# DECUS PROCEEDINGS

## FALL 1967

## PAPERS AND PRESENTATIONS

of

Digital Equipment Computer Users' Society

Maynard, Massachusetts

# FALL
# 1967

# PAPERS AND PRESENTATIONS

of

Digital Equipment Computer Users' Society

## FALL SYMPOSIUM

"COMPUTERS IN THE LABORATORY"

NOVEMBER 10, 11, 1967

Jolly Roger Inn–Anaheim Convention Center

Anaheim, California

CONTENTS

PAGE



---

* Abstract Only

# PREFACE

The Jolly Roger Inn and the Anaheim Convention Center were the settings for the sixth DECUS Fall Symposium held in Anaheim, California, on November 10 and 11. Approximately 200 users attended the two-day sessions on Computers in the Laboratory which included 30 papers, 5 workshops, and a tutorial on numerical analysis. The meeting opened with a keynote address by Digital's President, Kenneth H. Olsen, during which he commented on the growth of DECUS and reiterated Digital's support of the Society.

A new concept of tutorial workshops based around a software package and application area was well received. The workshops held on the PDP-9 Advanced Software and the PDP-8 Disc Software were ably presented by DEC people. Both included a question-and-answer session following the presentation.

The first module workshop was held on Friday afternoon. Approximately 24 people were in attendance. Sypko Andreae, Chairman of the Module Users Group (MUG), gave a short introduction reviewing the reasons and goals behind the establishment of a module group. This was followed by two users presenting several suggestions for new products in the M series and stronger lines of communication between users and DEC. An open discussion session followed. The overall feeling was that these workshops should be continued at future DECUS meetings.

Papers published in this volume have been printed as received from authors with no editorial changes. In some cases, papers were not received in time for publication and abstracts of these papers have been substituted. If the omitted papers are at some time submitted to the users group, they will be published in the newsletter, DECUSCOPE. Reprints of papers presented here are available from the DECUS Office, Maynard, Massachusetts 01754.

This proceedings also contains a list of meeting attendees, the program, and an author/speaker index.

Special thanks to meeting co-chairmen, Prof. Donald A. Molony of Rutgers and Prof. Philip R. Bevington of Stanford, for their able assistance.

Angela J. Cossette (Mrs.)
DECUS Executive Secretary

# A COMPUTER CONTROLLED DIFFRACTOMETER*

Howard A. Cohodas
Picker Instruments
Cleveland, Ohio

Abstract

In recent years X ray diffraction has become an important tool for the investigation of crystal structure. At the same time, the small computer has gained popularity in a systems control environment. Together they provide a powerful tool for scientific investigation.

One such system is the Picker diffractometer controlled by either the PDP-8 or PDP-8/S. Discussion will center around its hardware and software development, including examples of some useful programming techniques implemented on this system.

* This paper was not received for publication.

# THE INTEGRATOR/COMPUTER SYSTEM FOR GAS CHROMATOGRAPHY DATA AUTOMATION

A. T. Barrett, III
Infotronics Corporation
Houston, Texas

## Abstract

A small general purpose computer interfaced with an electronic digital integrator forms a flexible automatic analysis system for gas chromatography. The integrator performs the tasks of signal processing, integration and peak detection. The computer does the identification, computation and reporting of the final results.

## Introduction

There have been several developments in the past few years in the area of hardware which have a direct bearing on process chromatography. Since their introduction in late 1952, gas chromatographs have become widely accepted as an analytical tool in the chemical, petroleum and petro-chemical industries. Although their use is widespread, the methods of analyzing chromatograph output have been slow in developing.

The output of a chromatograph is an electrical analog signal which may range from microvolts to volts. The signal contains the qualitative and quantitative information about the sample which it represents but it does not present itself in a readily usable form. One method of presenting the signal is to attach a strip chart recorder to the chromatograph, which shows the chromatogram as a series of peaks. The speed of the recorder provides information about the elution time of a given peak but none concerning its magnitude or area. However, some information is available from the strip chart. The elution time of a peak is indicative of its identity and the peak heights can be used to approximate magnitudes.

Measurement of the area of a peak is the most difficult part of chromatogram processing. There have been many methods devised to compute these integrals. Perhaps the most primitive is cutting the peak out of the strip chart and weighing the paper. It is crude at best but it does provide usable results. Planimeters are an improvement but alignment and curve following make it almost as laborious as the previous method. Disc integrators are an improvement over both of these procedures and give results easily convertible to numerical information. Even though the disc integrators and other electro-mechanical devices of the same kind provide usable output, there are problems involved in their use.

In all three integration methods previously mentioned, the signal being analyzed is the trace of the pen on the strip chart recorder. The recorder is a fine device for representing the analog signal of the chromatograph but it also provides a source of error due to its susceptibility to noise, its calibration requirements and dynamic range problems. In addition, the use of capillary columns presents a new type of problem. Peaks eluting from a capillary column normally come in short intervals and are of narrow width. Unquestionably the methods of cutting out the peaks or tracing the peak on a planimeter become inadequate with output of this nature. Even the disc integrators become useless because of their slow response time to fast peaks.

The latest advancement is the electronic digital integrator. Here is a device which analyzes the output of the chromatograph. It can handle both slow and fast peaks and present a printed copy of the quantitative information, peak retention times and areas. However, it is only a partial solution to the total problem of automated analyses. There are still several manual steps to be performed with the integrator's output.

A plethora of paper is created by any chromatography operation. Once the pertinent information about a chromatogram is obtained, it still needs to be manipulated in such a manner as to obtain results in engineering units. A technician must sit down with his trusty slide rule or desk calculator and convert these data to mole percent, concentration or whatever units are applicable. Since the function of the control laboratory is to furnish information to the operators of the various processes throughout the plant, reports need to be generated in the proper format and transmitted to the cognizant personnel. Manual operations such as these are subject to the inconsistencies of the human. Not only is the accuracy of the results dependent upon the reliability of the human but the manual operations during this phase introduce a time lag which can become significant. Quite obviously, the elimination of as much human intervention as possible would decrease the turn around time of the data and make errors predictable.

A computer is the natural device to reduce these manual operations. However, until just recently computers were out of the question for such a dedicated operation. The small general purpose computers, such as the PDP-8 and PDP-8/S, which have become available in the past two years are priced so that they become very attractive as a chromatography laboratory tool. However, price alone does not constitute all of their beauty. These small computers are at home in the wide ranging environmental conditions of the chromatography laboratory. They can be easily interfaced to a variety of peripheral I/O devices and analytical instruments. Thus, the small computer avails itself to the chromatography laboratory as the right piece of equipment necessary to form the missing link in the quest for an automated analysis system. But what job or jobs is the computer to do?

## A-D System

There is one school of thought that the computer should perform all tasks, that is, signal filtering, peak detection, integration, normalization, reporting, etc. This approach is commonly used in laboratories with a large number of chromatographs where the inputs are connected to a bank of multiplexed amplifiers whose output goes to an A-D converter. It is an on-line system in the true sense of the word and it does provide for automatic analyses. However, such computers, though they be classed as small, take on proportions of a medium scale machine when one examines the interrupt, core and mass storage requirements. Core sizes of 16K and bulk storage of 256K are not uncommon. As for front-end hardware, sophisticated multiplexers with computer controlled amplifier range selection are needed since wide dynamic range devices are usually not available. Scanners need to be optimized so that sufficient samples are presented to define a given peak. Since the chromatograph signal is of a low level, it is very susceptible to interference as it goes to the digitizer. Relays, sample switches and the like can generate noise that can play havoc on a chromatograph signal. If the cables from the chromatograph to the digitizer invite the inherent noise in the laboratory, much time, effort and money may be spent before the problem is eliminated. Last but not least there is the amount of programming necessary to put such a system on-line. Different analyzers will require different analysis programs in addition to sophisticated digital filtering, baseline tracking, peak detection and integration routines. All in all, it is a high level programming job which requires the services of many people and much time to complete.

## Integrator/Computer System

A second type of system produces the same ultimate result yet in a different manner. Such a system is formed by interfacing one or more digital integrators to a small scale computer. Figure 1 shows a typical hardware configuration. Functionally the system is very simple. Each integrator performs the timing, detection, and integration of the constituent peaks eluting from its chromatograph. The computer receives the time and areas as they are transferred from the integrator and then performs the analysis and reporting when the run is terminated.

The integrator/computer system is not as susceptible to noise as is the A-D system. The integrator sub-system can be located next to the chromatograph yet the computer can be down the hall, or in the next building. By being close to the chromatograph, the input cable to the integrator is short and less responsive to background noise. The output of the integrator is a high level digital signal which can be transferred to the computer over a relatively long distance.

The entire system is built in a modular fashion so that each sub-system is independent. Although the computer may not be functioning, useful numerical results may be obtained from the integrators via their printers leaving only area corrections, normalization, etc. to be performed by the technician.

Figure 2 shows a block diagram of an electronic digital integrator. As the analog signal from the chromatograph is output, it is amplified and sent in parallel to a voltage-to-frequency converter and to the peak detector. The voltage-to-frequency converter output is a signal which is linearly proportional to the amplitude of the input voltage. Therefore, a certain frequency of a given time duration results in a specified number of counts. As the peak goes from baseline to baseline, the counts are summed over the various frequencies to give the peak integral.

The peak detector operates on the first derivative of the amplified signal. As the peak begins, the sensing circuit waits until the slope becomes greater than a certain level before initiating the integrate logic. The cycle of the peak detector logic goes from positive slope to zero slope to negative slope to zero slope in order to define a peak. The elution time is picked at the apex of the peak. At the end of the final phase of the cycle, the logic loads the contents of the data counter and the elution time of the peak into the buffer; thus freeing the counter for the next peak. The time and data are then strobed out of the buffer into the printer, or other output device.

Certain functions in the logic perform as a digital filter by rejecting peaks smaller than a specified number of counts. False peaks can be rejected by sensing the amount of time they are in zero slope and washing them out if they exceed a preassigned tolerance.

As for baseline correction, the output of the voltage-to-frequency converter, during times when no integration is occuring, should be a rate close to zero, possibly one or two counts per second. If the count rate increases and slope sensitivity does not initiate integration, the baseline drift corrector converts the frequencies back to voltages and, through a second input in the voltage-to-frequency converter, introduces a signal of opposite polarity. In effect, this produces a subtraction and the output of the voltage-to-frequency converter is again back to one or two counts. The drift corrections are suspended during integration and the baseline at the start of the peak is stored in an analog memory.

When all peaks of interest have been integrated and transferred to the readout device, the necessary data is available to complete the analysis. All that the computer must do is to properly identify the peaks, apply the respective response factors, complete the analysis in engineering units, and then form a report.

## Data Reduction

As was previously mentioned the trace of the chromatograph signal is a series of peaks eluted over a period of time. The area of each peak is proportional to the percentage that the peak comprises in the total sample. The sum of all the areas represents the total sample size. These can be easily expressed in mathematical form by the following equations. Given an area, $A_i$, and response or proportionality factor, $W_i$, the size of the total sample may be found by

$$A = \sum_{i=1}^{N} A_i W_i \qquad (1)$$

where N is the number of peaks in the chromatogram.

The mole percentage or normalized values of each component in the sample, therefore, becomes

$$\%_n = \frac{A_n W_n}{A} \times 100 \qquad (2)$$

Obviously, before any such computations may be performed, each peak must be present and properly identified for association with its corresponding response factor. In order to perform the identification, the retention times of the peaks, either absolute or relative to a reference peak, are used. In either case it is necessary to establish, the reference peak using absolute time. When the reference peak is established, the other peaks may be identified by their temporal position with respect to it. Then application of response factors may be applied and the results computed and reported. It sounds simple enough to do but what if the computer is performing such work?

When a new type of sample is being set up for running on the computer/integrator system the technician must supply the computer with certain pieces of information. These are stored in the computer as a table and consist of the absolute time of the reference, and the names, response factors and retention times of the expected peaks in the run. The times may be obtained from the printed integrator output by running a typical sample or by calculating mean values from several samples. Each table is identified by a code number and the address of the table stored in a directory in memory. A special program is provided which allows the operator to enter this information. The program assembles the tables in memory and punches a paper tape of what was stored. Several tables may be stored in core at any one time. The system is ready to operate when this information is available to the computer.

In the actual operation of the integrator/computer system, the technician injects the sample into the chromatograph and simultaneously presses the "start" button on the interface console. This action causes identification information such as sample type, sample number, time of run, date, analyzer number, etc. to be entered into the computer from a series of thumbwheel switches. The timing in the integrator is also begun at this point. As the constituents elute, their peaks are integrated, timed and the results transmitted in parallel to the printer and to the computer. The program determines the integrator used in the transfer and stores the data in an array allocated for that integrator.

When the final peak of interest has been entered into the computer, the technician simply presses the "reset" button on the interface console. An end-of-chromatogram signal is sent to the computer from the reset integrator notifying it that the identification procedures may begin for that integrator's data. An executive program examines the status of each integrator and when a "data completed" switch is found to be "on" the control is transferred to the data reduction and identification routines. The interrupt remains enabled during the identification and reporting so that peak times and areas from other chromatograms may be entered.

The operator console also has a button which allows the run on any given integrator to be aborted. By pressing this button the operator resets the integrator's clock and data counter and puts it into a standby mode. The computer therefore does not receive any new information from the integrator and no action is taken on data received up until the time of the abort. The start of a new run reinitializes the program, over-writing the identification information, peak times and areas of the aborted run.

Depending on the method of analysis desired by the technician, retention times may or may not be converted to relative values. Normally relative retention times are used because of the shifts in absolute times due primarily to column aging. In either case, the temporal position of a given peak to the reference peak is used to establish the identification of the former. Only those peaks whose retention times have been previously entered and stored in the computer will be identified. Extraneous peaks are considered unknowns and reported as such. The identification of the various peaks allows application of the proper response factor to obtain the corrected area. Response factors may vary over a wide range depending on the method of reporting the results. For normalization, equations 1 and 2 are applied and results reported in percentages. If an internal standard is used, the response factor may have units which give results in terms of concentration.

The final report includes the sample identification, analyzer number, time of run, date of run, etc. In addition, each peak is listed with its name, retention time, peak area and analysis results. The conclusion of the report transfers program control back to the executive routine to check for more completed data sets. And so the cycle repeats. A simplified block diagram of the computer program is shown in Figure 3. Figure 4 shows the format of the analysis report. From the start of the run until the final report is complete, the only manual operation consists of pressing two buttons.

## Computer Configurations

In comparison with the program needed for the complete A-D system, the integrator/computer approach requires much less complex programming. The identification, analysis and reporting routines virtually remain a constant size for any size system. The size of the interrupt routines and working storage requirements will naturally vary with the number of integrators in the system. Figure 5 relates the maximum configurations for 4K and 8K core memories. These figures are for the PDP-8 and PDP-8/S computers. The 4K core system can contain all the program, working storage and tables to manage up to 4 chromatographs with a maximum of 20 peaks per run. By adding a 4K memory extension module this configuration can handle up to 12 chromatographs with 30 peaks per chromatogram.

Where capillary columns are used, a large number of peaks usually results. If that is the case, a 4K computer is used in conjunction with a 32K disk. Data storage routines are changed so that the peak times and areas are stored on the disk rather than in core. In addition, the tables of retention times, response factors and names are stored on the disk. Not only does this configuration handle runs in excess of 100 peaks but it also is used where there are more than twelve chromatographs. As in the core only systems, the identification, analysis and reporting routines remain the same size in the disk system. Thus, expansion of the system program becomes a relatively easy task.

## Accuracy

Regardless of the computers configuration, the integrator is the main point of accuracy since it performs the reduction of the chromatograph signal. The best accuracy of the integrator is always obtained with well resolved peaks along a flat baseline. Under these conditions the integrator will give accuracy to .25% of peaks greater than one-half millivolt. Below that level, the accuracy degrades in proportion to the peak amplitude. The Infotronics Model CRS-100 integrator detects peaks of 10 microvolts and 100 seconds wide. That is a slope sensitivity of .2 microvolts per second, far better than the normal situation in the A-D system. In the case of unresolved peaks, the integrator drops a vertical to the baseline at the valley. This scheme has limitations since overlap errors are involved.Error is also propagated where the baseline changes during integration. The value of the baseline at the start of the peak is assumed at the end of the peak and a triangle of error is introduced proportional to the amount of drift.

## Conclusion

In summary, the integrator/computer system offers a flexible method for automating gas chromatograph analysis. It is particularly well suited to meet the demands of the process laboratory in both accuracy, reliability and speed. The computer necessary in such a system may be truly a small computer utilizing a minimum core and possibly a 32K disk. Program requirements remain the same on virtually every size system. Manual intervention is reduced to a bare minimum and time lags in reporting final results are eliminated. The system is modular in design and provides some back-up information in the event of a sub-system failure. The modularity also allows for easy expansion to fit the fluid environment of the process laboratory where instrument configurations and analytical methods often change.

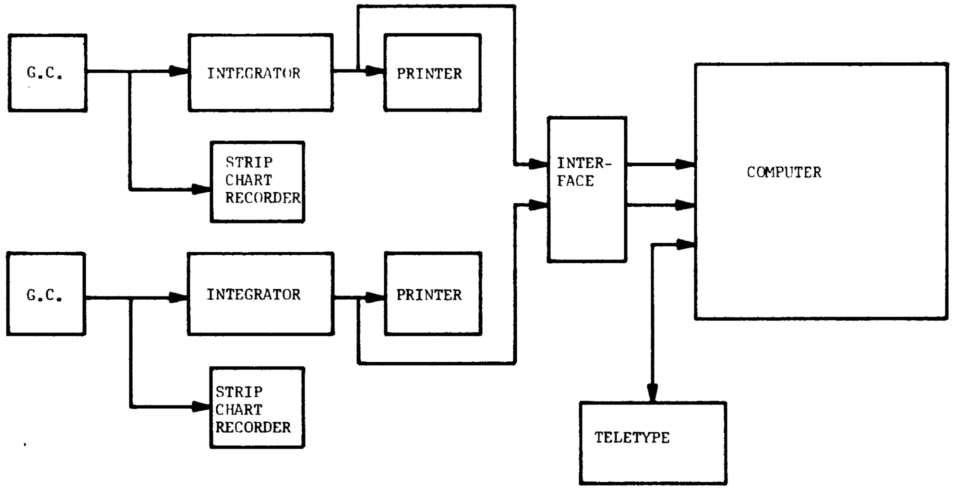



FIGURE 1 -

Hardware configuration and block diagram for a two integrator system. Additional inputs would require the integrator and printer.

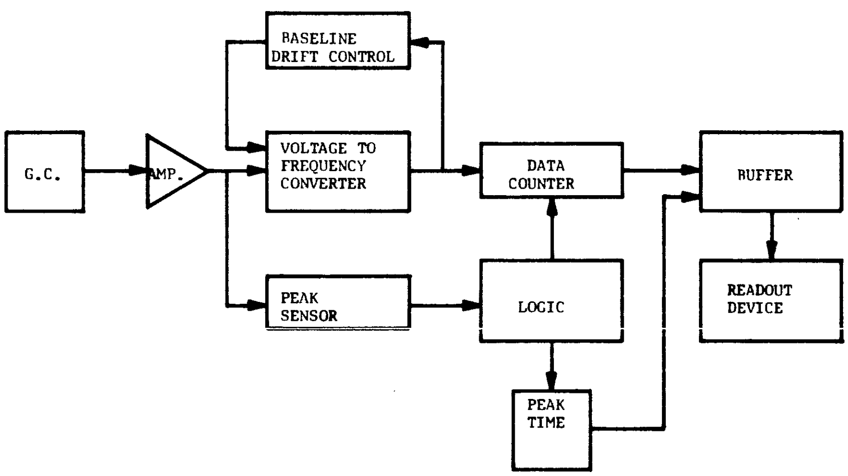



FIGURE 2 -

Block diagram of an electronic digital integrator.

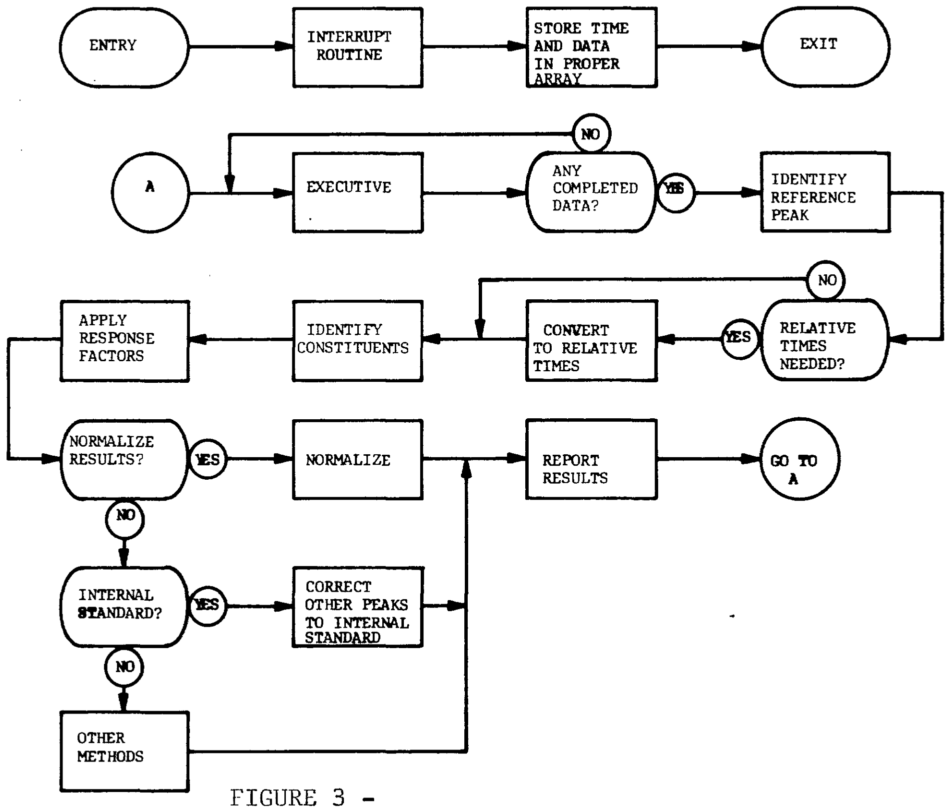



FIGURE 3 -

Integrator/computer program flowchart

DATE 11/02/67

SOURCE REACTOR PROD

SYSTEM 2

ANALYZER 12

| COMPOUND | TIME | AREA | PERCENT |
|---|---|---|---|
| ETHYLENE | 51 | 27200 | 13.81 |
| ETHANE | 78 | 13688 | 6.95 |
| PROPYLENE | 123 | 25953 | 13.18 |
| PROPANE | 153 | 21559 | 10.95 |
| ACETYLENE | 190 | 18089 | 9.18 |
| ISOBUTANE | 226 | 10076 | 5.11 |
| H2SULFIDE | 266 | 17437 | 8.85 |
| N-BUTANE | 303 | 7877 | 4.00 |
| BUTENE-1 | 348 | 24841 | 12.61 |
| UNKNOWN | 379 | 133 | .06 |
| T-BUTENE-2 | 418 | 19385 | 9.84 |
| C-BUTENE-2 | 465 | 4095 | 2.08 |
| UNKNOWN | 520 | 385 | .19 |
| ISOPENTANE | 552 | 6147 | 3.12 |
| | | | ------ |
| | | | 99.99 TOTAL % |

FIGURE 4 -

Report format of integrator/computer system

COMPARISON OF EQUIPMENT CAPABILITY

| MAXIMUM CORE | I MAXIMUM INPUTS (MEM=4OI) | N MAXIMUM CONSTITUENT NAMES (MEM=5N) | E MAXIMUM DATA TABLE ENTRIES (MEM=4E) | P MAXIMUM PEAKS/RUN (MEM=(4I+3)P) | K PROGRAM & COMMON MEMORY | TOTAL MEMORY REQUIRED |
|---|---|---|---|---|---|---|
| 4096 | 4 | 47 | 80 | 20 | 3000 | 4095 |
| 8192 | 12 | 250 | 400 | 30 | 3200 | 8060 |
| 4096 W/32K DISK | 35 | 400 (DISK STORED) | 3750 (DISK STORED) | 100 | 3300 | 4000 CORE 31,000 DISK |

FIGURE 5 -

Comparison of equipment capabilities

# DATA ACQUISITION AND ANALYSIS OF HIGH RESOLUTION MASS SPECTRA IN REAL TIME

M. L. Cramer and D. J. Waks
Applied Data Research, Inc.
Princeton, New Jersey

## Abstract

A multi-phase computer program operational on a minimum configuration PDP-8 performs data collection, compression, mass measurement and chemical composition determination while connected on-line to a high resolution mass spectrometer. An analog-digital converter, variable oscillator, adjustable threshold, bias control, and error detection logic are employed in a low-cost data acquisition interface specially designed for this application.

Considerable variation in adjustment and parameterization is permitted from run to run enabling the application of automatic measurement and analysis to a wide range of mass spectral data. The operator/experimentor is encouraged to control and steer the process in an on-line interactive fashion through the selection of run parameters, standard measurement criteria, composition and error limits, and final report format.

## Introduction

The mass spectrometer data acquisition and analysis system described is a multi-phase proprietary[1] computer program designed to operate on a DEC PDP-8 digital computer with 4096 words of memory, ASR teletypewriter and high speed paper tape input. A high speed paper tape output is optional to the user.

