.1.1 Summary

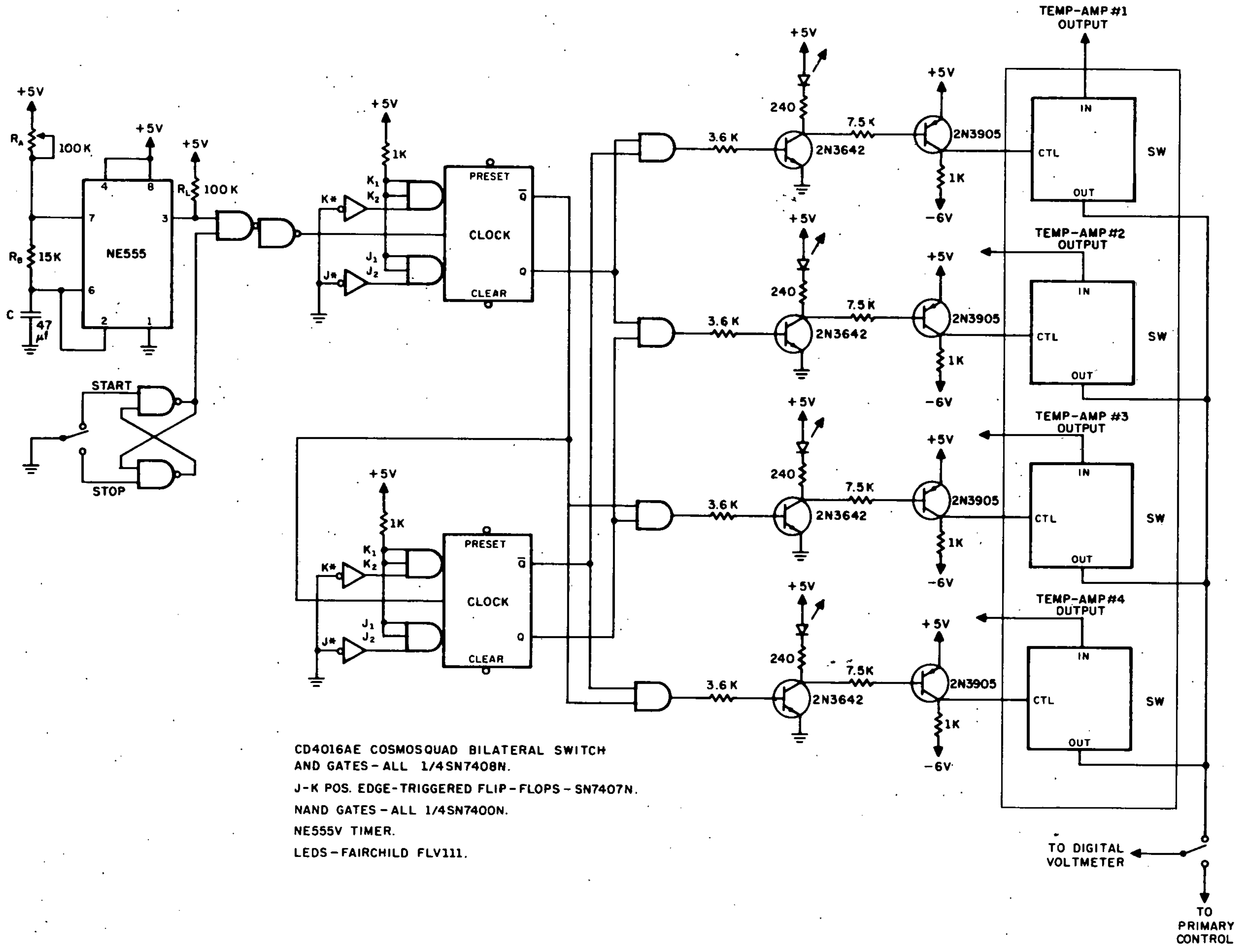
During this period the new temperature sensors were installed and the new circuits tested. After minor adjustements they functioned as well as had been expected. However, during subsequent cooling of the crystal a seeming inefficiency in one secondary thermoelectric module, which had been noted earlier, worsened. This problem prevented cooling of one edge of the crystal to less than -35° C. Upon opening the chamber for inspection, it was discovered the substrate had cracked along a line near the suspect T.E.M., causing an effective "open" thermal circuit. Thermal expansion and contraction along the T.E.M.-substrate interface is believed to be the primary cause of the problem. Ways to remedy the situation are presently being investigated.

Along with the new temperature-sensor network, a 4-channel analog multiplexer was designed and built (Figure 1) to multiplex each of the crystal-edge sensor readings (4 in number) to the digital voltmeter, every 5 seconds, sequentially, without operator intervention. This is a great aid in operation of the system, as formerly temperature readings were taken by manual probing of test points.

The new electron gun was installed, but has not been tested thoroughly yet due to the temperature-control problem.

The first stage of the dynamic-focusing arrangement is complete, with the isolator preamp-driver (Figure 2) designed. The actual focus supply modulator is in the last stage of design and will be fabricated soon.

Stan Kopec



CD4016AE COSMOSQUAD BILATERAL SWITCH
AND GATES - ALL 1/4SN7408N.
J-K POS. EDGE-TRIGGERED FLIP-FLOPS - SN7407N.
NAND GATES - ALL 1/4SN7400N.
NE555V TIMER.
LEDS - FAIRCHILD FLV111.

Figure 1. Four-channel Analog Multiplexer

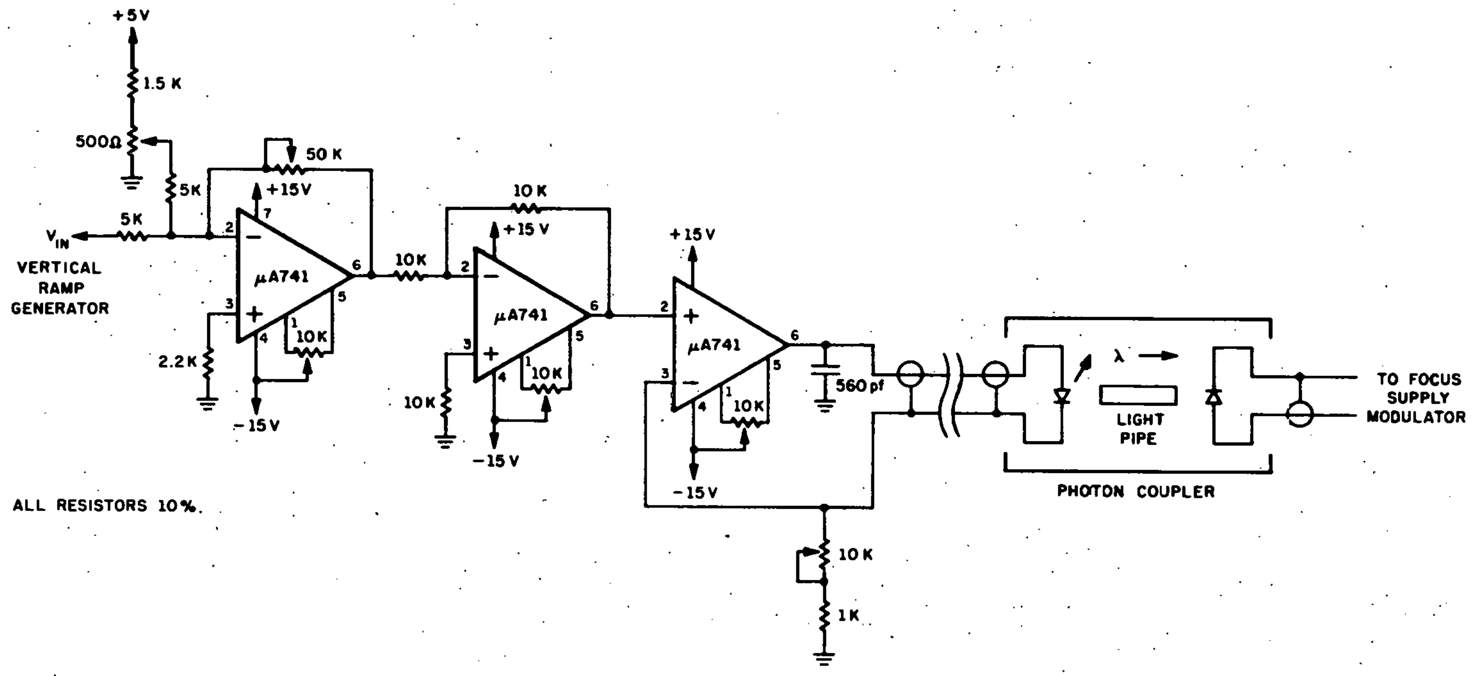


Figure 2. Photon-Coupler Preamp-Driver

2.2 Digital Radio Aids for a Flight Simulator

2.2.1 Summary

In the two preceding reports, the digital radio aids were discussed and one of their functions, the code keyer was presented. Two other functions, the radio memories and the display are now under study. In the following, the display will be presented.

2.2.2 The Display Configuration

The system implemented is hybrid, consisting of a set of binary and BCD counters for most of the conversions, and three 74185 ROMs when it is desired to simultaneously display the three Morse code letters.

The system comprises two parts: the conversion circuits and the control logic.

2.2.3 The Conversion Circuits (Figure 1)

The circuits for the Morse code letters consist of a set of latches: 2 hex latches (SN 74174) and one quadruple latch (SN 74175). They store the data coming from the data bus. The outputs of the latches are in groups of five and energize the 74185 ROMs used for the binary to BCD conversion.

The circuits used for all the other conversions consist of a set of binary counters (SN 74193) and a set of BCD counters (74192).

Quadruple 2 line to 1 line multiplexers (SN 74157) choose either the data coming from the BCD counters, or that coming from the binary to BCD ROMs. The multiplexers' outputs are fed into BCD to 7 segments decoders and drivers (SN 7447A). The decoder outputs drive the final stage, the 7 segment numeric display (type TIL 302). An additional circuit, the TIL 304 is used in order to indicate the sign. It uses a buffer driver.

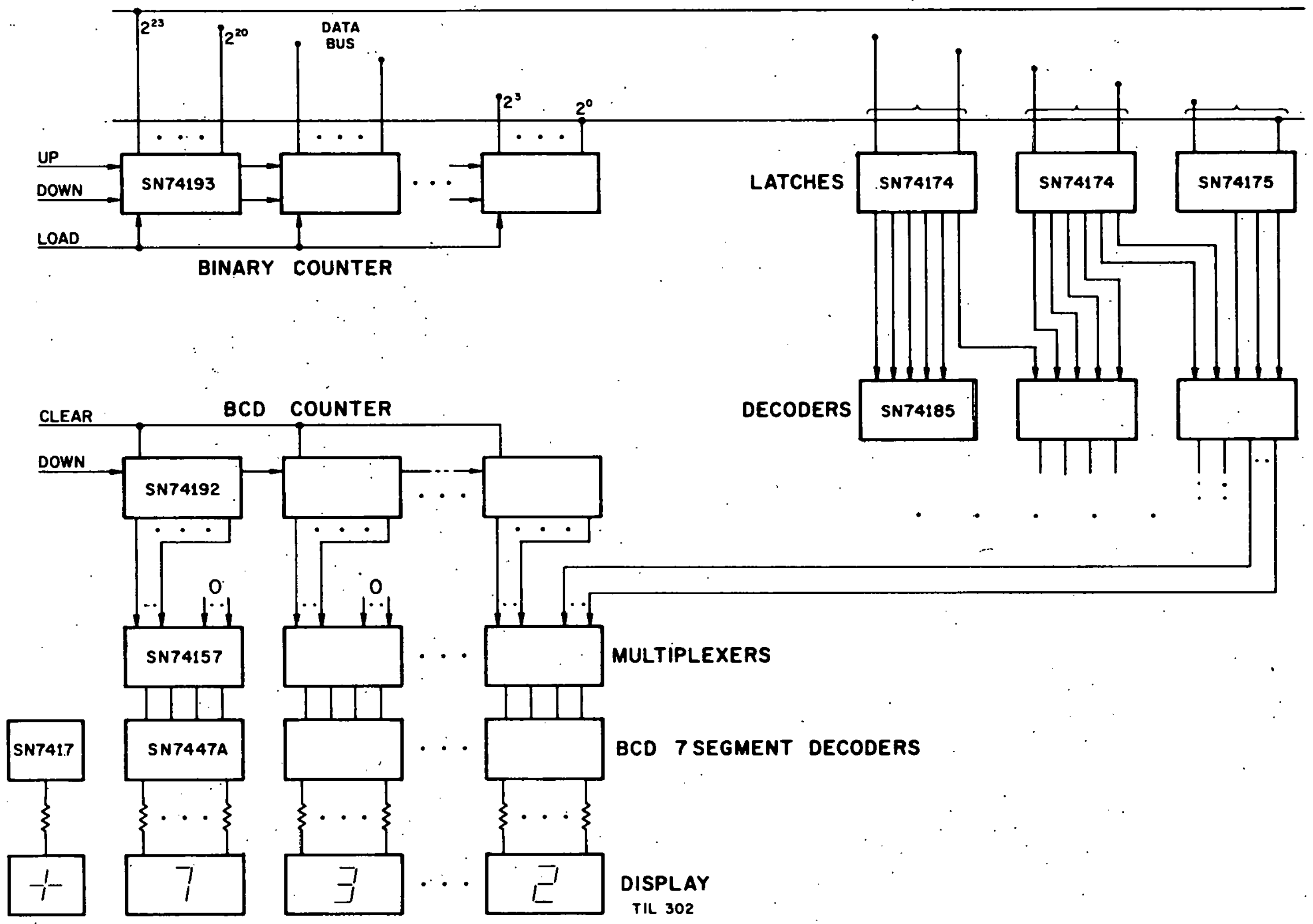


Figure 1. Conversion Circuits

2.2.4 The Control Logic (Figure 2)

When data is to be displayed, the instruction (a pulse on a given line) is decoded in order to perform particular functions which are of two types depending on whether the counters or the ROMs are to be used.

If we use the counters, three control lines are needed:

- counter start: this pulse loads the data into the binary counter, while clearing the BCD counters. Then it enables either the up or down line for the binary counter (for the BCD counter, only the down line is used).

  The carry and borrow lines of the binary counter are used to determine when it is empty. When this is detected, a pulse is produced which clears the flag and therefore disables the counting.
- special clock: this line is used when the data to be displayed requires a special conversion (for angles or conversion of distance to feet). The flag is reset to zero when the binary counter is empty.
- rate select line: this line determines which rate must be set into the rate multiplier. The selection between rate 1 and rate 2 is done through 2 to 1 line data selectors.

  The flag is cleared when the binary counter is empty.

If the Morse code letter circuits are used, one line is required:

- Morse letter line: this line feeds the clock input of the latches and sets a flag which maintains the select line of the multiplexers which transmit either the outputs from the BCD counters or the outputs of the ROMs at 1.

A fifth line is used for turning off the display by clearing a flag whose output is fed into the B1/RBO line of the 7 segment display circuits.

Lucien I. Facchin

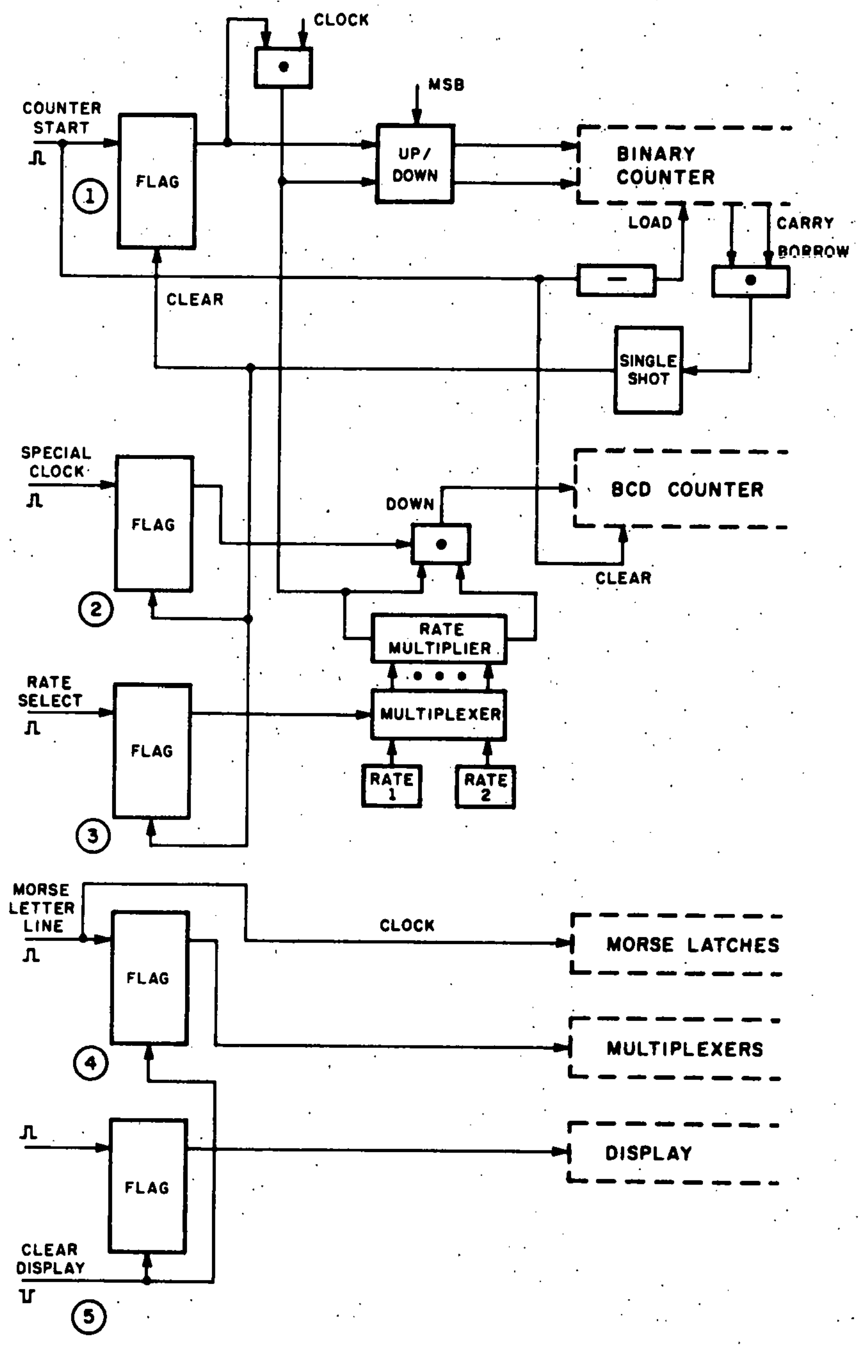


Figure 2. The Control Logic

2.3 FROG (Project No. 36)

2.3.1 Project Summary

FROG is a device which attempts to model the analysis performed by the brain of a small animal on visual clues in order to recognize the universals, prey and enemy. The details of the mechanism have been described in earlier quarterly reports.

2.3.2 Project Status

Of the nine basic logic elements which are widely used in the design of the mechanism, hardware design for the first eight have appeared in parts over the last four quarterly reports. In the following section we discuss the hardware implementation of the last element, namely, the T element. During the current quarter the principal macro-element, the T*-element, of FROG's "brain" has been implemented in digital logic. This element is designed in terms of the set of nine basic continuous-logic elements mentioned earlier. The T*-element is functionally somewhat like the unit cell (or neuron) of FROG's brain. The circuit has been tested in parts. A prototype layout is currently being made.

2.3.3 The T-Element

A T-element is a memory element. It is initially unset, i.e. its stored value is zero and its output is zero. At any later instant, $t = t_1$, its stored value is equal to the last non-zero input to the element multiplied by the quantity min(1, i/n), where i is the number of non-zero inputs till t_1 and n, an integer number, is a fixed parameter of the T-element. The output at any instant is equal to the stored value.

In the hardware form the circuit has a four-bit input, X, and requires a signal FRESH INPUT (FI) to signify the application of a non-zero

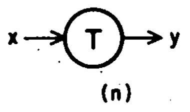
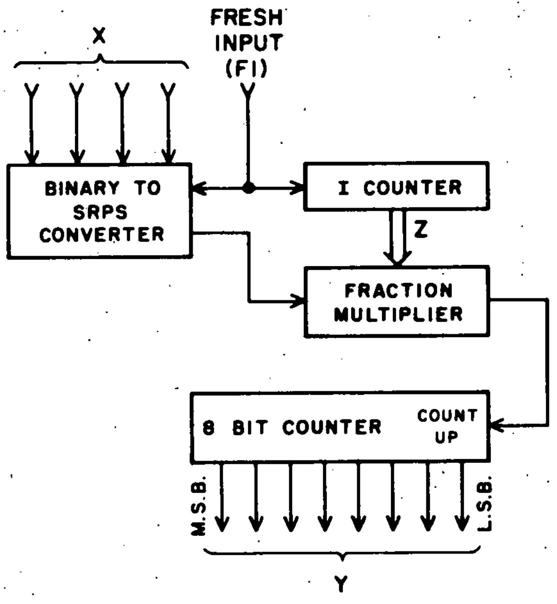
LOGIC ELEMENT		
NAME	SYMBOL	REALIZATION
TRANSMITTANCE		

Figure 1. Realization of a Transmittance (T.) Element (Schematic) (l.s.b. = least significant bit, m.s.b. = most significant bit).

input to the circuit (see Figure 1). The signal FI is utilized to increment a special counter called the I-COUNTER which maintains the current value of i till i = n, after which the counter value is kept frozen at n. Thus the output of the counter at any instant is x = min(i, n).

The FI signal initiates a conversion of the input binary number to a corresponding synchronous random pulse sequence (SRPS). The method of conversion is the same as that used in a P-element (cf. QTPR, Oct.-Dec. 1973, Section 2.1.2). This SRPS is passed through a circuit called the FRACTION MULTIPLIER which suppresses n - z pulses for every successive group of n input pulses (z = output of I-COUNTER) in order to achieve a multiplication of the input SRPS by the fraction multiplier z/n = min(i, i/n). The output SRPS is now converted back to an equivalent 8-bit binary number, Y, simply by accumulating the pulse sequence over a suitable length of time in an initially cleared 8-bit counter. Thus, the output of the circuit is Y = min(i, i/n) × most recent non-zero value of X.

2.3.4 Future Work

In the next quarter the circuit for the T*-element will be tested as a whole when the card for it is laid out, following which a requisite number of them will be manufactured and put together to assemble the MEMORY of FROG.

D. Bose

2.4 INCOM (Project No. 46)

2.4.1 Introduction

INCOM (INterface COMputer) is a graphics processor which recognizes hand-drawn symbols. It consists mainly of three modules: the PREPROCESSOR, the FEATURE EXTRACTOR and the CLASSIFIER.