The computer is connected to a high resolution mass spectrometer or analog FM tape recorder through a special data acquisition interface designed and built at Applied Data Research. It includes a 138E analog-digital converter, variable oscillator, adjustable threshold, bias control and error detection logic. Data collection is initiated/terminated by means of a start/stop switch which may be actuated either manually or automatically.

The overall system response is such that on-line scans of 40 seconds per decade of mass are possible with mass measurement accuracy of better than 10 ppm. Slightly under 600 spectral peaks can be digitized and 512 analyzed as part of any single data collection run. These data may be punched out and transferred to a larger computer if more elaborate analysis is desired.

## General Operation Description

The operation of the system is best described in terms of the phases through which the data is collected, compressed and transformed until the final results are output.

### Phase I -- Collection

The initial phase is concerned with the acquisition, compression and retention of the sample data obtained from the measuring instruments. During this phase, discrete sample voltages are transformed "on the fly" to centroid times for each spectral peak encountered. The number of samples and raw intensity is also recorded in the computer's memory for each peak. At the conclusion of the data collection the user has the option of punching the peak data onto paper tape for (1) processing on another computer or (2) subsequent processing on the PDP-8.

The printed results of the Collection Phase include run statistics (i.e., number of peaks, base intensity, etc.) and a specially formatted low resolution digital plot (LRDP) of the spectrum for identification of benchmark peaks of the reference compound and verification of the quality of the data collection.

### Phase II -- Conversion

The second phase performs the interpolation procedure for converting centroid times to mass values. The peak data set is made up of characteristic prominances of a reference compound superimposed on the spectrum of the sample. The researcher is required to identify from the LRDP three consecutive peaks of the reference and enter their respective mass numbers via the teletype. Once this information is supplied, interpolation proceeds automatically. The reference masses are always maintained on punched paper tape as part of the computer program and are read during the interpolation process. This tape also includes masses of minor peaks of the reference compound which are removed automatically from the final results before the last phase.

### Phase III -- Report

The final phase is designed to enable the chemical analyst to obtain a maximum of information from the mass results. He may edit and/or re-run the report many times thus gaining a variety of alternative analyses in a short period. In addition to the measured mass, logarithmic peak intensity and peak width (i.e., number of samples taken); a composition is determined for each measured mass within user specified limits. These limits include an error tolerance and basic molecular content -- C, 13C, H, N, O. Three additional user specified heteroatoms may be optionally included as part of the composition determination.

The report may be terminated at any time and a subsequent report begun immediately with modified parameters. Partial reports may also be obtained automatically.

## Operating Philosophy

This system is designed for use directly by an experimentor with little or no formal training in the operation and programming of digital computers. Familiarity with the setup of the PDP-8 console and input-output equipment is a basic requirement, however. This information is readily available in the paperback handbook.

The interface controls have been also greatly simplified and a series of "tuning" diagnostics has been included in the data collection phase program.

### Computer Setup

The procedures have been reduced to the following basic computer "skills":

1. Loading and unloading the high speed paper tape reader.
2. Typing into the teletype.
3. Actuating the teletype punch and reader if required.
4. Setting the PDP-8 console switches (in octal notation).
5. Actuating the "LOAD ADD", "START", and "CONT" keys on the PDP-8 console when necessary.

### Interface Setup

The control adjustments at the data acquisition interface are also easily understood:

1. A/D converter precision switch (12, 11, 10 --- 8 bits).
2. Sampling Rate Dial (continuous setting, two ranges).
3. Threshold Switch (five settings and off).
4. Analog Input Adjustment (continuous setting dial; three position bias control).

Once the above controls have been set properly no further alteration is required for subsequent runs from the same instrument. "Tuning" at the interface is necessary to create a proper match given the characteristics of the raw data. A complete set of diagnostic programs is built into the system so as to ease and standardize initial setup procedures.

### Tape Handling

The programmed operation proceeds through its phases without operator intervention except for the entry of requested parameters through the teletypewriter at certain points. The basic system is supplied as a single length of fan-folded paper tape which is loaded initially into the high speed paper tape reader. As the run progresses, the tape is read automatically until the hopper is exhausted. The tape is then easily removed.

## A Sample Run

### Initialization

The system tape is placed in the reader hopper and is set to load automatically. Once loaded the computer waits for the "start collection" signal from the attached equipment.

### Data Acquisition

When the start signal is sensed, the program proceeds to collect and initially compress the data. Peak data are stored sequentially in the computer's memory. This process continues until:

1. The stop signal is sensed.

2. Peak Data limit reached (560 peaks).

3. Sample limit reached (524, 288 samples).

About 10 - 30 digital samples are processed in calculating the centroid for a peak.

### End of Collection Results

At the end of data acquisition, the computer types a characteristic message containing the following information:

1. Peaks -- the number of peaks digitized whose width was greater than four samples.

2. Base Intensity -- a number which represents the sum of the voltages at the samples taken for the most intense peak found.

3. Samples Missed, Late -- an indication of the instantaneous response of interface as a function of the sampling frequency and data density above threshold.

4. Backup -- an indication of the overall response of the real-time program in on-the-fly data acquisition and compression.

### Punching Peak Data

If the user wishes to retain for later processing the information stored in the computer's memory as the result of this data collection, he may follow a pre-planned procedure to do so.

After punchout is complete, the user may continue to process the data which is still in the computer's memory or he may begin a fresh data collection by simply restarting the initial phase of the collection program.

### Low Resolution Digital Plot Output

The results of the data collection are output in a specialized format for identification of benchmark peaks of the reference compound (perfluorokersene):

```
#1  [14]
ØØØ459===6

#2  [864]
ØØØ389===7

#3  [867]
ØØØ342===6

#4  [83Ø]
ØØ3674===================22

#5  [2584]
ØØØ475===7
    .
    .
    .
    .
    .
    .
#NN  [TTTT]
IIIII===============SS
```

where #NN is the ordinal peak number, TTTT is the number of samples taken between adjacent peaks, IIIII is the absolute intensity of the peak, ======= is a line representing the intensity relative to the most intense peak (base intensity) and SS is the number of samples making up the peak.

The LRDP is not unlike the strip chart recording of the same data and a correlation between the two is often the means for positive identification of the characteristic spectrum of the reference compound.

The LRDP output may be suspended at any

time by striking the "ALT MODE" key on the teletype. The program automatically engages the paper tape reader to load Phase II once Phase I has been completed.

## Interpolation Parameters

After the Phase II program tape has been read in the computer requests the entry of the positions and masses of first 3 reference peaks. The operator must respond by typing three pairs of numbers, each pair representing an ordinate (peak number from the LRDP) and a mass number associated with that peak. The latter may be entered as an integer within one mass unit of the true value (i.e., 404.9760 may be entered directly, as 405 or as 404).

The computer then asks for the maximum allowable number of missed references and peak width. Here the user is allowed to modify the standard parameters associated with these interpolation criteria. These "built-in" constants are Ø (no missing reference peaks will be tolerated) and 40 (twice the number of samples taken for a strong peak).

The implications of changing these parameters is understood once the interpolation algorithm[2] is described. Given the centroid (time center) and mass of three reference peaks, and the mass of a fourth peak, the time of the fourth peak is predicted. A forward search is made in the peak data to find this value within one peak width. Once found, the true time is used together with the mass values to convert the centroids of the sample peaks lying between the third and fourth reference peaks to their respective measured mass values. The first reference peak is dropped, peaks 2, 3, 4 become 1, 2, 3 and the interpolation proceeds until:

1. limit of peak data is reached

2. limit of reference table is reached

3. a reference is predicted but not found within the stated peak width tolerance

If the user wishes to optionally modify the requirements for 3. above he may choose to relax the peak width tolerance or allow the interpolation to proceed even though some reference peaks were not found due to their absence from the digitized spectrum.

## Interpolation Subphase -- Reference Tape

The table of reference masses is stored on the program tape and is read sporatically during the interpolation calculation. This tape is prepared specially for the installation and contains two lists of mass values, sorted in descending order each list terminated by a zero (Ø) mass entry. The first list gives the mass values of the major peaks of the reference compound and the second list some minor peaks of the same compound usually present in the spectrum. The latter are sorted from the results prior to entering the final phase.

If the user desires to alternatively introduce his own reference tape into the information stream without modifying the system tape he may use the low speed reader on the teletype for this purpose.

## Final Phase

The Report Phase is entered automatically at the conclusion of the Conversion Phase. The user is given the opportunity to control the content, extent and in some sense the format of the results of the analysis, by completing a pre-planned dialog with the computer just before the report is printed.

## Composition Calculation and Mass Analysis Report

Content. The contents of the mass analysis report encompass all of the measurements taken for spectral peaks considered:

1. measured mass

2. peak width (i.e., number of samples)

3. logarithmic intensity (shown as a sequence of "+" characters)

For each measured mass not identified as a reference, a composition is calculated yielding an additional measurement:

1. calculated mass(es)

2. error(s) in mmu between calculated and measured mass

3. elemental composition(s) in terms of 13C, C, H, N, O

The composition calculation can be

extended, however, to include up to three additional elements specified by the analyst as part of the compositional limits (see below).

## Composition Calculation

Composition Limits. The composition calculation algorithm[3] is a combinational and highly iterative procedure. Without some logical means for limiting the number of combinations examined, excessive time can be consumed even on the fastest of computers.

These limits include:

1. a composition error tolerance (in ppm and/or mmu)

2. a maximum number of atoms anticipated per element (13C, N, O)

3. a maximum sum of heteroatoms (N, O)

4. if additional heteroatoms are considered, a maximum number of atoms anticipated for each

Entering the Composition Limits and Report Format Directions. These limits are entered via teletype as part of the analysis. The computer inquires for each of the composition limits and some additional information concerning the format of the final report:

1. extract mass limits -- the report results may be edited to list only measurements between certain mass values.

2. extract mass defect -- measured masses having a known mass defect value can be omitted from the report by entering the defect limits.

3. extract references -- the measures masses identified or determined as representing reference peaks can be omitted from the final report.

4. additional heteroatoms -- the elemental symbols and exact masses of up to three heteroatoms can be entered directly to augment the composition calculation and report output.

5. elemental compositional limits -- the maximum number of atoms expected for each of 13C, N, O and the three additional heteroatoms (if present) must be entered to enable efficient processing of the results.

6. heteroatom composition limit -- the total heteroatom content anticipated must be entered for added computational efficiency. It is not unusual for the number to be less than the sum of the individual limits for each of the heteroatoms.

7. compositional error limit -- the error in ppm and/or mmu tolerated for any composition is essential in rejecting unwanted combinations.

## Experience

The system as described has been in operation since August at Sadtler Laboratories, a commercial analytical firm. Operating on-line to an AEI MS902 mass spectrometer, excellent results are being obtained, even in the high mass range.

A second system is located at the research facility of Associated Electrical Industries, Manchester, England. Other systems are to be installed shortly at a number of laboratories in this country.

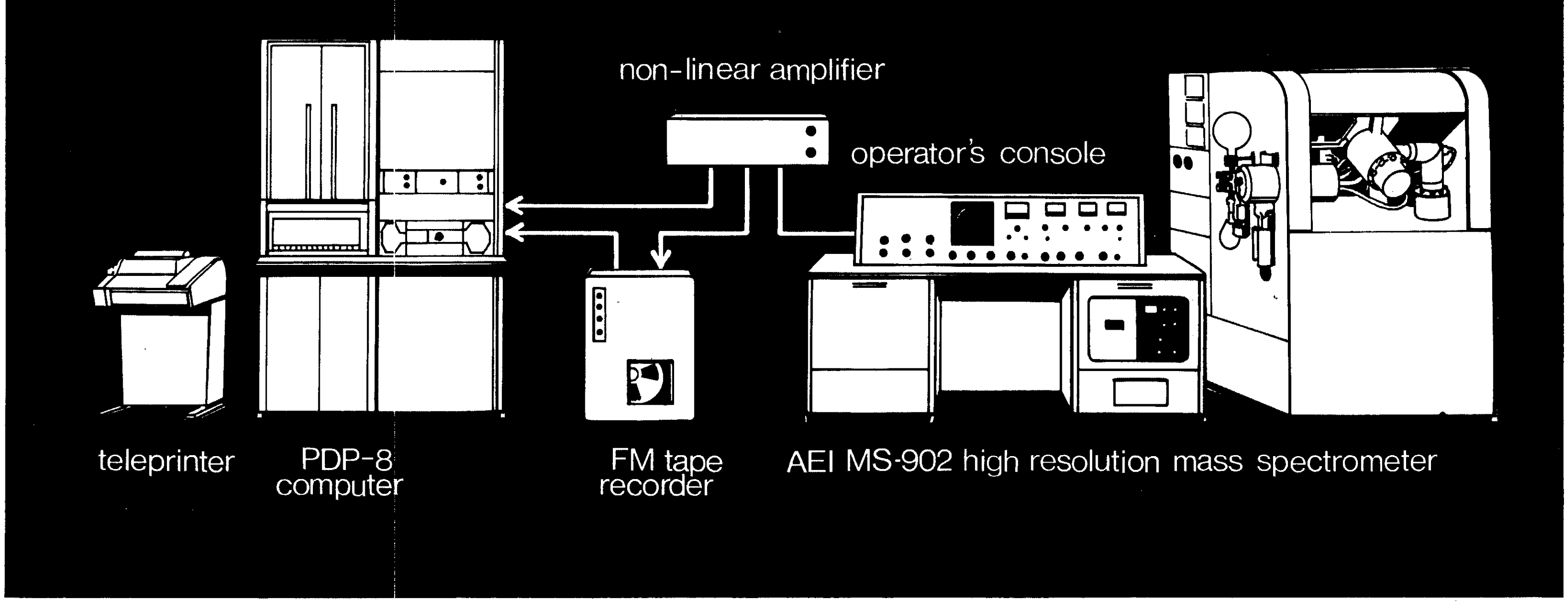
## References


1. Mass Spectrometer Data Acquisition and Analysis System Operations Manual, Picker Nuclear, White Plains, New York.

2. Fast Scan High Resolution Mass Spectrometry. Operating Parameters and its Tandem Use with Gas Chromatography, W. J. McMurray, B. N. Greene, and S. R. Lipsky, Anal. Chem., Vol. 38, Aug. 1966, pp 1194-1204.

3. Computer Techniques for the Fast and Facile Conversion of Line Positions to Elemental Compositions of Ions Recorded of Photographic Plates, D. Desiderio and K. Biemann, Proc. 12th Annual Conference on Mass Spectrometry, Montreal, 1964, p. 435.

**Mass Spectrometer Data Acquisition and Analysis System**



## PDP-9T: COMPATIBLE TIME-SHARING FOR THE REAL-TIME LABORATORY*

M. M. Taylor
Defence Research Establishment Toronto

D. M. Forsyth
Department of Psychology, Harvard University

L. Seligman
Digital Equipment Corporation

### ABSTRACT

Modifications have been made to a PDP-9 to permit real-time control of laboratory apparatus in a time-shared environment. The system is designed to accommodate about 6 independent real-time users, providing each with device service latencies of under 100 microseconds and response latencies of a few milliseconds. At the same time a similar number of interaction or background jobs may be sustained by this system (e.g., editing, assembling FORTRAN jobs). The PDP-9T system provides each user with a virtual memory space of 32K words. Physical core of the PDP-9T may be expanded to as much as 256K words. The virtual machine looks like an ordinary PDP-9 except that 1) a few instructions trap to the monitor (i.e., HLT, OAS); 2) an IOT instruction is decoded into 1 of 256 possible calls to the system monitor; 3) programs written to capitalize on the nature of the environment will run more efficiently than those which pretend to be in an ordinary machine.

An on-line computer in a research laboratory greatly increases the range of possible experiments. Indeed, experiments inconceivable without the computer may become routine.

The PDP-9T is a relatively inexpensive small time-sharing computing system based on the PDP-9. It includes hardware to permit true multiprogramming, and when a disc memory and sufficient core are added, is intended to handle about half-a-dozen real time users and a like number of users engaged in conventional conversational computing (e.g., editing, compiling, debugging, etc.) It is compatible with the PDP-9 in that almost all programmes written for the PDP-9 will run unchanged in a user machine in the PDP-9T, and in that user-designed hardware will interface to the PDP-9T exactly as it does to the PDP-9. Extra instructions are available in the apparent order code of the PDP-9T user machine. These are implemented by automatic calls to system subroutines.

*This is DRET Technical Paper #688.

### Current status of the PDP-9T

The PDP-9T is a DEC special system which starts life as a PDP-9. The hardware modifications which were designed by the designers of the PDP-9 are incorporated by the DEC Special Systems Group. The hardware is fully supported by DEC.

DEC has no responsibility for the time-sharing software which is a joint project of the Department of Psychology, Harvard University, and of the Defence Research Establishment Toronto (DRET), where the first two PDP-9Ts will be installed. A PDP-4 has been operating in the Psychological Laboratory at Harvard for four years. Since no PDP-9T has yet been delivered, software outlined in this paper must be regarded as illustrative rather than definitive.

### Configurations

The PDP-9T will accept all the standard DEC peripherals for the PDP-9, and will accommodate all the processor options except the standard (KX09A) memory protection and the parity options. The extended arithmetic element (EAE) and the Automatic Priority Interrupt (API) are standard. It will accept up to 256K (K = 1024) words of core as opposed to the PDP-9 maximum of 32K. A typical large configuration is shown in Fig. 1.

The minimum configuration for multiprogramming consists of a PDP-9T processor with EAE and API, 16K core, two special clocks, bulk storage equivalent to 4 Dectapes, and input-output terminals. The PDP-9T processor is a modified PDP-9 processor, and as a special system costs $25,000 more. One of the two special clocks has a counting period of 1 μsec. The other replaces the 16 msec standard clock and has a period of 200 μsec. In the minimum configuration control may be passed around freely among independent programmes. No core swapping is done and all simultaneous programmes must fit in core together.

To make full use of the time-sharing hardware and software, the configuration must include the standard PDP-9 disc, which is used for swapping programmes in and out of core. When the disc is added, only those segments requiring fast response to asynchronous signals need be in core permanently. Other programme segments are called into core as required. More core increases the power of the system dramatically. 256K can be accommodated, but 64K is probably adequate for most systems. A 340 display may be readily fitted, but a 339 display is added only with difficulty, and should not be considered a standard option for the PDP-9T.

## PDP-9T Hardware

The general requirements for a compatible time-sharing system operating in a real time environment include absolute protection of users from potentially destructive interference, both in respect of programme material and in respect of the operation of peripherals. The system should meet the timing requirements of any one user regardless of the requirements of any other user.

In the PDP-9T protection of core is accomplished by a mapping or paging scheme. A user writes as if his storage were a continuous region of core, whereas it is in fact divided into several pages of 2K words located in various places both in and out of core. Mapping hardware provides the linkage between the addresses written by the user and the physical addresses.

Protection of peripherals is accomplished by forbidding the user direct access to input-output (IOT) commands. An IOT instruction written by the user is interpreted by the hardware as a special kind of XCT (execute) instruction with the mnemonic XMR (execute monitor) which may lead immediately to an IOT command, but which will more often lead to a system service routine which performs the required function. Peripheral protection is thus permitted by the hardware, and implemented by the software.

Certain commands such as HLT (halt) permissible in the PDP-9, would cause difficulties to the system if available to a user of the PDP-9T. Such commands, as well as certain conditions involving the paging scheme, are trapped. The instruction causing the trap is not executed, and control passes to a system routine of top priority called the trap service routine.

### Modes

Most time-sharing systems permit operation in a variety of modes. The hardware works differently depending on the mode. The PDP-9T has three modes, of which one, normal mode, is used only to load the initial starting routines and the other two, monitor and user modes, alternate during time-shared operation. A mode flip-flop is set to 0 for monitor mode and to 1 for user mode.

Normal Mode. A switch on the console controls whether the PDP-9T operates in the time-sharing modes or whether it operates exactly as does a conventional PDP-9. When it is operating as a PDP-9, it is said to be in normal mode. When switched to "Time-Sharing" and the console locked, the system goes into monitor mode.

Monitor Mode. When the system is in monitor mode, the entire PDP-9 instruction set is available. In addition, indirect addressing can access 64K. Description of monitor mode addressing will be deferred until the mapping scheme has been described. Monitor mode is entered from normal mode, on a trap, on an API interrupt, and following some XMR commands.

User Mode. When the system is in user mode, the whole PDP-9 instruction set is available. However, HLT and OAS (Or the

Accumulator with the Switches) are always trapped, and their effect simulated by the trap service routine. They may be used in a programme, provided the programmer realizes that their effect is simulated and, in the case of OAS, that he must provide a simulated console switch register. A trap also occurs in the rare event that an interrupt request arrives during the operation of an XCT (execute) instruction whose target instruction is also an XCT. IOT commands are not trapped but are interpreted in a special way.

Mapping

In the PDP-9T, either in monitor mode or in user mode, no memory reference generated by the CPU ever addresses core directly. A programme runs in a virtual memory logically indistinguishable from a PDP-9 memory. All PDP-9 addressing techniques are legal although in monitor mode the indirect address field is 16 bits long instead of 15 bits. The concept of a virtual machine is useful in visualizing the operation of the system. The user sees a 32K virtual user machine whose mapping onto the physical machine is determined by the contents of the user map.

From the system's point of view, the virtual memory is split into pages of 2K words. Some of these pages are in physical core, some are in backup storage, and some may have no physical existence at all. The actual location of the pages of a virtual memory in no way affects their addressing in a programme.

In the PDP-9 each logical address is 15 bits long. The PDP-9T hardware interprets these 15 bits as containing a 4 bit page number and an 11 bit address within a page. The page number is used to select one of 16 mapping registers. The 16 registers together form a map of the virtual memory onto the physical memory. There are 2 maps, one generally active in user mode, and one generally active in monitor mode. The organization of the mapping function is shown in Fig. 2. The mapping process requires that the cycle time of the PDP-9T be slowed to 1.2 μsec from the 1 μsec cycle time of the PDP-9.

The contents of the selected mapping register indicate where the page is to be found, and whether or not it is a read-only page. If the page is in physical core, the least significant 7 bits of the contents of the mapping register indicate which page. These 7 bits are concatenated with the 11 bit address within the page to form a complete 18 bit physical address. The PDP-9T will thus accommodate 256K of core.

If the logical page is not in physical core, the most significant bit of the mapping register is set. This is the MO (Memory Out) bit. If a mapping register with this bit is selected, a trap occurs regardless of the machine mode.

The second bit of the register is the RO (Read Only) bit, and is set if the page is designated as a read-only page. If an instruction which would alter the addressed location selects a mapping register with the RO bit set, a trap occurs. This read-only facility is not available on the PDP-9. It permits a user a certain degree of protection against his own stupidities; a more important function is to permit the writing of software to be shared among users. Pure procedures may be written into RO pages and mapped into several users' programmes without mutual interference. The writing of pure procedure segments is extremely difficult in the PDP-9.

The map contents cannot be read into the machine. They are loaded from core through a cycle stealing facility like the DMA (Direct Memory Access). A map image is stored in a core buffer, two registers to each of 16 consecutive core words. The map image is loaded into the maps in 16 machine cycles, permitting easy changes among virtual machines.

Monitor Map and User Map. There are two maps, identical in operation, the monitor map and the user map. They differ only in that they are invoked in different conditions.

When the system is in monitor mode, either map may be invoked. The mode flip-flop is set to 0, and during any direct memory reference, the monitor map alone may be invoked. In an indirect memory reference, however, 16 bits (bits 2 - 17) of the indirect word determine the effective address. If bit 2 is zero, then the monitor map is used; but if bit 2 is 1, then the user map is invoked. The virtual monitor machine thus behaves in some ways as if it had a 64K memory space. It can address any word in the 64K space when the instruction being executed is in the lower 32K of the space. The upper 32K of the space coincides exactly with the 32K address space of the virtual user machine. If any command results in a jump into the user space, the mode flip-flop is set to 1. In other words, a transfer from monitor mode to user mode is accomplished by a jump into the virtual user machine. This is the only way the system can get into user mode.

When the system is in user mode (mode flip-flop set to 1 state), the user map is invoked except during an XMR command, when the monitor map is invoked using the special addressing structure of the XMR (see below).

If any operation involving a JMS (jump to subroutine) is performed, the state of the mode flip-flop is set in bit 2 of the return word. For example, if the system is in user mode when an API interrupt occurs, the JMS sets bit 2 in the return word of the monitor mode service routine, so that the return jump automatically returns control to the interrupted user mode programme. However, had the system been in monitor mode when the interrupt arose, the same return jump would have left the system in monitor mode to continue the interrupted operation.

XMR Vector and Vector Service Routines

The instruction which has the form of the PDP-9 IOT command has an entirely different function in the user machine of the PDP-9T. It is still used to request input-output operation, among other things, but its operation is like that of an XCT with special characteristics. It is given the mnemonic XMR (execute monitor).

The operation of an XMR is sketched in Fig. 3. The format consists of 4 bits of operation code, 8 bits of function code or address field, and 6 bits of microcode. In octal, it may be symbolized XMR abbxx. The function code is abb, the microcode xx.

When an XMR is executed, it addresses an instruction held in a vector or table in the monitor machine. The instruction is in location $(400 + abb)_M$ where the subscript M indicates Monitor. The instruction found is executed differently depending on which of three classes it belongs to.

1) All instructions except IOT and jump instructions are executed exactly as written and the user programme continues with the instructions after the XMR. The monitor map is used to determine the memory reference of the instruction in the vector, permitting the user access to certain useful data contained in the monitor space, (e.g., real time clock).