The PREPROCESSOR is the interface between the graphics terminal and the FEATURE EXTRACTOR. Its function is to normalize the binary image of the hand-drawn symbol and store the result in the image memory (IMAGE).

The FEATURE EXTRACTOR will apply several types of operators on IMAGE to extract the nodes (local features) from the binary image. These nodes are stored in the node memory (NODEM) before further processing. Another set of operators is applied on NODEM to merge superfluous nodes. The output of this module (FEATURE VECTOR) will be the local features with the interrelations between them.

The CLASSIFIER is the pluggable unit of the system. Each unit is designed to recognize a fixed set of symbols.

The CLASSIFIER will assign a unique character to the FEATURE VECTOR.

2.4.2 Project Status

Simulation programs have been run to test the validity of the operators involved. The results were successful. A new set of features were added to describe interrelations between the local features. These are included in the feature vector. The programs necessary to find these features were run successfully.

Another program was written to simulate the learning and recognition phases of INCOM. It accepts IMAGE and NODEM as its inputs and outputs a message. In this message it indicates if the character has been recognized correctly or not. In this program there is no a priori information stored. It learns from the input samples.

2.4.3 Future Work

All the simulation programs will be put together in one package to simulate the whole process of recognition.

Work is underway to find the minimum number of features required to classify a fixed set of symbols. Using these features the minimum number of classification rules are to be found.

The end result should be a package of simulation programs. The input to this package is the samples of a fixed set of symbols. The output is the minimum number of features and classification rules required to recognize this set.

Writing a detailed report on the work done on the project is underway. It will include descriptions of the simulation programs.

Mohamed El-Sonni

2.5 THESPIAC (Project No. 47)

During this quarter, minor changes were made in the operational specifications (and design) of THESPIAC. Prior to this quarter, THESPIAC's specifications called for a crossfader and from one to seven independent submasters. Consultation with lighting specialists outside the Krannert Center suggested that a more powerful control structure would result from subordinating the submasters to the crossfader. After evaluating this suggestion, it was found that it did provide a richer control capability and also simplified the operation of the system. Thus, the design and specifications were revised.

Most of the parts required for the reduced-size bench model were received during this quarter; printed circuit cards have been fabricated for console sections of the model and universal cards will be used for the remainder of the system. Actual construction will begin early next quarter.

Les Daley

3 SOFTWARE SYSTEMS RESEARCH

(Supported in part by the Atomic Energy Commission under grant No. US AEC AT11-1 2383)

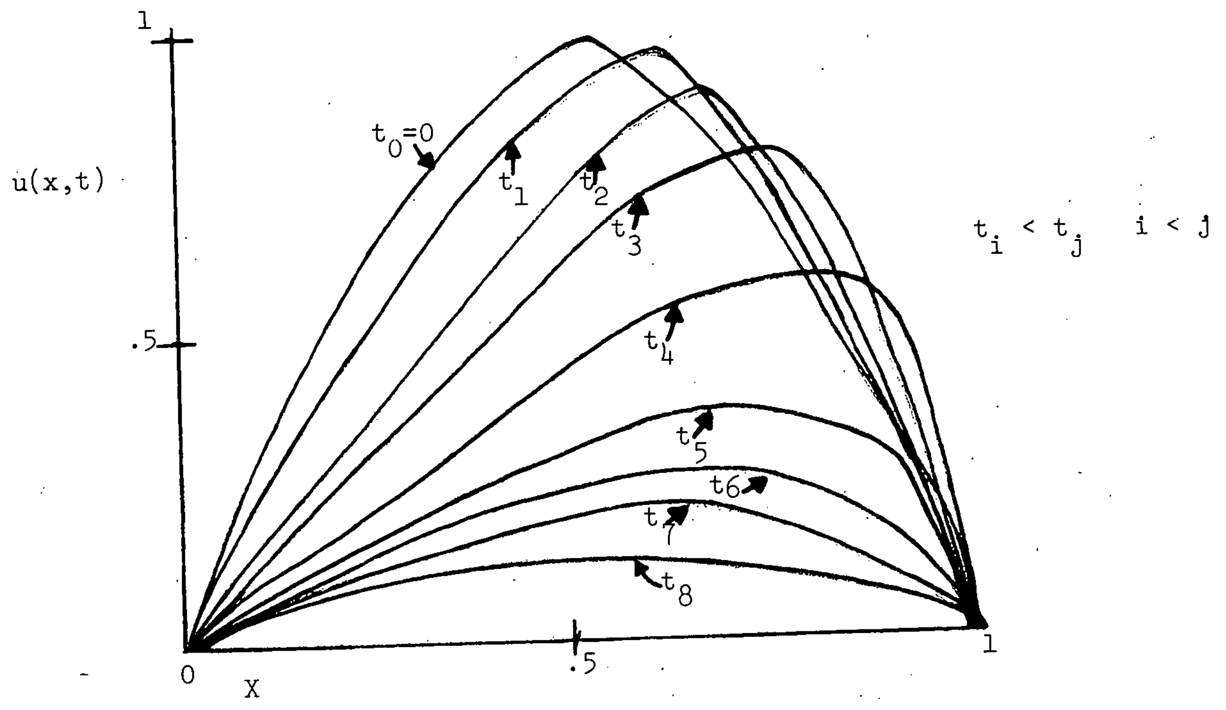
3.1 Numerical Processes

3.1.1 DIFSUB with partial differential equations (M. Ostrar)

Examination of the possibility of using time stretching with DIFSUB in the solution of partial differential equations was continued. A detailed examination of Burger's equation

$$u_t + uu_x = cu_{xx}$$

with $u(x,0) = \sin\pi x$ and $u(0,t) = u(1,t) = 0$ as initial and boundary conditions was begun. It is hoped that by studying this equation, a number of possibilities for suitable stretching functions might suggest themselves. The solution to the above equation decays with time and exhibits a wave-front to the right of $x = .5$ which steepens and moves to the right and then more slowly lessens and moves back toward $x = .5$. The sketch below illustrates this.



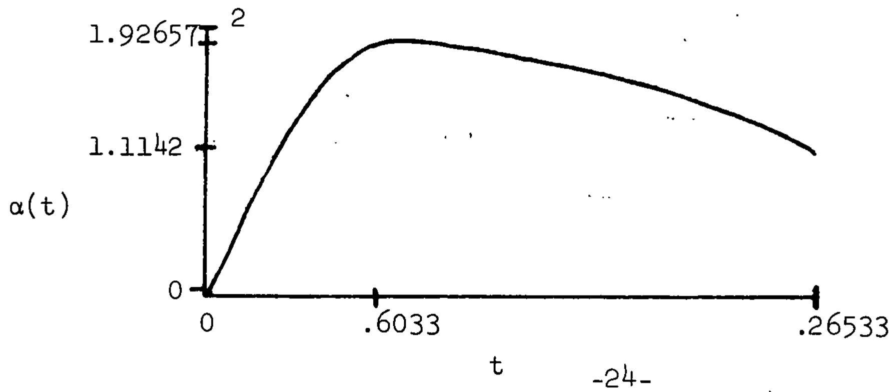
Plots of the solution and its various partial derivatives were made. Based on these plots, it was decided to try a stretching function of the form $x = \tanh(\alpha(t)y)/\tanh(\alpha t)$ where $\alpha(t) > 0$ for all t and $y_i - y_{i-1} = y_j - y_{j-1}$ for all i, j. This has the effect of making $x_i - x_{i-1}$ a monotonicly decreasing function of i. In short, the x axis is stretched apart near x = 0 and increasingly compressed as x increases.

Fixing λ at .05, a number of functions were tried for α. Results have been mixed and to some extent perplexing. No clear indication of which stretching functions to use has arisen. However, the results have yielded evidence that with the right choice of functions, time stretching can probably be an effective technique. Among the functions tried were:

$\alpha = k$ for k = .5, 1, 1.5, 1.92657, and 2.5: Under this circumstance where α is merely a constant, time stretching is equivalent to using a fixed, nonuniform mesh spacing. Results with k = 1 and k = 1.5 were quite good.

$\alpha = .1+(1-e^{-kt})1.82657$: Values fo 4 and 8 were tried for k. Integration up to t = 2 yielded fairly good results. At t = .2, the increase in accuracy when k was 4 was notable.

Several approaches were tried based on a stretching function which mapped the value of x for which u was maximum at a given time into y = .5. The true values of this function at various times were obtained by solving a nonlinear equation. A sketch of this function is shown below.



A quick and dirty model using a polynomial which was .1 at t = 0, 2 at t = .75, 1 at t = 2 and whose first derivative at t = .75 was zero yielded fairly good results. However, when I tried to interpolate using some of the actually known values, all of the polynomials which were obtained exhibited wild oscillations and crossed the t axis at least once. Since the stretching function involves 1/tanh(α(t), α too close to zero will cause problems.

A cubic spline was tried with abyssmally poor results. As these poor results may be due to lack of continuity beyond the second derivative, quintic splines were also tried using 6, 12, and 80 knots. Results were in general quite poor. This can be attributed at least in the case of the six knot spline (and probably the 12 knot spline too) to very large second and higher derivatives just beyond t = .8. Up until t = .2, the 6 knot spline was working fairly well.

The detailed results of the tests discussed above are given in the chart below. The columns headed "RATIO" contained the numbers $(KNT_M/KNT_S)^2$ ERR_M/ERR_S where KNT_M is the number of calls to DIFFUN using the particular time stretching method indicated and ERR_M is the maximum error at any of the mesh points inspected. KNT_S and ERR_S are the counterparts for the use of DIFSUB without time stretching. A RATIO less than 1 is a good result. In order to reduce the error by a factor of n^2, without time stretching, the number of mesh points would have to be increased by a factor of n, and the amounts of computation would increase by roughly the same factor.

t = .2			t = .4			t = .8			t = 2.			t = 3.			
KNT	MAX ERR	RATIO	KNT	MAX ERR	RATIO	KNT	MAX ERR	RATIO	KNT	MAX ERR	RATIO	KNT	MAX ERR	RATIO	Method
171	.7970-3		222	.6360-3		236	.6060-3		337	.8710-3	.819	384	.3410-3		No transformation
171	.4246-3	.532	222	.2190-2	3.443	236	.2190-2	.361	337	.7140-3	.497	384	.2590-3	.760	α =.5
171	.3380-3	.424	220	.7140-3	1.103	236	.2000-2	3.300	337	.4330-3	.359	386	.1210-3	.359	α =1.
171	.7560-3	.949	222	.5780-3	.909	238	.8300-3	1.393	339	.3090-3		386	.1540-3	.456	α =1.5
212	.6010-3	1.159	263	.4950-3	1.092	318	.7380-3	2.211	378	.3970-3	.573	427	.1550-3	.562	α =1.92657
212	.1720-2	3.317	263	.2410-2	5.318	357	.7150-3	2.700	457	.5110-3	1.078	506	.2310-3	1.176	α =2.5
177	.1590-3	.214	226	.4920-3	.802	320	.3970-3	1.204	419	.3750-3	.665				$\alpha=.1+(1-e^{-4t})1.82657$
265	.4120-3	1.241	316	.1960-3	.624	371	.5110-3	2.084	471	.3360-3	.753				$\alpha=.1+(1-e^{-8t})1.82657$
171	.6670-3	.837	220	.6740-3	1.041	433	.2740-3	1.522	1457	.2960-1	576.627				6 knot quintic spline
220	.6130-3	1.273	439	.5800-3	3.566	CORRECTOR FAILED TO CONVERGE									12 knot quintic spline
532	.1120-2	13.602	715	.8170-3	13.325	904	.6790-3	16.440	1005	.5850-3	5.966				80 knot quintic spline
575	.7310-3	10.371	721	.5170-3	8.574	991	.6590-3	19.175	1090	.4570-3	5.483				cubic spline
177	.2020-3	.272	269	.3800-3	.877	365	.5570-3	2.199	467	.2740-3	.603				crude polynomial

One problem with using tanh as the basis of the stretching function is that the x axis is increasingly stretched as x increases. This isn't really what is desired. The errors near x = 1 were drastically reduced by using tanh as a stretching function, but the worst errors then appeared for x between .4 and .8 (which is where the edge of the front in the solution is located). This suggests that letting $x_i = f_i(y_i, t)$ where the f_i's are not necessarily all the same, might correct such a situation.

3.1.2 Numerical Techniques for Periodic Solutions to Differential Equations (M. L. Schweitzer)

This quarter the main approach was to approximate the solution by a linear combination of continuous functions using quadrature to form the approximation. Both autonomous and non-autonomous systems of first order equations have been considered.

For the non-autonomous case let the differential equation be written as $\frac{dy(t)}{dt} = f(y(t),t)$. Given the period T, we seek $y_0(t)$ such that $y_0(t+T) = y_0(t)$. For a given set of functions $Q_i(t)$ defined over the interval [0,T], we wish to find α_6's such that $z(t) = \sum_{i=0}^{n} \alpha_i Q_i(t)$ minimizes $\frac{dz(t)}{dt} - f(z(t),t)$ in some sense.

In particular, we have used linear and cubic splines with n knots for the set of Q_i's. Using n+1 equally spaced knots t_i, i = 0,1,...,n where $t_0 = 0$ and $t_n = T$ we define the linear splines by

$$Q_i(t) = \begin{cases} \dfrac{t - t_{i-1}}{t_i - t_{i-1}} & t_{i-1} \le t \le t_i \quad 0 < i \le n \\ 0 & t_{i+1} \le t,\ t \le t_{i-1} \quad 0 < i < n \\ \dfrac{t - t_{i+1}}{t_i - t_{i+1}} & t_i \le t \le t_{i+1} \quad 0 \le i < n \end{cases}$$

we define the cubic splines as the basic splines $Q_i(t) = B_i(t)$, where

$$Q_i(t) = 0 \qquad t \le t_{i-2},\ t > t_{i+2} \qquad 2 \le i \le n-2$$

$$Q_i(t_i) = 1 \qquad 0 \le i \le n$$

and $Q_i(t_i)$ has first and second continuous derivatives at the knots t_{i-2}, t_{i-1}, t_i, t_{i+1} and t_{i+2}.

In order that $z(t)$ be periodic, α_0 must equal α_n. To find the α_i's we solve the n equations

$$I_i = \int_0^T \left[\frac{dz(t)}{dt} - f(z(t),t)\right] Q_i(t)dt = 0 \text{ for } i = 1,2,\ldots,n$$

Following are some results for the linear system

$$\frac{dy(t)}{dt} = \begin{bmatrix} -1 & 1 \\ 1 & -2 \end{bmatrix} y(t) + \begin{bmatrix} \text{sint} \\ 2(\text{cost}-\text{sint}) \end{bmatrix}, \quad T = 2\pi$$

which has solution

$$y_0(t) = \begin{bmatrix} \sin(t) \\ \cos(t) \end{bmatrix}$$

In the autonomous case, the differential equation is written as $\frac{dy(t)}{dt} = f(y(t))$. Since the period is generally unknown, one value of the solution was fixed and the period was allowed to vary. Again using linear and cubic splines, the α_i's were found by solving

$$I_i = \int_0^1 \left[\frac{dz(T\,\tau)}{d\tau} - T\cdot f(z(T\cdot\tau))\right] Q_i(T\cdot\tau)d\tau = 0$$

for $i = 1,2,\ldots,n$ by letting $\tau = t/T$.

The Van der Pol equation $\frac{d^2y(t)}{dt} - \mu(1-y(t)^2)\ \frac{dy(t)}{dt} + y(t) = 0$, was written as a first order system namely:

$$\frac{dy(t)}{dt} = \begin{bmatrix} y_1(t) \\ \mu(1-y_1(t)^2)y_2(t) - y_1(t) \end{bmatrix} \text{ where } y(t) = \begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix}.$$

This was then solved using various Van der Pol constants, μ, and fixing $y_1(t)$ at different values. Some results follow. The larger μ, the more nonlinear is the system and harder are the non-linear equations to solve. The period in all cases is the least accurate term. The calculated solution is very close to the solution found by using fourth order Runge-Kutta with a very small stepsize over the period calculated with the spline technique.

It is thought that using Hermite polynomials for the approximation may produce better results. This would require approximating $y_0(t)$ with $z(t) = \sum_{i=0}^{n} \alpha_i Q_i(t) + \beta_i \psi(t)$ where $Q_i(t)$ and $\psi_i(t)$ are the hermite interpolating polynomials over $[t_{i-1}, t_{i+1}]$.

Several integral equations can be used and are now being investigated.

Linear Splines

	4 knots	8 knots	16 knots
t	$y_1(t)$ $y_2(t)$		
$\frac{\pi}{2}$	1.25299249 .00027948	1.05400228 .00000067	1.01301139 +.00000000
π	.03896940 -1.23487053	-.00167882 -1.05316588	-.00010907 -1.01295687
$\frac{3\pi}{2}$	-1.25299249 .00027948	-1.05400228 .00000067	-1.01301139 .00000000
2π	.03896990 1.23487053	.00167882 1.05316588	.00010907 1.01295687

Cubic Splines

	4 knots	8 knots	16 knots
t	$y_1(t)$ $y_2(t)$		
$\frac{\pi}{2}$	1.01494334 .00000005	1.00061196 +.00000000	1.0003431 0.0
π	-.00043834 -1.01472439	-.00000236 -1.00061080	-.00000009 -1.00003427
$\frac{3}{2}\pi$	-1.01494334 -.00000005	-1.00061196 .00000000	-1.00003431 0.0
2π	.00043834 1.01472439	.00000236 1.00006108	.00000009 1.00003427

These results took around 3 iterations of Broyden's method to solve the n equations with I_i 10^{-13}.

Van der Pol constant = .01

	Period	Max. Amplitude of $y_1(t)$	Value of fixed point
exact value	6.28322457699	2.00000104165	
Cubic Spline 4 knots	6.27778309784	2.00000104165	2.00000104165
8 knots	6.28320017645	1.98758508495	1.41065954101
16 knots	6.28321578388	1.96284477662	1.41005954101
Linear Spline 4 knots	6.00722244795		
8 knots	6.26819870123	[fixed at exact max. amplitude]	
16 knots	6.282200996489		

Van der Pol constant = 2.0

	Period	Value of fixed point
exact value	7.62987448	
Cubic Spline 4 knots	7.06889452	1.2707293
8 knots	7.17395371	1.7401753
16 knots	7.21783722	2.0198138 (max. exact amplitude)

3.1.3 Higher Derivative Methods (R. L. Brown)

The higher derivative multivalue formulas reported in the two previous reports are defined by

$$0 = \sum_{i=0}^{k} \alpha_i^{(0)} y_{n-i} + \sum_{j=1}^{s} h^j \alpha_0^{(j)} f_n^{(j-1)}$$

for $s = 1$, $k = 1,2$;

$s = 2$, $k = 2,3,4$;

$s = 3$, $k = 4,5$;

$s = 4$, $k = 5,6,7,8$;

They have been incorporated in a variable order, variable stepsize predictor-corrector method defined by

$$\underline{y}_n = (y_n, y_{n-1}, \ldots, y_{n-k}, hf_n, \ldots hf_n^{(s-1)})^T$$

$$F(\underline{y}_n) = \sum_{i=0}^{k} \alpha_i^{(0)} y_{n-i} + \sum_{j=1}^{s} h^j \alpha_0^{(j)} f^{(j-1)}(y_n) \qquad (1)$$

$$\underline{y}_{n,(0)} = B\, \underline{y}_{n-1}$$

$$y_{n,(m+1)} = \underline{y}_{n,(m)} \quad \underline{e}_0 W^{-1} F(\underline{y}_{n,(m)})$$

$$+ \sum_{j=1}^{s} h^j \underline{e}_j (f^{(j-1)}(y_{n,(m)}) - f^{(j-1)}(y_{n,(m-1)})).$$

T is the transpose operator. B is a matrix representation of the predictor and will be defined below. W is an approximation of $\frac{dF}{d\,y_n}$ and causes Newton-like convergence of the iteration on $y_{n,(m)}$, the mth iterate of the approximation y_n to $y(t_n)$ where $y(t)$ solves

$$y' = f(y), \qquad (2)$$
$$y(t_0) = y_0$$

The vector $\underline{e}_i$ is the ith unit vector; all vectors are numbered from 0 to q.

By storing the variables in Nordsieck form as

$$\underline{a}_n = (y_n, hy'_n, \ldots, h^q y_n^{(q)}/q!)^T,$$

we get

$$\underline{a}_n = P \underline{a}_{n-1}$$

for

$$P_{ij} = \binom{j}{i}$$

where, for some non-singular matrix Q,

$$\underline{a}_n = Q \underline{y}_n.$$

Then we see that

$$B = Q^{-1} P Q$$

in (1), and the predictor-corrector method becomes

$$\underline{a}_{n,(0)} = P \underline{a}_{n-1},$$

$$\underline{a}_{n,(m+1)} = a_{n,(m)} - \underline{e}_0 W^{-1} G(\underline{a}_{n,(m)}) \qquad (3)$$

$$+ \sum_{j=1}^{s} h^j \underline{\ell}_j (f^{(j-1)}(y_{n,(m)}) - f^{(j-1)}(y_{n,(m-1)}))$$

for

$$G(\underline{a}_{n,(m)}) = F(Q^{-1} \underline{a}_{n,(m)})$$

and

$$\underline{\ell}_j = Q(\alpha_0^{(j)} \underline{e}_0 + \underline{e}_j).$$

This was incorporated in a FORTRAN subroutine D4 and tested on a set of stiff test problems due to Ehle ["A Comparison of Numerical Methods for Solving Certain Stiff Ordinary Differential Equations", University of Victoria, Department of Mathematics, report # 70, November, 1972]. The number of calls to DIFFUN, the differentiating subroutine that provides D4 with $h^j y^{(j)}/j!$, $j = 1,\dots,S$, is recorded for each test and appears in table 3.1. The number of calls to DIFFUN with S = 1 made by a recent version of DIFSUB (currently available in the FORTUOI library) is given for comparison. The 3 values given are for EPS = 1.E-3, 1.E-6, 1.E-9, for EPS the error per step allowed.

The distribution of the eigenvalues $\lambda_i(t)$ of the Jacobian matrix $\frac{df}{dy(t)}$ is an indication of stiffness. If all $\lambda_i(t)$ in the left complex half plane are contained in

$$S(\alpha) = \{z: \ |\arg(-z)| < \alpha\},$$

then DIFSUB, which is only stiffly stable at orders 3,4,5,and 6 should work well for $\alpha \doteq 0$, while D4, which is A-stable to order 11, should be competitive for $\alpha > 30^\circ$. In the test problems, $60^\circ > \alpha \geq 30^\circ$ for tests E,I, and J, and D4 is competitive with DIFSUB for these values. For F,G,H,K,L, and M, $\alpha \geq 60^\circ$, and D4 is competitive or better than DIFSUB for these tests, based on the number of calls to DIFFUN. Actually, when DIFFUN is called by D4 with S > 1, more actual computation is done than when DIFSUB calls DIFFUN with S = 1. Therefore, the results are only indicative of the power of D4 to solve such problems. A user version of D4 is currently being planned with particular attention to a simple calling sequence.

The test problems given by Ehle, are:

A. $\underline{y}' = \begin{bmatrix} -1 & 95 \\ -1 & -97 \end{bmatrix} \underline{y}, \ y(0) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \qquad t_f = 10.$

TEST	# CALLS: D4	# CALLS: DIFSUB
A	285, 775, 2318	122, 263, 380
B	79, 188, 322	124, 244, 333
C	462, 217, 407	92, 174, 224
D	243, 476, 1279	150, 354, 405
E	171, 294, 1255	173, 351, 538
F	478, 1204,3090	1756, 2164,3213
G	339, 806, 2003	3123, 5472,5354
H	446, 1199,3018	1150, 2232,3128
I	37, 28, 25	52, 48, 38
J	20, 25, 24	21, 21, 21
K	326, 233,7301	51, 312, 651
L	60, 75, 61	61, 71, 68
M	400, 257, 621	47, 121, 615

TABLE 3.1

B. $y_1' = (-1 + y_2^2)y_1 + (1 + y_2)y_2, \quad y_1(0) = -1,$

$y_2' = -y_1 + (-19 + y_1(2 + y_1))y_2, \quad y_2(0) = 1,$

$t_f = 16.$

C. $$\underline{y}' = \begin{bmatrix} -10^4 & 10^3 & 0 & 0 \\ -10^3 & -10^4 & 0 & 0 \\ 0 & 0 & -50 & 10 \\ 0 & 0 & -10 & 50 \end{bmatrix} \underline{y}, \quad \underline{y}(0) = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix},$$

$t_f = 3.$

D. $$\underline{y}' = \begin{bmatrix} 2-3Q & 4-4Q \\ 1.5Q-1.5 & 2Q-3 \end{bmatrix} \underline{y}, \quad \underline{y}(0) = \begin{bmatrix} 1 \\ 1 \end{bmatrix},$$

$Q = 10^5, \; t_f = 10.$

E. $$\underline{y}' = \begin{bmatrix} -10^4 & 9990 & 0 & 0 \\ -9990 & -10^4 & 0 & 0 \\ 0 & 0 & -.09 & .1 \\ 0 & 0 & -.1 & -.09 \end{bmatrix} \underline{y}, \quad \underline{y}(0) = \begin{bmatrix} 50 \\ 50 \\ 1 \\ 1 \end{bmatrix},$$

$t_f = 100$

F. $y_1' = 100(y_4 - y_2) - y_1 - 2(y_2 - y_4)^2$

$y_2' = y_1$

$y_3' = .01(100y_2 - y_4 + 2(y_2 - y_4)^2)$

$y_4' = y_3$

$$\underline{y}(0) = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \quad t_f = 64.$$

G. $\underline{y}' = \begin{bmatrix} -5 & -5 & -45.5 & 44.5 \\ -5 & -5 & -44.5 & 45.5 \\ 45.5 & 44.5 & -5 & 5 \\ -44.5 & -45.5 & 5 & -5 \end{bmatrix} \underline{y}$ $\qquad \underline{y}(0) = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$,

$t_f = 25.$

H. $y_1' = 100(y_4 - y_2) - y_1$

$y_2' = y_1$ $\qquad \underline{y}(0) = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$,

$y_3' = .01(100y_2 - 101y_4 - 10y_4^2)$

$y_4' = y_3$

$t_f = 64$

I. $\underline{y}' = B\,\underline{y} + f(t)$, $y(0) = [1 \quad 0 \quad 1 \quad 0]^T$

$B = \begin{bmatrix} -.1 & -.05 & 0 & 0 \\ .05 & -.1 & 0 & 0 \\ 0 & 0 & -100 & -100 \\ 0 & 0 & 100 & -100 \end{bmatrix}$, $\quad f(t) = \begin{bmatrix} 1 + .05t + 3t^2 + .05t^3 \\ -1 - .15t - 3t^2 - .15t^3 \\ 2t - 4t^3 \\ -2t - 200t^2 + 4t^3 + 200t^4 \end{bmatrix}$

$t_f = 100$

J. $\underline{y}' = UBU^T \underline{y} + U\underline{w}$, $y(0) = \begin{bmatrix} 0 \\ -2 \\ -1 \\ -1 \end{bmatrix}$, $U = \frac{1}{2} \begin{bmatrix} -1 & 1 & 1 & 1 \\ 1 & -1 & 1 & 1 \\ 1 & 1 & -1 & 1 \\ 1 & 1 & 1 & -1 \end{bmatrix}$,

$B = \begin{bmatrix} 10 & 10 & 0 & 0 \\ -10 & 10 & 0 & 0 \\ 0 & 0 & -1000 & 0 \\ 0 & 0 & 0 & -.001 \end{bmatrix}$, $\underline{w} = \begin{bmatrix} \frac{t_1^2 - t_2^2}{2} \\ t_1 t_2 \\ t_3{}^3 \\ t_4{}^2 \end{bmatrix}$ $\underline{t} = U\underline{y}$, $t_f = 100$

K. $\underline{y}' = UBU^T \underline{y} + U\ f(t)$, $\underline{y}(0) = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$,

$B = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & -100 & -900 \\ 0 & 0 & 900 & -100 \end{bmatrix}$, $f(t) = \begin{bmatrix} t^2 + 2t \\ t^2 - 2t \\ -800t + 1 \\ -1000t - 1 \end{bmatrix}$,

U as in J., $t_f = 25$.

L. $\underline{y}' = UBU^T \underline{y} + U\underline{w}$, $y(0) = \begin{bmatrix} 0 \\ -2 \\ -1 \\ -1 \end{bmatrix}$, $B = \begin{bmatrix} 10 & 100 & 0 & 0 \\ -100 & 10 & 0 & 0 \\ 0 & 0 & -100 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$,

U and $\underline{w}$ as in J., $t_f = 25$.

M. $\underline{y}' = A\underline{y} + f(t), \quad \underline{y}(0) = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \end{bmatrix},$

$$A = \begin{bmatrix} -100 & -1000 & 0 & 0 \\ 1000 & -100 & 0 & 0 \\ 0 & 0 & -.02 & 0 \\ 0 & 0 & 0 & -.01 \end{bmatrix},$$

$$f(t) = \begin{bmatrix} 100t^3 + 1000t^3/(1+t) - 3e^{-.01t} \\ -1000t^3 + 100t^3/(1+t) + (3t^2+2t^3)/(1+t)^2 \\ .02t^3 + 3t^2 \\ .01t^2 + 2t \end{bmatrix},$$

$t_f = 100.$

3.2 Graphical Support Programs

3.2.1 Plot Package; Automatic Knot Selection Algorithm (W. L. Chung)

Among the automatic knot selection algorithms tested in the last quarter, KNOT-D has been closely analyzed mainly because it gave reasonably good results. In other words, it is considered to be a satisfactory scheme in fulfilling the original objectives:

1) storage reduction,
2) computational efficiency,
3) shape preservation,
4) numerical accuracy

Of these, (1), (2), and (3) are rather general requirements which should be met for the description of line drawings. However, the need for numerical accuracy comes from the particular application that we want to work with.

So far, we have been interested in piecewise cubic polynomials to describe the curves. This piecewise cubic polynomial for an interval can be expressed in a convenient form:

$$p(x) = m_{j-1}\,\alpha(x) + m_j\,\beta(x) + y_{j-1}\,\gamma(x) + y_j\,\delta(x)$$

where

$$\alpha(x) = (x_j-x)^2(x-x_{j-1})/h_j^2$$

$$\beta(x) = (x-x_{j-1})^2(x_j-x)/h_j^2$$

$$\gamma(x) = (x_j-x)^2[2(x-x_{j-1}) + h_j]/h_j^3$$

$$\delta(x) = (x-x_{j-1})^2[2(x_j-x) + h_j]/h_j^3$$

If we fix the values y_{j-1} and y_j, as in KNOT-D, all we have to know to determine p(x) for each interval is m_j since the scheme is basically one-sided.

Therefore, it would be a reasonable approach to closely examine the behavior of m_j in order to get the idea of how the scheme will perform.

First of all, the error bound for m_j is obtained. And then the error bound for the interpolation scheme is obtained from this. These results are summarized in the following.

Lemma 1. Let $y \varepsilon C^4[I]$. Then for each knot x_j,

$$|y'_j - \bar{m}_j| \le \frac{1}{64} ||y^{(4)}|| h^3_{max}$$

where $\bar{m}_j$ is the value of m_j caluclated with $m_{j-1} = y'_{j-1}$.

Lemma 2. Let $y \varepsilon C^4[I]$. Then for each knot x_j,

$$|y'_j - m_j| \le 4|y'_j - \bar{m}_j| \le \frac{1}{16} ||y^{(4)}|| h^3_{max}$$

This lemma not only gives us the accuracy of m_j but also the stability of the scheme.

Lemma 3. For $y \varepsilon C^4[I]$,

$$||Hy^{(k)} - Py^{(k)}|| \le a_k ||y^{(4)}|| h^{4-k}_{max} \quad , \quad k = 0,1,2,3$$

where Hy is the 2-point Hermite interpolate of y, and $a_0 = \frac{1}{64}$, $a_1 = \frac{1}{16}$, $a_2 = \frac{3r}{8}$, $a_3 = \frac{3r}{4}$, and $r = h_{max}/h_{min}$.

From these lemmas and the triangular inequality

$$||y - Py|| \le ||y - Hy|| + ||Hy - Py||,$$

the theorem follows.

Theorem 1. Let $y \in C^4[I]$. Then

$$||y^{(k)} - Py^{(k)}|| \le M_k \, ||y^{(4)}|| \, h_{max}^{4-k}, \quad k = 0,1,2,3$$

where $M_0 = \frac{7}{384}$

$$M_1 = \frac{2\sqrt{3}+27}{532}$$

$$M_2 = \frac{2+9r}{24}$$

$$M_3 = \frac{2+3r}{4} .$$

It would be worthwhile to compare this bound with those of cubic spline interpolate which is a global scheme. That is,

$$||y - S|| \le \frac{5}{384} \, ||y^{(4)}|| \, h_{max}^4 .$$

In the new version of KNOT-D which takes input points one by one, YMAX is set to the maximum of y values so far instead of local maximum for each interval since the latter results in unreasonably small step size whenever y changes its sign.

This is now being tested with practical examples which are obtained from DIFSUB.

3.2.2 Operating Systems (L. Lopez)

The incorporation of the 2701 channel code into PLORTS together with a desire to allow batch operation of SPACT has caused the code formerly in SPACT to be partitioned:

SPACT

This module contains the routines to communicate to PLORTS read and write requests to the terminals. This module loads SPACTOBJ into core and calls it passing a parameter list whose format is as follows:

address	contents of word
R1 + 0	value of R1 on entry to SPACT
R1 + 4	address of READ routine
R1 + 8	address of WRITE routine
R1 + 12	address of PROMPT routine
R1 + 16	amount of core to be left free after swap core is getmained.

When SPACTOBJ returns: the interfaces to PLORTS are cleared and the message 'NORMAL END OF SPACT' is sent via a WTO macro.

Read routine

This routine initiates a read from the PDP-8. The format of the parameter list is as follows:

address	contents of word
R1 + 0	address of a word into which the routine will place the address of the ECB which will be posted upon completion of the read.

address)(con't.)	contents of word
R1 + 4	length in bytes of buffer
R1 + 8	address of buffer

Write routine

The routine initiates a write and waits for completion of this write.

address	contents of word
R1 + 0	address of write buffer

The length of the buffer is implicit in the record.

PROMPT routine

This routine sets the return code to "not equal" and returns. PROMPT is called when SPACTOBJ has nothing to do. A condition code of not-equal means that SPACTOBJ should do a WAIT. A condition code of equal means that READ ecb has been posted.

BATCH

This module provides the same parameter list as does SPACT to SPACTOBJ however the WRITE routine writes directly to the printer and the READ routine merely records the parameter list passed to it.

When SPACT has nothing to do it (as before) calls PROMT then performs the read from the card reader. PROMT then posts the read ECB and returns with a condition code of equal.

3.2.3 PLW (L. Lopez)

The experimental compiler PLW is up and running. Users, although few, have had fair success in using this compiler. Presently PLW is being recoded in PLW.

3.2.4 The Analysis System (L. Lopez)

The analysis system was converted this quarter. Three modules were created to facilitate conversion:

RUNITEM

All the programs in the system require the old SPACT's console vector (OCV). Any program started from the new SPACT is passed the new Spact's console vector (NCV). It also places the address of the NCV in the word before the OCV and does a XCTL8 to ITEM passing the OCV's address.

CRUDINIT

Any program which uses the storage management, I/O or program fetch facilities of SPACT must initialize the module WSVC by calling IN8 passing in R1 the NCV. CRUNINIT extracts the NCV from the OCV and passes the NCV to IN8.

CRUDLINK

This module does a link from the USER glist in the NCV to the CV in R1.

3.3 User Oriented Software

3.3.1 Network Analysis (T. Runge)

A PLW program to perfomr network analysis and replace the analysis package of ITEM, GLOBAL, WEED, and PARSE is being written. The program will take a user's definition of a network from cards and produce the output routines S1, S2, and DIFFUN for the numerical simulation of the network.

Network analysis consists of three process:

1) An input routine. This reads the cards, translates the information from a pseudo-PDP-8 format to an acceptable PLW format, and interprets the information to create PLW data structures representing the network.

2) A pre-end order traversal of the tree-like structure of pictures and subpicture instances which is the user's network. The traversal produces an intermediate representation of the network's equations and gathers information to be used to print a network variable map upon request.

3) The output routine builder. This processes the intermediate representation and determines the output classes to which the variables belong. It then constructs S1, S2, and DIFFUN from the network equations.

<u>Input Routine</u>

The user's definition of a network consists of terminal type definitions and a set of input blocks for each picture in the network. These are read one block at a time (the set of terminal type definitions is considered to be one block) and translated into PLW format as the block is interpreted.

From the terminal type definitions is built a Terminal Type Table which holds, for each type of terminal defined in the network:

1) The terminal type name

2) The number of and names of all E type variables defined on this terminal type

3) The number of an names of all I type variables defined on this terminal type

The input routine also builds an Input Location Array. This array holds, for each picture defined in the network, a set of pointers to the PLW data structures created from the input blocks for that picture. These input blocks are of five types:

1) Header block - this indicates the name of the picture being defined and signals that all other blocks for this picture follow in the input stream.

2) Terminal-Connection block - this indicates the terminal type names used locally in this picture, specifies connections of local terminals, and describes all terminals defined on this picture. This information is placed in a PLW Terminal-Connection block for this picture.

3) Subpicture block - this indicates the set of subpicture instances invoked by this picture and specifies the parameter equations and the type names of terminals which are internally connected to terminals on the subpicture for each subpicture instance. This information is placed in a PLW Subpicture block for this picture.

4) Declaration block - this lists all global and local variables defined on the picture. These are placed in a PLW Declaration block for the picture.

5) Parameter-Equation block - this lists all parameters, parameter equations and general equations for the picture. These are placed in a PLW Parameter-Equation block for the picture.

If any of blocks 2) through 5) is not present for a particular picture, the input routine builds a null block of that type for the picture. This simplifies the later processing of pictures.

The Input Routine is coded and is to be debugged in the immediate future.

Network Traversal

The network traversal produces a binary tree representation of the general tree corresponding to the user's network and maintains a stack which at any given time consists of entries for all picture instances which are ancestors of the picture being visited. The preorder visit of a picture is performed by the routine Visit Down and the endorder visit by the routine Visit Up. The traversal consists of these two routines and a driver to invoke them in the proper sequence.

<u>Visit Down</u>

The Visit Down routine builds the tree entry and stack entry for the picture being visited, and links the tree entry into the tree, and pushes the stack entry onto the stack.

Into the <u>tree entry</u> is placed the picture name, the tree links, the information for the network map (not all of which can be determined during Visit Down), and a symbol table containing all variables defined in the Declarations block for this picture.

Into the <u>stack entry</u> is placed a link to this picture's father's stack entry, a link to this picture's tree entry, a node table, an instance block, and an equations block.

The <u>node table</u> is constructed from the Terminal-Connection block for this picture and contains, for each terminal defined on the picture:

1) the terminal identification number

2) flags indicating whether this is an external, internal, or local terminal

3) (for external terminals only) a pointer to the node table entry of the externally connected terminal [in this picture's father's stack entry]

4) the terminal's type if it can be determined at this point

5) a ring link for building rings of connected terminals

Visit Down builds connection rings in the node table according to the local connections specified. All terminals on a common ring are assigned the same type (if any terminal on the ring had a type assigned to it in the input). Connections between terminals of different types are omitted with a warning message. This is also true of external connections and types are passed along <u>all</u> lines of connection between a terminal of unassigned type and one with a type assigned.

The <u>instance block</u> is constructed from the subpicture block for this picture and contains, for each subpicture instance invoked by this picture:

1) the subpicture name

2) the number of internal terminal connections from this picture to this subpicture and a pointer to the node table entries of the internal terminals

3) the number of parameters defined on the subpicture and a pointer to their symbol table entries - this information is passed up to this picture during the operation of Visit Down on the subpicture.

4) the non-default parameter equations for this subpicture instance's parameters

The <u>equation block</u> is constructed from the Parameter-Equation block for this picture and contains the default parameter equations and the non-parametric equations which are specified on this picture.

<u>Visit Up</u>

The intermediate representation of the equations specified on a picture is created by Visit Up on that picture.

As in Visit Down, terminal types are again passed along all lines of terminal connections. At this point the type of every terminal on this picture should be assigned. Using these types, E variable sets are assigned to rings of connected terminals. One E set is assigned to the entire ring due to the implicit equation E(A) = E(B) for conencted terminals A and B. These implicit equations are also used for external connections and the same E variable set is assigned to the externally connected terminal (ring).

I variable sets are assigned to each terminal. Then, using the implicit equation for current summing around a connected ring, we generate equations involving the I variables in the set for each ring. If there are K I variables in the set for a ring of given type and if there are i internal terminals $A_1, \ldots A_i$, and j external terminals, $B_1, \ldots B_j$, on the ring then the generated equations are

$$\sum_{t=1}^{i} I_1 (A_t) - \sum_{t=1}^{j} I_k (B_t') = 0$$

$$\vdots$$

$$\sum_{t=1}^{i} I_k (A_t) - \sum_{t=1}^{j} I_k (B_t') = 0$$

where the B_t's are the terminals externally connected to the B_ts.

These generated equations are placed directly into the intermediate representation. Then all parameter equations in the instance block and all equations in the equation block are checked for syntax errors and intermediate representations of them are created.

In the intermediate representation of an equation, each occurrence of a variable is replaced by a pair of symbol table pointers. The first points to the symbol table entry for the variable while the second points to the symbol table entry for a preceeding variable which is a coefficient of this variable occurrence (if there is any such). This structure is used in the determination of the output types of variables.

Visit Up discards all equations of only one or two variables and encodes these equations in an equivalence ring structure in the symbol table.

Equations of the form

'variable' = 'constant'

are replaced by marking the variable as set to a constant and linking it to a source representation of the constant. Then all occurrences of the variable will be replaced by an occurrence of the constant. An attempt to set a variable to a constant C_2 after the variable has already been set to another constant $C_1 \neq C_2$ will generate an error message and the second equation will be discarded.

Equations of the form

'variable 1' = ±'variable 2'

are replaced by joining the symbol table equivalence rings of the two variables. These rings have a sign attached to the ring links and the sign between 'variable 1' and 'variable 2' is as in the equation.

Then all occurrences of variables on a common ring will be replaced by occurrences of a single variable from the ring (with the proper sign inserted). An attempt to join two equivalence rings where each has some variable set to a constant and where the constants are not the same (with proper allowance for sign) will generate an error message and the equation will be discarded.

The network traversal is coded with the exception of the syntax checking of equations.

Output Routine Builder

The Output Routine Builder will consist of routine to step through the intermediate representation of the equations, gathering information about the interrelationships of variables until no more such information can be gained. At that point variables will have their output types assigned and the equations will be partitioned by these types into the outputs routines S1, S2, and DIFFUN.

The Output Routine Builder has not yet been coded.

3.4 Satellite Processor Software and Hardware

3.4.1 Using the GRASS Terminal Connection to PLORTS (R. Whyte)

As a result of the connection of GRASS terminals to PLORTS, truly interactive graphics are possible and have been realized in FORTRAN using subroutines in CALLCOM. To bring up PLORTS on the Grass terminal one loads the Grass system and enters REACT the 360/Communication program and types " on the PDP-8 console to start the monitor. The terminal now behaves like a PLORTS terminal.

When 32 lines have been displayed the screen will erase and the line will begin again at the top. However if the first line begins at the top of the screen, 32 lines will be displayed and the terminal will wait for the user to type carriage return before it erases the screen and displays another 32 lines. The top of the screen is reached by making a joystick hit above the Draw Area (which causes an exit from REACT), and then re-entering REACT.

II. Callcom

The PDP-8/360 Interactive Communications package is comprised of two programs: REACT, the generalized communications program segment which runs in the PDP-8, and Callcom, a group of Fortran subroutines which run in the 360 and are callable from Fortran programs running uder Call/OS Fortran.

Callcom provides the user with routines to perform basic input/output functions with the terminals, including capability to receive joystick hit coordinate information, to receive text lines typed by the user, and to send text lines and line blocks to the terminal.

3.4.1.1 REPLY

Input to a user program from the terminal is performed by calls to REPLY. When REPLY is called, the next input from the terminal, after being filtered*, is returned to the program, along with information as to input type and length.

The Fortran calling sequence is:

```
CALL REPLY(ITYPE,LEN,IDATA)
```

IDATA is an INTERGER*4 array with 18 elements. The type and data contained in the input are determined as follows:

* For filtering commands see "Using Callcom with Call/OS Fortran"

ITYPE	- 1	=	input was a joystick hit
	- 0	=	input was a user typed text line
LEN	-	=	# of elements of IDATA containing valid data. (i.e. if a 15 character text line were returned, LEN would equal 5; for a joystick hit, LEN would be 3)

IDATA

FOR A JOYSTICK HIT

IDATA(1)	=	x coordinate of joystick hit in SCREEN increments (0 to 1023)
(2)	=	y coordinate of joystick hit in SCREEN increments (0 to 799)
(3)	=	screen segment hit (1-9)
IDATA(1)	-	IDATA(LEN) contain the returned EBCDIC characters packed four per array element. If the total number of characters in the typed text line was not a multiple of four, IDATA(LEN) is padded with blanks to fill out the element.