2) If the command in the vector is IOT cddyy, the instruction actually executed is IOT cdd-(xxVyy). The device code of the vector entry is used, and the microcode of the XMR is ORed with the microcode of the vector IOT. Normally the vector IOT would have a zero microcode, permitting the user to programme the peripheral device directly. Through this facility, users may maintain direct control over their own peripherals with only 1 cycle overhead for each IOT issued. IOT instructions in the vector are changed with the users.

3) If the command is a JMP or JMS, or an XCT of either of these, priority is raised to API level 4 (top software priority) and the jump executed. If the command is JMS, the return word will have bit 2 set, ensuring that the return jump from the subroutine will pass control back to the user machine. The JMS provides an entry into a VSR (Vector Service Routine).

Whenever a VSR is entered, the function code and the microcode of the XMR are stored in a special register which can be read by the VSR programme. XMRs thus provide entries into potentially 256 different VSRs, and each VSR can perform up to 64 different functions under control of the microcode field of the XMR. Standard VSRs intended for the PDP-9T system include floating point arithmetic operations, communication functions, string processing operations, and file manipulation, particularly input and output operations. VSRs are also used for communication between user programmes and the system.

The VSR system is the heart of the PDP-9T. The system service VSRs deal with all control aspects of a programme. They handle the real-time requests of the user (e.g., Output current AC contents to device 23 at time T + 35 msec.). They permit the reassignment of virtual memory (e.g., Make page 5 read-only; or Attach page 3 to this segment). They permit system resource reservations, which can also be made through the command language at the user's control teletype.

The group of functional VSRs provide an apparent large extension to the PDP-9 order code, facilitating assembly level programming and reducing the amount of core needed by the user for his own programme. The user machine appears to have floating point and string processing hardware, as well as pushdown stacks and jumps. These operations naturally take up processor time, but very little more than would be needed if the subroutines were in the user's space and deliberately used by him as library routines.

Traps

The action of a trap is like that of an API interrupt at level 0, the highest priority in the API. When a trap condition arises, control passes immediately to a trap service routine common to all traps, and priority goes to level 0. The type of condition causing the trap is held as a bit or bits in a special register known as the trap register, which can be read into the AC. Possible trap causes are MO (Memory Address not in physical core), RO (Attempt to change a read-only location), HLT (Attempt to execute a halt instruction), OAS (Attempt to address the console switches) and $XCT^2_I$ (an interrupt request arrives during the execution of an XCT instruction whose target is an XCT). In the case of HLT and OAS which are microprogrammed instructions, all instructions microprogrammed with them are completed before the trap occurs.

In the case of an MO or RO fault, the trap register holds a bit indicating whether the error occurred during the Fetch phase of the instruction. The least significant 5 bits of the trap register also contain the number of the mapping register causing the fault, so that quick action to clear the error may be made.

The trap service routine does enough at level 0 to determine the cause of the trap. It then in effect drops its priority so that correction measures may be completed at a lower level, and hardware interrupts may proceed unhampered. Eventually, the cause of the trap will be cleared, and the offending routine may proceed if it needs to do so.

Hardware Summary

Three major modifications to the PDP-9 afford the facilities for flexible multiprogramming. Mapping the logical addresses onto physical core gives memory protection, and permits parts of programmes to be held elsewhere than in core when they are not immediately needed. The use of the IOT command as a monitor execute (XMR) instruction gives the user access to system service routines, while permitting him to control his own peripheral devices without danger to other users. The trap system acts in potentially dangerous situations, and permits them to be corrected. All PDP-9 programmes will run, although programmes written especially for the PDP-9T will normally run much more efficiently than will standard PDP-9 programmes.

PDP-9 Software

Since no PDP-9T has been delivered,it is too early to go into the details of the software. Although the main structures of the monitor system have been designed and to some extent coded, there may yet be significant changes when the designs are exposed to the text of actual operation.

Files and the Command Language

The user machine in the PDP-9T communicates with the outer world by means of files or through directly controlled peripherals. Directly controlled peripherals are handled through the XMR command, the monitor having entered the appropriate IOT commands in the transfer vector when the user machine was activated. IOT commands may be assigned to XMR commands through console interaction using the command language. Most interaction with the outer world is done through file handling VSRs. The regular input-output VSRs available to the user machine take items from the head of a sequential file or append items to the end of a sequential file. Any item to which the user has access before his programme or a software subsystem begins to run is a file. His programmes are files, lists of data are files, chapters of his uncompleted book are files, directories to files are files. Some peripherals such as the teletype or the line printer are files. The files all have names, either given by the system or freely constructed by the user.

A file is known by number to the binary user programme, and by name to the system. Input-output devices have names which may be used as if they were file names, so that input/output connection with the real world is accomplished simply by defining the file to be a peripheral device. For example, while a programme is being debugged, its output may be sent to the control teletype, or possibly to the line-printer. Later, its output is needed as input to another programme, and will be sent to a named file stored on the disc or on Dectape. The linkages between the numbered "ports" in the user programme and the different destinations are made at load time, through the command language.

The Operator and the command language

The user controls his virtual system through the command language. He can regard this language as permitting him to talk to an operator who gets files from storage and mounts the proper tapes if they are not already installed, who makes reservations for system resources, who activates major software systems such as the Fortran compiler or the editor, who sets up file linkages, and so forth. Some of the functions of the operator are performed by a human operator in the computer room, some by a software module within the computer. The larger the system, the less of the operator's work is performed by the human. For example, a system using Dectapes as the major backup storage medium will frequently find that wanted files are on unmounted tapes, whereas a system with larger tape files, or a large mass storage device, will usually find all files on line.

The command language has macro facilities. A sequence of elementary commands may be defined as one single new command. Operands of the elementary commands that vary from call to call of the macro may be defined as parameters, which appear in the macro call as operands of the new command. A user may define a macro for his own benefit, accessible only to him or to those specifically denoted by him to have access; he may also use any of many system macros whose operation is the same for all users. An example of a system macro is the EDIT command. EDIT FROM JOE TO BILL is equivalent to the elementary sequence:

ASSIGN JOE AS INPUT TO EDITOR

ASSIGN BILL AS OUTPUT FROM EDITOR

ASSIGN HERE AS FILE 1 FOR EDITOR

LOAD EDITOR

START EDITOR

HERE is the control teletype, and denoting it as file 1 means that it serves as both input 1 and output 1. Non-standard editing may not require the use of the control teletype, in which case the macro would not be used; rather, the proper sequence of elementary commands would be written in full.

Time allocation in the PDP-9T

The natural unit of programming in the user machine is the task. A task is a series of instructions performed sequentially. In an interrupt environment in a regular PDP-9 the main programme would be one task and the interrupt handler another. There is no meaningful sequential relationship between instructions in the two tasks.

In the PDP-9T there are 5 different types of task differentiated by their timing requirements. The fastest and slowest tasks are reserved for system functions, the other three being available to the user.

1) Hardware interrupt level routines
All device interrupts come through the API system. The PDP-9T has no ordinary PI line, and the API is standard. The API defines only 4 priority levels for hardware interrupts, but any of an indefinite number of devices can raise an interrupt which transfers control to a location unique to the interrupting device. An interrupt waits for as long as the flag is raised. If the service routine does not clear the device flag, the interrupt request will be renewed immediately the routine terminates. A hardware level interrupt service routine is a system routine which simply transfers whatever data the device requires and clears the device flag. If the device requires further service, a request is entered for a low priority service routine. Responses to hardware level interrupt requests should be just as fast as they are on a standard PDP-9.

2) Guaranteed maximum latency (GML) tasks. When a task must be completed within a specified period of the wakeup signal, the user requests a GML task reservation. GML reservations are available in a few precisely defined grades. Within each grade of GML service, only a finite amount of computing time is permissible, which must be shared among all tasks having reservations at that grade. After subtracting system overhead and a small allowance for autonomous memory transfers and hardware level interrupt handling routines, up to 100% of the remaining CPU time is shared among the GML task reservations. A task requiring service of a particular grade must specify at reservation time the maximum duration of a computing burst, and this duration in combination with the maxima declared by all the other tasks at the grade must not exceed the computing time allotted to that grade.

3) Requested Latency (RL) Tasks. The organization of RL tasks is essentially the same as that of GML tasks. There are specified grades of service. The tasks have to specify the duration of a service burst and the minimum interval between calls.

RL tasks are executed in the time left over when a GML task does not use its full allotment of time, or is not ready to go during some queue scan. Although the GML tasks have first call on 100% of the machine time, it is extremely unlikely that over any extended period they will use an appreciable fraction of the time. Hence, in practice, almost all the computing time will be available to the RL task group.

All of the computer time after overhead is available for reservation by RL tasks, just as with GML tasks. Also as with GML tasks, it is extremely unlikely that the RL tasks will use up any appreciable fraction of their allotment over any extended period. On any occasion when a task uses less than its burst declaration, or is invoked less often than it has allowed for in its reservation, time is left over for background tasks.

4) Background Tasks. After 100% of the CPU time has been allotted to GML tasks, and another 100% to RL tasks, considerable time will generally be left over. This time is devoted to background tasks.

Background tasks are run in the time not used by RL tasks. The scheduling algorithm is a modified round robin. It tends to give tasks that finish quickly many short bursts of computation, and long grinding tasks few long bursts. When a new background task is introduced, the monitor assumes it is a quick task, and puts it in a queue for quick but short service. If it terminates before the end of the service burst, well and good. The task will be in the same queue next time around. Interpretation of a line of code in a compiler might be such a task. If it does not terminate but is stopped for time overflow, the next time around it will be in a queue for less frequent, but longer service. In this way, tasks at background level all have the same total amount of computing time permitted them, but those that finish quickly get more frequent opportunities to compute.

Almost all user tasks will be at background level. All those not used in running an experiment in real time will be at this level, including those involved with conversational operation. It seems realistic to assume that response times should be not much more than 1 second with the designed complement of 6 conversational users operating simultaneously.

5) Twiddle tasks. The user never sees twiddle type tasks, and they are described here only for the sake of completeness. Twiddle tasks are tasks essential for the maintenance of the system, but which can be done in almost any time that is left free. They include garbage collection on the disc (organizing disc files for easy recovery, and erasing obsolete and possibly misleading information), linearizing Dectape files (which are initially written out in the speediest fashion without regard to linear order of blocks), running system diagnostic programmes (the lowest priority task of all, and therefore probably running for the longest time) and so forth.

## Summary

The PDP-9T is a small time-shared computer designed to work in real time for several simultaneous users. It is a modified PDP-9 with memory mapping in pages of 2K words, traps for potentially damaging instructions, and a versatile hardware system of interpreting input-output instructions as calls to a wide range of system routines. The software system permits users to programme in terms of tasks which have independently specified requirements for response latency and core usage, and which may communicate with the real world as well as with each other.

The PDP-9T in a reasonable configuration is expected to handle about 6 users running independent experiments with response time in the tens or hundreds of milliseconds, plus a like number of users operating in more conventional conversational mode. The input-output flexibility of the PDP-9 is retained, so that experiments are very readily interfaced with the system.

## Acknowledgment

The time allocation scheme outlined in this paper is due to R. Strom of the Psychology Department, Harvard University.

The work at Harvard was supported by : Grant I R01 GM 15258.

TYPICAL LARGE PDP-9T CONFIGURATION

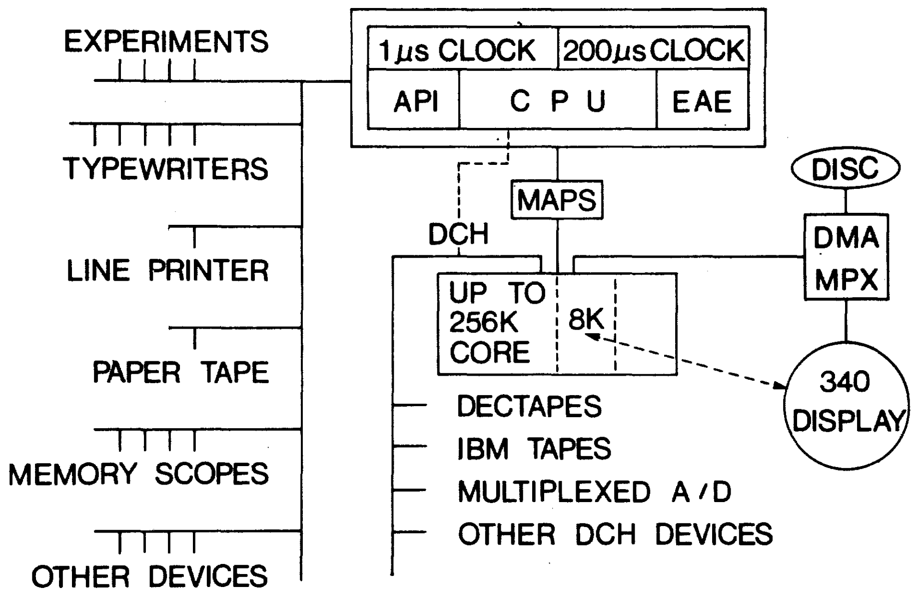



Fig. 1 Typical large PDP-9T configuration, showing the variety of peripheral devices that may be attached to the system.

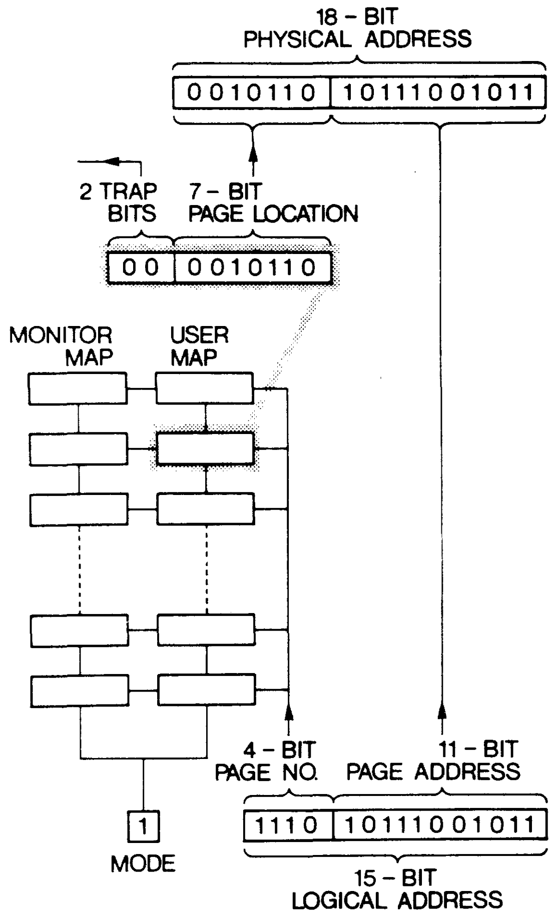
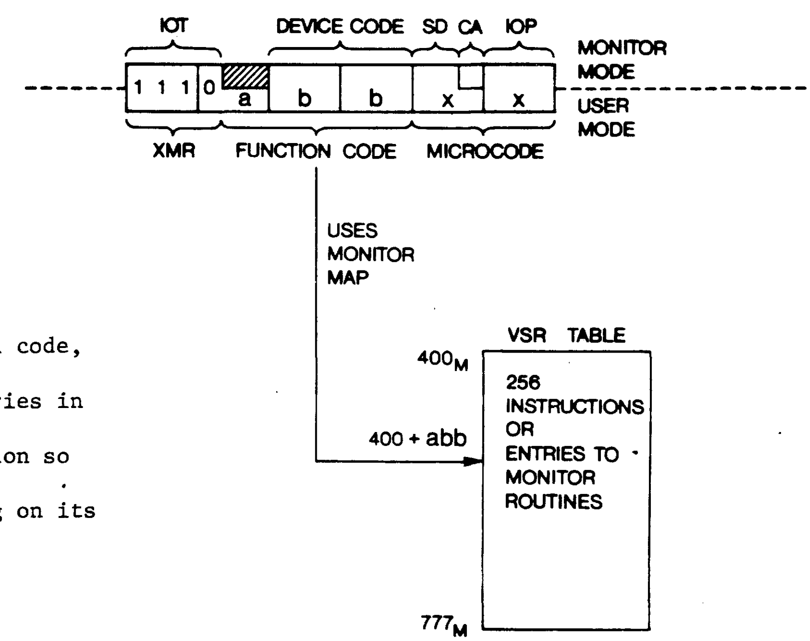



Fig. 2

The mapping system, showing how a logical address provided by the CPU is mapped into a physical address which locates the word in physical core. The logical address is $72713_8$ in the user machine. The map register addressed is number $36_8$ (user map register $16_8$). Its contents indicate that the page is physical page number $026_8$, and the complete physical word address is $132713_8$.

Fig. 3

Operation of the XMR command. The function code, or address field, points to one of 256 entries in the VSR table or XMR vector. The instruction so addressed is executed in a manner depending on its class.




Functions of code 70 in monitor and user mode

# A DIGITAL ELECTROCHEMICAL CONTROL AND DATA ACQUISITION SYSTEM

George Lauer and R. A. Osteryoung
Science Center
Aerospace and Systems Group
North American Rockwell Corporation
Thousand Oaks, California 91360

## Abstract

Real time, on line data acquisition, is in itself not a novel concept. However, the majority of work reported to date has been concerned with either some time sharing scheme or with a small computer dedicated to a given task or instrument. We report here a system built around a standard PDP-8, which is sufficiently versatile that a number of standard laboratory instruments and techniques can be directly attached with a minimal amount of wiring.

## Introduction

Our area of interest is the investigation of electrochemical phenomena at the electrode-electrolyte interface. The nature of the experimental problems is such that the system must be fairly versatile.

The majority of techniques which we use to investigate interfacial phenomena require fairly fast rising waveforms which must be absolutely clean (no overshoot, ringing, etc.). Timing is very critical, and a large number of data points are desired over a relatively short time. The experiments are usually non repetitive, *i.e.*, single shot. The chemical system is placed into an equilibrium or steady state with respect to the parameter of interest and is then perturbed; the response to the perturbation is measured and fitted via standard regression techniques to the model being considered.

A typical technique is illustrated by the method of potential step chronocoulometry. The waveforms observed during a run using this method are shown in Figure 1. The top is the perturbation pulse with $\tau$ (tau) being on the order of 10-20 milliseconds, and the total experimental time being on the order of 20-60 milliseconds. The lower waveform is the response, which is the charge passed as a function of time. In order to obtain a meaningful fit to the model and to obtain reasonable statistics a minimum of 150 points must be gathered during the course of the run. The time at which each data point is obtained as well as the value of $\tau$ (tau) must be known with a very high order of accuracy as these values are used in the analysis of the data.

The basic control circuit used for such an experiment is shown in Figure 2. This type of analog system is commonly employed in electrochemical studies. The perturbation pulse,as well as any steady state bias, is fed in at the input of the control amplifier (1) and the response is measured at the output of the integrator amplifier (4). It should be noted that the input pulse could be supplied with a simple pulse generator, and the response obtained with an oscilloscope and a camera. In fact, during the original development of the technique, such a method was used. However, the data obtained from pictures is far from accurate. Reading data off oscilloscope pictures is both inaccurate and tedious; particularly as a large amount of data is required.

## Digital System

We have assembled the system described below to improve the accuracy of our experimental data as well as to make getting the data a less tedious task. We purchased from Digital Equipment Corporation a PDP-8 programmed data acquisition unit (NOT A COMPUTER) with type 182 EAE, a type 138 E analog to digital converter, a PC01 high speed reader-punch, and we recently added a DF-32 disc pack. We built and interfaced a real time clock with 1 microsecond resolution, three digital to analog converters, an integrating analog to digital converter, a relay driver, four general purpose field effect transistor switches, a solenoid driver, and a laboratory interrupt system. We also added a Burr-Brown type 1673 sample and hold module to the type 138 ADC.

Standard DEC R, W, and A series modules were used throughout. A block diagram of the digital to analog converters is shown in Figure 3. Each of these can be set by a jam transfer of the information from the PDP-8 accumulator. The measured rise time (5% to 95%) is 450 nanoseconds and no overshoot. The output of two of the D to A's is fed into high speed operational amplifiers (Analog Devices type 201) to allow level shifting. We do not use the DEC type 702 or 704 power supplies as we have found it far more economical to use a standard laboratory type to run all three converters. These also have a lower equivalent noise at the output. Care was taken to separate "digital" from "analog" ground and these were brought together at only one point in the system. The "intensify" control is used for oscilloscope displays in conjunction with the converters. This arrangement is considerably cheaper than any display commercially available and performs the same function. There are many instances when a hard copy graphical display of the data is extremely desirable. To obtain such hard copy we attached a Moseley type 7000 X-Y recorder with type G2B null detector to the computer and used it in conjunction with the digital to analog converters. This arrangement is cheaper by a factor of at least four than purchase of a standard digital plotter and additionally it is far easier to program. It is admitted that we are unable to reproduce the Mona Lisa with our plotting system; however, we feel, with some surety, our requirements will not be that exacting in the foreseeable future.

A block diagram of the real time clock is shown in Figure 4. The base period of the clock is set externally via a potentiometer over the range of 1 microsecond to 10 milliseconds. The number of base periods desired is set into the clock register from the accumulator. The clock counter is cleared and started by an IOT pulse; when the contents of the clock register equals the contents of the clock counter a pulse is output and the counter is cleared. Thus, pulses can be programmed over the range of 0 to 4096 times the base period. The clock can be started and stopped by external events and by IOT commands, and the contents of the clock counter can be transferred to the accumulator. Thus, external events and internal programs can be accurately timed if desired.

The field effect switches are simply an A121 multiplexer module tied to R204 flip/flops controlled by IOT pulses. These allow very fast switching of signals where the 480 ohms "on" resistance is not a major factor. The relay control consists of an R204 flip-flop controlled by IOT pulses controlling a W051 driver. This driver is used to set or reset a mercury wetted relay. We have added a skip command to the relay controller so that actual contact closure may be sensed.

The integrating analog to digital converter consists of a Vidar type 241 voltage to frequency converter whose output is connected via a W520 comparator to a 12 bit up counter made up of R202's. The output of the counter is tied to the PDP-8 accumulator input. The counter is gated on for 4.096 milliseconds during each acquisition via an IOT. We thus have 12 bit accuracy as the full scale output of the voltage to frequency converter is one megahertz. Again the savings obtained by building this unit from components as opposed to purchase of a complete integrating ADC is considerable.

A block diagram of the interrupt system is shown in Figure 5. The system was designed to operate in a manner similar to the operation of the trigger circuits of conventional oscilloscopes. Standard negative going pulses (0 to -3 volts) or positive going pulses (-3 to 0 volts) as well as level changes on a given slope can be used to interrupt. These signals set a flip/flop, which in turn causes the interrupt. This flip/flop must be reset under program control after each interrupt is received. The signal can be blocked by setting the blocking flip/flop under program control. This feature is useful if another device on the computer capable of interrupting is to be used. We have found it advantageous to disarm, via switches all other interrupts such as the ADC, high speed reader, etc. This is simply accomplished by grounding one input of the proper gate attached to the PDP-8 interrupt buss.

A block diagram of the system as used for double potential step chronocoulometry is shown in Figure 6. We have, to a certain degree, "closed the loop." The computer applies the perturbation, and accepts the response data. The analog system, with its very high bandpass is ideal for actual control of the desired parameters, the digital system is ideal for diming, application of the pulses, and for data acquisition. In effect, this is a relatively happy marriage of the best features of the analog and digital functions.

We have found that the PDP-8 is quite useful for processing of the data and general purpose computations. In the particular experimental procedure described we ran a standard linear regression analysis on the raw data using the floating point package and obtained the desired answer,in the proper units, immediately. The programming is not particularly difficult, and we conserve core by keeping the data stored in fixed point and floating it whenever necessary. Only the minimal number of floating point numbers - constants, running sums, etc. - are stored. Normal computation time for 500 points is on the order of one minute. Precision is greater than that obtained using the IBM 360 equipment in single precision mode.

We have mounted the whole system on a cart and thus we can bring the PDP-8 to the instrument, rather than laying long lines. The interfacing we have described makes it a fairly simple task to attach a mass spectrometer, a gas chromatograph, or any of a large number of other analytical instruments. The prime effort then becomes a matter of software. It is not often that a requirement exists such that two instruments in separate laboratories require digital control at precisely the same time. One can usually wait for an hour or two. The system we have described offers many advantages over time sharing. It allows the experimentalist a great deal of flexibility without having to consider other users at the actual run time.

The cost of the system, while considerable in terms of a single laboratory instrument, is reasonable when considered as a general purpose device. In matter of fact, the cost is of the same order as standard hardware programmed acquisition devices such as multichannel analyzers, computers of average transients, etc.

The primary disadvantage we have encountered so far is that the experimentalist must become aware of the system's capabilities and interfacing in order to use the system to advantage. This sometimes results in a reorientation of primary interest to computing; the authors are not excluded from this problem.

The other disadvantage is political. Those colleagues who have used the system find it an extremely useful tool, and one must often reject kind offers from them to increase the working space in the laboratory by placing the system permanently in their own labs.