Output to the terminal from the user program is performed by calls to LINE, TEXT, SEND, and ERASE.

3.4.1.2 <u>LINE</u>

Line allows the user to add a line to the current block. If the block was previously empty, a new one is initialized. The "beam" position is moved to the specified coordinates, drawing a line if specified.

The X and Y coordinates must be specified as integer within the range 0 - 1023 and 0 - 799 respectively.

When the current block contains 25 lines it is output to the terminal and a new block is initialized.

The FORTRAN calling sequence is:

```
CALL LINE (X,Y,INTEN)
```

where:

X = X coordinate of line end

Y = Y coordinate of line end

INTEN - 1 = draw line from current beam position to (X, Y)

- 0 = move beam without drawing to (X, Y)

3.4.1.3 LINED

Lined is similar to LINE, except X and Y represent displacements off of the current beam position. After a call the new x position will be x + X from this call.

3.4.1.4 SEND

Send outputs the current line block and reinitializes it. This is generally called when the user wishes to output the final line block which may contain less than 25 lines. The FORTRAN calling sequence is:

```
CALL SEND
```

3.4.1.5 TEXT

TEXT allows the user to display text lines at a particular X, Y location on the screen. The FORTRAN calling sequence is:

```
CALL TEXT(X,Y,ITEXT,IDIM,LEN)
```

X,Y - integer coordinates of the text, as in LINE

ITEXT - an INTEGER*4 array containing the text packed 4 characters per element

IDIM - dimension of ITEXT

LEN - the number of characters in ITEXT to be displayed

If X = Y = 0 and LEN > 0, the specified text appears in a new line immediately below the previous one. If X = Y = 0 and LEN < 0, ABS(LEN) is the number of characters and the characters are appended to the current line. If LEN = 0 a blank line is generated.

3.4.1.6 ERASE

ERASE causes the display screen to be erased. The FORTRAN calling sequence is:

```
CALL ERASE
```

3.4.1.7 Extended Input/Output Facilities

The standard READ/WRITE statements of CALL/OS FORTRAN can also be used, with unit designations of 6. After every 32 lines set by WRITE statements, the display screen will be erased and the subsequent lines will begin at the top.

3.4.1.8 Using Callcom with CALL/OS FORTRAN*

To append the Callcom subroutines to a Fortran program contained in a User PLORTS file the user must use the external copy command of PLORTS:

```
COPYE 0072.HASKIN.FILENAME
```

where

```
FILENAME = CO.LINE   for LINED and LINE and SEND
           CO.TEXT   for TEXT
           CO.REPLY  for REPLY
           CO.ERASE  for ERASE
```

this appends the subroutines to the user's program.

NOTE: LINE and TEXT use subroutines which are contained in CO.LINE. Therefore if the user wishes to use TEXT he must also copy LINE.

The user should then type #D BLANKS which commands the interface to ignore all one byte blocks which are sent. This is necessary since CALL/OS outputs blanks at line terminations and while this does not affect the picture it may cause the screen to be erased after 32 have been sent.

To initialize joystick hit filtering the user types # RESET to initialize the mask to accept all screen areas. To suppress a hit in a certain area the user types #AREA X where X is the screen segment to be filtered out. To retrieve a filtered segment type #D AREAX where X is the screen segment to be enabled. Finally the program is executed by typing FORTRAN TIME = XX where XX is the time limit (<99) in seconds the user wishes the program to run. At the end of this time CALL/OS will ask CONTINUE? A response of Y will extend the time while an N will terminate the run.

---

* Since CALL/OS does not support named COMMON statements it is necessary for a user program which contains a COMMON block to begin COMMON with a dummy area of 57 integer*4 locations to skip over COMMON used in CALLCOM routines.

Using Callcom with FORTRAN IV

If interaction is not desired the user may elect to run his program under hasp from PLORTS and send the output to a data file in his disc space. The picture may then be displayed using the LIST function of PLORTS.

The unit designations must be changed to 8 in all write statements and the carriage control character should be dropped. A data file must be opened in the user's program by

```
CALL OPEN(8,'FILNAM','OUTPUT',IC)
'FILNAM' = Name of file
'OUTPUT' = Type of file
 IC      = Test code
```

Also since FORTRAN IV does not initialize variables to zero, the variable INDEX in COMMON must be initialized to zero in the main program before the first call to the CALLCOM routines.

Callcom Program Implementation

Callcom consists of the 5 previously mentioned subroutines, two utility functions ICODE and LSPLIT, and an error exit routine ERCHEK.

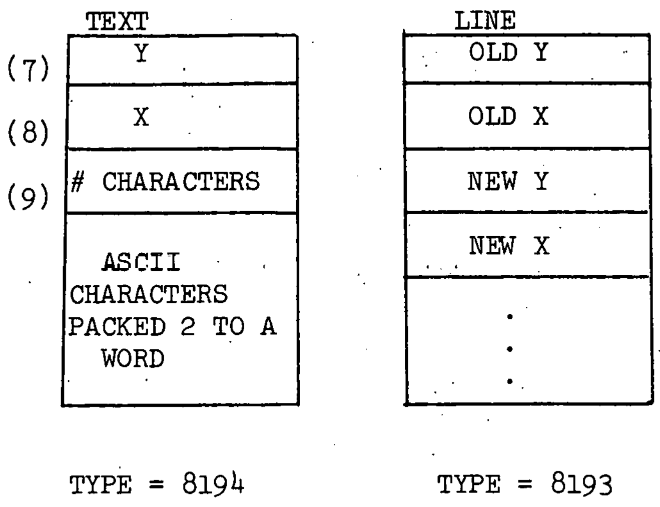
BLOCK STRUCTURE

Data is transferred in the form of LINE and TEXT blocks. A block is realized as an array of INTEGER*2 words the first 6 of which comprise the block header.

The block header has the form:

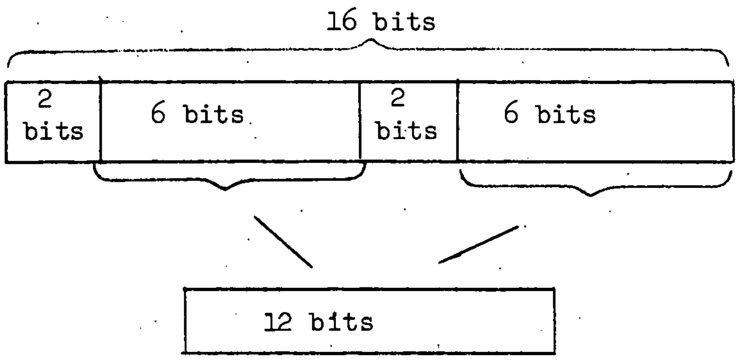
(1)	#B
(2)	LK
(3)	BLOCK TYPE
(4)	# WORDS - 1 IN REST OF BLOCK
(5)	POINTER TO END OF BLOCK
(6)	POINTER TO END OF ENTRY

This header is followed by the data which is of the form:



DATA TRANSMISSION

The 16-bit half words from the 360 are transformed to 12-bits for use by the PDP-8 in the following manner:



The function LSPLIT in Callcom performs the transformation from a standard half-word to the split form above for transmission to the PDP-8. In addition both 2-bit unused areas are set. This is necessary to assure that valid data will appear to be alphabetic as all output to the terminals is done in A-format. In a similar manner ICODE converts the split 16-bit word to a standard one.

Finally EDCDIC characters are converted via a table lookup by CNVARY.

3.4.2 COM10 - Interactive Communication - PDP-10 (R. Whyte)

The necessary software to allow communication between FORTRAN and MACRO programs running in the PDP-10 and the Grass terminals was implemented this quarter. The communication is accomplished through two main programs; COM8, a terminal monitor in the PDP-8, and COM10, a group of MACRO-10 routines callable from user programs running from the GRASS terminals.

The calling sequences for the routines, TEXT, LINE, REPLY, and ERASE from FORTRAN programs are the same as those formerly used in COMMUNE. The calling sequence from MACRO-10 programs is

```
     JSA A16, ROUTINE
       EXP P1
       EXP P2
        .
        .
        .
```

where A16 = register 16

ROUTINE = the desired routine

P1... = parameters for the call in the order they appear in a Fortran call.

(Refer to GRASS: Remote Facilities Guide Report # 466, pp. 3-8). A text line sent from the PDP-10 has the form #BLKT***<8 bit ASCII text> CARRIAGE RETURN where *** represents three 8 bit words containing the X and Y coordinates of the text line packed as follows:

$\underbrace{\text{YYYYYYYY}}_{\text{WORD 1}}$	$\underbrace{\text{XXXXYYYY}}_{\text{WORD 2}}$	$\underbrace{\text{XXXXXXXX}}_{\text{WORD 3}}$
Low order 8 bits of Y	Low order 4 bits of X & high order 4 bits of Y	high order 8 bits of X

A line has the similar form #BLKL***!!! where *** and !!! are respectively the initial X, Y and the final X, Y coordinates of the line. A joystick hit from the terminal is sent back to the PDP-10 as #BLKJ $X_1X_2Y_1Y_2$ where X_1X_2 are the high and low order 6 bits of the X coordinate of the joystick hit.

To create a module using COM10 the user must link the COM10 subroutines to his FORTRAN or MACRO program. This is easily accomplished by typing EXECUTE MYPROGRAM, COM10 from the terminal.

The PDP-8 side is brought up by loading ACID10 into the 8/I and COM8 in the 8. To load ACID10 refer to LD8I in the last quarterly. COM8 is loaded with the run command of OS/8. Once these programs are operating the GRASS terminal resembles a CRT console. It is necessary however to initialize the channel by typing SET TTY NO ECHO and SET TTY PAGE to suppress echo and allow control over the output stream from the 10. In addition when the screen is full it will 'wait' for the user to type CR before erasing and displaying more lines at the top of the screen. To send a control character it is necessary to hit the INTER key followed by the desired character. To send an ALT MODE one must hit INTER and then NOT ('¬').

This facility allows the user to create, edit, debug, and execute interactive programs on the PDP-10 from the GRASS terminals.

3.4.3 <u>Graphics Interpreter Language</u> (J. Stynes)

The Graphics Interpreter Language, which was originally presented in preliminary form in the Quarterly Report for October, November, and December of 1973, has been completed. A report will be published, under a separate cover, describing the instruction set, the reasons for choosing that particular set of instructions, and some examples of its usage.

3.4.3.1 Interpretive Graphix (N. Hennegan)

This past quarter the data structure management routines (here after called DSM), display routines, and the command decoder have been coded (and debugged to a reasonable extent).

The greatest problem encountered with the implementation of these routines was the setting of pointers to the current file, block, entry and text line--FP, BP, EP and LP respectively. When each interpreter instruction is executed, the assumed file is the file associated with FILNUM in the console vector. The assumed block is the block associated with the current block number in the file head of the current file. The assumed entry is the nth entry in the current block where n is in the entry number location of the current block header. Likewise, the assumed text line (if any) is the nth text line in the current entry where n is in the current text line number location in the block header. (See the sample file--figure)

At the time of execution it would be highly inefficient to recalculate each of these pointers every time they are used. To overcome this each time an interpreter instruction is executed which changes the assumed file, block, entry or line; or the locations of any of the said, the pointers are also changed. For example, if the instruction SETBN (set block number) is executed, the current block number is changed in the file head then the entry pointer EP and line pointer LP are set to the entry and line whose numbers are in the block head. The routines which calculate the pointers are SETFP, SETBP, SETEP and SETLP. Each of these routines call the routine one level lower such that in the example, only a call to SETEP is necessary to set the EP and LP.

In addition to instructions which change assumed sub-structures the absolute location of a file way change when returning from a wait since a new copy of the file may be swapped in. In this case the displacement of the old address of the file (FP in the C.V.) to the new address is added to each of the remaining pointers (BP, EP, and LP).

The code of the interpreter instructions is simple and well documented. The reader is urged to consult the listings. The basic core to most of the interpreter instructions are a few utility subprograms. A brief description of the major ones follow:

<u>SETFP</u> The file number is obtained from FILNUM in the C.V. A call is made to SETFIL which establishes the file in core and sets FP and CDFFBK (to 62X1--CDF!XO where X is the bank of the file) in the C.V. The contents of CDFFBK is placed in the routines TADATP and DCAATP for their use (see later).

<u>SETBP</u> The current block # in the file is obtained from the file head. FP + (BNOFST = address of block number. The address of the first entry is at FP + (FILHED. Each block's number at address of block +1 is then compared to the current block. When found the address is placed in BP and SETEP is called. If not found an error condition is signalled and control returns to the monitor. The displacement to the first text line (if present) within an entry is calculated from the # of interpreter words per entry and the number of XY quads per entry and is loaded into EHEADL in the C.V. Note--the displacement to the text coords is at BP + EHEADL - 2.

SETEP The entry number is found at BP + (ENDFST. The entries within the block are counted until the number is reached. If the entry number is 0, control returns to the calling program. EP is then set to the address of the entry and SETLP is called with a -1 in the AC. If the entry number exceeds the number of entries within the block, an error is signalled and control passes to the monitor.

SETLP This routine is very similar to SETEP. The text line number is at BP + (LNOFST in the block head. Because not every entry contains text or the same number of text lines, the line number in the block head may exceed the number of text lines within the new entry. If SETLP is called with a -1 in the AC and the number exceeds the number of text lines, the current line number is set to 0 and control returns to the calling program. If however the program is entered with a 0 in the AC, the calling routine was SETLN so that if the line number exceeds the number of text line, an error is signalled and control passes to the monitor.

TADATP (TAD at pointer) The address of a pointer is in the word following the call..

```
        JMS   TADATP
         TEMPTR
           .
           .
           .
```

The contents of the address pointed to by TEMPTR in the bank of current file is added into the AC. The change to the data field of the file bank is set by SETFP.

<u>DCAATP</u>	Similar to TADATP except that the contents of the AC is deposited into the address contained in TEMPTR.
<u>STKPOP</u>	The contents of the location pointed to by STKPTR is <u>added</u> to the AC and STKPTR is decremented by 1.
<u>STKPSH</u>	STKPTR is incremented and the AC is deposited into the location pointed to by STKPTR.
<u>STKGET</u>, <u>STKSTO</u>	(Stack get and stack store) Similar to the push and pop instructions except the level relative to the top of the stack is in the word following the call. The STKPTR remains the same.
<u>GETEHD</u>	(Get entry head length) The length of each entry head varies with each block. The entry head length (EHEADL in the C.V.) is equal to the number of interpreter words per entry plus four times the number of XY quads per entry plus one word for the displacement to end of entry plus two words for the text XY coordinates.
<u>ALOCAT</u>	This routine allocates or disallocates space within a file. IR is notified by DMSALO of the change in length of the file. The remainder of the file is moved either up or down depending upon whether or not space was to be added or deleted. Upon entry to ALOCAT the AC contains the number of words to be allocated (+) or deleted (-). The word following the call is the address after which space is to be added or the address before which space is to be deleted.

```
/SAMPLE FILE USED WITH ACID2       PAL8-V7               PAGE 1

                 /SAMPLE FILE USED WITH ACID2
                 /
         0000    *0000                               CORRESPONDS TO FILE NUMBER IN CONSOLE
                                                       VECTOR  (FILNUM)

00000    0165    FILE,    165                /DISP TO END OF FILE
00001    0000             0                  /CHECK SUM
00002    0000             0                  /TYPE
00003    0000             0                  /FRONT POINTER
00004    0000             0                  /REAR POINTER
00005    0000             0                  /ACTIVE FILE #
00006    0000             0                  /LAST REFERENCE
00007    0000             0                  /WRITE MODIFICATION FLAG
00010    0050             50                 /BLOCK NUMBER
00011    0077    BLOCK,   77                 /DISP TO END OF BLOCK
00012    0050             50                 /BLOCK NUMBER
00013    0001             1                  /ENTRY NUMBER
00014    0001             1                  /CURRENT TEXT LINE NUMBER
00015    0000             0                  /NUMBER OF INTERPRETOR WORDS/ENTRY
00016    0000             0                  /NUMBER OF XY QUADS/ENTRY
00017    0000             0                  /DY
00020    0000             0                  /DX
00021    0007    ENTRY,   7                  /DISP TO END OF ENTRY
00022    1040             1040               /Y COORD OF TEXT
00023    0532             532                /X COORD OF TEXT
00024    0001             1                  /CHAR COUNT
00025    1700             1700               /'0'
00026    0004             4                  /CHAR COUNT
00027    4011             4011               /' I'
00030    6061             6061               /'01'
00031    0007    ENTR2,   7                  /DISP TO END OF ENTRY
00032    0220             220
00033    0532             532
00034    0001             1
00035    1700             1700
00036    0004             4
00037    4011             4011
00040    6062             6062
00041    0007    ENTR3,   7
00042    1040             1040
00043    1352             1352
00044    0001             1
00045    1700             1700
00046    0004             4
00047    4011             4011
00050    6063             6063
00051    0007    ENTR4,   7
00052    0220             220
00053    1352             1352
00054    0001             1
00055    1700             1700
00056    0004             4
00057    4011             4011
00060    6064             6064
00061    0007    ENTR5,   7
```

MOVE Routine to move code within the bank of a file. Calling sequence:

```
        JMS  MOVE
          0            /From Addr
          0            /To Addr
          0            /# of words to move
```

If the to address is less than the from address, auto increment registers are used as pointers. But if not, normal core is used for the pointers since the pointers must be decremented.

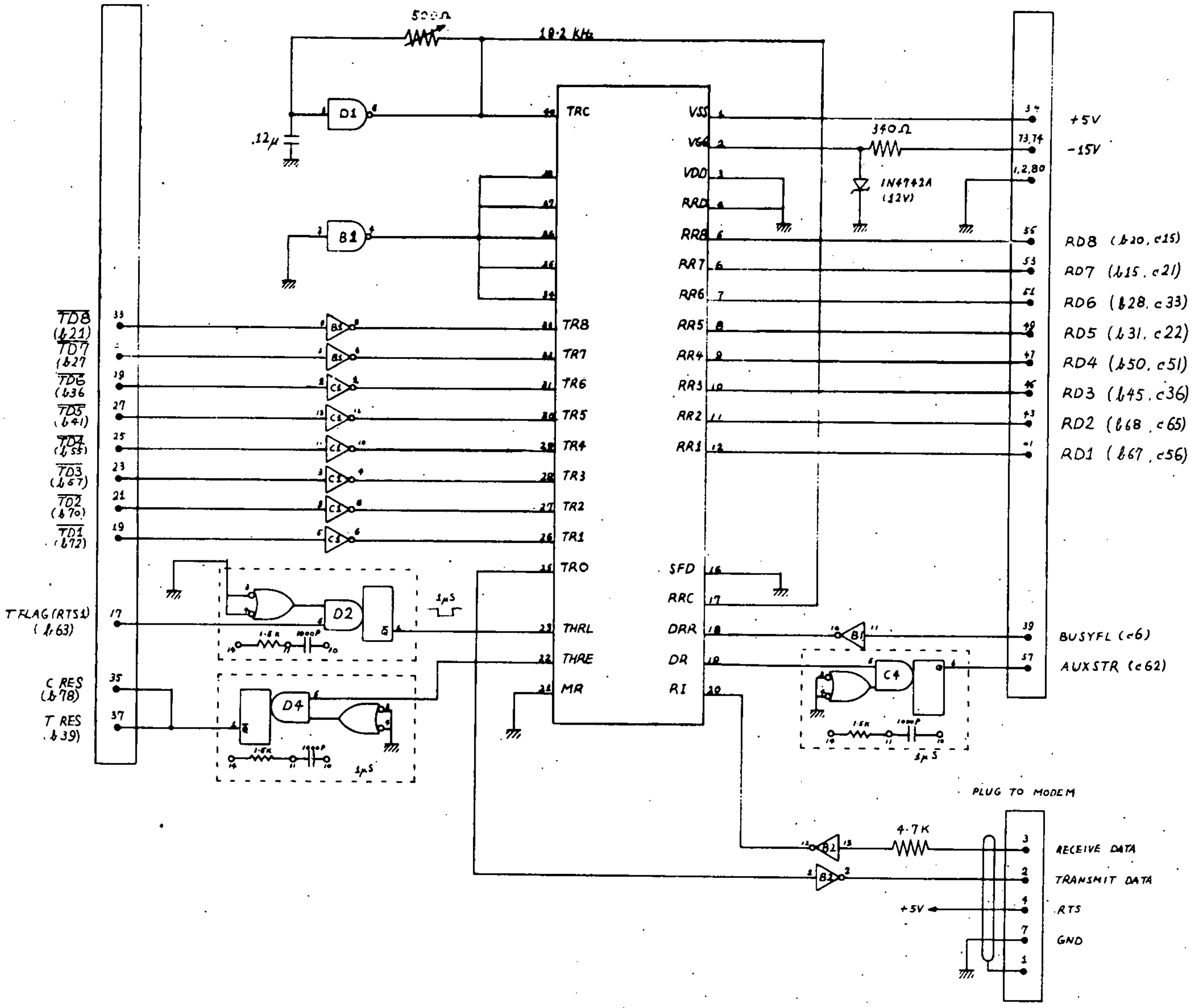
CNGDSE The subroutine is entered with the change in the displacement to the end of the entry in the AC. This change is added to the current displacement and CNGDSB is called with the change in the AC.

CNGDSB, CNGDSF Similar to CNGDSB except the change in the displacement to the end of the block and file are changed. CNGDSB calls CNGDSF so that by calling CNGDSE, all higher level displacements are likewise changed.

3.4.4 Serial Interface to the Computek Terminal (W. Tam)

The Computek terminal can now be connected to a 1200 baud modem via a serial interface. The interface is interchangeable with the existing interface between the PDP8/I and the Computek. This enables remote connection of the Computek terminal than telephone lines.

The interface board is designed to replace board g in the Computek during serial mode. An LS1 chip performs most of the serial/parallel conversions. One shot multivibrators are used to do logic level/pulse conversion for control signals.



	1	2	3	4	5	6	7	8
A								
B	7404	TR1802A						
C	7404			74121				
D	7413	74121		74121				

COMPUTEK BOARD 8

MODEM INTERFACE

4. SWITCHING THEORY AND LOGICAL DESIGN

(Supported in part by the National Science Foundation under Grant Number U.S. NSF-GJ-40221)

A report on minimal networks under general cost functions was prepared. The report is to show that conventional design objective, i.e., minimization of the number of gates first and the number of connections second would lead to a compact network implemented in integrated circuit in most cases of NOR networks.

Work on Tison's method and transduction of networks was continued.

S. Muroga

The study of discrepancies in certain computational results (statistics for obtaining optimal NOR networks with 3 external variables using both ILLIP with a feed-forward inequality formulation and a more specialized approach with an all-interconnection formulation) contained in various reports and papers published by this group was concluded. The study was written up as a DCL report which also contains a recent study of the effects of "chain-checking" and "additional inequalities" on the computation speed of ILLIP with the feed-forward inequality formulations.

Several of our transduction (<u>trans</u>formation and <u>reduction</u>) programs for NOR-gate networks were changed in preparing for the addition of the fan-in, fan-out restriction capability. At the same time, these programs were altered so that they may now handle 60-gate (actually the number of gates plus external variables is limited to 60) networks (previously, 40 gates was the maximum).

These programs were interfaced with Legge's program (changed from an 80-gate to a 60-gate capacity) which generates fan-in, fan-out restricted networks from unrestricted networks.

Storage areas in the various segments of each program were overlapped to save memory space - recently the most critical factor in obtaining faster turn-around time.

An initial consideration was made of the problem of modifying the transduction programs to allow them to handle incompletely specified functions. The required changes do not appear too difficult at this time.

(J.N. Culliney)

Over semester break I read and analyzed Theorie' des Consensus (part I) by Pierre Tison. Some theorems by D.M.Y. Chang and T.H. Mott, Jr. were then investigated, and an apparent error was uncovered in one of the proofs. Before continuing further investigation, I coded and debugged a program implementing Tison's algorithm for finding prime implicants. Currently, progress is being made in finding an efficient algorithm for multiplying large boolean functions in order to complete implementation of another of Tison's algorithms.

(R. Cutler)

An important application of the network transformation approach to logical design (see previous Quarterly Reports) is the synthesis of networks which satisfy fan-in and/or fan-out requirements. For his masters thesis Jeff Legge programmed several procedures which take a network and solve its fan-in and fan-out problems. Because of the existence of Legge's programs, the following approach was used for the fan-in, fan-out problem: obtain an initial network, apply Legge's programs, and then apply modified versions of the transformation procedures which maintain the fan-in, fan-out requirements. During this quarter a transformation procedure known as PROCII was modified so that it will maintain the fan-in, fan-out requirements of the networks it transforms. The modified version of PROCII is known as PRIIFF.

A number of experiments have been run on the computer to test the effectiveness of using PRIIFF in conjunction with Legge's programs. Not all of the results from these experiments have been tabulated, but preliminary results indicate that networks derived using the combination of Legge's programs and PRIIFF contain considerably fewer gates and connections than networks derived using Legge's programs alone.

(K. Hohulin)

In this quarter, I modified the Error-Compensation procedure (one of the network transformation methods) so that it can treat the problems with fan-in and/or fan-out restrictions. Obviously there are many different ways to do this, but the way currently used is the most convenient and straight forward one. For a given switching function, we begin with Legge's program (called JEFF) to get a fan-in and fan-out restricted network, then call the modified Error-Compensation procedure PROCCE trying to eliminate the redundancies as much as possible.

There are totally 13 subprograms in the original program PROCCE and 6 of them needed to be modified. After testing 30 5-variable functions, we found that the results by PROCCE are better than those by

PRIIFF and JEFF, but the average computation time is longer. This means that PROCCE is still the most powerful procedure among transformations methods if we do not care about the time used.

(J.K.C. Hu)

A report was prepared, discussing that all minimal networks under general cost functions can be obtained by solving integer programming problems. Minimal NOR networks under generalized cost functions for all three-variable functions and certain four-variable functions requiring 5 or fewer NOR gates are included in the report as examples.

The problem of minimizing the number of inverters in a feed-forward combinational network consisting of AND, OR and NOT gates was investigated. A writing on some interesting results related to this investigation is currently under preparation.

(H.C. Lai)

5. MACHINE AND SOFTWARE ORGANIZATION STUDIES

(Supported in part by the National Science Foundation under Grant Number US NSF GJ-36936

The following is a collection of related work aimed at improved designs for computer and software systems. We are interested in parallel and pipeline processors, small primary memories, effective use of rotating memories, and some questions concerning user languages for problems including typical FORTRAN type calculations, simulation languages, and a variety of file processing problems.

(D. Kuck)

5.1 PROGRAM ANALYSIS

5.1.1 Speedup of Iterative Programs - (S-C. Chen)

The theoretical and practical results of parallel computations of general linear recurrence systems and m-th order linear recurrence systems have been summarized in a report [1]. The studies of these results applied to standard numerical algorithms such as a direct Poisson equation solver have been conducted.

The work will continue to formalize these results as building blocks which can be used for more general problems of parallel computation.

5.1.2 IF Analysis - (R. Towle)

Work continued on algorithms to detect removable IF's in DO loops and algorithms to remove IF's from DO loops.

5.1.3 EISPACK Program Analysis

Automatic Array Operations Detection - (D. Y. Chang and K. Y. Wen)

Work was done on the recognition of various array operations. Assuming the dependency graph of an ordinary Fortran program is given, we can generate an algorithm to detect the types of array operations existing in the Fortran program and produce counts on the various operation types, such as those shown in Table 6 of Wen's thesis--"Machine Parameter Deduction by Program Analysis." Count results can eventually be used in a general purpose simulator to determine some parameters of a computer organization. This work will be continued in the next quarter in conjunction with some preliminary work on the simulator stated above.

5.1.4 COBOL Analysis - (R. Strebendt)

A number of COBOL programs have been analyzed. As a result of these analyses a number of parameters have been determined for the design of a machine for concurrently processing records. By the end of the next quarter it is expected that this investigation will be completed with the following work accomplished:

1) Compiler algorithms designed
2) Machine structure designed
3) Machine parameters determined
4) An estimate of the cost of the machine made
5) A comparison of the cost/performance ratio made with that of existing machines.

5.2 FILE PROCESSING

5.2.1 Information Retrieval Computers - (W. Stellhorn)

Initial design studies are complete for a parallel processing computer dedicated to information retrieval systems using inverted files. Systems of different sizes having from one to 512 parallel data paths have been examined with special emphasis on 16- and 256-path designs. It is concluded that all th hardware required for these two systems can be built and can be made to operate within the time constraints which have been established. A suitable head-per-track disk with parallel transmission capabilities and a fast, wide-band data memory have both been demonstrated in other systems; and all required components except the disk are available commercially at relatively low cost. Other mass storage devices with high transmission rates, including possibly a suitably modified movable head disk, may be substituted for the head-per-track unit.

In simulation tests with a large search involving the coordination of 70 terms and a total of 67,000 document identifiers, speed-up factors ranging from 12 to 60 have been observed for the new design relative to a conventional machine. Also, a large number of smaller searches can be processed simultaneously with very little degradation in the performance of the parallel machine.

Algorithmic studies are continuing in an effort to optimize the system and to develop a theoretical description of its performance.

5.2.2 Disk Routines - (K. Morgan)

Work continued on developing software for file handling and disk space allocation/deallocation. The disk space allocation/deallocation routines have been completed and I/O routines will be developed during the next quarter.

5.2.3 File Processing Software - (B. Hurley)

Work continued on the Information Retrieval Computer Simulator. Specifically, we developed algorithms used to select postings lists from a disk in an optimal manner with respect to the merge network which processes them. Also, the effect of delay times in the merge hardware upon the simulator were studied.

5.2.4 Inverted File Retrieval System - (M. Milner and J. Rinewalt)

During this quarter, design of experiments with users was begun. Initial experiments are planned for early next quarter but difficulty in obtaining copies of the original documents may cause a delay.

Work on the PDA has been delayed due to changes in 360/75 system software and delays in parts delivery. A device driver for our Centronics 306 printer was written, debugged, and installed. Several minor changes to DOS were made to allow dumps to the 306 printer, console inspection of device registers, and hexidecimal representation of quantities previously expressed in octal. This form is better for byte inspection because it is invariant with respect to the parity of the byte's address. An executive to manage the IR system has been designed and divided into phases. The first phase, currently being coded, will handle a single user on a single machine using 28K of core and measure various quantities of interest as the user works. Later phases will handle multiple users, control access to additional core beyond the 28K DEC and DOS support, and distribute processing over both PDP-11/40s.

5.2.5 System Clock - (P. Krabbe)

A Programmable Real-Time Clock has been designed similar to D.E.C.'s KW11-P. The principle difference between these clock modules is that the model made in house has 32-bit registers instead of 16.

Maintenance

Disk drive CDS114 #2875 malfunction trouble was a small piece of wire shorting out pins of backplane wiring.

Disk drive CDS114 #2874 malfunction trouble was a load NAND on PCB AL13 (Position B2 output 8).

5.2.6 Hardware Support - (J. Bengtson and L. Hollaar)

Work has proceeded on the construction of the memory management unit and various device interfaces for the PDP-11/40 system, slowed somewhat by delivery delays on some components. The Centronics printer has been interfaced, and is now in regular service, and the malfunctions in the Century 114 drives have been corrected.

The design of an experimental merge/coordination unit to operate with the PDP-11/40 has been started. This unit, when used in conjunction with the IR system software being developed, will be used to investigate the structure and performance of system hardware enhancements.

References

[1] S. C. Chen and D. J. Kuck, "Time and Parallel Processor Bounds for Linear Recurrence Systems," submitted for publication.

Publications

D. Kuck, P. Budnik, S-C. Chen, E. Davis, Jr., J. Han, P. Kraska, D. Lawrie, Y. Muraoka, R. Strebendt and R. Towle, "Measurements of Parallelism in Ordinary FORTRAN Programs," IEEE Computer, pp. 37-46, Jan. 1974.

6. NUMERICAL ANALYSIS

(Supported in part by the National Science Foundation under Grant Number U.S. NSF-GJ-36393)

During this quarter I collaborated with Y. J. Kim to prove that Kim's approximate factorization of an irregular finite element system is positive definite, a necessary and sufficient condition for the convergence of an adaptive chebyshev iteration for solving the system.

Paul Saylor

7. ASPECTS OF COMBINATORIAL COMPUTING

(Supported in part by the National Science Foundation under Grant No. US NSF GJ-41538)

Recent work includes study of scheduling algorithms, selection algorithms, backtrack programming, and various problems in combinatorics.

1. Scheduling Unit-Time Tasks with Limited Resources (A. C. Yao)

A set of tasks are to be scheduled on a computing system with s kinds of resources. Each task takes one unit of time to complete, and requires certain amounts of these resources. A schedule for the execution of these tasks is to be consistent with a prescribed partial ordering relation on the tasks, and the total demand for each kind of resource must not exceed a fixed amount at any instant. Several heuristic scheduling algorithms were studied in terms of their worst-case behavior.

Let ω be the total time for executing all the tasks according to a priority list, and ω_o be that according to an optimal schedule. It was shown that, independent of the number of processors, $\omega/\omega_o \leq cs + O(1)$, where c is a constant. Tighter bounds were obtained for the case in which the partial ordering relation is empty.

When preemption is allowed, it has been shown that $\omega_o(\text{non-preemptive})/\omega_o(\text{preemptive}) \leq cs$. Thus, the added freedom of preemption might not lead to a significant reduction in completion time when s is small. The situation is drastically different in the more general case in which the execution times of tasks are not uniform. In that case, we were able to construct examples with $s = 1$ and the ratio $\omega_o(\text{non-preemptive})/\omega_o(\text{preemptive})$ being essentially n, the number of processors in the system.

2. A New Scheduling Algorithm (N. F. Chen and C. L. Liu)

A new algorithm for scheduling jobs with equal execution time in multiprocessor computing systems was studied. This algorithm is called the level-and-count algorithm, or briefly LCA. We were able to prove:

1. The level-and-count algorithm produces optimal schedules for 2-processor systems.
2. The level-and-count algorithm generates a best "level schedule" for n-processor systems if the union of the sets of successors of any two jobs in one level contains all the jobs in the next level.
3. For 3-processor systems,

$$\frac{\omega_{LCA}}{\omega_{BLA}} \le \frac{5}{4}$$

$$\frac{\omega_{CGA}}{\omega_{BLA}} \le \frac{4}{3}$$

Where ω_{CGA}, ω_{BLA}, ω_{LCA} are the total execution times using the Coffman and Graham algorithm, a best level algorithm, and the level-and-count algorithm,respectively. Moreover, these bounds are best possible.

3. Scheduling in a Real-time Environment (S. Dhall and C. L. Liu)

The problem of scheduling jobs with "hard" deadlines in a real-time environment was studied. In particular, the case in which a multiprocessor computing system serves a certain number of jobs making periodic requests was investigated.

4. Selection Algorithms (J. A. Koch)

Research was carried out on improving the upper bound on the number of comparisons required to determine the t^{th} largest elements of n elements, $V_t(n)$. S. Lin's technique for lowering the bound of $V_3(2^k+2)$ was extended. For t odd (even, respectively) take t+1, (t+2), of the n elements and find the smallest one using t, (t+1), comparisons. Since this element is less than t, (t+1), elements, it cannot be one of the top t elements and can be ignored. Of the t, (t+1), comparisons used, the first $\frac{t-1}{2}$, $(\frac{t-2}{2})$, comparisons can be "reused" in finding $V_t(n-1)$. Thus, $V_t(n) \leq \lceil\frac{t+1}{2}\rceil + V_t(n-1)$ and for $n = 2_k + t-1$ there will be a saving of $\lceil\frac{t-1}{2}\rceil$ comparisons over tree selection.

Another method for selectively improving the upper bound was found which extends the following technique for lowering the bound of $V_3(n)$. Take two groups of 4 elements from the n elements and find the largest of each group. Suppose the two largest elements are a and e:

$$\begin{tikzcd}
b \arrow[r] & a & f \arrow[r] & e \\
c \arrow[u] & d \arrow[u] & g \arrow[u] & h \arrow[u]
\end{tikzcd}$$

Clearly, the smaller of the two elements b and f is less than three elements and cannot be one of the top three. Thus, this element together with c, d, g, h can be ignored. All but three of the comparisons made can be "reused" in the tree selection for $V_3(n-2)$. It follows that $V_3(n) \leq 3 + V_3(n-2)$ and the bound is lowered for $n = 2^k+2$ and 2^k+3. The extension is as follows: Let $x = \lfloor \log_2 n \rfloor$. For $n \leq 2\lfloor t+3/2 \rfloor$ take 2^x elements and eliminate 2^{x-2} of them from contention as members of the top t elements. This will cost $2^{x-2} + \lceil \frac{t-1}{2} \rceil$ comparisons that cannot be used in the tree selection for $V_t(n-2^{n-2})$. This method will lower the upper bound for $n = 2^k + t-1$, $2^k + t, \ldots, 2^k + t-2 + 2^{x-2}$. It is useful for small values of t since the minimum number of elements initially selected grows exponentially ($n \leq 2 \lfloor \frac{t+3}{2} \rfloor$). But it does yield a new upper bound,

$$V_3(n) \leq n-3 + \lceil \log_2 (n-1) \rceil + \lceil \log_2 (n-1 - 2^{\lceil \log_2 n \rceil - 2}) \rceil$$

which is less than all known upper bounds for certain values of n and equals the other bounds for the remaining values of n.

5. Speed of Selection Networks (F. F. Yao)

From a set of n elements, it is desired to find the t largest elements by means of a network built of comparators. An interesting question is to determine the minimal delay of such a network. This question was settled recently for small t and large n. However, the general question remains open. We obtained a lower bound $\log_2(t(1 - \frac{t}{n}))/(\log_2 3-1)$ on the number of delay levels. This bound is valid for all t and n. In particular, when $t = n/2$, the formula yields a lower bound $1.7 \log_2 n$ which shows that the simple information theoretic bound $\log_2 n$ is unattainable for finding the median of a set of elements. The possibility of constructing a fast median-finding network that achieves our lower bound is being explored.

6. Finding the Maxima of a Set of Vectors (F. F. Yao and A. C. Yao)

The computational complexity of the problem of finding the maxima of a set of n d-dimensional vectors was investigated.

1. It was established that exactly $S(n) + n-1$ comparisons are needed for the case $d = 2$ ($S(n)$ is the number of comparisons needed to sort n numbers).

2. For $d \leq \log_2 n$, it can be shown that at most $c_1(n^{2 - 1/d-2} \log n)$ comparisons are needed (c_1 is a constant). This implies that for $d \leq c_2 \log_2 n$ for some constant c_2, there are algorithms better than the simple-minded algorithm which compares every pair of vectors component by component.

7. A Postage Stamp Problem (A. C. Yao)

Suppose that N letters are to be mailed, and the amounts of postage requires for the N letters are 1¢, 2¢, ..., N¢, respectively. If a letter cannot use more than k stamps and the total postage used for each letter must be exact, what is the minimum number of different denominations we must have? Let $d(N,k)$ denote this number. It has been shown that $(k!)^{1/k} N^{1/k} \leq d(N,k) \leq kN^{1/k}$. Some initial data indicate that the exact value of $d(N,k)$ is closer to the lower bound than the upper bound shown above.

It seems to be difficult to evaluate the limit $\lim_{N\to\infty} \frac{d(N,k)}{N^{1/k}}$ (if the limit exists at all). Even for the case $k = 2$, little is known about the limit $\lim_{N\to\infty} \frac{d(N,2)}{\sqrt{N}}$, although some additional information on the behavior of the function $d(N,2)/\sqrt{N}$ is available.

8. Backtrack Programming (J. Bitner and E. M. Reingold)

During the last quarter, work has been carried out on the use of macros in backtrack programs. In specific, the queen's problem, a tiling problem with pentominos, and the problem of searching for optimal difference preserving codes were considered. The use of macros proved to be an effective technique in backtrack programming. As an example, a program using macros found all solutions to the queen's problem on a 15 x 15 chessboard in 25 minutes. This was significantly faster than an earlier program that did not use macros which took 3 hours to solve the problem.

9. Minimal Total Separating Systems (A. C. Yao)

A solution to a combinatorial problem due to Katona was obtained. Let S be an n-element set. We want to determine the smallest number $f(n)$ for which there exists a family of subsets of S, $\{A_1, A_2, \ldots, A_{f(n)}\}$, with the following property: Given any two elements $x, y \in S$ $(x \neq y)$, there exist k, ℓ such that $A_k \cap A_\ell = \emptyset$, and $x \in A_k$, $y \in A_\ell$. In particular, it is shown that $f(n) = 3 \log_3 n$ where n is a power of 3.

10. The Average Size Theorem (H. P. Tsao)

A theorem due to Kleitman and Milner, known as the Average Size Theorem, states that if $F \subseteq P(n)$ is a Sperner family with $|F| \geq \binom{n}{k}$ and $F \neq F_k$, $k < \lfloor \frac{n}{2} \rfloor$, then $|F|^{-1} \sum_{A \in F} |A| > k$, where $P(n)$ is the set of all subsets of an n-element set and F_k is the set of all subsets of size k in $P(n)$. A stronger version of the Average Size Theorem for a certain class of partially ordered sets including the lattice of subsets of a finite set , the lattice of subspaces of a finite vector space over a finite field, and the lattice of divisors of a positive integer, has been derived.

11. Graph Theory (P. Mateti)

A revision of the report "On finding all circuits of a graph" by P. Mateti and N. Deo, UIUCDCS-R-73-585, was completed.

8. THEORY OF DIGITAL COMPUTER ARITHMETIC

(Supported in part by the National Science Foundation under Grant No. US NSF GJ 38204.)

In this quarter, an algorithm was developed for on-line multiplication in a one-dimensional iterative array* of finite state machines. The algorithm has the feature that all the finite state machines (cells) including the initial cell are identical. A small note was prepared for publication and will be submitted soon. A more detailed technical report is being prepared and will be published in the next quarter.

(L. N. Goyal)

*A. J. Atrubin, "A One-dimensional Real-time Iterative Multipler," IEEE Trans. Elect. Comput., Vol. EC-14, pp. 394-399, June 1965.

9. ACSES --

The Automated Computer Science Education System

Supported by NSF under grant
EC-41511

During the first three months of the NSF contract period, our project to automate introductory computer science courses on PLATO has made a fair amount of progress. More specifically:

A. Library of lessons

The library has grown from about 50 to 60 lessons, and, more importantly, many of the lessons have been improved. In particular, the function of the keys that allow a student to control his path through a lesson have been standardized among all CS lessons, so that a student who has gone through a few lessons has become familiar with most of the control options in all lessons and can proceed through subsequent lessons faster. These conventions are described in a lesson "csauthors", (F. Izquierdo) and standard pieces of code are collected in lesson "cslibrary" (H. G. Friedman).

B. Language processors

A table-driven program entry, editing and syntax analysis program has been completed (T. R. Wilcox, Al Davis, M. Tindall). Tables for PL/1 are complete. Tables for FORTRAN, BASIC, COBOL, Snobol, APL, and MIX have been started.

The run time systems for each of these languages are different. The PL/1 run time system works for a subset of PL/1 suitable for the first half of one semester's instruction, and is being used this semester in one introductory course, CS 106. Run time systems for FORTRAN and BASIC are in progress.

C. Conversational information and advising system

The first half of the guide, the translator, has been completed as part of a Ph.D. thesis by Jean Pradels. A request in English is translated into an internal language and the latter is translated back into English, so that the user can judge for himself whether his request has been properly understood.

Planning for the second half of the guide, the request process is complete, and implementation has just begun (Dave Eland).

D. Communication system

Lessons cscomments, cstalk, and csnotes have been completed (H. G. Friedman). They allow communication between users and the management of the system in real-time (when a human monitor is present at a terminal), as well as communication with delayed response.

E. Automatic judging of student programs

Planning is essentially completed for two lessons that ask the student to write fairly sophisticated programs and attempt to judge these programs interactively, according to correctness as well as elegance (good structure).

One of these (Ron Danielson) asks the student to write a PL/1 program for symbolic differentiation; the other (Prabhaker Mateti) is intended to be able to judge sorting programs.

F. Reports

- J. L. Pradels, "The guide: an information system", Ph.D. thesis, UIUC DCS-R-74-626, March, 1974.

Ph.D. thesis proposals:

- Ron Danielson, "Application of artificial intelligence techniques to an automatic tutor of programming"
- Alan Davis, "Execution-time error analysis in a computer-assisted instruction environment"
- Dave Eland, "An information and advising system for an automated introductory computer science course"
- Prabhaker Mateti, "A sorting program verifier: A tutoring system for sorting programs"
- Michael Tindall, "Table-driven compiler error analysis"

G. New project: automation of exams

A program called "examadmin" has been written (Dave Eland) to handle all security and data collection aspects of exams given on PLATO.

An automated exam must be able to present such a great variety of individual problems that the only practical approach is to write problem generators, each of which is capable of producing many different problem instances. The difficulty lies in the requirement that these instances should not be too similar to each other.