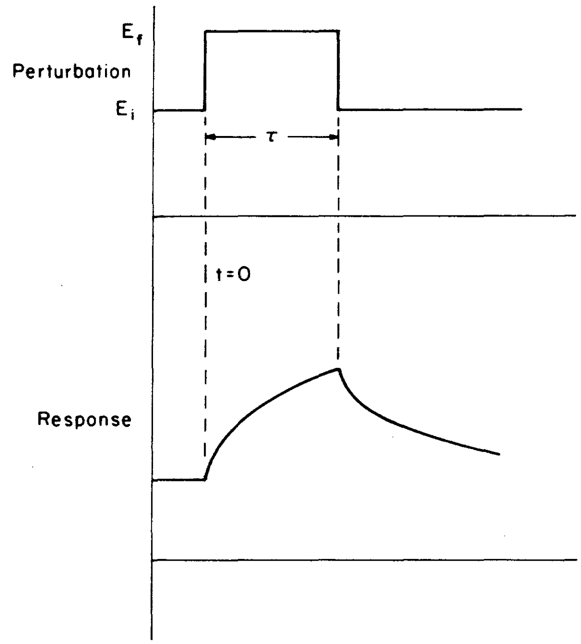



Figure 1 Perturbation (tau) and response (bottom) waveforms observed in double potential step chronocoulometry.

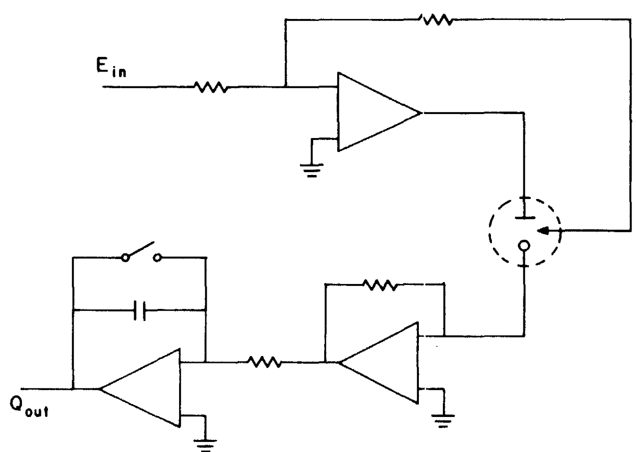



Figure 2 Analog electrochemical potentiostatic control circuit. Operational amplifiers are all analog devices type 201.

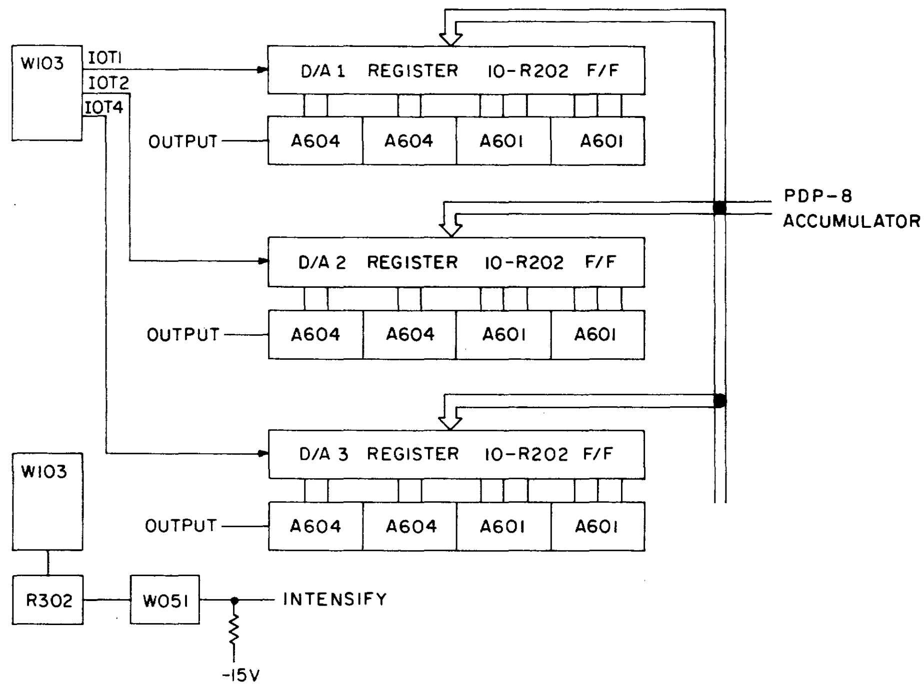



**Figure 3** **Block diagram of digital to analog converter and control. Numbers refer to DEC module types.**

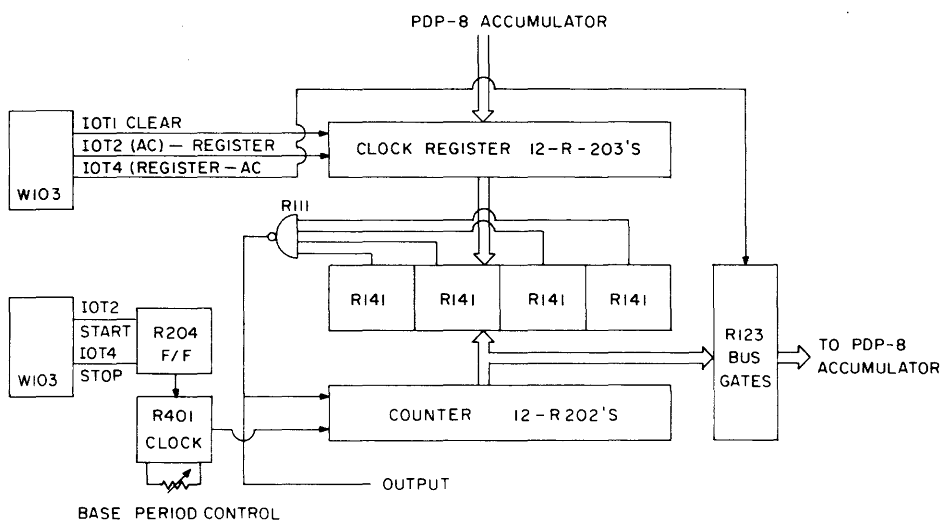



**Figure 4** **Block diagram of programmable real time clock.**

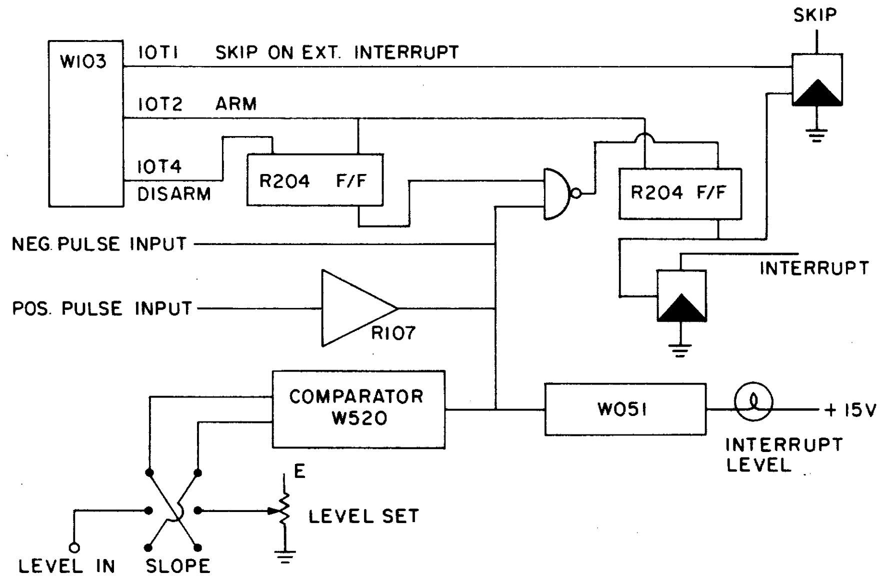



**Figure 5 Block diagram of laboratory interrupt circuitry.**

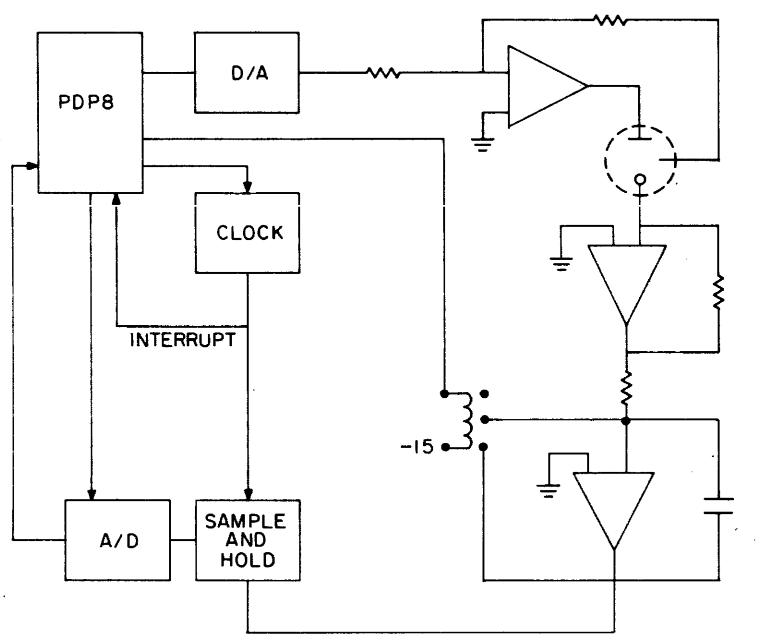



**Figure 6 Block diagram of complete set up to perform double potential step chronocoulometry.**

UCRL-70639

# TIME-SHARED COMPUTER CONTROL IN ANALYTICAL CHEMISTRY

Jack W. Frazer
Lawrence Radiation Laboratory, University of California
Livermore, California

## Abstract

The PDP-7 system in Analytical Chemistry at the Lawrence Radiation Laboratory illustrates the real-time, time-shared instrument control application. The monitor program to allow multiple-asynchronous operations to function without interaction will be described. Various analytical applications that are in current use will be abstracted. In addition, future concepts will be described and anticipated limitations discussed.

## INTRODUCTION

Approximately three and one-half years ago the Analytical Chemistry Section at the Lawrence Radiation Laboratory embarked on a very limited program to automate certain analytical instruments.[1,2] In January of 1966 we received a PDP-7 control computer to implement the automation effort. This computer was to be used for the control of laboratory equipment, data acquisition, and on-line computations. It was anticipated that these computer functions would improve accuracy, add new dimensions to chemical experimentation and provide more time and facilities for research. This paper outlines our progress to date.

The first automation undertaken was carried out via the time-shared mode as implemented by the PDP-7. This computer is a standard model with 8K of core memory, a 1KHz clock, a variable boundary register for memory protect, oscilloscope display and a single level interrupt. Since computer-control of laboratory equipment was a new and relatively unexplored field of endeavor, we started by automating some of our more routine analyses. As anticipated, the automation has envolved problems in all areas; hardware interfacing, software, and design of experimental apparatus. The statement that "automation is a systems problem" is too often ignored, and the experimenter attempts to find a solution using only that discipline, such as software, that he personally likes and best understands. The three apparatuses to be described briefly here were all designed and built at LRL. In order to accomplish the automation, it was necessary to develop special hermetically-sealed vacuum valves and programmable cold/hot traps, to rebuild mass spectrometers so that all variable functions required to perform routine analyses could be controlled by the computer, and to develop a digitizing circuit to be used in conjunction with pressure transducers. These are only a few examples of the equipment designed and built specially for this project.

## APPARATUS

Figure 1 is a schematic of the present system. As indicated, three units are presently operative. The other four are nearing completion or are completed and now require hardware debugging and appropriate software. Figures 2, 3 and 4 show the three apparatuses now in operation.

### Determination of Carbon, Hydrogen and Nitrogen in Organics

Figure 2 shows the vacuum apparatus used for the simultaneous determination of carbon, hydrogen and nitrogen in organic compounds. A schematic of the unit is shown in Figure 5. All valve[3] manipulations, Toepler pump control, cold trap temperature control and pressure measurements are performed under computer control. As a general description[4-7] of the analytical procedure, 3-5 mg of the sample to be analyzed (gas, liquid or solid) is weighed into a quartz tube along with one gram of copper oxide. The tube is then evacuated, sealed and fired at 925° C. Upon cooling, the bomb is transferred to the vacuum analytical apparatus, opened and the combustion products, namely the $CO_2$, $H_2O$ and $N_2$, are separated and measured as $CO_2$, $H_2$ and $N_2$.

A summary of this data is given in Table 2. Note that the lack of any inherent integrating characteristics in the particular hardware used for data acquisition can result in increased error.

The automation of this apparatus was also evaluated and executed as a system. As examples, the gray to binary encoding is a relatively simple task to perform by means of hardware. By so doing, computer operating time was saved as well as core memory. When the various methods of data acquisition were being tested, the change in bandwidth response of the acquisition system disclosed additional noise not detected with a 1 sec. full scale recorder. Rather than use hardware or software filtering techniques, the noise was reduced at the source by over a factor of $10^5$. This was accomplished by extensive shielding and the modification of all the proportional temperature controllers used on this apparatus. These seemingly trivial details have been mentioned because it has been our observation that too often automation is being undertaken almost entirely from the computer's point of view, without due consideration being given to the system as a complete functional unit.

The third apparatus automated was the vacuum fusion unit used to determine trace amounts of oxygen, hydrogen and nitrogen in metals. This unit (Fig. 4) is shown schematically in Figure 8. The computer performs all necessary valve operations, sample selection, sample admission to the system, pressure measurements via a Baratron pressure transducer, and also operates and performs data acquisition for the mass spectrometer.

A typical analysis is performed by dropping the metal specimen into a graphite crucible contained in the furnace (Fig. 8). The crucible is preconditioned by outgassing at ~2200° C in a hard vacuum. Next, a loading of 20-30 grams of platinum is added to the crucible and outgassed at 1900 to 2100° C. Upon the completion of this step the crucible and platinum bath are controlled at a fixed temperature in the range of 1900 to 2100° C. Metal specimens weighing from approximately 100 to 500 mg are then added. The CO, $H_2$ and $N_2$ obtained from the reduction are measured and the corresponding oxygen, hydrogen and nitrogen content in parts per million calculated. As an indication of the sensitivity of this particular apparatus, oxygen in tungsten can be detected at the 0.2 ppm level.

Many problems were encountered in the construction and operation of this apparatus. The automation of the mass spectrometer (residual gas analyzer) resulted in some "noise" problems which often resulted in halting the computer. However, the electrical driven valves (CCV) produced the most noise, and extensive electrical filtering was required before they could be controlled by the computer. Also, there were many problems associated with the mass (m/e) range scanning by the mass spectrometer. Although the unit is now operative, a systematic study of its operating characteristics and accuracy have not been completed.

<u>Monitor</u>

The basic format of the control monitor is shown in Figure 9. As the boundary register, trap, and disc facilities are added, certain minor changes will be required; however, the main structure and logic presented here will be maintained. This monitor is designed to perform control and data acquisition functions for ten analytical apparatuses.

As stated above, there is only one interrupt level. Upon execution of an interrupt all pertinent information is saved and then the clock is interrogated. If it is not a clock interrupt, the individual I/O units are checked and an appropriate software flag set in the waiting loop portion of the monitor. If it was a teletype interrupt, the information is taken and used as the flag. This is followed by a standard interrupt exit.

If the interrupt resulted from the clock, all software clocks associated with apparatuses currently being controlled are incremented. Should service be required for any unit, an appropriate flag is set in the waiting loop. If high speed data acquisition is required, that is performed immediately. The interrupt exit is saved in a push-down stack and the exit then is to the beginning of the waiting loop.

The priority of data and control processing is currently that shown in Figure 9. No waiting loops are allowed in the control or communications program. All waiting must be done in conjunction with the waiting loop of the monitor. This loop dictates the priority of operations. This monitor does not dynamically allocate time. Instead, time is prescheduled by the program structure.

The temperatures of the traps B and D are controlled by the computer.[8] As an example of their versatility, trap B can be changed to any desired temperature between $-180^\circ$ C and $+100^\circ$ C merely by changing one word in the program. In addition, the inlet of each trap contains a short zone where multiple sublimation-condensation stages are performed under computer control. This last feature is used as a means of obtaining large separation factors for the three gases.

Pressure measurements as required for the manometric determinations are made in the gas buret G by a 10 torr full scale MKS Baratron* pressure transducer. As mentioned above, a digitizing circuit was developed for this transducer. By carefully controlling the transducer temperature, pressure measurements reproducible to better than 1 bit in 15 bits are obtained.

The entire analysis is performed under computer control with the single exception of opening the quartz bombs. From years of experience, we know that once in a great while the bomb holder will be broken in the process of opening the bomb. Should this happen under computer control it would be necessary to have appropriate sensors on the apparatus inlet to avert major failure and breakage of the main analytical apparatus. The required safety system is far more expensive than the convenience warrants. As an alternative, a manual system was devised. An intermittent buzzer is turned on 60 seconds before a quartz bomb is to be opened. At the end of the 60 second warning period the chemist has 20 seconds to open the bomb, during which period all valves are closed. Should the holder be broken during this operation, the chemist has merely to operate one toggle switch to avert a major accident. Compromises of this type are often necessary in the construction of an efficient system.

In Table 1 are listed some typical results obtained under partial and full computer control. It is our opinion that with more experience, we will be able to routinely obtain determinations having a standard deviation of 0.03% or less.

The chromatograph that has been automated (Fig. 3) is the latest in a series of multi-column systems to be constructed at LRL. These have been built for special analytical purposes.[9-11] A schematic of this particular unit[12] is shown in Figure 6. The unit has three columns, two detectors and an automatic inlet system that can accept any of three samples or standards. The columns can be temperature programmed from $-80^\circ$ C to $+250^\circ$ C and they can be run in many parallel-series configurations. The computer controls the inlet system, the carrier gas flow path as related to the three columns, measures the sample or standard pressure before injection, and performs data acquisition from the thermistor detector circuit.

The construction of this apparatus is similar to that found in ultra-high vacuum systems: to obtain an analytical sensitivity range of $\sim 10^\circ$, mechanical leaks could not be tolerated. Carrier gas flow rates and pressures are precisely controlled. The temperature of the detectors is very carefully controlled; it has been shown that the long-term stability is better than $\pm 0.001^\circ$ C.[13]

All of the 6- and 7-way valves shown in Figure 6 are controlled by the computer. This control allows the chemist to program for the particular chemical resolution desired. Figure 7 is a chromatogram which shows the versatility of this apparatus. The unit is designed and constructed so that the carrier gas flow and pressure can be dynamically balanced. All valves, flow-pressure regulators, and the pressure transducer are contained in a temperature controlled oven. As an example of one of the desirable characteristics of this system, a 20 ft. chromatographic column can be added to or taken out of the main carrier gas stream and the thermistor detector signal (at maximum sensitivity) changes by $\leq 5$ μv as recorded on a 1 mV full scale recorder.

Three methods of data acquisition have been tested under computer control. In the first, the signal of the thermistor-detector bridge combination was measured by a servo-driven potentiometer and shaft angle encoder combination. The output of the shaft angle encoder is converted from gray to binary code by a hardware system. The unit has a resolution of 1 part in 21,000. The other two methods tested were 1MHz voltage-to-frequency converter and counter combination and an analog-to-digital converter preceded by X1000 amplification.

---

*MKS Instruments, Inc., 45 Middlesex Turnpike, Burlington, Mass.

Preliminary data indicates that this monitor will allow us to control simultaneously up to ten analytical apparatuses while taking and preprocessing data at rates of $\geq$ 3KHz. For very high data rates, such as those used for fast-scan high-resolution mass spectrometry, the computer will have to be dedicated to that task during the duration of the experiment.

## REFERENCES

*This work was performed under the auspices of the U. S. Atomic Energy Commission.


1. Analytical Chemistry Section Progress Report, Jan. 1 - Dec. 31, 1966, UCRL-50006-67-1, p. 14-16.

2. J. W. Frazer and R. K. Stump, Mikrochim. Acta (Wien) 1967, 651-657.

3. J. Frazer and R. K. Stump, Lawrence Radiation Laboratory, Livermore, Calif., Report UCRL-50317 (1967).

4. J. W. Frazer, Lawrence Radiation Laboratory, Livermore, Calif., Report UCRL-5134 (1958).

5. J. W. Frazer, Mikrochim. Acta (Wien), 1962, 993.

6. J. W. Frazer and R. W. Crawford, Mikrochim. Acta (Wien) 1963, 561.

7. J. W. Frazer and R. W. Crawford, Mikrochim. Acta (Wien) 1964, 676.

8. R. K. Stump and J. W. Frazer, Lawrence Radiation Laboratory, Livermore, Calif., Report UCRL-50318 (1967).

9. J. W. Frazer, Lawrence Radiation Laboratory, Livermore, Calif., Report UCRL-6244 (1960).

10. J. W. Frazer and K. Ernst, Explosivstoffe, 1964, 4-9.

11. J. W. Frazer, V. L. DuVal and C. J. Morris, Lawrence Radiation Laboratory, Livermore, Calif., Report UCRL-12169 (1965).

12. J. W. Frazer and V. L. DuVal, Lawrence Radiation Laboratory, Livermore, Calif., Report UCRL-70704. To be published.

13. K. Ernst, Lawrence Radiation Laboratory, Livermore, Calif., Report GCTN-216.

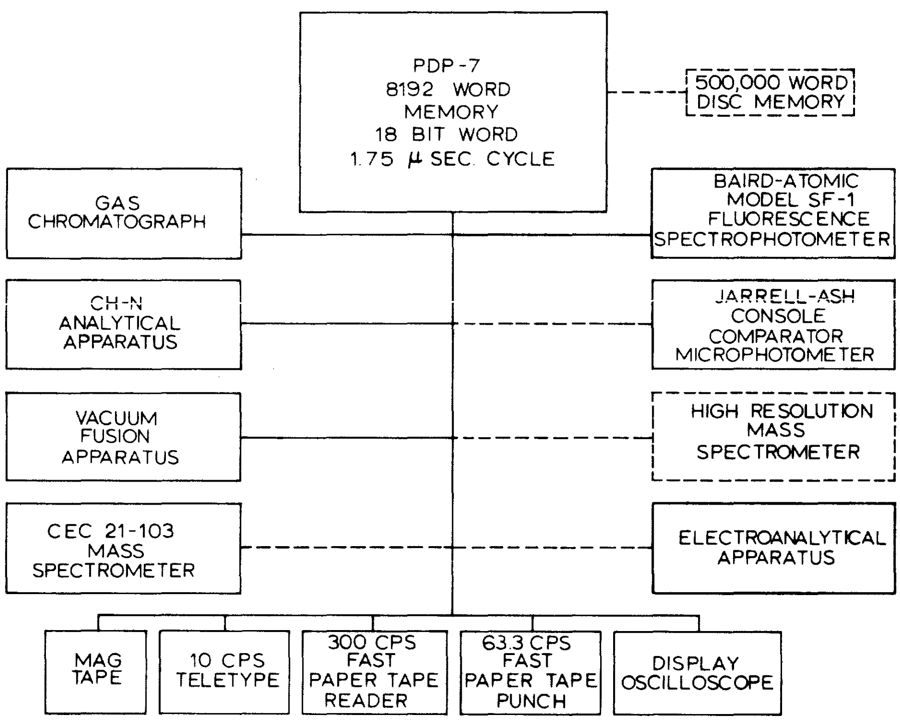




Figure 1 PDP-7 and Instrumentation System

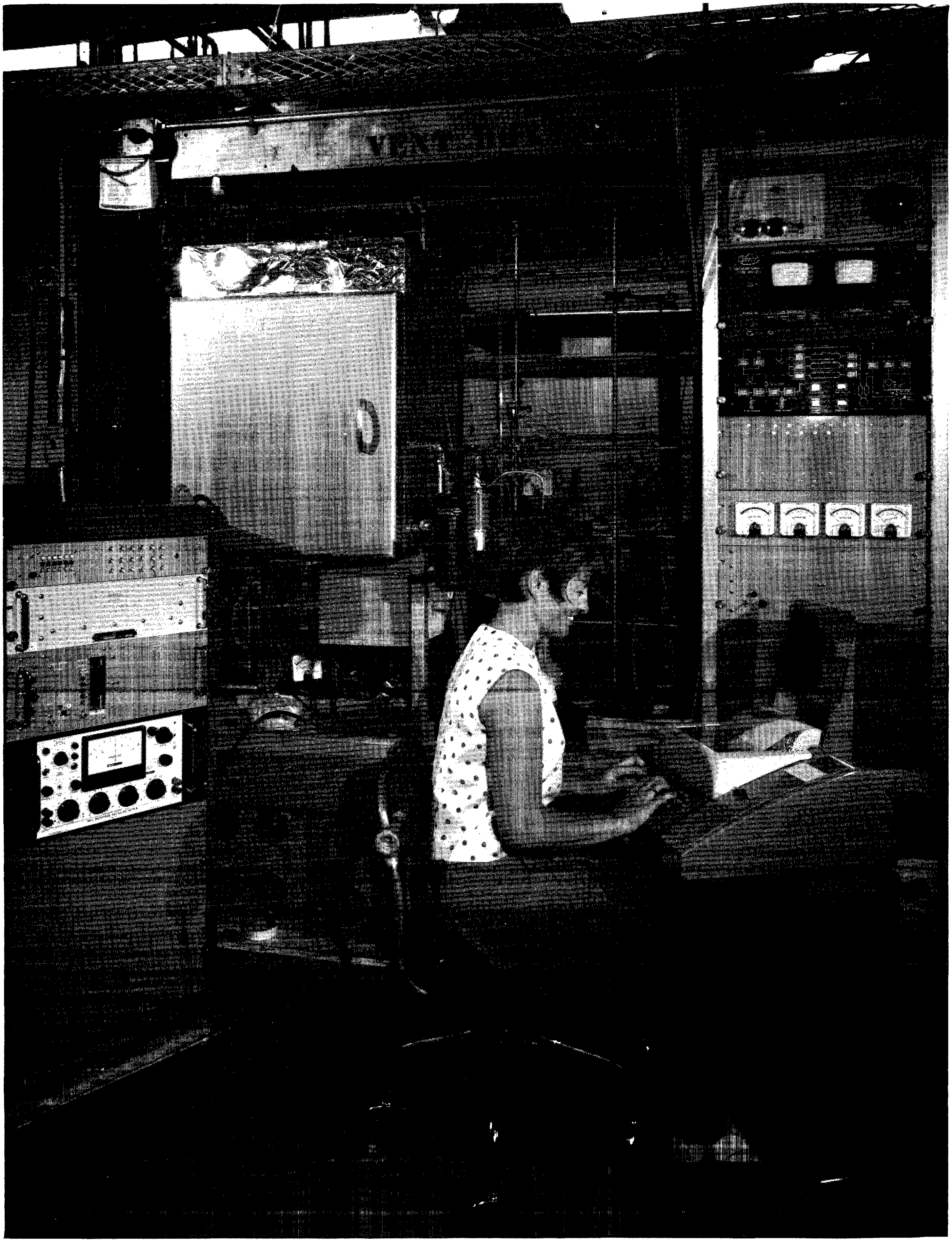


Figure 2 Automated Apparatus Used for the Simultaneous Determination of Carbon, Hydrogen and Nitrogen in Organic Compounds