Where the problems concern programming language questions, we want to use the syntax tables of the compiler and have table-driven problem generators capable of posing (and grading) problems about any of the programming languages for which we have tables. (Ben Barta, Fred Hansen, Francisco Izquierdo, Sylvian Ray, Larry Whitlock).

Goal for summer: a Fortran hour exam complete, so it can be used this fall in CS 103, when Rich Montanelli will carry out an experiment teaching Fortran on PLATO.

Jurg Nievergelt

10. MACHINE PERCEPTION AND INTELLIGENCE
(AQVAL Project)

1. Theory and Implementation of Variable-Valued Logic (Michalski, Yuan, Yip, Peterson, Hamilton)

The concepts of a symmetric selector and the exception and separation operations have been introduced and incorporated in the system VL_1. Michalski has written a paper 'VARIABLE-VALUED LOGIC: System VL_1' and submitted it for presentation at the 1974 International Symposium on Multiple-Valued Logic, West Virginia University, May 29-31, 1974. Yuan has debbuged the procedure for star generation for the uniclass cover algorithm. Peterson has been working on the development of an entirely new input data format for AQVAL/1-version 7, and new version 8 (which is under preparation by Kurt Hirchert). The new format will be much more user-oriented (e.g., will permit a user to use non-numerical names of object classes, $VL_\bullet$-variables, and values of the variables, as well). Yip has been working at the algorithm for VL_1 formula minimization.

2. Applications of VL Systems (Michalski, Yuan)

In cooperation with Dr. Barry Jacobson, University of Illinois, Department of Plant Pathology, there has been performed an experiment on the application of the AQVAL/1 program to the development of an automatic classification system for the diagnosis of 20 diseases of soybeans. Prelimir results are very encouraging.

√

11. MESH RESEARCH PROJECT

(Supported in part by the National Science Foundation under Grant No. US NSF GJ 36265.)

11.1 PASCAL Compiler

Progress on the PASCAL compiler for a minicomputer network included considerable refurbishing of and improvement on the bootstrap assembly language version. In addition to being more thoroughly debugged, a number of features were added to enable external procedures and improved file accessing, both of which are necessary in order to write the new compiler completely in PASCAL.

The lexical and syntactic analysis phases of the new PASCAL compiler were written and debugged and the semantic routines were modified for improved error handling. Certain major language modifications were agreed upon and incorporated in the code, including CASE and IF expressions, and fast procedure calls. In all cases, the modifications attempt to create a superset of the existing PASCAL without invalidating any of its earlier features.

Since it is envisaged that PASCAL is to be used to write operating systems, certain basic operations, like input/output would have to be handled, either by adding special constructs to PASCAL, or enabling calls to assembly language written routines. The latter has been made possible in a fashion which treats assembly language modules as external procedures.

(J. Krishnaswamy)

11.2 Microprogramming

In the last quarter the Lockheed SUE microinstruction simulator has been rewritten in PASCAL. This has been done for the following reasons:

1. To allow the simulator to run under batch on the ED-11 to allow classroom use of it.
2. To ease the modification and further debugging of the simulator.
3. To make the internal operation of the SUE processor, as represented by the simulator, more apparent through the use of structured programming in PASCAL.

The PASCAL version of the simulation is compatible with the assembler language version both with respect to user instructions and to internal and disk file formats with the following exceptions:

1. ↑<CR> has the effect as a line feed to the old simulator.
2. Interrupts through the keyboard to the user are not yet accepted.

Further modifications may be made in the next quarter, especially with respect to simulating another SUE processor and the writable control store hardware.

Also in the last quarter a survey of the architectural characteristics of minicomputers has been completed. The results of this survey indicate a trend towards more registers and fewer interrupt levels, while most other minicomputer parameters have remained the same over the past 8 years.

All in all, however, there seems to be a relatively small variation in these parameters among a large majority of the minicomputers surveyed. Thus no problem is foreseen in simulating the hardware of minicomputers in an unrestrictive range.

In the next quarter, an attempt will be made to find a transparent and structured method for defining the instruction set of a minicomputer. A general purpose emulator would therefore be in effect a compiler that would map this specification into an emulation program in microcode.

(S. Davidson)

11.3 Distributed Network Operating System

During the first part of this quarter, efforts were made to perfect the reliability of a multiprogrammed I/O monitor. This software provides foreground/background capability on the PDP-11/20. It forms the nucleus of a distributed software system on the experimental MESH network. Unfortunately these efforts were frustrated by apparently random hardware errors on the DEC UNIBUS which were not properly diagnosed until late April. Current efforts are devoted to testing the multiprogrammed monitor with the normal student job load. It is expected that the network version of the system will be perfected during the summer.

(G. Chesson and E. McClary)

11.4 Writable Control Store (WCS)

Printed circuit board wiring was completed on WCS this quarter. The following work remains to be done:

1. board checkout,
2. modification of Interconnecting Modules (ICM) such that boards may be connected,
3. modification of SUE processor (Lockheed still has not delivered our second processor).

11.5 System Clock

A Programmable Real-Time clock, similar to DEC's KW11-P (ours has 32 bit registers instead of 16 bit registers), has been designed. Fabrication is in progress.

11.6 Maintenance

Testers for the PDP-11 Address (M105) and Interrupt (M7820 and M7821) boards were designed and built this quarter. Most of the Address and Interrupt boards have been repaired. DIP switches have been installed on some boards to make their programming easier.

(P. Krabbe)

11.7 Bootstrap PASCAL and Run-Time Package

During the last quarter, the bootstrap version of the PASCAL compiler has been completed, and work is progressing on a compiler written in PASCAL itself. The new version will include the entire defined language PASCAL, together with the extensions already present in the bootstrap version, and is being designed so as to make future extensions extremely easy. The intention is, in part, to allow experimentation with language constructs to be as simple as possible.

Another design constraint is that the code-generation routines should be easily converted to generate code for other machines, such as the Lockheed SUE.

The bootstrap compiler has been frozen at its present level, so that work can progress on version 2.

The differences between the language accepted by the bootstrap version and the language defined in the PASCAL report are as follows:

1. Variables of type SET are not currently permitted.
2. Only FILES of type CHAR may be declared. However, a FILE may be declared as a BINARY FILE, in which case it may be treated as a FILE of INTEGER.
3. RECORDS may not include ARRAYS as subfields, and a declared TYPE may not include an ARRAY.
4. PROCEDURE/FUNCTION parameters are not yet implemented.
5. FUNCTIONS may return only simple variables.
6. The GOTO statement has been sub-divided into two forms. The first allows branches within the current block, and is invoked by

   GOTO <LABEL>;

   The second allows only branches out of the current block, to a declared label, and is invoked by

   EXIT <DECLARED-LABEL>;

This makes programs easier to comprehend, and removes some possible ambiquities.

7. POINTER types are not implemented.

8. PACKED ARRAYS are not explicitly implemented. However, if a variable's values fall in a sub-range of -128..127, the variable will be stored in a byte, so some packing is done implicity.

9. ARRAY and RECORD parameters may not be called by value.

10. PROCEDURE and FUNCTION parameters are not yet implemented, but it is expected that they will come up prior to version 2 of the compiler.

11. The TYPE 'REAL' is equivalent to INTEGER.

12. Any PROCEDURE may be declared 'FORWARD'. This allows mutual recursion of PROCEDURES. The parameters of the procedure must be declared at the first declaration of the procedure. If a PROCEDURE is declared FORWARD and not supplied, a runtime error is caused on the first attempt to execute it.

13. A PROCEDURE may be declared EXTERNAL. This implies that the body of the PROCEDURE is resident on disk, and should be loaded. This permits compilation of programs which are too large to be compiled as a whole. It also permits a program to overlay itself in a natural manner. Currently, an EXTERNAL PROCEDURE can only communicate with its caller through the parameters on the program statement.

The PROCEDURE QWERTYUIOP, declared EXTERNAL, will be searched for until the title of QWERTY.COD, under firstly the current user, and secondly [1,1]. This allows public program libraries to be set up.

E.g.,

```
PROGRAM MAIN
VAR
        I:INTEGER;
PROCEDURE QWERTYUIOP(ASD: INTEGER; VAR FGH: INTEGER);
EXTERNAL;
BEGIN
        QWERTYUIOP(4,I);
        WRITE(I,EOL);
END.
```

is a possible main program. If the program

```
PROGRAM Z(I: INT; VAR J: INT);
BEGIN
        J←I*I;
END.
```

is supplied on disk under the title QWERTYUIOP. COD. The output will be

```
   14
```

Notice that the parameters declared should correspond, in number, order, and type, but the names provided need not agree.

14. The CASE statement has been extended to allow 'ELSE' as a CASE selector. The statement after the 'ELSE' is executed if the case variable takes on none of the other case selectors.

15. The statement WRITE(X: 0), where X is an integer, causes X to be printed with no leading spaces. Thus

```
WRITE(2: 0, 4: 0)
```

causes output

```
 24
```

Further extensions, to permit octal and hexadecimal formats are being considered.

16. The WRITE and READ statements have been extended to allow them to apply to an arbitrary file. The syntax is

```
    write MYFILE(A,B,C);
```

17. The FILE declaration statement has been extended. The current syntax is

```
  ZXC:  FILE[<DIRECTION>, <FILETYPE>, <DEVICE>] OF CHAR;
```

The external name of this file will be <PROGRAMNAME>. ZXC.

The parameters take on the values

```
<DIRECTION> :  can take on values
        IN  - the file can only be used for input.
        OUT - the file can only be used for output.
        EXT - the file will be opened extend, if it exists,
              and output, if it does not already exist.

<FILETYPE>  :   can take values
        ASCII - the file is a DOS ASCII file.  Any integers
                transferred to/from it will be converted
                to/from ASCII.  The default files
                input/output are ASCII files.
       BINARY - the file is a DOS binary file.  It
                essentially consists of a bit stream.
                Reading/writing chars transfers a byte
                from/to the stream, while reading/writing
                integers transfers two bytes.
<DEVICE> : This field can take on the name of any available
           DOS device.  If a non-existent device is specified,
           a fatal error will be caused, and the program
           terminated.
```

18. The standard functions EXTEND(FILE) and CLOSE(FILE) have been added.

19. The assignment operator may be used inside an expression. Thus, A[I←I+1]←J←K←1; is legal.

20. A string is treated as a constant array, and may be passed as a parameter.

21. The assignment operator may operate on arrays. Thus

```
A: ARRAY [0..20] OF CHAR;
A←'THIS IS A STRING';
```

is a valid statement.

22. The READ/WRITE statements have been extended to allow specification of an array argument.

E.g., `VAR A: ARRAY [0..79] OF CHAR; B: ARRAY [0..10] OF INT;`
`READ(A,B);`

will read 80 characters from the input file into A, and will read the next 11 integers on the input file into B. If a character array is read from an ASCII file, the read is terminated by an EOL, or by the end of the array, whichever occurs first. Writing a character array onto an ASCII file is terminated by an EOL, FF, VT or any negative character.

23. It is possible to READ/WRITE to/from an ARRAY instead of a FILE. An ARRAY is assumed to be an ASCII FILE.

There are several routines in the runtime package for which suitable language constructs are not yet available. These include:

A. Core allocation/deallocation procedures, in readiness for the implementation of pointers.

B. A loader, currently used by external procedures, and by overlays in the runtime system itself. There will eventually be some type of construct to permit run-time correspondence between a PROCEDURE and a FILE.

C. A program may start up a procedure as an independent job, or as a dependent, asynchronous process. This is one form of allowing multi-tasking.

In addition to the work on PASCAL, DEC's MACRO ASSEMBLER has been modified so that the permanent symbol table may include registers, constants, and pre-digested macros. A PASCAL program has been written to take a macro library as input, and produce an object module suitable for linking to MACRO.OBJ as output.

(I. Stocks)

11.8 Deadlock Problems in Networks

In a large computer network in which processors have common access to resources, resource management becomes a significant problem. In particular, the problem of deadlock must be considered in the allocation of resources. Centralized resource management schemes impose an inherent size limitation on the system, for if the system is too large, the single resource manager becomes a bottleneck. Thus resource management in a large system must be distributed. The problems related to the distribution of resource management were studied and the results are the subject of a master's thesis (UIUCDCS-R-74-619 to appear July 1974).

The various types of centralized deadlock-related algorithms were surveyed in terms of their adaptability for distributed control. The main problems encountered in adapting these algorithms are related to the interference among processes performing resource management tasks simultaneously. Three distinct types of interference problems were identified and solutions were given. These solutions are demonstrated in some very fast distributed resource management procedures which were designed.

In the event that deadlock does occur in a distributed network, compatible recovery procedures must be available. Guidelines for the design of these recovery procedures were given with further examples.

Finally, since no single set of distributed resource management procedures can perform well in every system, a discussion is given on how to select the most cost-effective resource management scheme for any given system.

(T. Miller)

11.9 SUE Microassembler

The final draft of the SUE Microassembler language manual was completed and the language design was frozen. As for the coding of the microassembler in PASCAL, several major routines were rewritten entirely, and large blocks of code were broken down into procedures for modularity. The end result was a shorter, and hopefully faster program. As debugging of procedures continued, the program developed into two passes. When intermediate files came into existence in PASCAL, the debugging continued. Test programs have been set up. Maintenance manuals and other documentation are being completed. The microassembler is expected to be fully operational by the end of spring semester.

12. CENTER FOR ADVANCED COMPUTATION

(Supported in part by the Advanced Research Projects Agency under contract No. DAHC04 72-C-0001 and National Science Foundation grant No. GI-35179X)

REPORT SUMMARY

This is a progress report for the Center for Advanced Computation, University of Illinois at Urbana-Champaign. During this period there was research in the following areas:

1. Applied Mathematics
2. Pictorial Pattern Information Processing
3. Distributed Systems
4. Network Terminal Systems
5. Energy Research
6. Application Systems Group

12.1 APPLIED MATHEMATICS GROUP

12.1.1 Exponential Approximation

A document [1] was prepared describing work through October, 1973, on an algorithm involving differential corrections and linear programming. Suggestions were made in that document as to possible improvements in the algorithm. Some progress has been made in carrying out those suggestions, but program debugging is still necessary before extensive trials can be carried out and conclusions reached.

12.1.2 Large Matrix Problems

The spectral synthesis method [2] developed in collaboration with members of the Chemistry Department has been tried on large problems of physical importance. The matrices for which the generalized eigenvalue problem had to be solved were of order 256. A variation of the QZ algorithm developed by Sameh and Chang [3] was used and very good results were obtained, the largest residuals being of the order of 10^{-11}. Some thought is being given to iterative methods, which should be readily adaptable to parallelism and ILLIAC IV.

The Simultaneous Iteration Method [4, 5] for finding the leading eigenvalues and corresponding eigenvectors, has been generalized to obtain the eigenvalues of a large sparse symmetric matrix in any interval $[a, b]$, [6]. The new method is quite suitable for dealing with such sparse matrices on a parallel computer. The algorithm has been adequately tested on a serial machine and we are in the process of writing an ILLIAC IV program.

REFERENCES

1. G. Belford and J. F. Burkhalter, "A Differential Correction Algorithm for Exponential Curve Fitting," CAC Document No. 92, November, 1973.

2. G. Belford, R. L. Belford, and J. F. Burkhalter, "Eigenfields: A Practical Direct Calculation of Resonance Fields and Intensities for Field-swept Fixed-frequency Spectrometers," J. Magnetic Resonance, v. 11 (1973), pp. 251 - 265.

3. C. Chang, "An Algorithm for the Symmetric Generalized Eigenvalue Problem $Ax = \lambda Bx$," Master's Thesis, Department of Computer Science, University of Illinois, Urbana, Illinois.

4. H. Rutishauser, "Computational Aspects of F. L. Bauer's Simultaneous Iteration Method," Num. Math., 13 (1969), pp. 4 - 13.

5. H. Rutishauser, "Simultaneous Iteration Method for Symmetric Matrices," Num. Math., 16 (1970), pp. 205 - 223.

6. A. Sameh, J. Lermit, and K. Noh, "On the Intermediate Eigenvalues of Symmetric Sparse Matrices," CAC Document No. 91, Center for Advanced Computation, University of Illinois, Urbana, Illinois 61801.

12.2 PICTORIAL PATTERN INFORMATION PROCESSING

12.2.1 Introduction

With continued support from ARPA, NASA and USGS, development of ARPA Network and ILLIAC IV multispectral image processing facilities continues. At the request of Information Sciences Institute (ISI) the interactive TENEX multispectral image editing system has now been moved from ISI to Bolt, Beranek & Newman (BBN) where it will be generally accessible to ARPA Network users at BBN commercial rates. The Statistical Reporting Service (SRS) of USDA plans to use this system for editing and small-scale analysis of ERTS data corresponding to ground truth areas over two states. The ARPA-supported image modeling group of Purdue will also be using this software to achieve immediate ARPA Net image processing capabilities.

12.2.2 ILLIAC IV Pattern Information Processing

Both multivariate cluster analysis and statistical classification ASK algorithms are now operational on ILLIAC IV for large-scale analysis of ERTS multispectral imagery. Direct comparisons between the ILLIAC IV and the IBM 360/67 of LARS/Purdue indicate processing speed ratios between two and three orders of magnitude for these two algorithms. The availability of instruction overlap on ILLIAC IV and more comprehensive data management systems will greatly improve existing ILLIAC IV processing rates and throughput capabilities. NASA support has been secured for development of ERTS data management and editing on the IBM 360/67 at Ames Research Center.

12.2.3 Image Processing Graphics Support

Software has been developed for plotting in color interpreted pictorial information using the Zeta plotter. ERTS data interpretations can now be plotted as colored resource maps equivalent with respect to scale and geography to USGS 7½' quadrangles. CAC Imlac image processing software will be shared with NASA/Ames and Purdue. Extensive use of the DICOMED film scanning output device at NASA/Ames is being planned.

12.2.4 Pattern Information Processing Algorithms

Research continues toward the development of pattern information correlation algorithms. Current work involves the development of efficient procedures for determining maximum-entropy minimum-energy information distributions within networks. The MEME pattern correlation coefficient is being researched as a mathematical basis for stereopsis. A report in this area is forthcoming.

12.3 DISTRIBUTED SYSTEMS

12.3.1 Introduction

The Distributed System Group conducts research in distributed computational systems of heterogeneous computers. The principal effort in the six month period has been directed to outline the scope of the problem and to determine the range of costs and benefits accrued by a single software environment that spans several computers. Preliminary results indicate that a large class of systems can be implemented as distributed systems on dissimilar computers and have higher performance, higher reliability and lower cost than would be achievable on a single computational facility.

12.3.2 PL/1 Compatibility

The PL/1 compatibility study was completed in December of 1973 with the exception of the study of I/O compatibility. It appears reasonable to use PL/1 as a compatible language for distributed systems. PL/1 multi-tasking and compile time facilities are currently planned to be dropped from the final PL/1 ANSI standard. Therefore, the incompatibility of those facilities, as they currently exist on different vendor equipment, becomes a moot question. At the beginning of the project, PL/1 was considered as a good general choice due to its availability on Multics, IBM, and Burroughs equipment. Since then, PL/1 compilers have become available on or will shortly become available for Honeywell GCOS, UNIVAC, Control Data, and Data General Equipment.

12.3.3 Multi-machine Education

An education effort was undertaken to improve the fluency of distributed system group programmers in the use of Multics, the B6700, TSO on OS/360, and TSS on the 360/67. That program has been completed. The programmers are fluent in the generation of software on each of those systems at the level of interaction with the network control program (when such interactions are permitted).

12.3.4 Network Protocols

The study of process-to-process protocols that live on top of standard ARPA network host-to-host protocols was begun this period. This study was conceptualized as an investigation of process control and data transfer protocols. It has evolved into an identification of the need for network-wide utilities and the provision of protocols to support those utilities.

The concept of a data management protocol to study a resilient process-to-process protocol has been chosen. The data management protocol under study would be data management system independent. It would recognize the difference between the transmission of update requests vs. the transmission of read requests. The protocol may be multi-level and take into account that parts of it might be executed in a communications front-end, like ANTS, a large host, or even an intelligent local terminal with an imbedded micro-processor.

12.3.5 Network Economics

A study of the performance and economic implications of distributed systems was begun and some preliminary results generated.

These results indicate that, for certain tasks up to 2 orders of magnitude, cost and performance improvement can be achieved by the correct choice of machine on the ARPA network. Furthermore, some machines which are the very best at some task (e.g., the 360/91 at number crunching) are the worst at other tasks (e.g., the 360/91 at data management). Distributed computing seems to be feasible for surprisingly small tasks.

A program of benchmarking various machines on the network has begun. The purpose of the program is to quantify the effectiveness of each of the hosts in a full cost recovery heterogeneous environment. A detailed set of numerical benchmarks has already been run and some preliminary data management benchmarks have been generated to produce the conclusions in the first paragraph of this section. More detailed benchmarks in the areas of number crunching, bits and character flogging, file flogging, and console management are nearly complete.

12.3.6 Network Consultation and Assistance

Due to its unique experience with distributing computing and expertise on a multiplicity of ARPA network computational facilities, the Center for Advanced Computation has been frequently called upon to provide general consultation and assistance to other ARPA contractors and Department of Defense agencies who wish to use the ARPA network in a more production oriented mode. In particular, the Center has provided extensive assistance to the ARPA Biocybernetics project to ease their use of the Multics based Consistent System and to link Consistent System capabilities with the BIOMED statistical package available at the

CCN 360/91. The Center has also consulted with the Joint Technical Support Activity on a set of WWMCCS related problems. Of particular interest have been the provision of distributed data management facilities in a hostile environment.

12.4 NETWORK ACCESS SYSTEMS

12.4.1 Introduction

The Network Terminal Systems (NTS) Project in the Center for Advanced Computation has the following objectives: (1) Design and Development of the ANTS, MARK II system, (2) Support of selected ARPA contractors in the installation of ANTS systems, (3) Participation in network protocol development, (4) Graphics support for programs in the CAC and Laboratory for Atmospheric Research, and (5) Operation of the University of Illinois ANTS system.

12.4.2 ANTS, MARK II

During the first half of the six-month period, coding of level 0 and 1, and checking out of all level 0 and a few level 2 functions was completed. In the second half of this period, coding of level 2 was also completed, including the terminal handler and NCP, as well as TELNET. These, along with level 1 modules, were in a checkout phase at the end of this period.

The user TELNET version of ANTS, MARK II is expected to be making connections to the network by mid-April with server TELNET providing login capability from the network by early May.

A debugging system was developed which consists of two facilities: a run-time interactive debugger similar in function to DEC DDT, and a disk-image patcher which runs independently of ANTS. The system has facilitated the debugging process and greatly reduced the number of required recompilations.

12.4.3 ANTS Steering Committee

The ARPA-IPT office established an ANTS Steering Committee to help guide the ANTS, MARK II development and assess its suitability as a network access mechanism. In January of 1974, the ANTS Steering Committee met with the members of the NTS staff for three days of technical discussions and planning sessions. The members of the Steering Committee are Ken Pogran, Jerry Burchfiel, Tom Boynton, and Dave Crocker.

A part of the technical discussion was centered around the ANTS, MARK II system design and use of the PEESPOL compiler for implementing site-specific applications on the PDP-11. The consensus of the Steering Committee was that "ANTS is an excellent general purpose network access device, and that it offers a good base upon which to build specialized network access facilities." It was also generally agreed that "the development of PEESPOL had contributed greatly to the cleanliness and extensibility of the ANTS system."

Another topic of the technical discussion was directed at establishing a set of general specifications for ANTS. The specifications call for individual ANTS systems to be tailored according to the hardware on the system and the software modules desired. Obviously, for any given system, the software which can be included depends on the hardware available. The specifications outline the user-level features available in ANTS, MARK II, including TIP-emulation, command abbreviation/completion, rich terminal support and other generally friendly features.

The determination of priorities by the ANTS Steering Committee was very closely tied to the technical discussion. The outcome of those sessions was a plan for the production of the initial working version of ANTS, MARK II and an ordering of priorities for subsequent required features. These plans called for the installation of the first version to be installed at UCLA-NMC in April supporting basic TELNET. In terms of user-level features, this first installation will support:

command abbreviations;
character and line modes;
line mode input editing;
line feeds inserted after carriage control functions;
variable timing delay for terminal carriage control functions;
diversion of terminal output to the printer; and
multiple connections per terminal.

Activities which we are trying to complete in the current fiscal year include:

user and server FTP;
NETRJE;
device drivers for the DH11, RK11, DL11, and DC11;
documentation; and
extensions to PEESPOL to support separate module compilation and linkage.

Other activities, to be jointly supported by ARPA, Lincoln Labs, and NASA Ames include: device drivers for the MT11, DT11, PC11, and possibly a LP11. These are expected to be completed during the summer.

The ANTS Steering Committee strongly recommended to ARPA that continued support be given to such activities as fancy terminal support, command completion, and extended TELNET; documentation and user support, changes and future developments to network protocols, ANTS reliability enhancements, maintenance and slight improvements to the PEESPOL compiler, system measurements, and various cooperative efforts with other network activities.

12.4.4 ANTS, MARK I

During the months of December, January, and February, an effort was launched to make a number of improvements and enhancements to the current MARK I prototype ANTS system. This system was then installed at a number of sites on the network as an interim measure to provide network access before the initial delivery of MARK II.

A separate information bulletin has been sent to all sites receiving MARK I installation detailing the differences between the latest version and previous versions. In addition, a user guide has been written detailing the current commands and available capabilities of MARK I ANTS. It was distributed the week of March 4, 1974.

The following sites on the network are running the MARK I system:

Center for Advanced Computation - University of Illinois
Network Terminal Systems Group - University of Illinois

Army Materiel Command - Ft. Belvoir, Virginia
Ballistic Research Laboratory - Aberdeen Proving Grounds, Maryland
Network Measurement Center - University of California, Los Angeles
Lawrence Livermore Laboratory - RISOS Project

12.4.5 PLATO/ANTS

The experimental linkup between a modified PLATO IV plasma terminal at UCSB and the PLATO system at Illinois appears to be successful. The terminal at Santa Barbara is attached to the IBM 360/75 (it is standard ASCII compatible) and from there can access the network. At the Illinois end, the PLATO system is connected to ANTS through a direct phone line. As an interim measure, the software of MARK I was experimentally modified to listen on preassigned sockets for the PLATO connection. The user at UCSB must set up two simplex connections to these numerically adjacent sockets.

In mid-March, the PLATO software was installed in the production MARK I system at Illinois on an experimental basis. It will remain there so long as it does not appear to unduly degrade system performance. The multi-user PLATO module may be written for MARK II. In MARK II, the module would be overlayable and the connection would be initiated through a normal ICP connection.

12.4.6 PEESPOL

Work on PEESPOL over the past six months has been a mixture of debugging of new features implemented in the previous six months, and adding new features to the compiler. Among the new features are:

Miscellaneous macro generator improvements and enhancements;
Up-level addressing;
Allocation of separate code and data;
Greatly enhanced loop control facilities;
New patch merging facilities;
Miscellaneous listing readability improvements; and
Separate compilation and binding facilities

12.4.7 Hardware

During this reporting period, several IMP Interfaces were delivered. These went to Ames, UCLA, and NBS. Along with the Ames interface, our first "Distant Host Adapter" was delivered.

A new, improved "IMP Indicator Panel" was delivered during this period. The panel consists of 143 light emitting diodes that display the state of the major registers and control elements in the IMP Interface. Currently, Ames is the only site with this new panel.

Improvements were made to the IMP Interface to improve its operation with both the PDP-11 and the IMP. These improvements are manifest in the form of Engineering Change Orders Nos. 4, 5, 6 and 7, which have been incorporated into most interfaces.

A report is in preparation for ARPA's Interface Standardization Committee. This report, to be completed in early April, describes the history of the Illinois interface, its users, desirable engineering changes which could be issued, and a detailed response to the Committee's questionnaire of 29 March.

12.4.8 Documentation

A concentrated effort in documentation was begun in December. A preliminary plan for desirable documentation has been outlined. Several new documents are being written at the present time, and revisions to existing documents are either under way or contemplated. Existing and planned documentation are summarized below.

As one might expect, the major problem with providing documentation is the pressing need for programmers to move on to their next task. This will be alleviated by assigning personnel to the documentation section at appropriate milestones in their work, until the required documentation is completed.

Summary of Available Documentation

Bouknight, J. W., Gary Grossman, and Dave Grothe, "A New Approach to Network Access," NTS Document No. 1. Paper presented at the Data Communication Interface '73 Symposium, St. Petersburg, Florida, November 1973. Urbana, Ill.: Center for Advanced Computation, University of Illinois at Urbana-Champaign, 1973. 7 pages.

This document is an overview of the ANTS MARK II system. It is slightly out of date, but the best public summary.

Bouknight, Jack, "IMP Interface Manual," NTS Document No. 2. Urbana, Ill.: Center for Advanced Computation, University of Illinois at Urbana-Champaign, 1973. 150 pages.

A technical description and installation manual for the PDP-11 to interface designed by the University of Illinois.

Bouknight, Jack, "ANTS Mark I Support System." Urbana, Ill.: Center for Advanced Computation, University of Illinois at Urbana-Champaign, 1974. 65 pages.

This is a new document which is presently in first-draft form. It describes the programs and procedures used to maintain ANTS Mark I through network connection to the UCSD B6700.

Grothe, David, "PEESPOL: A Metaprogramming Language for Operating Systems." Urbana, Ill.: Center for Advanced Computation, University of Illinois at Urbana-Champaign, 1974. 17 pages.

A technical paper describing the metalanguage facilities of PEESPOL.

Grothe, David, "PEESPOL Reference Manual," NTS Document No. 5. Urbana, Ill.: Center for Advanced Computation, University of Illinois at Urbana-Champaign, 1973. 164 pages.

This is the definitive statement of the PEESPOL language. The entire language syntax is specified in BNF, with prose descriptions of semantics. It is not intended as an introduction to the language. This document will be revised shortly to expand the number of illustrative figures, particularly in the appendices. Efforts will be undertaken to provide an alphabetized index of the BNF for easy reference purposes.

Kelley, Karl, "ANTS Mark II System Description." Urbana, Ill.: Center for Advanced Computation, University of Illinois at Urbana-Champaign, 1974. 60 pages.

This new document is presently in first draft form. The early chapters give an overview of the ANTS Mark II system. Later chapters will provide increasingly detailed descriptions of major elements of the ANTS code.

Kelley, Karl, "Plan for Documentation of ANTS." Urbana, Ill.: Center for Advanced Computation, University of Illinois at Urbana-Champaign, 1974. 15 pages.

This is a new and temporary document used for planning purposes only.

Network Terminal System Group, "ANTS Mark I User Guide," NTS Document No. 6. Urbana, Ill.: Center for Advanced Computation, University of Illinois at Urbana-Champaign, 1974. 55 pages.

This is a new document, created from the outdated "Getting Started on the ARPANET". It will be distributed shortly to all Mark I users. A limited set of examples will be expanded in future revisions.

Network Terminal System Group, "Network User's Handbook," NTS Document No. 3. Urbana, Ill.: Center for Advanced Computation, University of Illinois at Urbana-Champaign, 1973. 150 pages.

A handbook designed primarily for use at Illinois, this document describes the user interface to ANTS, and introduces various user sites on the network. The document is intended primarily for ANTS users at the University of Illinois. It is in need of revision for reasons of outdated personnel information and changes to ANTS MARK I.

Sher, M. S., "A Case Study in Networking," NTS Document No. 4. Urbana, Ill.: Center for Advanced Computation, University of Illinois at Urbana-Champaign, 1973. 18 pages.

An article first published in the Fall 1973 issue of the EDUCOM Bulletin and later revised for publication in the March 1974 issue of Datamation. It describes the experience at Illinois of moving from a local B6700 operation to almost total dependency on ARPANET service sites.

Sher, M. S., "Retail Outlets for Network Computer Services," NTS Document No. 7. Urbana, Ill.: Center for Advanced Computation, University of Illinois at Urbana-Champaign, 1974. 14 pages.

An invited paper presented at the AAAS symposium, "Organizing Computer Networks for Science," San Francisco, February 26, 1974.

12.5 ENERGY RESEARCH

The Center is developing, under a recent grant from the National Science Foundation, a program to evaluate the environmental and economic impacts of energy resource consumption. The program studies the use of energy in today's goods and services, and seeks lower energy alternatives. Also, working closely with members of the Center's Economics staff, the project is utilizing computer technology to quantify and analyze the effects of various energy alternatives on the labor market.

A number of studies have been conducted in such areas as measuring energy and employment effects of mandatory deposits on beverage containers, comparison of various transportation modes, and the dollar, energy and employment impacts of consumer options.

In the area of energy supplies, staff members have conducted studies on low sulfur coal reserves and resources, and nuclear power availability to 1985. In the demand area, a study was done on the composite demand of energy by fuel, user, and PAD (Petroleum Administration for Defense) district to 1985.

Energy usage in the commercial and industrial sectors of the U. S. Economy was examined, and the interindustry energy model developed at CAC was modified to evaluate fuel substitution alternatives.

12.6 APPLICATION SYSTEMS GROUP

12.6.1 Introduction

The application systems group has a service orientation. A principal activity is to maintain software systems developed at the Center and help external groups to use them. A major secondary activity is liaison with state, regional, and local government staff to explore computer-based support and to offer technical consultation.

12.6.2 Natural Resource Information Systems

The IRIS (Illinois Resources Information System) was developed at the Center to give environmental planners easy and rapid access to a wide range of information on natural resources. Data on geology, hydrology, forestry and vegetation, soil characteristics, and land use patterns are maintained by the system.

During this period, staff members continued in their assistance of IRIS users, particularly with the Northeastern Illinois Planning Commission.

12.6.3 Socio-Economic Data Base Management

The University of Illinois' Department of Agricultural Economics, together with other units of the College of Agriculture, has established ISEIRD (Illinois Socioeconomic Indicators for Rural Development), a data collating and retrieval system.

The objective is to provide a set of continuing and updated information on initial aspects of Illinois agriculture and rural life.

CAC is developing interactive computer tools for researchers that will be utilizing ISEIRD data. Facilities will be provided for creating, manipulating, displaying and maintaining data archives. The Center is building a computer system that will then provide rapid access to data sets for extension, education, planning and research purposes.

12.6.4 Land Use and Related Information Support

Staff members have served as technical liaisons to state government agencies on several land use and environmental issues. A primary contact is the Bureau of the Budget, an agency that is concerned with state planning and the role of Illinois with respect to federal legislation. The Center has been represented in efforts to discuss a statewide legislative information system and other data oriented projects of such a comprehensive scope.

REPORTS AND PUBLICATIONS

Alsberg, Peter A., "Accounting System Primer," CAC Document No. 89. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, November 1973.

Alsberg, Peter A., "Distributed Data Management in the WWMCCS Environment," CAC Document No. 115. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, April 1974.

Babcock, Michael, Keith Erickson, Hugh Folk, and Walter Nidzieko, "Supply and Demand for Engineers in Illinois: 1975 and 1980," CAC Technical Memo No. 14. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, October 1973.

Belford, Geneva, "Some Experiments in LaPlace Transform Inversion," CAC Technical Memo No. 22. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, April 1974.

Belford, Geneva G., and John F. Burkhalter, "A Differential Correction Algorithm for Exponential Curve Fitting," CAC Document No. 92. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, November 1973.

Belford, Geneva, R. Jonathan Lermit, George Purdy, and Ahmed H. Sameh, "Computational Mathematics Abstracts," CAC Document No. 90. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, November 1973.

Bezdek, Roger H., and Bruce Hannon, "Energy and Manpower Effects of Alternate Uses of the Highway Trust Fund," CAC Document No. 101. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, December 1973.

Bullard, Clark W., "Energy Conservation through Taxation," CAC Document No. 95. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, December 1973.

Bullard, Clark W., and Robert A. Herendeen, "Energy Use in the Commercial and Industrial Sectors of the U. S. Economy, 1963," CAC Document No. 105. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, December 1973.

Bullard, Clark W., "Energy in the Planner's Mind," CAC Document No. 116. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, May 1974.

Center for Advanced Computation, "Semi-Annual Technical Report, October 1, 1972 - March 31, 1973," CAC Document No. 74. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, April 1973.

Center for Advanced Computation, "Semi-Annual Technical Report, April 1, 1973 - September 30, 1973," CAC Document No. 93. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, October 1973.

Chang, Chang-Chung, "An Algorithm for the Symmetric Generalized Eigenvalue Problem for $Ax = \lambda Bx$," CAC Document No. 110. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, February 1974.

DauffenBach, Robert C., "The Structure of Occupational Mobility in the U. S. Economy," CAC Document No. 103. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, December 1973.

DauffenBach, Robert C., "Supplementary Appendix to CAC Document No. 103: Occupational Code Transformations, and Probability Transition and Recruitment Dependence Matrices," CAC Document No. 104. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, December 1973.

Dunwoody, J. Ernest, Francisco Puleo, and Bruce Hannon, "Urban Bus-Car Substitution: Dollar, Energy, and Labor Impact," CAC Document No. 98. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, December 1973.

Frate, Dennis A., "Geophagy: A Dietary Practice in Holmes County, Mississippi," CAC Document No. 85. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, September 1973.

Hannon, Bruce M., "The Structure of Ecosystems," Journal of Theoretical Biology, 41 (Part 3), 1973, 535-546.

Hannon, Bruce M., and Roger H. Bezdek, "The Job Impact of Alternatives to Corps of Engineers Projects," American Society of Civil Engineers, Engineering Issues, Proc. Paper 10080, 99, October 1973, 521-531.

Hannon, Bruce M., "An Energy Standard of Value," The Annals of the American Academy of Political and Social Science, 410, November 1973, 139-153.

Hannon, Bruce M., "Options for Energy Conservation," Technology Review, February 1974.

Holmgren, Steve, "The Development of a System for the Application of Grey-Scale Computer Graphics to Presentation to Three-Dimensional Structures for Computer-Aided Ship Design. Interim Report. Part I: Display Recommendation," CAC Document No. 83. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, September 1973.

Holmgren, Steve, and Suzanne Sluizer, "The Development of a System for the Application of Grey-Scale Computer Graphics to Presentation to Three-Dimensional Structures for Computer-Aided Ship Design. Interim Report. Part I: Programming the Linescan Algorithm," CAC Document No. 84. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, September 1973.

Herendeen, Robert A., "The Dollar, Energy, and Employment Impacts of Certain Consumer Options," CAC Document No. 97. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, December 1973.

Huang, Hui-Ming, "A Parallel Algorithm for Symmetric Tridiagonal Eigenvalue Problems," CAC Document No. 109. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, February 1974.

Kimberly, John R., and David B. Rottman, "County Jails in Illinois: A Comparative Analysis of Local Contexts, Inmate Populations, Jail Characteristics and Staff Orientations," CAC Document No. 87. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, August 1973.

Lermit, R. Jonathan, "Numerical Methods for the Identification of Differential Equations," CAC Document No. 86. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, October 1973.

McDaniel, Lawrence M., "Mathematical Software and Computer Networks," CAC Document No. 94. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, October 1973.

Ray, Robert M., "ILLIAC IV Multispectral Image Processing Research," CAC Document No. 112. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, March 1974.

Ray, Robert M., and John Thomas, "ILLIAC IV Execution Times for Two ERTS-1 Data Interpretation Algorithms," CAC Technical Memo No. 13. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, September 1973.

Rieber, Michael, "Low Sulfur Coal: A Revision of Reserve and Supply Estimates," CAC Document No. 88. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, November 1973.

Rieber, Michael, "U. S. Energy and Fuel Demand to 1985: A Composite Projection within PAD Districts," CAC Document No. 108. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, January 1974.

Roistacher, Richard C., "A Microeconomic Model of Sociometric Choice," CAC Document No. 107. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, January 1974.

Sameh, Ahmed H., R. Jonathan Lermit, and Killion Noh, "On the Intermediate Eigenvalues of Symmetric Sparse Matrices," CAC Document No. 91. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, October 1973.

Sebald, Anthony, and Robert A. Herendeen, "The Dollar, Energy, and Employment Impacts of Air, Rail, and Automobile Passenger Transportation," CAC Document No. 96. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, December 1973.

Talaat, Moustapha, "Using Microprogramming to Implement Graphics Systems on Network Computers," CAC Document No. 117. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, January 1973.

Thomas, John, "An ILLIAC IV Algorithm for ERTS-1 Data Cluster Analysis," CAC Technical Memo No. 17. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, March 1974.

Thomas, John, "An ILLIAC IV Algorithm for Statistical Classification of ERTS-1 Imagery," CAC Technical Memo No. 18. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, March 1974.

Willauer, Edward T., "The Principal Components of Occupational Structure: A Factor Analysis Approach," CAC Document No. 111. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, February 1974.

THESES

Chang, Chang-Chung, "An Algorithm for the Symmetric Generalized Eigenvalue Problem for $Ax = \lambda Bx$," M.S. thesis in Computer Science, A. H. Sameh, advisor. May 1974.

DauffenBach, Robert C., "The Structure of Occupational Mobility in the U. S. Economy," Ph.D. thesis in Economics, Hugh Folk, advisor. October 1973.

Huang, Hui-Ming, "A Parallel Algorithm for Symmetric Tridiagonal Eigenvalue Problems," M.S. thesis in Computer Science, A. H. Sameh, advisor. January 1974.

13. GENERAL INFORMATION

13.1 Personnel

The number of people associated with the Department in various capacities is given in the following table:

	Full-time	Part-time	FTE
Faculty	25	3	26.50
Visiting Faculty	3	0	3.00
Research Associates and Instructors	0	0	-----
Graduate Research Assistants	0	54	24.53
Graduate Teaching Assistants	0	31	15.25
Hourly Junior Academic	0	21	3.82
Professional Personnel	1	0	1.00
Administrative and Clerical	15	0	15.00
Nonacademic Personnel (Monthly)	12	0	12.00
Miscellaneous Supporting Staff (Hourly)	0	17	5.36
TOTAL................	56	126	106.46

The Department Advisory Committee consists of Professor J. N. Snyder, Head of Department, Professors E. K. Bowdon, D. F. Cudia, M. F. Faiman, H. G. Friedman, C. W. Gear, D. B. Gillies, W. J. Hansen, W. J. Kubitz, D. J. Kuck, C. L. Liu, R. S. Michalski, M. D. Mickunas, R. G. Montanelli, S. Muroga, T. A. Murrell, J. Nievergelt, J. R. Phillips, W. J. Poppelbaum, S. R. Ray, E. M. Reingold, J. E. Robertson, A. H. Sameh, P. E. Saylor, D. L. Slotnick, D. S. Watanabe, and T. R. Wilcox.

13.2 Bibliography

During the first quarter, the following publications were issued by the laboratory:

Report Numbers

No.	608	Mueller, David H., "The ICGS System: Users Manual," February, 1974.
No.	609	Mueller, David H., "The ICGS System: System Tables Manual," February, 1974.
No.	622	Hansen, Wilfred J., "Transportation of Higher-Level Language Programs: Exemplified by an ALGOL 68 Transportation Representation," January, 1974.
No.	624	Mickunas, M. Dennis, "On the Covering Problem for Unambiguous Context-Free Grammars," February, 1974.
No.	625	Mickunas, M. Dennis, "On the Strong Covering Problem for LR(k) Grammars," February, 1974.
No.	629	Yao, Andrew Chi-Chih, "On a Problem of Katona on Minimal Separating System," March, 1974.
No.	630	Cutler, James R., "ERGODIC: Computing with a Combination of Stochastic and Bundle Processing," March, 1974.
No.	632	Liu, Jane W. S. and Liu, C. L., "Bounds on Scheduling Algorithms for Heterogeneous Computing Systems," to be published.

Theses

No.	617	Tao, William Yan-Yuen, "A Firmware Data Compression Unit," January, 1974. (M.S. Thesis)
No.	623	Kawasaki, Tsuneo, "Optimal Networks with NOR-OR Gates and WIRED-OR Logic," January, 1974. (M.S. Thesis)
No.	626	Pradels, Jean Louis, "The Guide - An Information System," March, 1974. (Ph.D. Thesis)
No.	631	Budzinski, Robert L., "OCOMO: Optical Correlation MOdulation," March, 1974. (M.S. Thesis)

13.3 Abstracts

Alsberg, Peter A., "Distributed Data Management in the WWMCCS Environment," CAC Document No. 115. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, April 1974.

Abstract:

The Distributed Data Management problems of the World Wide Military Command and Control System (WWMCCS) ADP network are discussed. The application of recently developed data compression and query tuning technology to this problem is described. The concept of a self monitoring and self restructuring data management system is described. The self organizing system, if successful, would have a significant impact on the ADP network performance and reliability. Initial areas of research and development are identified; a research and development program to address those areas is presented; and a plan is described to integrate the research and development program with the ongoing activity to develop a World Wide Data Management System (WWDMS).

Belford, Geneva G., and John F. Burkhalter, "A Differential Correction Algorithm for Exponential Curve Fitting," CAC Document No. 92. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, November 1973.

Abstract:

This report discusses a new approach to the construction of best uniform approximations of multi-term exponential form. The method used is a combination of differential corrections and linear programming. Results of a number of computer tests are discussed in detail.