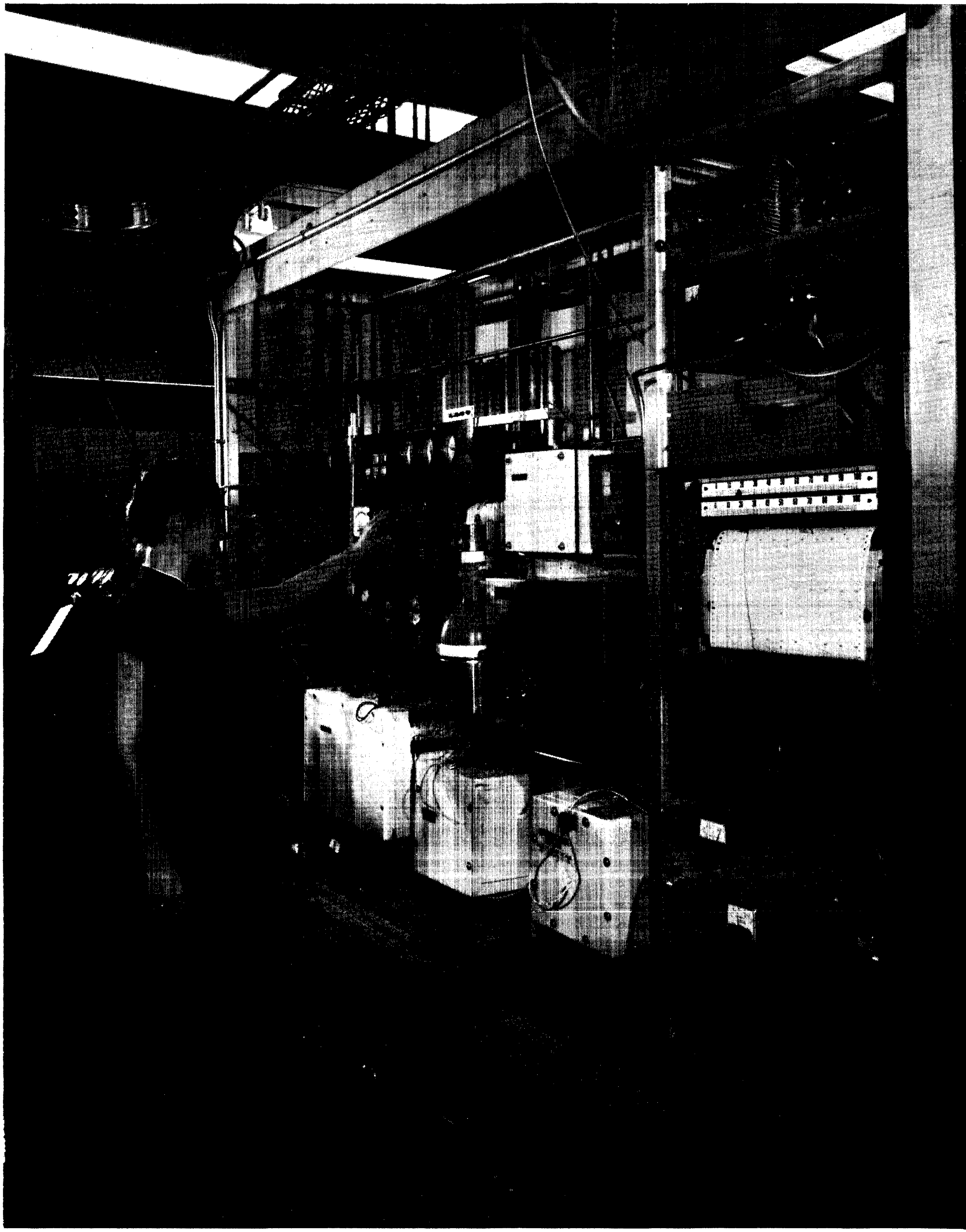


Figure 3 Three-Column Gas Chromatograph

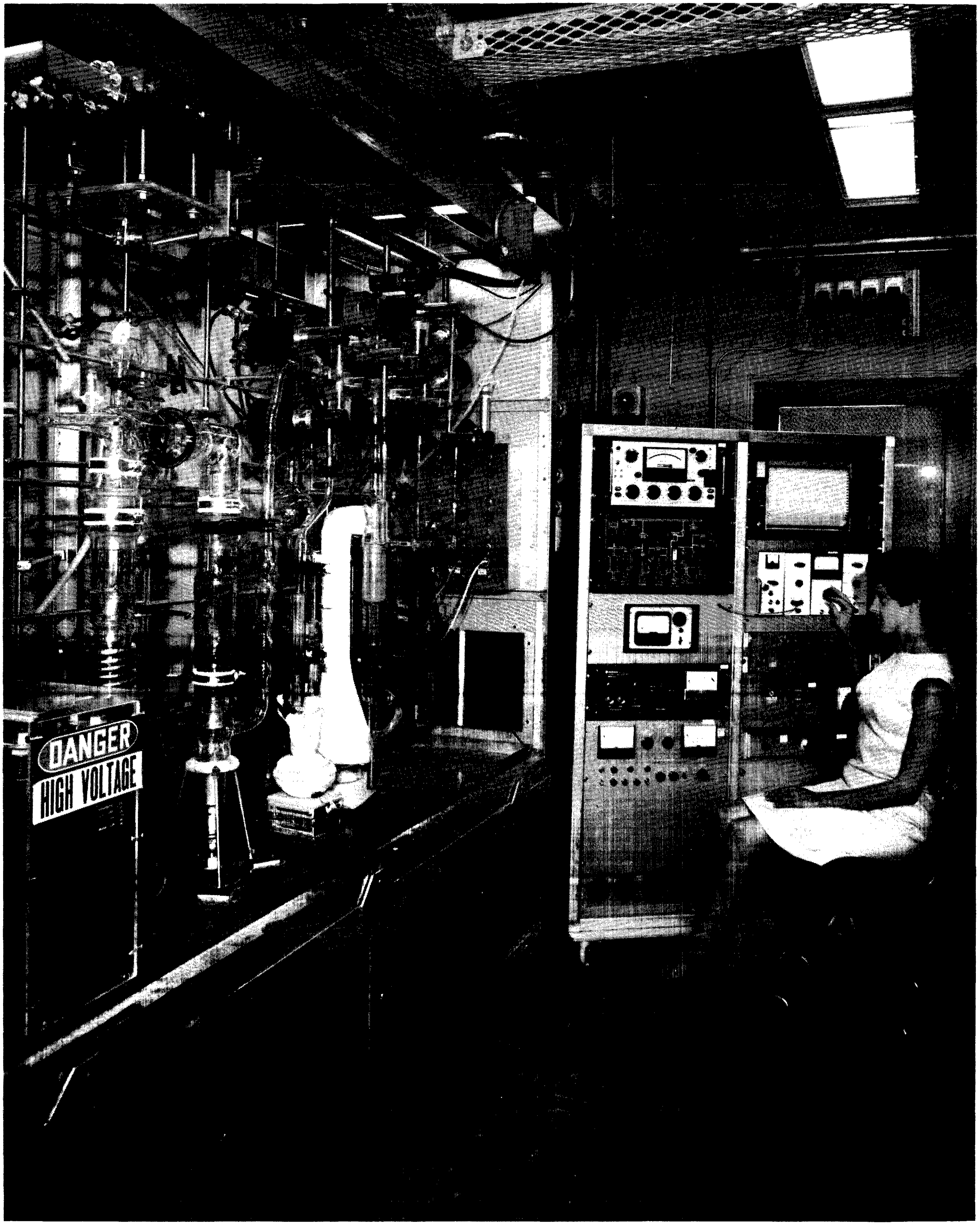
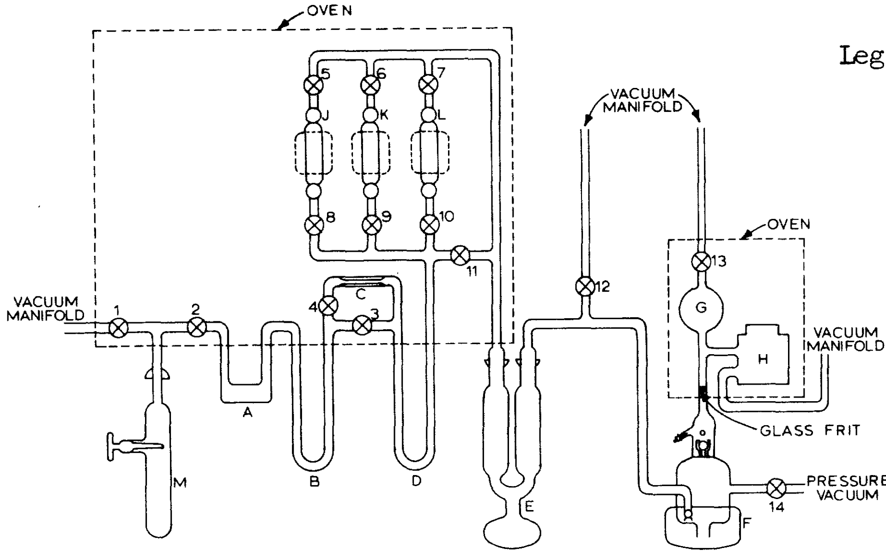



Figure 4 Automated Vacuum Fusion Apparatus for the Determination of Oxygen, Hydrogen and Nitrogen in Metals




Legend:

A. Cu-Hg Amalgam
B. -100°C Cold Trap
C. Glass Restrictor
D. -196°C Cold Trap
E. Hg Transfer Diffusion Pump
F. Rodder Type Toepler Pump
G. Glass Burrette
H. Pressure Transducer
J. 250°C Uranium Trap
K. 800°C Uranium Trap
L. 150°C Manganese Dioxide Trap

Valves 1-13 are automated, hermetically sealed glass valves.

Figure 5 Schematic of the Apparatus for the Simultaneous Determination of Carbon, Hydrogen and Nitrogen in Organic Compounds

Table I. Typical results of CHN analysis via computer control.

| Compound | Carbon % Found | Carbon % Theoretical | Nitrogen % Found | Nitrogen % Theoretical | Hydrogen % Found | Hydrogen % Theoretical |
|---|---|---|---|---|---|---|
| | | Manual readout | | | | |
| p-Dinitrobenzene | 42.88 | | 16.65 | | 2.30 | |
| | 42.88 | | 16.65 | | 2.45 | |
| | 42.82 | | 16.60 | | 2.35 | |
| Average | 42.86 | 42.87 | 16.63 | 16.67 | 2.37 | 2.40 |
| | | Fully automatic | | | | |
| p-Dinitrobenzene | 42.90 | | 16.64 | | 2.38 | |
| | 42.89 | | 16.62 | | 2.46 | |
| | 42.89 | | 16.66 | | 2.50 | |
| Average | 42.89 | 42.87 | 16.65 | 16.67 | 2.45 | 2.40 |
| Acetanilide | 71.02 | | 10.30 | | 6.56 | |
| | 71.06 | | 10.41 | | 6.75 | |
| | 71.04 | | 10.38 | | 6.79 | |
| Average | 71.04 | 71.09 | 10.36 | 10.36 | 6.70 | 6.71 |

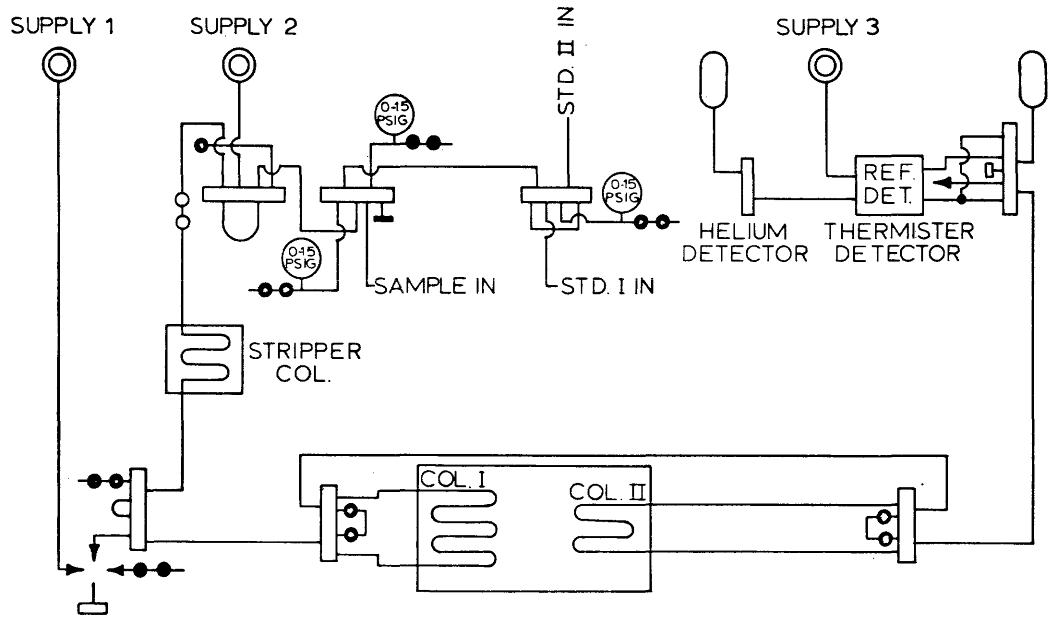



Figure 6 Schematic of the Fully-Automated Three-Column Gas Chromatograph

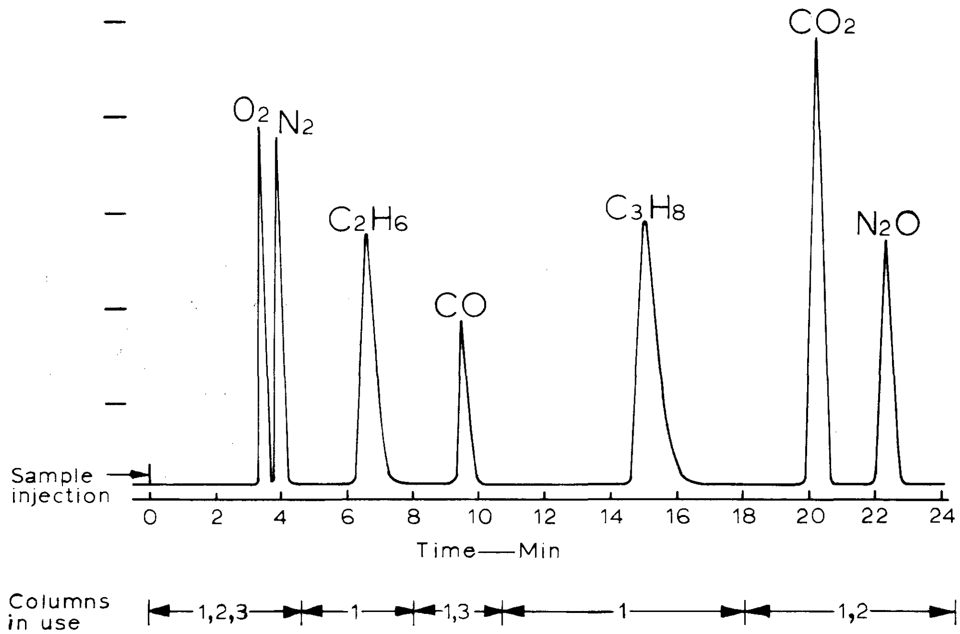



Figure 7 Typical Chromatogram Illustrating the Capabilities of a Three-Column Chromatograph

Legend:

| Columns | | Mesh Size | Operating Conditions Temp.(°C) | cc He/min. |
|---|---|---|---|---|
| 1 - Chromosorb 102 | 6 ft 1/8 in S.S. | 60-8 | 27 —> 50 | 50 |
| 2 - Porapac Q | 6 ft 1/8 in S.S. | 120-150 | 27 | 50 |
| 3 - 13X molecular sieve | 6 ft 1/8 in S.S. | 30-60 | 27 | 50 |

## STANDARD DEVIATION

LINEAR TIME FIT OF RESULTS
NO SMOOTHING OR FILTERING

| DIGITIZING METHOD | PEAK HEIGHT | INTEGRAL |
|---|---|---|
| MILLIVOLT-POTENTIOMETER SHAFT ENCODER | 0.02 % | 0.08 % |
| VOLTAGE TO FREQUENCY CONVERTER | 0.04 % | 0.55 % |
| ANALOG TO DIGITAL ENCODER | 0.12 %* | —— |

GLL-678-314

* NO APPARENT TREND

Table 2 Results Obtained by Three Methods of Data Acquisition from the Gas Chromatograph

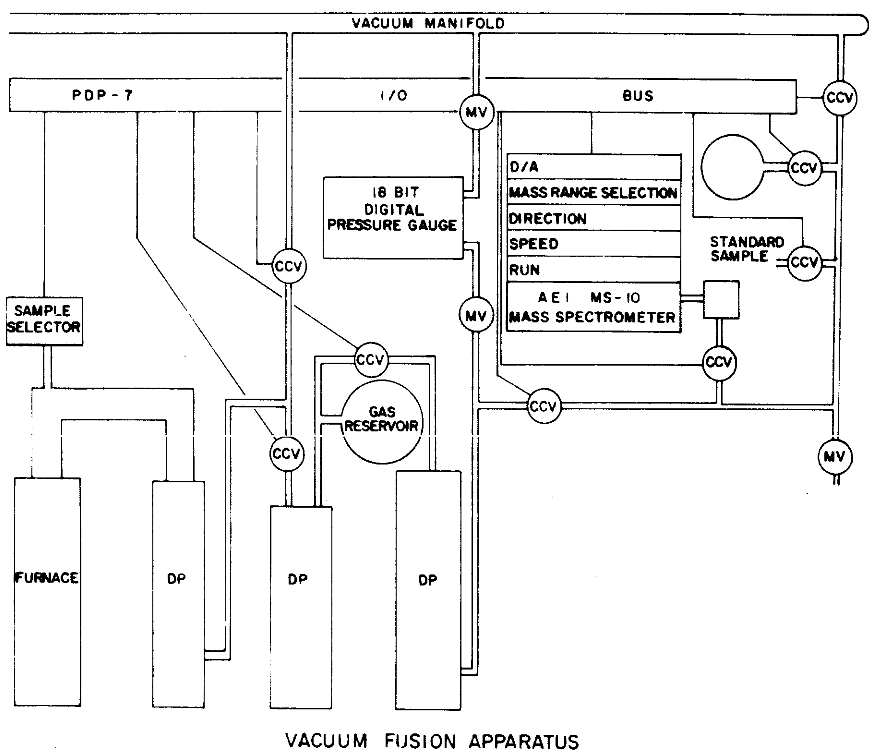



Figure 8 Schematic of Vacuum Fusion Apparatus

Legend: CCV - Computer Controlled Valves

MV - Manually controlled valves

DP - Diffusion Pump

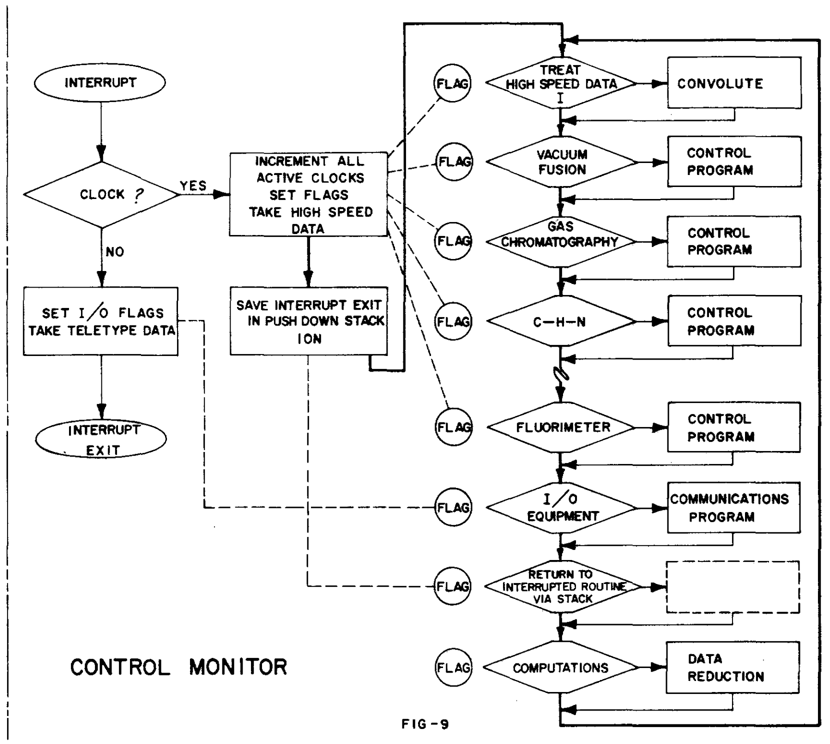



Figure 9 Control Monitor

# DEDICATED COMPUTERS FOR INSTRUMENT CONTROL † *

Roger E. Anderson
Chemistry Department, Lawrence Radiation Laboratory
Livermore, California

## Abstract

The low-cost control computer can be readily interfaced to laboratory instruments using breadboard circuitry. This paper will describe several such applications using the PDP-8/S. Emphasis will include software techniques, system capability and versatility, and experimenter interaction. These systems will be discussed as an alternate to a larger time-shared system. Specific examples will be used to illustrate this philosophy.

† This work was performed under the auspices of the U.S. Atomic Energy Commission. (UCRL-70638)

* This paper was not received for publication.

# A REAL TIME AIR POLLUTION ANALYSIS SYSTEM

Carter L. Cole
Automatic Information Management
Encino, California

## Abstract

Chemical sensors are placed on-line to the PDP-8/S in this real-time micrometeorology application. The operator can select logging of readings as they currently are, as averaged over an interval of up to four hours, and as compared to predetermined calibration values. The system is initially implemented for twelve stations, each of which measures sulfur dioxide, carbon monoxide, temperature, wind speed, wind direction, and smog.

## Introduction

The purpose of this Real-Time Air Pollution Analysis System is to provide the City of New York with a real-time system for the acquisition and logging of data taken from chemical sensors placed at many remote stations. This system is now being assembled by Packard Bell Electronics, Space and Systems Division. It includes a variety of chemical sensors, the PDP-8/S, and a Packard Bell product, Automet III, which is a heavy duty teleprinter. Automatic Information Management has done all of the system analysis for computer selection and all implementation of the software required for system operation.

The system will initially be delivered with 6 sensors implemented at each of 10 remote stations. The sensors measure carbon monoxide, sulfur dioxide, temperature, wind speed, wind direction, and COHS, which is a measurement of smoke (smog) units. The program has been designed to permit expansion to 12 sensors at 30 remote stations.

The requirements placed on the system are not very demanding. The purpose is to implement a scan cycle which must guarantee that in any 5 minute interval every sensor at every station on the system has been sampled. The time allotted for the processing of any one sensor reading from its initial input until output logging if any is approximately 500 microseconds.

## System Description

### Typical Sensor Processing

The typical processing of one reading during this cycle is somewhat as follows. The data reading is input in binary coded decimal, converted to binary, adjusted by linear scaling or some other computational function, checked against a danger limit which will permit an output alarm signal to be generated if a reading (for example, smog) has been detected to be in excess of some level requiring attention. The computed reading is then integrated or averaged with readings since the last logging cycle, and output if the system is in an output cycle.

### Determining An Output Interval

Output can be selected in three ways. First, a thumb-wheel switch exists to select a number of scans between output intervals. We will recall that each scan cycle means five minutes. This switch permits selection of a number up to 48, which provides for a four hour interval between output logging. Output in this case then, represents the average reading since the last output log selected by this thumb-wheel switch.

A second method of causing output is by a demand log button which will cause the program to output the instantaneous reading taken during the current scan. In this way, the present reading of all sensors in the system can be examined. This will be used for example when the computer has generated an alarm signal to signify that a particular parameter is out of bounds.

Finally, when the system has been set in calibrate mode, it is known that the sensors are being placed in known samples. The program automatically enters the demand log mode so that the printout can be compared to the expected results. Calibration permits sampling of the sensor limit conditions at either the high limit or the low limit. It is not really significant to the program that these represent extremes on the sensor. It simply permits the computer to choose a comparison between the present sensor reading and one of two pre-computed sensor readings expected. In initial use, however, it will be used to measure the ability of the sensor to perform according to its range specification. Readings taken during a calibration cycle are not integrated with the normal readings.

As a final comment, it can be noted that the system can provide a continuous logging of readings by selecting an output interval of one cycle. If one is using the demand log to investigate instantaneous readings of any particular sensor, the logging cannot start until the beginning of the next scan cycle. That is, until the computer is next processing the first sensor at station 1. This delay is due to the necessity of synchronizing the system with automatic formatting done by the Automat III for features like page headings and station identification.

## Interesting System Components

Double Precision Routines - The basic arithmetic of the computer system uses principally 24 bit fixed point arithmetic. AIM has, therefore, made minor adjustments to provide for a double precision pseudo accumulator in page zero and has modified the DEC single and double precision multiply and divide routines to automatically leave their results in this pseudo accumulator. In addition, double precision add, load, store, and Log 10 routines were implemented so that nested computations were facilitated.

Interrupt Processing - The Packard Bell interface provides 2 different external sources which can cause a program interrupt. The first condition is the start of a scan cycle which means that the system is now positioned at the first sensor at station 1 . The second condition is data ready, which means that the system has now got a data reading from a sensor ready for input. The interrupt routine then must determine the cause of the interrupt. It does this by inputting an interrupt word which provides up to 12 bits of identifying information. These bits tell the system whether the interrupt occurred for start scan cycle or data ready. If the interrupt was for data ready, then the system will input another 12 bit word to get the data, and will not examine any further information in the interrupt word. If the interrupt occurred to signify the start of a scan cycle then the remaining bits of the word are picked apart to determine the system options such as demand log, calibrate scan, and the thumbwheel switch setting. The system will then set up the necessary control flag information based on the present setting of these options. It is also a standard feature in the interrupt routine to check whether the system was prepared to process the interrupt, i.e. whether the previous parameter has been successfully processed in the 500 microseconds allotted. It is possible then to measure the system performance, and to know the percent loading of each parameter. This permits a realistic evaluation of what additional demands can or cannot be placed upon the system with regard to duty cycle and computational requirements.

Control Routine - The control of the system is implemented at two levels. First of all there is a very brief control program to provide for normal operation of the system during a scan. This routine, for example, simply routes the input reading into the appropriate parameter processing routine and cycles the reading through the normal parameter flow. The second and more complicated area of control is that required to modify system operation at the start of a new scan cycle. This is accomplished in the system idle routine when it is waiting for a data ready interrupt to occur. This routine does the detailed unpacking of the interrupt word itself for the start scan cycle, resolving all possible flag conflicts and priorities associated with various combinations. It then changes the system flags from their setting on the last scan cycle to what is required to control the system for the new scan, for example, clearing out the accumulated tables. This method of organization generally permits the more complicated processing to be done during the time left over from the least demanding parameter, with the result that system performance is not degraded during this hierarchical decision process.

Additional I/O Interface - For purposes of programming the system, Packard Bell Space and Systems Division implemented some additional IOT instructions to permit system operation. One of these is used to input the interrupt word which permits the program to identify which external source is causing the interrupt. An additional IOT is used to input the 12 bit data word.

An IOT is used to generate the alarm pulse which of course would call attention of the system operator to the desirability of taking a demand log to find out which station and which sensor is now out of the tolerable range. An IOT has been implemented to permit output of a 12 bit binary coded decimal reading to the Automat III.

An IOT has also been implemented to permit the computer to sense a change from station to station. This is required by the teleprinter to generate a carriage return-line feed and by the program to recycle back to the first parameter processing routine. Similarly a skip on output ready IOT has been implemented to permit the system to resolve the difference in processing speeds required by the different parameters.

## Summary

This system has provided an extremely flexible, expandable, low cost system for data logging where input, output, and computational requirements are not demanding. Automatic Information Management has found the PDP-8/S very satisfactory for such systems. Although the addressing limitations complicate software generation for systems which require large data bases, it is still a good buy when combining total hardware-software costs when compared to using a comparably interface-oriented computer with far more extensive instruction repertoires.

# AUTOMOBILE EXHAUST ANALYSIS SYSTEM

ROBERT JAHNCKE
BECKMAN INSTRUMENTS, INC.
FULLERTON, CALIFORNIA

## ABSTRACT

Government standards have specified a procedure for measuring the carbon monoxide and hydrocarbons emitted from the tailpipe of an automobile. The procedures require the use of non-dispersive infrared analyzers and a series of computations to be applied to the response as measured to yield a final answer.

The operation is automated by the use of a PDP-8/S general-purpose computer to compute the results on-line. The analyzer data is presented to the computer via a 10-channel relay scanner and a 10-bit analog to digital converter. The computer then performs linearization of the ADC data, weighting, and correcting the data to meet government requirements.

## INTRODUCTION

In recent years the government has developed a testing procedure to analyze the major pollutants in automotive exhaust. The problem is very complex due to the large variety of components in the exhaust, the wide range of exhaust temperatures, the wide range of exhaust gas escape velocity at various speeds, and the presence of particulate matter in the exhaust. However, extensive testing has shown that the problem can be reduced to a repeatable test yielding a significent measure of the level of exhaust emission. The test procedure that has evolved is now known as the California 7 Mode-7 Cycle Test, - or simply the 7 Mode-7 Cycle Test, now that the Federal Government has adopted the test. The exhaust emission components that are measured in the test are unburned hydrocarbons, carbon monoxide, and carbon dioxide.

## TEST PROCEDURE

The test procedure is to exercise the test vehicle on a dynamometer test stand through a 7 mode cycle as described below:

Mode 1 Idle
" 2 Accelerate to 30 mph
" 3 Cruise at 30 mph
" 4 Decelerate to 15 mph
" 5 Cruise at 15 mph
" 6 Accelerate to 50 mph
" 7 Decelerate to 0

Throughout the test cycle the exhaust gas is processed by extensive sample handling equipment. The exhaust gas is separated and fed into four separate non-dispersive infrared gas analyzers. Each analyzer is sensitized to respond to only one of the components in the exhaust. The 4 instruments required are for carbon dioxide, carbon monoxide, and 2 levels of concentration of hydrocarbons. The non-dispersive infrared analyzers were chosen because of their rapid response to the input gas. This allows the reaction of the vehicle to be measured on an individual mode basis. Thus, for each of the 7 modes of the cycle, a measurement is made on the level of each gas concentration for that mode. (See Exhibit I, "Test Calculations"). The CO2 measurement is used only in the computation to correct the data for air dilution by normalizing both the CO and CO2 to 15%. The correction factor obtained is applied to both the CO and HC data. Finally, unique mode weight factors are applied to each set of mode data. The mode weights are in proportion to the occurance of each mode in a statistically determined average automobile trip. Thus, the mode weight for Mode 6 is 0.455, which indicates that nearly half of the time we are accelerating a vehicle from a non-zero speed. The sum of these weighted values will give an overall measure of vehicle performance to which the Government Standards apply. These

Government Standards are 1.5% carbon monoxide, and 275 ppm unburned hydrocarbons. The 7 Mode cycle is repeated for 7 cycles as follows:

> The first 4 cycles are called warm-up cycles, and their average response is only 35% of the final value. The 5th cycle is used to clear the instruments with nitrogen gas, and the last 2 hot cycles are used to complete the test, and their average response is weighted at 65% for the final value.

In the past the only means of arriving at the final 2 numbers has been the "eyeball" averaging of the data from a strip chart recorder, then the conversion of the deflection to gas concentration level, and finally, the calculation of the corrected data. The process required 2 hours for a clerk to do, and even then the answers were questionable due to the typical introduction of clerical errors.

## SYSTEM DESIGN

Our answer to the problem was an on-line computation machine that would produce the results within seconds of the completion of the vehicle test. Our design goals were:

1. On-line computations to yield immediate results.

2. Simple operation to minimize the need for a highly trained technical person to operate the equipment.

3. Convenient packaging to allow mobility and durability in a garage climate.

4. Extremely low failure rate in the equipment because of the lack of strong technical capability within the customer's facility. This was necessary because any failure, no matter how small, would be catastrophic as it would mean complete system shut-down until someone capable could be found to fix the problem.

5. A system to handle 2 tests simultaneously, in order to remain competitive with an analog system currently on the market.

With these design goals in mind we felt we could offer a system that would feature the following benefits to our customers:

1. Immediate test results,
2. Error free computations,
3. Improved repeatability in the test results,
4. Long term product life, or low obselescence due to the programmability of the digital computer,
5. A simple set-up and test procedure, as compared with the competitive analog computational device.

The following system was configured to meet these design requirements:

1. A 10 channel scanner to allow 10 discrete input signals, each to have a 0-1 volt range, and each channel to be scanned at the rate of 10 times per second.

2. A 10 bit analog to digital converter (ADC), to allow 0.1% resolution of the data.

3. A special computer interface unit was designed to allow the computer to input the ADC data, as well as contact closure information for mode start signals.

4. A PDP-8/S digital computer to receive the data, perform the necessary computations, and print the results in an acceptable format.

5. An optional 2nd teletype unit to allow the option of independent test operation for each test stand.

6. A computer program to connect all the foregoing into a workable system.

Of the above, programming was the major cost item, requiring approximately one man/year of effort. The problem was made difficult because of the restrictions of memory, computational speed, and further compounded due to particular customer requirements.

## PROGRAM DESCRIPTION

The program as developed had five levels of priority jobs allocated on the basis of real time requirements. The ADC data interrupted the processor at 10 millisecond intervals; and the typewriter input interrupted the processor for each operator input. The resultant levels of priority are listed below from highest to lowest priority:

1. An Interrupt Executive routine to decide which job should be executed next on the basis of interrupt flags set. This routine is entered for every interrupt.

2. An Input Data Processor routine to process the ADC data and formulate a partial integral of the data.

3. A final Integration Processing routine to average the mode data at the

end of each mode.

4. An Operator Communications routine to respond to operator input requests.

5. The Output Report routine, which printed the results of the test.

Each level of priority could be entered at independent moments of time, therefore, any routines used by one priority level could not be used by another level unless the routines were made recursive. Consequently, several utility routines were made recursive to overcome this problem.

With this program logic, the maximum duration for the interrupt to be turned off was 500 microseconds. However, each routine entered had to be completed before another interrupt from the same source could be acknowledged. Thus the ADC data processor had to complete all of its operations in less than 10 milliseconds.

To facilitate the operation of the machine, an Executive routine was designed which contained a loader, a tape dump routine, and a few other miscellaneous items. This enabled the operator to lock out the control switches as soon as the short Executive tape was loaded. After the machine is started in the Executive, all further operator control is done via the teletype. This eliminates the possibility of program failure from the misuse of the control switches by unauthorized personnel, or improper switch settings.

The computer program can be functionally broken down into 2 separate operations: 1) the calibration and setup mode, and 2) the on-line data reduction and output mode. The calibration procedures require that the operator insert into the machine via the teletype unit, all the necessary information to allow the machine to convert the raw ADC counts into meaningful gas concentrations. The non-dispersive infrared analyzers have a non-linear calibration curve. However, the non-linearity is small enough to allow a 5 straight line segment approximation to be used without degrading the accuracy significantly. It is up to the operator to select the 6 points that describe the curve to minimize the approximation error introduced. The operator must then enter the information to the computer through a prescribed calibration procedure, and the computer will then convert the information into meaningful computer words to be used during the on-line data acquisition mode. The process for entering the points must be done for every instrument, requiring up to 48 numbers to be typed in by the operator.

The calibration procedure requires positive actions on the part of the operator to make any changes or requests. Thus, the procedure to calibrate would follow this sequence:

> The operator types K followed by a space to cause the processor to stop and enter the calibrate mode. Then the operator would select a channel by an abbreviated gas name, as HHC. Then a point to be entered is requested by the operator. The computer would print the previous value and accept a new one or not, depending on the operator entry. A space with no digits would leave the old number undisturbed; typed digits followed by a space would insert the new one.

Once the information is in the memory of the machine, the operator may then preserve the information on a reloadable paper tape.

In addition to the entry of information for calibration, there is capability within the program to make a full correction for instrument amplifier drift and/or gain change. The total calibration procedure was designed so that an overall systems calibration is easily accomplished by measuring the instrument response via the ADC for an injected calibration gas.

The on-line processing mode has several optional features, selectable by the setting of the switch register on the front panel. Thus the operator has the choice of:

1. Running the entire 7 mode-7 cycle test, or terminating at the end of the current cycle.

2. The operator has the option of selecting all of the printout, some portion of it, or else inhibiting all printout except the final 2 number answer.

Regardless of the output selected, the program is interrupted at 10 ms intervals to input the individual ADC values. The program then converts the individual ADC number into a gas concentration level and formulates an accumulation of each gas to be averaged at the end of the current mode. An external operator mode start switch cycles the program in accordance with the test. As the data for each mode is available, it is printed according to the option selected.

Two tests may be run simultaneously with the system. The only restriction is that the test which is started first must be completed before any data is printed from the second test. All of the data for the second test is stored in the memory, waiting for the completion of the first test.

## SUMMARY

In conclusion, the system has done well in meeting our design goals.

1. Test results using the equipment have shown that the repeatability is 1-2%, which is good, considering that we are testing automobile exhaust. Correlation of data with hand calculated data has been within the same error range, indicating a significant improvement in total testing techniques.

2. Ease of operation has been proven by having trained several engineers in 2 hours or less on the full operation of the equipment. However, a garage mechanic will remain mystified with the equipment after 2-4 weeks exposure. Fortunately, nearly every one of our customers has had at least one individual capable of learning the full operation in less than a week.

3. The packaging convenience has been proven by the fact that one of our customers has placed one of these devices in a mobile van for testing of vehicles in the field.

* * * * *

Nov 67

EXHIBIT I

Example: Test Calculations

| Mode | Average As Measured HC | CO | $CO_2$ | Correction Factor | Corrected HC | CO | Weighting Factor | Weighted HC | CO |
|---|---|---|---|---|---|---|---|---|---|
| Idle | 650 | 5.2 | 10.1 | $\frac{15}{15.3}$ | 640 | 5.1 | .042 | 27 | .22 |
| 0-25 | 430 | 2.0 | 12.1 | $\frac{15}{14.1}$ | 460 | 2.1 | .244 | 112 | .51 |
| 30 | 390 | 2.5 | 12.1 | $\frac{15}{14.6}$ | 400 | 2.6 | .118 | 47 | .31 |
| 30-15 | 2600 | 4.6 | 8.6 | $\frac{15^*}{14.8}$ | 2600 | 4.7 | .062 | 161 | .28 |
| 15 | 470 | 3.7 | 11.4 | $\frac{15}{15.1}$ | 470 | 3.7 | .050 | 23 | .18 |
| 15-30 | 350 | 1.3 | 12.5 | $\frac{15}{13.8}$ | 380 | 1.4 | .455 | 173 | .64 |
| 50-20 | 4600 | 3.9 | 6.3 | $\frac{15^{**}}{13.0}$ | 5300 | 4.5 | .029 | 154 | .13 |
| | | | | | | | Sum - | 697 | 2.27 |

* $\frac{15}{6X.26 + 4.6 + 8.6}$

** $\frac{15}{6X.46 + 3.9 + 6.3}$

By a similar procedure the hot portion of the test yields composite values of 680 and 2.21. The reported overall composite values are:

0.35(697) + 0.65(680) = 686 ppm HC

0.35(2.27) + 0.65(2.21) = 2.24% CO

# LINC-8 TEXT-HANDLING SOFTWARE FOR ON-LINE PSYCHOPHYSICAL EXPERIMENTS

B. Michael Wilber
Stanford Research Institute
Menlo Park, California 94025

Abstract

A complete text-handling system (LUCIFER) has been developed for the LINC-8. All communication between LUCIFER and mortal man is carried on through a Teletype medium, so that hard copy is always produced, and one need never invoke scope, switches, and lights. Along with LUCIFER have appeared subroutines by which experiment-running programs can do input and output of data with text files or the Teletype. This paper discusses the philosophy of LUCIFER and includes examples of the use of LUCIFER and the running of a typical experiment.

We are using a LINC-8 computer for presenting stimuli and recording responses in psychophysical experiments. This use is characterized by extremely low data rates over long sessions. For example, experimental sessions typically take twenty minutes to an hour, with data rates of 30-180 bits per minute in each direction. Since much, if not all, of the computer's time is taken with running experiments, and because of the availability of commercial remote-access time-sharing computer facilities using ASR-33 terminals, we have decided not to do processing on the LINC-8 that can be done remotely. Communication between computers is via punched paper tape, and since this is our only use of that medium, it is not viewed as onerous. (Eventually we may be able to eliminate this use with a connection of the LINC-8 to the telephone lines.) Since the output of almost all the experiment-running programs is to other programs, output formatting is considerably simplified.

Because of the low data rates involved in our experiments, it is practical to input and output data in the form of text files called manuscripts. This greatly simplifies the problems of preparing input (stimulus) files and making sense of output (response) files because our regular text-handling programs can be brought to bear on these files. These programs, part of the LUCIFER[1,2] system, are needed for preparing the programs to run the experiments, so a great saving is realized by using the same programs to handle the data as much as it is handled on the LINC-8. A family of text-handling subroutines has grown out of the family of programs and has been made a part of LUCIFER, giving it the structure shown in Fig. 1. Experiment-running programs can handle text by merely incorporating the subroutines and using simple calling sequences. An example of this is shown in Figs. 2 and 3. We should note in passing that LUCIFER includes a program to convert almost any tape from LAP4 format to LUCIFER format, but there is no such program for LAP6. It might also be mentioned that the LUCIFER programs use the PROGOFOP typeout instruction and assumes the keyboard and typing mechanism are connected, so they probably could not be modified to run on the so-called classic LINC.

## The LUCIFER Philosophy

Part of the philosophy of LUCIFER is that a typewriter-like device is a good medium for interacting with computers. With such a device, one has hard copy of what one did just before somebody walked in with an interesting question. Since one is often referring to typed and written material, a typed page in a well-lighted room seems to cause a good deal less eyestrain than a flickering scope in a dark room--and well-lighted rooms are easier to come by than dark ones. When one is interacting with an active special-purpose program, one is less tempted to push the wrong button than with a passive, general purpose bank of switches and lights, because only logically "correct" commands need be accepted by the program; and one can easily arrange that the consequences of a mistaken command are easier

1. LINC unrelenting console interception and file editing routines.
2. See LUCIFER documentation.

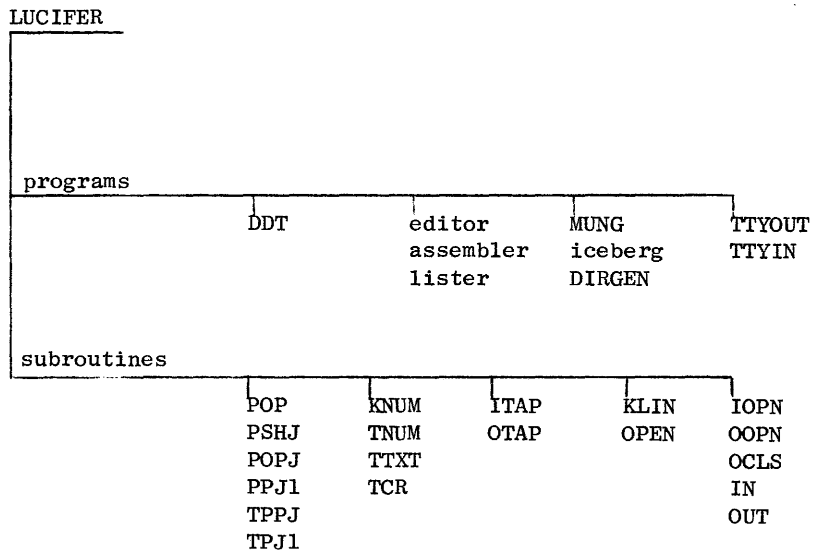



Fig 1 The overall organization of LUCIFER.

```
VISIØN↓

OUTPUT TØ *WØRK↓
STIMULI *J00↓
RANDØMIZATIØN •1 FRØM *ØRDFIL↓

EXIT

ØUTPUT TØ * Q↓
MUNG↓

MUNG FRØM *WØRK↓
TØ *269↓

MUNG FRØM * Q↓
TTYØUT↓

STRAIGHT CØPY FRØM *269↓
R 3
4030
4013
4013
0167
0071
  •
  •
  •
4315
4026
0067
QUIT•
```

TA-746522-26

Fig 2 Running a typical experiment, including the output of data via the Teletype. Here and in all the following figures, keyboard type-in is underlined.

to recover from. See Fig. 4. Again, hard copy is produced automatically as a by-product of initializing a program instead of (hopefully) by somebody noting down on paper what is put into certain magic locations, hopefully without making mistakes. There also seems to be some advantage to leaving the file name permanently and automatically recorded on paper instead of hoping the right paper or magnetic tape or card deck was put in the right place. Operating instructions can almost completely be dispensed with if the program gives an idea of what parameters are needed, and in this mode, one can easily arrange that no steps can possibly be forgotten. Our experiment-running programs run through a set dialog and do not start the experiment until the end of the dialog.

It is felt that interactive programs can have more natural and easy-to-use command structures than currently appearing in many distributed programs. The LUCIFER programs are few in number, interactive and frequently used, so ease of memorization is a minor consideration. This is also helped by the fact that they have as many commands in common as apply.

In most situations not every command makes sense at all times. For instance, a line of the current file cannot be examined if there is no current file. It would help if one could obtain a priori knowledge of what sort of commands are acceptable in a given situation. In the LUCIFER and experiment-running programs, this consideration has given rise to the concept of prompting, i.e., all type-in is in response to some type-out from the program, and this type-out is somehow indicative of the nature of response expected. For example, as illustrated in Fig. 5, if the last thing typed was an asterisk, then the program expects a file name, consisting of up to six characters terminating on a carriage return. Furthermore, rubout functions as a backspace and types as a backslash, and the special name blank Q(" Q") causes the program to return to GUIDE.

In this connection, a little can be said about the LUCIFER programs command structures and methods of operation. First, one cannot modify text, directories, or core locations without first seeing what one is modifying. This is in contrast to some systems in which one cannot see one's text and change it at the same time. For example, to kill or replace a line of text, one directs the editor's attention to that line, which causes its contents to be typed out, and only then will the editor accept a command to replace or kill it. There is a certain amount of protection from careless errors afforded by the fact that the positions of the keys on the keyboard were considered when choosing the commands to be associated with certain actions. For instance, the editor kills and replaces lines and inserts lines before others (the commands are K, R and B) instead of changing, deleting and inserting forward of a given line (the commands might be C, D and F) or killing, inserting and overlaying, with commands K, I and O.

All the LUCIFER programs and all the experiment-running programs always refer to files by name, and so the actual location of a file is not relevant unless one wishes to add new files to the directory or expand ones already there. On the other hand, the format and location of the directory are well noted in the documentation, and the only program which lists the names in the directory also lists sufficient additional information to reconstruct parts of the directory or even the entire directory.

The editor edits a file in place and is a random access editor. Thus opening a file involves ascertaining its location, validity and length, requiring the inspection of exactly two tape blocks, instead of copying the file into some "working area" and/or inspecting it to build a directory. Also, one has no need to edit a file serially and/or be continually copying it between two temporary files and finally rename one with the original's name and kill the other one and the original source.

The LUCIFER Programs

The purpose in writing the LUCIFER programs was primarily to facilitate the process of forming programs on the LINC-8. This imposed two requirements on LUCIFER. First, it had to have a much more tractable overall organization and command structure than other available systems. Also, it was felt mandatory that there should be no overlaying. This not only greatly enhances response time of the programs and greatly facilitates the process of debugging them, but it makes programs more readable, permits them to be ordinary GUIDE programs, facilitates their assembly, and otherwise facilitates local and remote program updating. A secondary purpose was to consolidate action from the switches, lights, scope and Teletype to just the Teletype. The reason this was felt desirable is a foundation of the philosophy of LUCIFER. LUCIFER includes DDT, a debugging program, and four kinds of text-handling programs: the programs of primary interest, some bookkeeping programs, two programs whose sole purpose is to allow our experiments to communicate experimental data with other

```
L↓

*J00↓
0002
•G
0001/
0002/
0003/  764   1364  1764  2364  2764  3364  3764  4364  4764  5364
0004/  5764  6364   764  1364  1764  2364  2764  3364  3764  4364
0005/  4764  5364  5764  6364   764  1364  1764  2364  2764  3364
0006/  3764  4364  4764  5364  5764  6364   764  1364  1764  2364
0007/  2764  3364  3764  4364  4764  5364  5764  6364   764  1364
0010/  1764  2364  2764  3364  3764  4364  4764  5364  5764  6364

•Q
*ØRDFIL↓
0016
•21L
•31U
•G
0021/   R 3
0022/   3    4    25   71   53   67   42   37   73   52
0023/   22   45   54   60   41   21   72   5    61   64
0024/   47   33   57   10   12   40   17   34   56   13
0025/   2    16   43   14   24   46   65   6    55   70
0026/   20   74   23   11   63   1    26   27   30   31
0027/   7    62   66   50   36   15   51   44   32   35
0030/

•Q
* Q↓
```

TA-746522-27

Fig 3 Typical input files for the experiment of figure 2.

```
PART2↓

N-TUPLES *3↓

N-TUPLES *P/C↓
ØUTPUT TØ *WØRK↓
0014-TUPLES, 0102 ØF THEM.
DELAY 0..DELAY 0.DELAY 0.3
MARKERS .MDELAY 0.3
MARKERS .1 FRØM *MARK↓
C.R. FØR ERASE.P?
```

TA-746522-21

Fig 4 An experiment-running program rejecting "logically meaningless" parameters.

```
PART2↓

N-TUPLES *P/C↓
ØUTPUT TO *WØRK↓
0014-TUPLES, 0102 ØF THEM.
DELAY 0.2
MARKERS .3 FRØM *MARK↓
C.R. FØR ERASE.P?

ICEBRG↓

*SEED1↓
.B=0101.S=0073.4C.S=0067.4H:ZØT↓
.B=0164.S=0004.R:FØØBAZ↓
.Q
*FØØBAZ↓
.B=0164.S=0004.4X.S=0010.KQ
*SEED1↓
.B=0101.S=0063.10X.S=0073.Q
* Q↓
```

TA-746522-22

Fig 5 Prompting by an experiment-running program and the iceberg computers, and one program which does not exist.