Bezdek, Roger H., and Bruce Hannon, "Energy and Manpower Effects of Alternate Uses of the Highway Trust Fund," CAC Document No. 101. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, December 1973.

Abstract:

The direct and indirect dollar, energy, and employment costs of reinvesting the $5 billion (1975) Highway Trust Fund in six alternative federal programs are determined using a large linear computer model. These alternative programs are: Railroad and Mass Transit Construction, Educational Facilities Construction, Waste Treatment Plant Construction, the Law Enforcement Program, National Health Insurance Program, and Tax Relief Program.

Energy consumption would be reduced by shifting the Highway Trust

Fund to all of these categories except the Tax Relief Program. Employment would be increased in all cases. Energy consumption impact by type of energy and employment impact by occupation are given.

The highway and railway transport systems are compared in detail, and an energy-conserving, employment-increasing tax is suggested.

Bullard, Clark W., "Energy Conservation Through Taxation," CAC Document No. 95. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, December 1973.

Abstract:

The growth of energy productivity has slowed in recent years. This trend implies that energy demands will rise at increasing rates if economic growth is sustained. In turn, environmental, fiscal, and national security problems associated with growing energy demands may increase faster than our ability to cope with them.

To provide incentives for increasing energy productivity, an energy conservation tax is proposed. A linear model is used to estimate the impacts of such a tax on prices of final products. It is shown that an ad valorem tax could be more regressive than a specific tax based on energy units.

Bullard, Clark W., and Robert A. Herendeen, "Energy Use in the Commercial and Industrial Sectors of the U. S. Economy, 1963," CAC Document No. 105. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, December 1973.

Abstract:

This report presents detailed analyses of energy use in the 368 commercial and industrial sectors of the U. S. economy in 1963, and of intersector dependence in energy terms. Besides direct use, full attention is paid to the flow of non-energy goods and the energy thereby implied. The approach, which is based on energy Input-Output analysis, is described. Results: In section 2, sectors are ranked according to several energy-use criteria: 1) direct energy use, 2) energy intensity of the sector's output, and 3) energy required in the actual economy to provide the sector's total deliveries to final demand. In section 3, a detailed energy accounting of all inputs is given for each sector.

Bullard, Clark W., "Energy in the Planner's Mind," CAC Document No. 116. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, May 1974.

Abstract:

A land use planner can substantially affect energy demand in many ways. The planner designs the constraints within which the residents of an area choose to adopt energy conserving or energy wasteful lifestyles.

This paper defines the energy impact of planners' decisions, and develops a way of thinking about energy so it can be considered at every stage of the comprehensive planning process. Suggestions are offered for designing features into a community to insulate it from the instabilities of energy shortages and expected price increases.

Center for Advanced Computation, "Semi-Annual Technical Report, October 1, 1972 - March 31, 1973," CAC Document No. 74, Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, April 1973.

Abstract:

This document reports progress on ARPA contract DAHC04-72-C-0001 entitled "ILLIAC IV Applications Research at the Center for Advanced Computation, University of Illinois at Urbana-Champaign." The principal objective of this program is the development and testing of numerical techniques and software systems for use of ILLIAC IV over the ARPA Network. This is being accomplished through activities in the following areas:

1. Development of numerical techniques suitable for parallel processing in the areas of:
   a) Computational methods and linear algebra
   b) Linear programming
   c) Combinatorial algorithms
   d) Approximation of functions
   e) Integral equations
   f) ARPA Network
2. Development of an ILLIAC IV multispectral image processing system
3. Development of ILLIAC IV language for the phase II system
4. Development of programs in large scale calculations such as input-output economic modeling, quadratic assignment algorithms for various classes of spatial allocation problems, and atmospheric dynamics
5. Development of Network Access Computer System--ANTS

Center for Advanced Computation, "Semi-Annual Technical Report, April 1, 1973-September 30, 1973," CAC Document No. 93. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, October 1973.

Abstract:

This is a progress report on ARPA contract DAHC04-72-C-0001, entitled, "ILLIAC IV Applications Research at the Center for Advanced Computation, University of Illinois at Urbana-Champaign." During this period there was research in the following areas:

1. Development of numerical techniques suitable for parallel processing in the areas of:
   a) Linear programming
   b) Algebraic eigenvalue
   c) Approximation of functions
2. ILLIAC IV mulitspectral image processing
3. Enhancements to the PEESPOL compiler
4. Network Terminal Systems project
5. Distributed computational systems of heterogeneous computers

DauffenBach, Robert C., "The Structure of Occupational Mobility in the U. S. Economy," CAC Document No. 103. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, December 1973.

Abstract:

Knowledge of the extent and character of occupational mobility is basic to efficient and effective manpower planning and forecasting. This study is a highly detailed approach to such extent and character research questions. The fundamental purpose of this investigation is the identification of mobility-related groups of occupations through use of a neutral methodology. A cluster configuration of occupations, based on mobility patterns and linkages, enables a test of the worth of the recently proposed "Job Family" basis of occupational classification. Also, it provides information on the nature and kind of families of jobs--thus, there is a typological dimension to this study.

A theoretical approach to labor market dynamics through occupational mobility is provided in Occupation System Theory--a synthesis of Vocational Development and Labor Market Structure theories in conjunction with the Job Cluster concept. Two mobility models are employed: (a) the probability transition matrix (P) and, b) the recruitment dependence matrix (R). A separate report, CAC Document No. 104 (NTIS Report No. UIUC-CAC-DN-73-104), is a Supplementary Appendix containing Occupational Code Transformations, Probability Transition (P) Matrix, and Recruitment Dependence (R) Matrix.

Several important conclusions evolve from this study: aside from identification of interesting supply interrelationships between the diverse job categories and amplification of the dynamics of labor market operation, the fundamental conclusion is that the job family model is the relevant basis of occupational classification.

DauffenBach, Robert C., "Supplementary Appendix to CAC Document No. 103: Occupational Code Transformations, and Probability Transition and Recruitment Dependence Matrices," CAC Document No. 104, Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, December 1973.

Abstract:

This appendix, which is supplementary to CAC Document No. 103 entitled "The Structure of Occupational Mobility in the U. S. Economy," includes a listing of the occupational code transformations of Census to "use" codes and computer print-outs of the probability transition (P) and recruitment dependence (R) matrices. Computational methods are discussed in chapter five of the aforementioned document. In order to increase readability of the matrices, coefficients identical to zero appear as blanks; those which round to zero are printed, however.

Dunwoody, J. Ernest, Francisco Puleo, and Bruce Hannon, "Urban Bus-Car Substitution: Dollar, Energy, and Labor Impact," CAC Document No. 98. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, December 1973.

Abstract:

The direct and indirect dollar, energy and labor cost of the average urban bus and car is calculated. The net costs are determined for prescribed transfers from cars to buses. Schemes for detecting the impacts of respending any dollar savings are examined. The model is statistically based to allow impact calculations of changes in any particular bus system or of car operation in any particular city.

Frate, Dennis A., "Geophagy: A Dietary Practice in Holmes County, Mississippi," CAC Document No. 85. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, September 1973.

Abstract:

Geophagy has been observed and documented for over fourteen centuries. From the initial reportings till the present geophagy has been considered an example of a perverted appetite, and accordingly has been classified under pica. An indepth study in Holmes County, Mississippi, revealed that at least 80 percent of all women will consume clay at least once in their adult life. The practice, though centered on pregnancy, occurs from about the time of the first mentrual period till menopause. The practice itself is perpetuated culturally and not biologically as a mother will introduce her young daughter to the practice. Although mineralogical and chemical analyses on clay samples reveal that many elements necessary for body functions are present in clays, hematocrit data and other biological data indicate that clay consumption has minimal physiological effects.

Hansen, Wilfred J., "Transportation of Higher-Level Language Programs:

Exemplified by an ALGOL 68 Transportation Representation," Department of Computer Science Report No. 622, University of Illinois at Urbana-Champaign, January 1974.

Abstract:

Problem: Modern higher level languages, for example, ALGOL 68, must be defined to allow many alternate taken representations to accommodate the variety of input/output equipment available. Because of this variety, it may not be possible to transport programs from one hardware representation to another by any simple process of substitution.

Solution: As an example of the proposed approach, this paper defines a six-bit encoding for ALGOL 68 program text. It is an encoded character prefix representation, so it can be decoded without lookahead. The encoding is defined in detail and a decoder for a specific hardware representation is given.

Herendeen, Robert A., "The Dollar, Energy, and Employment Impacts of Certain Consumer Options," CAC Document No. 97. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, December 1973.

Abstract:

Impacts are compared for a spectrum of home consumption options, such as home vs. laundromat washing, hand vs. machine dishwashing, fresh vs. frozen vs. canned food, returnable vs. throwaway beverage containers, and various sets of kitchen appliances.

Kawasaki, Tsuneo, "Optimal Networks with NOR-OR Gates and WIRED-OR Logic," Department of Computer Science Report No. 623, University of Illinois at Urbana-Champaign, M. S. Thesis, January 1974.

Abstract:

Using gates (ECL) with dual outputs and WIRED-ORs, an algorithm to get the optimal networks, i.e., those which have a minimum number of NOR-Or gates and, as the secondary objective, a minimum number of connections, for a given arbitrary function, is discussed in this paper, under the assumption that only non-complimented variables are available as the network inputs. Only NOR-OR gates are used in these networks, but this algorithm can also be applied to networks with NAND-AND gates and WIRED-ANDs.

Based on this algorithm, optimal networks for all functions of three variables and also some functions of four variables are found.

Kuck, D., P. Budnik, S-C. Chen, E. Davis, Jr., J. Han, P. Kraska, D. Lawrie,

Y. Muraoka, R. Strebendt and R. Towle, "Measurements of Parallelism in Ordinary FORTRAN Programs," IEEE Computer, pp. 37-46, January 1974.

Abstract:

This paper reports the results of a measurement of parallelism at the statement level in 86 FORTRAN programs. The amount of parallelism is determined by an analyzer program and is measured in terms of speedup over serial execution, the number of independent processors required, the efficiency of parallel execution and other measures.

The analysis techniques are only sketched in this paper, details may be found in the references. We also outline some machine organization assumptions.

Lermit, R. Jonathan, "Numerical Methods for the Identification of Differential Equations," CAC Document No. 86, Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, October 1973.

Abstract:

Numerical methods are given for finding unknown functions contained in ordinary differential equations when a solution of the equation is known. These are iterative methods giving a best fit to the ℓ_2 norm to the known solution, which may contain random errors. A stability requirement on the numerical methods is proved.

Liu, Jane W. S. and C. L. Liu, "Bounds on Scheduling Algorithms for Heterogeneous Computing Systems," Department of Computer Science Report No. 632, University of Illinois at Urbana-Champaign.

Abstract:

The problem of job scheduling in a multiprocessor system containing processors of different computing speeds is studied. In particular, bounds on the worst case performance of several scheduling algorithms that can be implemented easily are obtained. Such bounds also provide us with information concerning how the performance of these scheduling algorithms vary as functions of the speeds of the processors. The trade-off between the speed and the number of processors in the system is also discussed.

McDaniel, Lawrence M., "Mathematical Software and Computer Networks," CAC Document No. 94. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, October 1973.

Abstract:

This paper discusses the impact of computer networks relative to the portability and accessibility of mathematical software. An application of resource sharing on computer networks is discussed--the integration of two resources on the ARPA Network: SPEAKEASY, an interactive system for researchers and EISPACK, a library of eigensystem routines.

Mickunas, M. Dennis, "On the Strong Covering Problem for LR(k) Grammars," Department of Computer Science Report No. 625, University of Illinois at Urbana-Champaign, February 1974.

Abstract:

A formal definition of one grammar, G' "strongly covering" another grammar, G is presented. Conditions under which the ability to parse G in an LR fashion implies the ability to parse G' in an LR fashion are developed. Constructive proofs are presented for the following formerly open problems. Every LR(k) grammar, G is strongly covered by an LR(1) grammar. Moreover, if L(G) is strict deterministic, then G is strongly covered by an LR(0) grammar.

Mickunas M. Dennis, "On the Covering Problem for Unambiguous Context-Free Grammars," Department of Computer Science Report No. 624, University of Illinois at Urbana-Champaign, February 1974.

Abstract:

It is shown that every unambiguous grammar which does not generate the empty string is covered by a Λ-free grammar. Every unambiguous grammar which does generate the empty string is covered by a grammar which is partitioned into a Λ-free portion and a portion which generates only the empty string. Finally, every unambiguous grammar is covered by such a partitioned grammar in operator form.

Mueller, David H., "The ICGS System: System Tables Manual," Department of Computer Science Report No. 609, Univeristy of Illinois at Urbana-Champaign, February 1974.

Abstract:

The Illinois Computing Graphics System (ICGS) is a batch system for a minicomputer. It executes a limited number of system programs, and provides facilities for rapid overlaying of these programs. This manual is intended to aid the system programmer in setting up the system tables.

Mueller, David H., "The ICGS System: Users Manual," Department of Computer Science Report No. 608, University of Illinois at Urbana-Champaign, February 1974.

Abstract:

The Illinois Computing Graphics System (ICGS) is a batch system for a minicomputer. It executes a limited number of system programs, and provides facilities for rapid overlaying of these programs. This manual is intended to aid the user in preparing system modules and executing the system.

Ray, Robert M., "ILLIAC IV Multispectral Image Processing Research," CAC Document No. 112. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, March 1974.

Abstract:

This report summarizes ILLIAC IV multispectral image processing research conducted during the last twelve months by the Center for Advanced Computation (CAC) of the University of Illinois in collaboration with the Laboratory for Applications of Remote Sensing (LARS) of Purdue University. The research reported has focused on the design and partial implementation of a comprehensive ILLIAC IV software system for computer-assisted interpretation of multispectral earth resources data such as that now collected by the Earth Resources Technology Satellite (ERTS). Research to date suggests generally that the ILLIAC IV should be as much as two orders of magnitude more cost-effective than serial processing computers for digital interpretation of ERTS imagery via multivariate statistical classification techniques. The potential of the ARPA Network as a mechanism for interfacing geographically-dispersed users to an ILLIAC IV image processing facility is discussed.

Rieber, Michael, "Low Sulfur Coal: A Revision of Reserve and Supply Estimates," CAC Document No. 88. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, November 1973.

Abstract:

Conventional estimates of both known resources and known recoverable reserves of low sulfur coal ($\leq$ 0.7 percent sulfur) are grossly overstated. On a standardized base of 22.6 million Btu/ton, the electric utility average in 1970, known recoverable reserves of low sulfur coal are reduced from the conventional estimate of 68.2 billion tons to 16.4 billion tons on a consistent Btu sulfur adjusted basis. The reduction amounts to 76 percent of the U. S. conventional low sulfur estimate and 85 percent of the western estimate. At a 7 percent growth rate in production, known recoverable reserves fall short of maximum cumulative production by 1985. To counteract this problem, several short and long-run policy options and alternatives are presented.

Rieber, Michael, "U. S. Energy and Fuel Demand to 1985: A Composite Projection within PAD Districts," CAC Document No. 108. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, January 1974.

Abstract:

This study covers the period from 1970-1986. Consumer groups are divided into: residential/commerical, industrial, transport, electric utility and non-energy. Energy demand for each of these groups is evaluated for each of the five PAD districts. Demand is stated in both BTU's and fuel units with respect to petroleum and petroleum products, coal, natural gas, nuclear power, and hydropower.

Roistacher, Richard C., "A Microeconomic Model of Sociometric Choice," CAC Document No. 107. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, January 1974.

Abstract:

The behavior of a person selecting a set of friends from a larger set of acquaintances can be analyzed as a consumer choice problem. The person can be regarded as a consumer allocating his income among a set of goods which he must purchase in quantities which will maximize his utility. An increase in utility can come either from an increase in expenditure or from a better allocation of resources. Results of an unlimited-choice sociometric questionnaire administered to 1204 boys at eight junior high schools showed that well-liked boys received the same number of choices as others, but had a higher proportion of reciprocated responses. It appears that social success results from lower costs of obtaining information about potential friends and better allocation of effort, rather than from making contact with more people.

Sameh, Ahmed H., R. Jonathan Lermit, and Killion Noh, "On the Intermediate Eigenvalues of Symmetric Sparse Matrices," CAC Document No. 91. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, October 1973.

Abstract:

An algorithm has been developed for finding the eigenvalues of a symmetric matrix A in a given interval [a, b] and the corresponding eigenvectors using a modification of the method of simultaneous iterations with the same favrable convergence properties. The technique is most suitable for large sparse matrices and can be effectively implemented on a parallel computer such as the ILLIAC IV.

Sebald, Anthony, and Robert A. Herendeen, "The Dollar, Energy, and Employment Impacts of Air, Rail, and Automobile Passenger Transportation," CAC Document No. 96. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, December 1973.

Abstract:

Dollar cost, and total energy and employment impacts of the three modes for intercity travel are compared for 1963 and 1971. "Impact" includes indirect as well as direct effects. Results are given on a passenger mile-basis, and for three specific trips of length 200-400 miles.

Tao, William Yan-Yuen, "A Firmware Data Compression Unit," Department of Computer Science Report No. 617, University of Illinois at Urbana-Champaign, January 1974.

Abstract:

In this report, a data compression-decompression algorithm has been developed. The compression technique employed is the Huffman coding method which minimizes the average length of the codewords for encoding. This algorithm is table driven. It is developed in firmware on the Lockheed SUE microprogrammable minicomputer as a standalone unit. Due to the removal of the data compression-decompression function from the main CPU and the high speed of execution of micro-program, overhead in performing the data compression-decompression algorithm is minimized tremendously. The result of this is a net gain in precious on-line storage area and I/O time with no loss in main CPU overhead.

Willauer, Edward T., "The Principal Components of Occupational Structure: A Factor Analysis Approach," CAC Document No. 111. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, February 1974.

Abstract:

The empirical results of a factor analysis of labor market characteristics by occupations are reported in this document. The purpose of the analysis is to identify the principal components of labor market segmentation by occupation. The results of the analysis can be used to construct a number of occupational systems whose contents are uncorrelated with one another. The data base for this study was constructed from the 1970 Bureau of the Census tabulations.

Yao, Andrew Chi-Chih, "On a Problem of Katona on Minimal Separating Systems," Department of Computer Science Report No. 629, University of Illinois at Urbana-Champaign, March 1974.

Abstract:

Let S be an n-element set. In this paper, we determine the smallest number f(n) which there exists a family of subsets of S $\{A_1, A_2. \ldots, A_{f(n)}\}$ with the following property: Given any two elements $x, y \in S$ $(x \neq y)$, there exist k, ℓ such that $A_k \cap A_\ell \neq 0$, and $x \in A_k$, $y \in A$. In particular it is shown that $f(n) = 3\log_3 n$ when n is a power of 3.

13.4 Colloquia

"Design and Control of Pipelined Processors for Masimum Cos-Effectiveness with Case Studies of the Fast Fourier Transform and Searching," by Professor Edward S. Davidson, Department of Electrical Engineering and Coordinated Science Lab., University of Illinois at Urbana-Champaign, January 28, 1974.

"Problems and Solutions in the Design of Primary Memory Systems for Array Processors," by Research Assistant Professor Duncan H. Lawrie, Department of Computer Science, University of Illinois at Urbana-Champaign, February 4, 1974.

"Microprogrammed Sequential Logic Design: From Flow-Chart To Digital Machines," by Dr. K. Hwang, Department of Electrical Engineering, University of Miami, Coral Gables, Florida, February 18, 1974.

"Parser Construction with Ambiguous Grammars," by Mr. Alan J. Demers, Department of Electrical Engineering, Princeton University, Princeton, New Jersey, February 21, 1974.

"Program Behavior Modeling," by Dr. Jai R. Rao, Holmes and Narver, Inc., Anaheim, California, February 25, 1974.

"Nordsieck Methods," by Dr. Robert D. Skeel, Department of Computer Science, University of Illinois at Urbana-Champaign, March 4, 1974.

"The Hardest Context-Free Language," by Dr. Jonathan Goldstine, Committee on Information Sciences, The University of Chicago, Chicago, Illinois, March 7, 1974.

"SIBYL: A Formally Defined Interactive Programming System Containing an Extensible Block Structured Language," by Mr. Garry Kampen, Department of Computer Science, University of Washington, Seattle, Washington, March 13, 1974.

"The Formal Semantics of Operating Systems," by Professor E. J. Neuhold, Institut fur Informatik, Universitat Stuttgart, 7 Stuttgartl, Herdweg 51, Germany, March 13, 1974.

"Program Working-Information Monitoring by a Space-Time Duality Approach," by Mr. Felix L. Lam, Department of Electrical Engineering and Computer Science, University of California, Berkley, California, March 14, 1974.

"Digital Signal Processing from a Hardware Point of View," by Mr. Bruce Kieburtz, Bell Telephone Labs., Holmdel, New Jersey, March 25, 1974.

"Minimal Switching Networks for Performing Shifts," by Professor Foong Frances Yao, Department of Computer Science, University of Illinois at Urbana-Champaign, March 28, 1974.

13.5 Drafting

During the first quarter, a total of 369 drawings were processed by the general departmental drafting section:

Formal Drawings	
Large Drawings	118
Medium Drawings	71
Small Drawings	52
Layout Drawings	41
Report Drawings	39
Change Order Drawings	41
Miscellaneous Drawings	7
Completed Total Drawings	369

(M. Goebel)

13.6 Shop's Production

Job orders processed and completed during the first quarter of 1974 are as follows:

	Navy Grant	NSF Grants	Other
Electronics Shop	67	5	13
Chemical Shop	50	5	14
Photographic Shop	46	3	10
Layout Shop	38	2	2

(F. P. Serio)

BIBLIOGRAPHIC DATA SHEET	1. Report No. UIUCDCS-QPR-74-1	2.	3. Recipient's Accession No.
·:le and Subtitle Quarterly Progress Report			5. Report Date
			6.
7. Author(s)			8. Performing Organization Rept. No.
9. Performing Organization Name and Address Department of Computer Science University of Illinois Urbana, Illinois 61801			10. Project/Task/Work Unit No.
			11. Contract/Grant No.
12. Sponsoring Organization Name and Address Department of Computer Science University of Illinois Urbana, Illinois 61801			13. Type of Report & Period Covered Quarterly Progress Report Jan. - Feb. - March
			14.
15. Supplementary Notes			
16. Abstracts Not Applicable			
17. Key Words and Document Analysis. 17a. Descriptors Not Applicable			
17b. Identifiers/Open-Ended Terms Not Applicable			
17c. COSATI Field/Group			
1 .lability Statement RELEASE UNLIMITED	19. Security Class (This Report) UNCLASSIFIED		21. No. of Pages 149
	20. Security Class (This Page UNCLASSIFIED		22. Price