DDT is a simple program to examine and change storage locations in octal and exert some control over the execution of a program. It always works on the most recently assembled program, and there is no provision for saving (with GUIDE's FILEBI) this program in any but its pristine state--it is possible to override this, but it is often easier to bring the manuscript up to date, thus facilitating later changes.

The programs of primary interest are the editor, the assembler, and the lister. The salient features of the editor have already been discussed. The assembler inputs the names of the manuscripts composing a program, types out all symbol (tag) definitions and assembles the binary for the program. The language it processes is an extension of a restriction of LAP4. Details may be gotten from its documentation, but some of the salient features are that lines may be as long as the editor will handle (something like sixty characters), symbols (tags) are up to four letters and/or digits with at least one letter, comments can be on the same line as anything else, and arbitrarily complex, logically meaningful expressions can be used for equalities and origins. Checking is much tighter, with almost any situation the assembler cannot correctly handle giving rise to a message containing the name of the situation, the current core location in the object program and the current line number. The lister simply produces a listing of a manuscript, with options to select only parts of the text and to form $8\frac{1}{2}''$ by $11''$ pages.

There are three bookkeeping programs included in LUCIFER: MUNG[1], the iceberg and DIRGEN[2]. The latter is used to convert a LAP4 tape to LUCIFER format if that is possible. All programs and subroutines concerned with the text in a manuscript use information stored in the manuscript's directory, which tells how many blocks are actually occupied by the text and the highest line number for each block. A manuscript in this form is called normal, but it is much more convenient for experiment-running programs to write text in another form, called abnormal. Also, pursuant to the second law of thermodynamics, the manipulations of the editor are quite likely to lower a manuscript's packing density, but they are somewhat less likely to raise its density. For these two problems, we have a program MUNG, which reads a normal or abnormal text file and writes a normal text file with the highest possible density. Finally, all responsibility for the tape's directory of files is vested in the iceberg whose operation is shown in Fig. 6. This program accepts explicit commands to change the directory but of course rejects any commands which would result in the directory becoming potentially invalid.

In running our experiments, we have found that additional processing should be done on other computers. With our present hardware and with the particular other computers used (dial-in with a Teletype), the only means of communication is ASCII-coded punched paper tape. This particular medium is less onerous, however, when it is viewed as backup to the storage of data in the more accessible forms and when one realizes that a very small absolute amount of data is handled this way and quite seldom at that. These programs are TTYOUT, which punches the contents of a file onto a paper tape with blank leader and trailer, and TTYIN, which inputs a paper tape into a file. For timing reasons, the latter will not run with PROGOFOP, but requires our own corruption of that program, which is named PROTOCROCK[3]. Since it is only intended for data, it does not handle the full character set, and for timing reasons, it is extremely limited in the length of a tape it will handle.

The last part of LUCIFER consists of a nonexistent program--a mythical beast. It may not be a program at all--it may simply be a nonexistent feature of MUNG. Due to its lack of existence, one cannot say very much about it, and whatever one does say about it may be of indeterminant validity and concreteness. One can, however, say that this program, which has no name, has the property of merging several files, or possibly arbitrarily or otherwise selected portions thereof, into one file. Despite the obvious usefulness of this program, it has only been used once, so it has never been written. Probably the principal reasons for this state of affairs are that the assembler is nearly indifferent to the number of manuscripts composing its object program, and that the lack of real pages makes short manuscripts desirable.

## The LUCIFER Subroutines

While the LUCIFER programs were being written, it was realized that their id and parts of their preconscious could be unified, generalized and quickened in manuscripts, so the experiment-running programs could easily communicate with LUCIFER and each other. The

1. Manuscript ultra-normalization and generation.
2. The directory generator.
3. PDP-8 routine to oversee tape operations and cooperative routines which obtain console-type knowledge (the program of total crockery).

NO. OF MS ON THIS TAPE IS 22

| NAME | B | N |
|---|---|---|
| SEED1 | 101 | 73 |
| 69 | 174 | 4 |
| FØØBAZ | 200 | 10 |
| GØRP | 210 | 20 |
| ZØT | 230 | 10 |
| MUMBLE | 240 | 10 |
| FØØBAZ | 250 | 10 |
| PDLMAN | 260 | 10 |
| IØSUBS | 270 | 10 |
| TAPEIØ | 300 | 10 |
| MSFIND | 310 | 10 |
| IØGENS | 320 | 10 |
| CØRE | 330 | 4 |
| SEED5 | 501 | 17 |
| RANDØM | 520 | 10 |
| LXIX | 530 | 10 |
| FIE | 540 | 33 |
| ZILCH | 573 | 15 |

```
ICEBRG↓

*FIE↓
•B=0540•S=0033•5X?•35C?•13H:YECH↓
•B=0560•S=0013•15H?•R: YECH↓
?•Q
*YECH↓
•B=0560•S=0013•3C•R:FRØDØ↓
•Q
*FRØDØ↓
•B=0560•S=0010•KQ
*FRØDØ↓

*FIE↓
•B=0540•S=0020•15X?•S=0020•↓
•13X•S=0033•Q
* YECH↓

* CØRE↓

* Q↓
```

Fig 6   A typical manuscript directory and iceberg manipulations on it.

```
0001/  [HERE IS THE PREPRØCESSØR
0002/
0003/  SET & PDP [SET PUSHDØWN PØINTER
0004/   PDL-1
0005/  SET & IØSW[I/Ø WITH TELETYPE
0006/   0
0007/
0010/  JMP PSHJ  [TYPE A CARRIAGE RETURN FØR GØØD MEASURE
0011/   JMP TCR
0012/
0013/  JMP TTXT  [ØPEN STIMULUS FILE
0014/   316     ['N-TUPLES '
0015/   255
0016/   324
0017/   325
0020/   320
0021/   314
0022/   305
0023/   323
0024/   240
0025/  JMP PSHJ
0026/   JMP IØPN
0027/    JMP 20   [TRY AGAIN IF NØ SUCH FILE
0030/
0031/  JMP TTXT   [ØPEN RESPØNSE FILE
0032/   317     ['ØUTPUT TØ '
0033/   325
0034/   324
0035/   320
0036/   325
0037/   324
0040/   240
0041/   324
0042/   317
0043/   240
0044/  JMP PSHJ
0045/   JMP ØØPN
0046/    JMP 20  [IT DØESN'T EXIST-TAKE IT FRØM THE TØP
0047/
```

Fig 7   Example.

logical outcome of this idea is the LUCIFER subroutines. For the first few months of their existence, they were highly evolutionary, but as experiment-running programs were written around them, they gradually coalesced into a unified, modularly useful whole.

The structure of the subroutine package is that the subroutines are distributed across five interrelated manuscripts in such a way that at least six subsets of the manuscripts are conceivably useful. The selected manuscripts are assembled along with one or several other manuscripts in which reside the conscious part of the program and its own special-purpose preconscious and id. These other manuscripts have to set aside certain locations and areas with specified names, and some small amount of initialization is required, but otherwise one need not consider the internal mechanizations of the id of LUCIFER.

The specific functions represented on the manuscripts are the following. The first contains basic pushdown list manipulative functions. Although this makes recursive functions possible, only one program actually does have a recursive section, and that program is not often used. On the other hand, the pushdown list is seen as a good discipline for the allocation of temporary storage, and the handling subroutine returns (always a problem on the LINC) is uniformized and considerably simplified.

The next manuscript contains subroutines to type text displayed in the calling sequence and to input and output numbers in octal. In our use, it would seem that octal numbers are not objectionable even to people having their first contact with computers, because <u>all</u> numbers handled by any part of LUCIFER are octal, so it is seldom necessary to convert between octal and decimal.

On the next manuscript there is a subroutine to buffer a line of input, up to a preset maximum length and process backspaces. Since almost all input is performed by this subroutine and the number input subroutine on the previous manuscript. it does not take new people very long to learn the interactive characteristics of our programs. The other subroutine on this manuscript is a subroutine to find a file in the manuscript directory. This subroutine uses the line input subroutine and does its own prompting, so there is some uniformity gained by this device. In addition, this subroutine detects names with lead blanks (these have significance as commands instead of names) and also obeys the command of this form which means that the current program should be terminated and GUIDE should be restored. Thus it is no accident that all our experiment-running programs at least exit in the same fashion.

A tape file can be treated as a character-oriented serial access input or output medium by use of the subroutines on the next manuscript. There are two completely independent subroutines here, one for input and one for output. The input file may be normal or abnormal, but the output file will be abnormal because it is very hard to write a normal file. In fact, only the editor and MUNG write normal text.

The upper levels of the id and the lower levels of the preconscious of LUCIFER are contained on the previous manuscripts; the last one contains subroutines from the upper levels of the preconscious and the lower levels of the id. There are subroutines to open and close a file to the output routine and to open a normal file to the input routine of the previous manuscript. In practice these special-purpose file opening subroutines are used instead of the general purpose one, which they call, on the third manuscript. All the previous subroutines do character input and output with instructions not intrinsically defined in the assembler. These instructions could be defined as operate-class instructions, if the Teletype is to be the medium for all character input and output. However, this final manuscript defines these instructions as funny subroutine calls. The subroutines, depending on the state of a flag in memory, cause either the Teletype or the current open input and output files to be used. Thus we have a degree of device independence. Device independence in itself is often used as a selling point for computers or software, but here, it is precisely what makes the subroutine package useful in the experiment-running programs. It is what enables the same set of subroutines to be used for communicating with the two media, and this makes the calling sequences tractable as well as cutting core requirements considerably.

## Examples

Some examples of the use of LUCIFER are in order here. The Fig. 7 shows the initialization necessary before using the full generality of the LUCIFER subroutines. The pushdown pointer (PDP) must be loaded with the address of the cell below the first cell of the area (PDL) reserved for the pushdown list, and the I/O medium switch (IOSW) must be set to indicate which medium is to be used first. Both of these cells could be preset at assembly time, but then one would not

```

#NXST   [MAIN LØØP ØF THE EXPERIMENT
JMP PSHJ     [GET NEXT RANDØM NUMBER
 JMP KNUM
  JMP '-2    [  IGNØRING THINGS THAT DØN'T MAKE SENSE
JMP PSHJ     [ØUTPUT FILE WANTS A C.R.
 JMP TCR

LDA &        [WE WANT TØ ØUTPUT AND WAIT
 4003
SCR 14
LDA &        [PICK UP A STIMULUS
 4003
ADD NMBR     [ RANDØMLY CHØSEN
STC TEMP
LDA TEMP
ØPR 6        [AND ØUTPUT
 NØP         [IGNØRE 0.1 SWITCH

SET & TEMP [HE PUSHED THE BUTTØN.
 FACT        [  GØØDIE.
LDA &        [ØUTPUT ZERØ
 3
SCR 14
ØPR 6
 XSK & TEMP [WILL MARK IFF NØT 0.1

SAM 10       [READ THE KNØB
ADA &        [AND
 377         [ ØFFSET
AZE &        [  PRØPERLY
 CLR
BSE TEMP     [NØW MAYBE MARK RESPØNSE
JMP PSHJ     [AND ØUTPUT
 JMP TNUM

XSK & CNTR   [MAYBE CØNTINUE
 JMP NXST    [ THE EXPERIMENT


```

TA-746522-24

Fig 8  Example.

necessarily be able to abort execution at any arbitrary point and restart the program and have it retain a semblance of sanity. Next a carriage return is output, followed by a call to the text typing subroutine and a call to the input file opening subroutine. If the file does not exist or does not contain valid text, the subroutine does not skip on return, and the program's option is to restart from the beginning. Otherwise, another message is typed out, and the output file opening routine is called. Again, if the file does not exist, the subroutine does not skip. Otherwise it skips and the only further step necessary before the input and output files are the source and destination characters is to change the setting of the I/O switch.

The next example (Fig. 8) illustrates the types of calling sequences used for the subroutines. In this passage, the I/O medium switch is set to the current input and output files. The subroutines used are the same as those used for Teletype I/O, but their id is under the influence of the medium switch. This is currently the main way device independence is useful, although programs could conceivably accept a fake file name, say blank T (" T") to mean the Teletype should be selected for a given activity. This is currently planned, but only for a program of low priority. In the passage at hand, the first activity is to read a number from the input file. If no number is found, the subroutine does not skip, and we call it again. The effect is to ignore anything not making sense as a number. The break character is returned in the accumulator, but we ignore it and output a carriage return to the output file. The carriage return subroutine returns with the accumulator clear, and then we pick up the number previously input. Next there is some calculation peculiar to the experiment this program runs, but finally the response is reduced to a number, which is then output onto the output file.

Figure 2 illustrates the entire process involved in running an experiment and obtaining a punched paper tape for graph plotting and further analysis. The program is loaded from GUIDE and runs through a set dialog in which it is told the names of the file containing the stimuli and responses as well as the name of a file containing randomizations and the position of the particular randomization to be applied to the stimuli. After this dialog, the program runs the experiment and, after writing the responses into the output file, restarts itself. This time, it is given the phony file name " Q", which directs it to return control to GUIDE. The next step is to MUNG the responses from the abnormal temporary file to the normal permanent file. Finally, since the response text has been normalized, we can output it onto a paper tape, as is shown in the illustration. Paper tape input is not shown because usually this is done concurrently with the final debugging of the program which runs the experiment.

A final example (Fig. 9) is an illustration of the edit-assemble-debug loop. The program is a simple adding machine, residing in the manuscript FOO. At the beginning of the example, the assembler (named A) is called from GUIDE and directed to assemble the program, consisting of the manuscripts FOO, PDLMAN and IOSUBS, in that order. It lists the symbol definitions and notes an undefined symbol in the program. The next step is to direct the assembler to reload GUIDE, which is then told to load and start the editor (named E), whose attention is first directed toward the file FOO (which actually occupies three blocks of its allocation) and then toward the line mentioned in the assembler's message. This line contains a typing error (pahj for pshj), and this is corrected. Then GUIDE is reloaded and the program is reassembled, this time with no comment from the assembler. Now GUIDE is restarted, from which DDT is loaded. DDT automatically loads the most recently assembled program, which is the program of interest, before requesting a command. We change the location after the call to the number input subroutine to a jump to DDT, which starts, restarts and extends down to location 1400. Then we ascertain that the program will be started at location 20 and start it. It identifies itself and awaits input. Numbers are given it, and it types the running sum and then a letter is typed with no preceding number after which DDT signals its readiness. The accumulator is perceived to contain the character code for the letter, the altered instruction is restored, the starting location is again verified, and the program is restarted. This time the subtotal and total features are seen to be working, and the program is finally commanded to return to GUIDE. This example was typed in to the computer and debugged and embellished in twenty minutes, and the sample run shown here lasted about eight minutes.

## Critique

The LUCIFER system, being a real system in constant use for four to ten months and having an evolutionary background, is possessed of some shortcomings, and it is felt that some space could be devoted to exposing them. The most fundamental fault is that, for historical reasons, two different

```
A↓

ASSEMBLE FRØM *FOØ↓
AND FROM *PDLMAN↓
AND FRØM *IOSUBS↓
AND FRØM * F↓

NMBR0251
   ⋮
LPKH0102
SUBT0075
FIE 0071
LXIX0053
UDS AT 00610043

ASSEMBLE FRØM * 0↓
E↓

*FOO↓
0003
.43/   JMP PAH.JB
JMP PSH.J↓
0044/    JMP KNUM↓

.Q
* 0↓
A↓

ASSEMBLE FRØM *FOØ↓
   ⋮
LXIX0053

ASSEMBLE FROM * 0↓
DDT↓

.63/1000 #
0062/6071 7400↓

.P=0020
.G
CHEAP ADDING MACHINE.
+5 0005
+3 0010
+44 0054
+S
.A=0046
.62/7400 6071↓

.P=0020
.G
CHEAP ADDING MACHINE.
+5 0005
+10 0015
+44 0061
+S0061
+14 0014
+52 0066
+12 0100
+S0100
+T0161

+Q
```

Fig 9   The edit-assemble-debug loop.

character sets are used-- the LINC character set and DEC's ASCII character set. Fortunately, one of these is only used in outplotting to the Teletype. The full set of subroutines, though composing a comprehensive, easy to use package, require almost 380 locations in LINC lower memory and two LINC memory quarters for buffers. Of course, the buffers need only be respected while they are in use, a fact which is exploited in one of our experiment running programs.

There is some conspicuous room for improvement in the manipulatons one can perform with the LUCIFER programs. The editor will not append to a manuscript but will only insert before a line. In practice, this means that all manuscripts have an empty line after the last meaningful one. Also, the editor will not make a new manuscript, but only edit an old one, so one usually has a manuscript, usually named SEED1, whose contents are exactly one empty line, which one MUNGs into a file before editing new text into it. The editor has two further properties which were included to increase its safety but which make its interactions take longer than would otherwise be necessary. First, it will not accept any commands pertaining to a line of text without first opening the line and typing out its entire contents. Also, whenever a change is made to a line, both the block containing that line and the manuscript's directory are written out onto the tape. Finally, editing would be greatly facilitated if it were possible to alter a line without retyping it in its entirety.

Other criticisms of the LUCIFER programs are more general. The editor, the iceberg, and the assembler are all incapable of handling the null case (i.e., empty manuscript, empty file directory and empty file directory and empty program). The editor and the iceberg both have safeguards built in so that they will not make an empty manuscript or directory from a nonempty one, and the action of the assembler on an empty program is harmless. There is no facility in LUCIFER for merging or dividing manuscripts. So far, LUCIFER is used mainly for preparing programs, and the assembler accepts a program spread out over many manuscripts, so the lack has not been objectionable enough to be cured. In the lifetimes of most data files, they are usually not changed enough that such a facility would be a great convenience. The iceberg is very crude--it does little more than accept a human-oriented command language for a minimal set of atomic operations on the file directory and perform consistency checking between those commands and the directory. The intent was (and still is) to have it automatically invoked to create and extend output files as necessary, and to have it automatically handle the case where a file cannot be expanded without running into another file, but where there are free blocks on the tape. One final criticism which applied to all the LUCIFER programs is that their command languages are very tight, in that any would-be command which is not of exactly the correct format is rejected. For instance, in contrast to assembly language, blanks are forbidden wherever they are not mandatory.

These criticisms are presented to show the other side of LUCIFER. Without this section, we would have just been extolling the favorable aspects of LUCIFER and ignoring the basic fact that ideal systems exist only before their logical consequences are attained and that as soon as a system is realized (in the form of running programs in production use, in this case), the logical consequences are hard to ignore. In considering the critique, one should bear in mind not only that it is offered by the author of LUCIFER, but that any sufficiently severe faults would be (and have been) corrected in the evolution of the system.

# THE LINC-8 IN RESEARCH ON SPEECH

Richard Harshman and Peter Ladefoged
University of California Los Angeles
Los Angeles, California

Abstract

Two applications of the LINC-8 computer to research on the analysis and synthesis of speech are described. AVG 1 is a program which averages and processes acoustic and electromyographic data. TALK is a program which facilitates creation and manipulation of sets of speech parameter curves. It displays and stores these curves, and generates from the curves coordinated varying voltage outputs which are used to control a terminal analog speech synthesizer. The role of such programs in phonetic research is discussed briefly.

Computer averaging of repeated analog signals is a fairly common procedure. Our Average 1 program, the first of two LINC-8 programs developed for speech research, demonstrates the capability of the LINC-8 computer to provide the base for a convenient and flexible system for averaging and processing repeated analog signals.

TALK, the second program to be described, uses the computer to control a speech synthesizer, and to facilitate the creation and manipulation of parameter curves for the generation of synthetic speech.

## AVG 1

In one part of our speech research we are attempting to analyze the involvement of certain facial, lip, and tongue muscles in the production of different components of speech. To accomplish this we compute averages of the electrical activity generated by these muscles along with averages of the audio activity due to speech. Electrodes for recording the electromyogram of the muscles under study are implanted in the lips or face of the subject. He is then instructed to repeat certain words or sounds 15-30 times, while both the audio activity and the emg are recorded on tape.

The trouble with such studies of speech is that every time you say a word, you say it slightly differently. These variations are irrelevant for most aspects of speech research, and by reducing the reliability of the record they obscure important differences among the analog signals corresponding to utterances of different but related phrases, or utterances of the same phrase by different speakers.

What is needed is a typical, representative curve for each set of repetitions of a phrase. In this typical curve, the effects of irrelevant variations among the repetitions should be reduced to a minimum, while those characteristics which are consistently present and thus characteristic of the phrase should be amplified and show up clearly. This typical representative curve is secured for each set of repetitions by averaging together the curves of that set, using the LINC-8 computer, and the program AVG 1.

Audio and/or emg signals are fed into two of the analog input jacks on the LINC-8 console. To establish a "trigger event", the computer samples a third input channel and tests to see if the voltage goes above a threshold level set by the experimenter. When this happens the sampling and adding to memory of 512 values is begun in the other two channels, and continued until 512 points have been sampled for each curve. The computer then returns to sampling the trigger channel, waiting for another "trigger event" to start the cycle over again. Each cycle adds to the sums for every point.

The threshold level for trigger is adjustable by a knob on the computer console. In addition, the numerical value of the trigger level may be displayed on the oscilloscope screen (figure 1). This allows subsequent resetting of the trigger to the same level, and allows computation of the voltage level and percentage of full scale at which triggering occurs.

Typically, we want to average the electromyographic activity which occurs before a particular speech event with the activity which occurs at similar moments in repetitions of the same speech event. Accordingly we want to start sampling the input signals a constant amount before the speech activity begins. This may be achieved by reproducing the audio or emg signal not only synchronously with the signals being averaged, but also a known amount earlier, by means of an additional playback head on the tape recorder, placed so that the tape passes it before passing the in-line head used for reproducing the other signals. The reference point for each curve is thus the beginning of an utterance (as signified by audio activity above the chosen threshold), but the computer initiates the sampling sweep a constant interval before then.

While the sums for the averages are being formed, the computer will also display (if requested) the number of tokens summed since the memory was last cleared. Alternatively, the experimenter can instruct the computer to stop summing after a specified number of tokens (sweeps) have been input, in which case the computer will first display this number preceeded by a minus sign and then count one place backwards with each sweep, from the number specified to zero (figure 1). Upon reaching zero it will enter a state where it will not accept any more data until the operator intervenes. Using the LINC-8 relays, a control feature has also been included, which remotely halts the tape recorder playing the data when the desired number of tokens have been summed.

The sweep rate, or rate of sampling the analog curves, is adjustable from approximately 1/10 second to 4 seconds for the total sweep. (This range can be expanded further if required.) Thus the maximum sample rate is around 200 microseconds per point, (5 Khz). The sweep rate currently being used in the emg experiments is usually one second per sweep. This sets the sampling rate at 512 points per second and limits the highest unambiguously distinguishable frequency to 256 cycles/second. Higher frequencies are present in the emg, and these are recorded, but confused with lower frequency activity. This is a desirable approach since we are concerned with the envelope of emg activity amplitude rather than the precise spectral characteristics of the muscle activity generating the envelope.

Since emg activity is an AC signal of approximately equal positive and negative deflections, summing of the gross envelope characteristics requires rectification of the incoming signal. Full wave rectification may be selected by raising a sense switch. This instructs the computer to reverse the sign of negative digitized values before they are added to the average. Alternatively a hardware rectifier, which in our case is combined with an integrating circuit, may be inserted in the line between the tape recorder and the computer. The integrator is under computer control; 15 microseconds after it has been sampled it is discharged and then starts summing again. Accordingly it is accumulating virtually all and only the energy between each digitized point and the preceding one. This neatly overcomes the time constant problems of most integration systems.

The sums for each point are in double precision. On the 12-bit word length of the LING-8 this means that 24 bits are available for expressing the value of each point. The A-D converters present each sample as a 9-bit number. Consequently, thousands of tokens may be summed without danger of overflowing the memory. Thus Average 1 stores two curve-sums at high capacity and high resolution both in time and magnitude.

When the desired sums have been accumulated, they can then be displayed and processed. Either curve may be displayed alone, or both may be displayed together. Both curves are scaled by the same amount, by simply setting a value in the left switches on the computer console and pressing the teletype key "V". Scaling is by a power of two, reducing the vertical dimension by 1/2, 1/4, 1/8, etc.

As an aid in accurate measurement of the scaled sums (averages), Average 1 provides a cursor (figure 2), selectable by sense switch, which can be moved over either curve accompanied by a numerical display of its location in both the x and y dimension. Typing "Z" on the keyboard rezeros the numerical display at any location of the cursor, allowing the experimenter to measure the height of peaks, the duration of waves, etc. directly.

Variations in the repetition of a phrase, and high frequency components in the emg and audio data cause the resulting average curve to be noisy. A procedure for *smoothing* the curves is therefore provided. It performs a running average of adjacent points each time "S" is pressed on the keyboard. The smoothed curve is then displayed on the oscilloscope. Usually 10-15 applications of the running average will sufficiently smooth the curve. Either curve can be smoothed independently of the other. Examples of a curve smoothed to various degrees are shown in figure 3. Smoothing acts as a non-linear low pass filter. An extensive series of empirical measurements have been made to determine the low-pass frequency characteristics of specific numbers of smoothings. Some of this data is summarized in the curves of figure 4.

Excessive smoothing can be corrected by simply repeating the scaling process from the original raw sums, which are still in memory, and beginning the smoothing over again. The allocation of memory for Average 1 allows for maximal flexibility in inputting and processing the data being averaged. The space allocated for the 24-bit sums is separate from the portion of memory for the scaled and smoothed curves which result from processing these sums. The 24-bit sums are not disturbed by processing and display of the curves. Thus: 1) different scalings and smoothings can be tried without the necessity of writing the raw sums onto magnetic tape; and 2) a number of tokens can be summed, displayed, smoothed, and then a number of additional tokens can be added to the same sum, then this new sum can be smoothed, scaled, etc.

Both filing of curves onto magnetic tape and later retrieval are quick and flexible, using simple keyboard commands. Curves stored on tape may be further smoothed or scaled at a later date. Either the double precision raw data or the single precision processed data may be stored.

The principal output device for immediate monitoring and measuring of the curves is the computer's oscilloscope. Curves displayed in this way may be labeled (using the GRAPHA program) and photographed. It has often been found preferable, however, to make large (8 1/2 by 11) detailed plots of the curves by using a subroutine of Average 1 and a Mosley x-y plotter. Average time to complete a plot of two curves on a sheet of paper is about 30 seconds.

The plotting subroutine is written to run entirely in the PDP-8 mode of the LINC-8. This leaves the LINC accumulator and memory buffer registers free to serve as buffers for the 9-bit D-A converters that the LINC uses to run the display oscilloscope. The PDP-8 inserts sequential x values and correct y values into these buffers using the commands for intercommunication between the two central processors.

In order to speed plotting and make it more accurate, the plot subroutine calculates the first derivative of the curve being plotted (i.e. the average of the absolute values of the difference between the point being plotted and the two neighboring points), and uses this number to adjust the plotting rate so that the pen travels at a fairly constant speed over the paper. Thus it speeds up for smooth straight horizontal lines, and slows down for sharp peaks.

AVG 2 and 3 are planned as not too difficult extensions of this basic package. AVG 2 will process four curves at once. AVG 3 will provide standard deviation and standard error computation and display.

## The TALK Program

The second program is concerned with *synthesizing* analog voltages, rather than analyzing them. It is designed to offer us computer control of one of our speech synthesizers. We are currently using it with the Terminal Analog Speech Synthesizer described by McKinney et al. in *Working Papers in Phonetics 4*. This synthesizer requires the specification of time varying voltages for: (1) fundamental frequency; (2), (3), (4) the center frequencies of the first three formants; (5), (6), (7) the amplitudes of the formants; (8) the amplitude of the noise generator; and (9) the center frequency of the noise band. The TALK program is designed to facilitate the production and manipulation of sets of voltages such as these. It is thus part of our scheme for producing high quality synthetic speech in experiments where a precisely specified stimulus is required, rather than for use in experiments on speech synthesis by rule.

Although in our current work we are using only 9 parameter specifications, the TALK program will actually handle up to 20 curves in its present form, or up to 24 curves if more use is made of magnetic tape. This should be sufficient for high quality synthesis in terms of acoustic parameters. Our more elaborate requirements for the computer control of a speech synthesizer which models the articulatory processes will be discussed at the end of this paper.

The TALK program has four modes of operation, selected by the operator by means of the teletype. (1) *Read* allows new curves, or modifications of old curves, to be created. (2) *Tape control* permits the storage or retrieval of sets of curves. (3) *Display* enables up to four curves to be displayed simultaneously, together with appropriate time and magnitude calibration lines, and labels. (4) *Play* generates the sets of voltages for controlling the speech synthesizer at a rate which may be specified. When the program is first started, or during operation, whenever the user types "M" on the teletype, the CHOOSE MODE display will be presented on the oscilloscope as in figure 5. The operator can then select one of the modes.

## Reading Curves into the Computer

When the read mode is requested the computer generates the display: "Input new values for??" The operator types in a two letter curve name, and the current values for that curve are displayed. This may be noise if a curve had not been read into memory previously. The display on the screen also includes a V shaped cursor and two (octal) numbers which show the position of the cursor on the time axis and on the vertical axis.

The position of the cursor can be controlled either by the use of two knobs on the computer console, or by an x-y "reader" device which produces voltages corresponding to the x-y coordinates of a pointer. We have modified a Moseley x-y plotter so that it can be used as a "reader." Curves can be drawn on graph paper (or on existing data such as spectrograms) and then placed on the reader. When the stylus is manually moved over the curve the position of the cursor in the computer display is moved accordingly. When the experimenter types "I" movements of the cursor cause new values for the curve to be stored in memory. The curve may be modified completely, or just a small section can be changed leaving the rest unaffected.

Each curve is 512 points long. Points are specified with 6-bit accuracy and stored two to a memory location. Thus any point can have an octal value between 0 and 77 (0-63 decimal). This exactly matches the accuracy of our digital-to-analog converters.

## Storage on Tape

Sets of curves, stored in this fashion in memory, may be transferred to magnetic tape by one simple command that specifies the unit and the location in which they are to be stored. Sets of curves may be easily retrieved from tape, one or more of the curves modified, and then returned to the same or a different location on magnetic tape.

## Display

To display several curves at once, or just enlarged portions of one or more curves, a special display mode is provided. The display mode allows the user to select, by name, up to four curves which he wishes to be displayed. They are then shown from top to bottom, in the order he specified them. They are automatically positioned and scaled to the largest scale allowed by the available space. If desired, the user may display only a portion of the curves, by specifying the first and last points he wishes to display.

The names of the curves may be displayed along with the curves. Vertical time lines corresponding to 1/2 and 1 second intervals can be displayed. Horizontal amplitude lines can be displayed at 0, 1/4, 1/2, 3/4, and full voltage, for each curve. A typical display is shown in figure 6.

Curves are normally displayed each on its own scale, but an option is available for displaying the three formant frequency curves on the same scale, and for varying their brightness according to the value for the corresponding point in the formant amplitude curve. In this manner an artificial "spectrogram" is recreated, as shown in figure 7.

## Playback of Curves

To generate synthetic speech, the curves are used to generate control parameters for a speech synthesizer. The computer reads the curves, turning their values into simultaneously varying analog output voltages. The rate at which the computer reads the curves is determined by an external pulse input. Each pulse advances the computer to the next point on each of the curves. There is no minimum rate of "play"; the maximum rate is around 300 points per second.

While the curves are being read and synthetic speech generated, any one of the curves may be displayed. A cursor is shown on this curve at the point where values are currently being read from the curves. A numerical expression of the location and the value for the curve being displayed is also provided. This allows the user to coordinate the speech he hears with the characteristics of the curves in memory. Playback may be paused at any time by turning off the pulse input. Speech then stays at the point where it was interrupted, and the numerical octal values of the control voltages at that moment can be noted.

Transition between functions such as the read in mode and the playback mode in this program can be completed in less than a second, so that rapid alterations and checks by playing back, can be performed.

Figure 1: A single token as displayed during input to the AVC 1 program. (All photographs are taken from oscilloscope display, then reversed white for black.) The lefthand number (at the top) is the trigger level, and the righthand is the number of tokens that remain to be summed before automatic halt and display.

Figure 2: Combined display of average audio and emg waveforms, and the measurement cursor ("V"). The numbers give the x and y displacement obtained by zeroing the cursor at the bottom of the second peak (upper curve) and then moving it to the top.

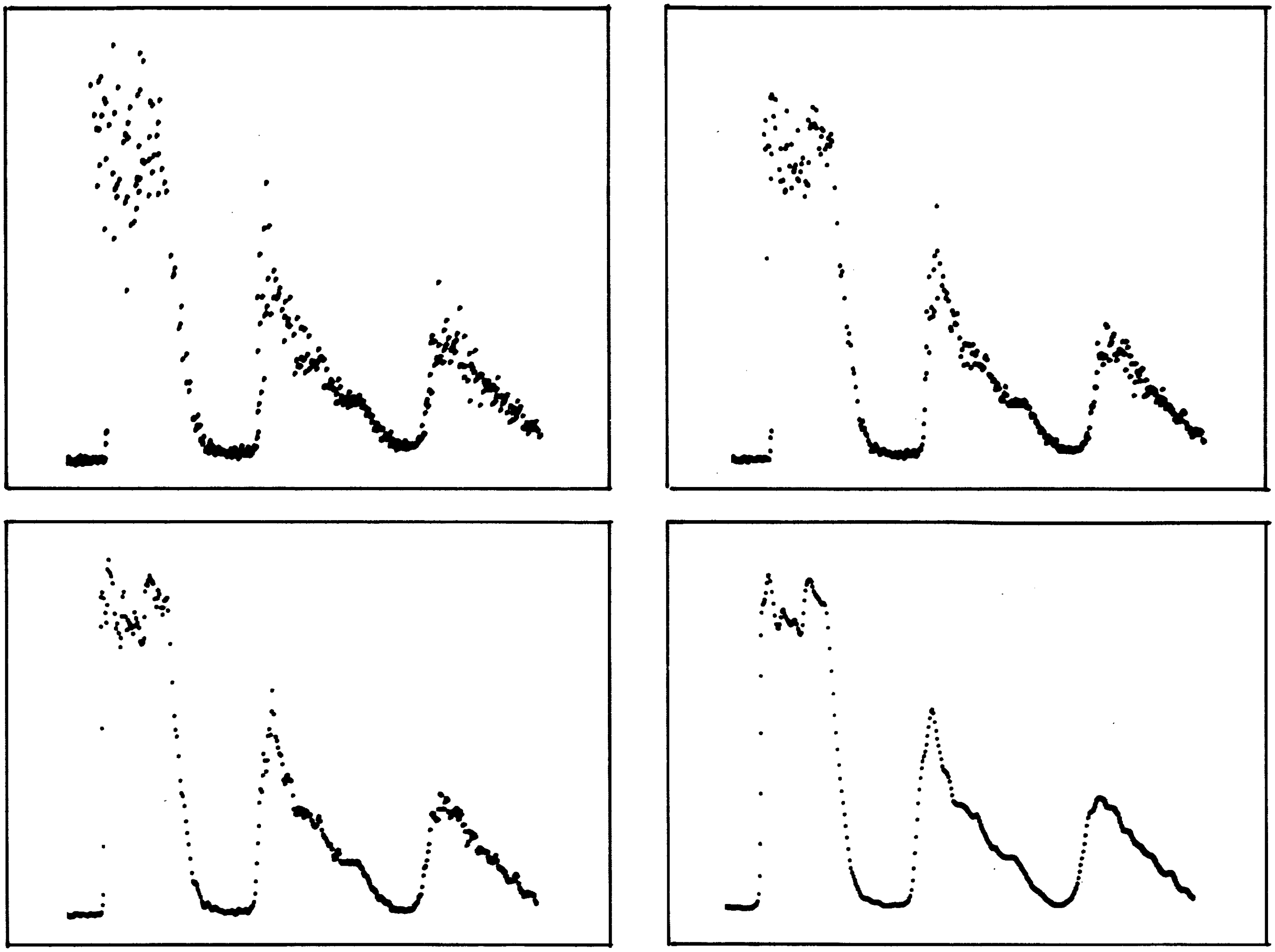


Figure 3: An unsmoothed average of 25 audio tokens, followed by the same curve processed by 1, 3, and 15 applications of a running average smoothing procedure.

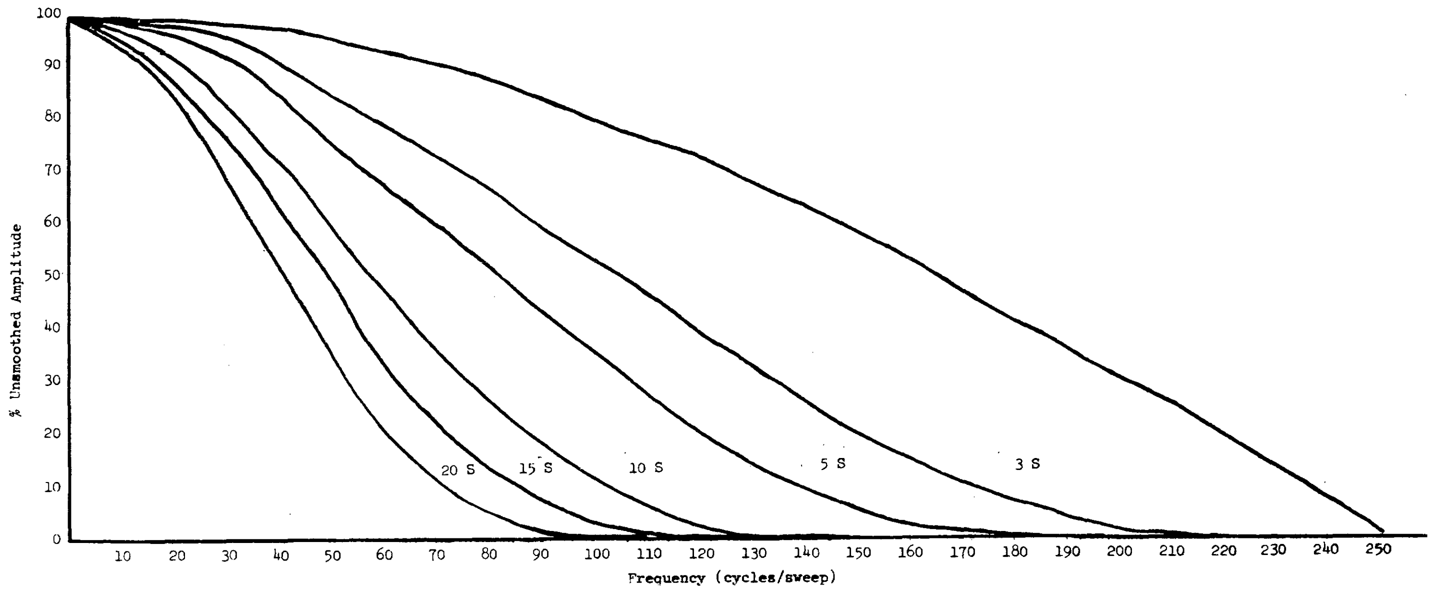



Figure 4: Effect of successive applications of the smoothing procedure on frequency components of the data. Numbers to the right of each curve represent the number of applications of the smoothing procedure. In each application adjacent points of the digitized emg are averaged together as follows:

(a) on even numbered applications

$$\text{New } P_i = \frac{\text{old } P_i + \text{old } P_{i+1}}{2}$$

where $i = 1, 2, 3, \ldots 511$

(b) on odd numbered applications

$$\text{New } P_i = \frac{\text{old } P_i + \text{old } P_{i-1}}{2}$$

where $i = 512, 511, 510, \ldots 2$

Smoothing procedes from left to right on even and right to left on odd numbered applications. The leftmost point is unaffected on even numbered applications, rightmost is unaffected on odd.

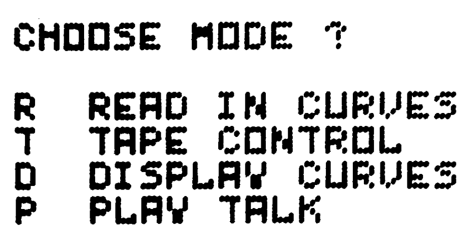



Figure 5: Display produced by the computer while waiting for the operator to choose one of the four modes of operation of the TALK program. This is one of 6 displays which help the operator "find his way around" and facilitate selection of program options.

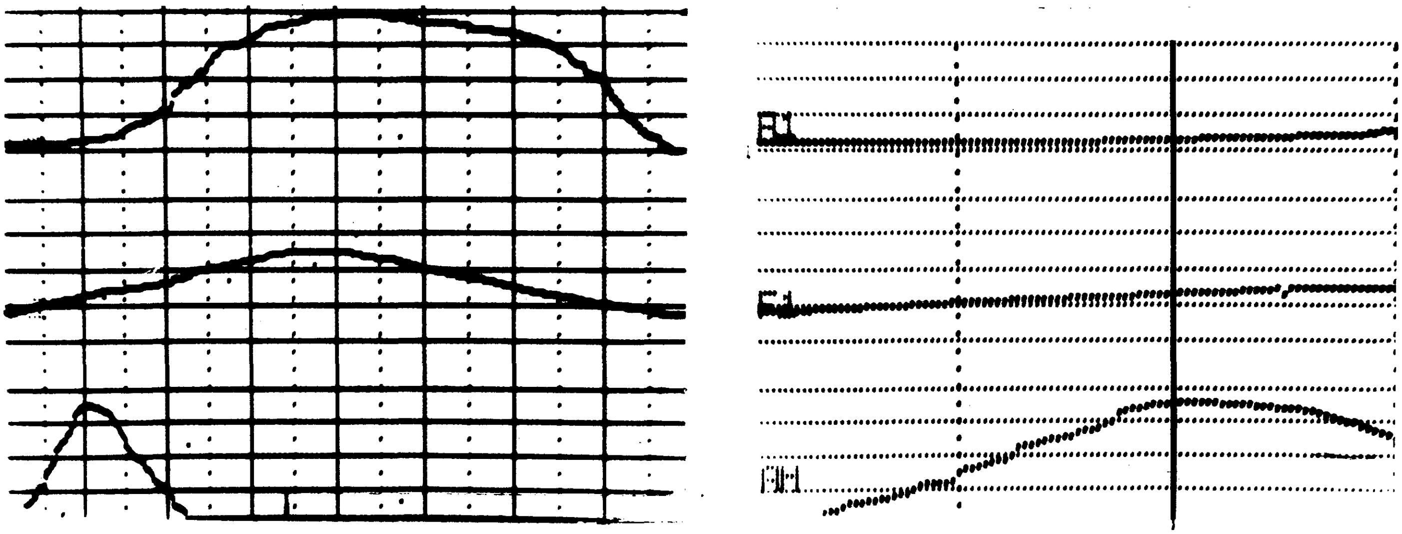


Figure 6: (Left) A display of three of the parameters for control of a terminal analog speech synthesizer (A1, F1, and AH).
(Right) A display of a portion of the same curves on an expanded time scale.

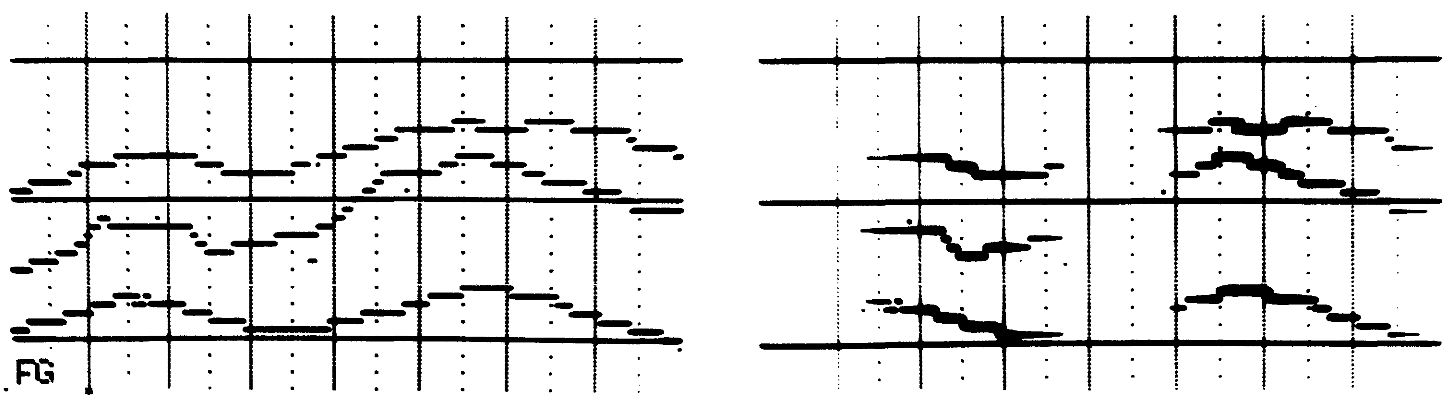



Figure 7: (Left) A display of the formant frequencies (F1, F2, and F3) on the same scale. (Right) The same group of formants with their amplitudes indicated by varying their intensity in accordance with the corresponding A1, A2, and A3 curves.

PREPROCESSING PHYSIOLOGICAL SIGNALS

Miss Maxine L. Paulsen
Medical Systems Development Laboratory
2121 K Street, N. W.
Washington, D. C. 20037

Abstract

A PDP-8 program has been developed to receive and preprocess as many as eight physiological signals simultaneously from a monitored patient (or signals from eight patients). Input to the multiple channels is analog signals, which are sampled 500 times/second and digitized. The program performs a code recognition of each signal and stores this and the subsequent data valued temporarily on a drum until the number points required for the analysis has been accumulated. Concurrently with other instructions, the data break facility allows short blocks of data to be written on the drum or long blocks read back into core. This long-block data, needed to do the analysis, is relayed via an interface to a second computer (CDC-8090), which transfers it to magnetic tapes. These tapes can be used as input to a third computer (CDC-160-A), which consolidates, analyzes, and interprets the signals for each patient. The operations performed by these three computers can be carried on simultaneously once the first input tape has been written.

## Introduction

One of the functions of the Medical Systems Development Laboratory has been the development of computer programs to input, identify, and analyze physiological signals. This work has been centered about a CDC 160-A computer system with a single channel input from an A/D converter.

New commitments have required multichannel input and some means of partially processing the data, to free the 160-A for analysis of the signals.

## The Preprocessing Hardware

To meet these needs, the MSDL purchased a Preprocessing SystemF from the Digital Equipment Corporation. The equipment includes the following:

* 1. 12-K PDP-8 Processor
* 2. 1 Type DM01 Data Channel Multiplexer
* 3. 1 Type RM08E Serial Magnetic Drum
* 4. 1 Type 139E General Purpose Multiplexer Control
* 5. 1 Type 138E General Purpose Analog to Digital Converter
6. 1 Type TC01 DEC tape Control Unit
7. 2 Type TU55 DEC tape Transports
* 8. 1 Programmable Real Time Clock
9. 1 PC01 High-Speed Paper-tape Reader and Punch
* 10. 1 Signal Input Routing Net work package (includes 24 type A103 Multiplexer Seitches and 64 amplifier Mounting Boards)
* 11. 1 High-Speed, two-way Interface to CDC 160-A Computer
* 12. 1 Interrupt and Skip Logic with 8 Program flags

* Equipment used in the program described in this paper.

## Input-Output Specifications for PDP-8 Program

The remainder of this paper will describe a PDP-8 program, which I have written for multichannel input.

The input to this program can be any physiological signal that is preceded by 3 or 13 BCD digits represented by square waves. Data can be received on up to 8 channels simultaneously. The data are digitized to a precision of 10 bits while being sampled at the rate of 500 times/sec. The output from this program is a "long-block" transferred to our CDC-8090 (a stripped 160-A) via the high-speed interface. A CDC program inputs the "long-block" data (referred to as a "lead"), writes it on magnetic tape, and lets the PDP-8 know when it is ready for another "lead" to be sent. This "long-block" or "lead" data has 2 words for channel identification, 13 words for the recognized BCD digits, and 2032 data points from the signal. These data points are used in the analysis of this signal by a CDC Program.

## Timing and Storage Considerations

We chose an input block size of 128 words. Eight blocks this size (allowing one from each channel) can be transferred from the core to the drum faster than the next eight blocks will be brought into core from the A/D Converter. It takes the same amount of time to input one or eight blocks into core, because of the sampling rate. We chose this block size from the timing table compiled by Mr. F. D. McLernon. (See figure 1) Two blocks, 128 words each, are reserved for each channel. Data coming into the core from a given channel are collected into Block 2 for that channel. Block 1 for that channel is full and ready to be written on the drum and, because of our block size, there will be enough time to write it before Block 2 is full. Then, Block 1 and Block 2 can switch roles; Block 2 has been written to the drum so that new data can be accumulated into it and Block 2 is full and ready to be written. The total storage required for these input blocks is:

$$8 \text{ channels} \times 2 \, \frac{\text{blocks}}{\text{channel}} \times 128 \, \frac{\text{words}}{\text{block}} = 2040 \text{ words}$$

Associated with these blocks is the "Block Ready Table". This table is used to keep a record of all the blocks to be written. Thus, only one of the two blocks for any given channel should be recorded in the table at any time. The entries recorded in the table are of the form XXOC, where XX00 is the starting address in core of the block to be written. C is the associated channel and is used indirectly to determine the starting location on the drum where that block is to be written. Once the block has been written, that entry in the table is replaced with 0000. There are two pointers associated with this table. The pointers advance to the end of the table, then circle back to the beginning. Pointer 3 follows pointer 2. Pointer 2 gives the location in the table where the next block ready is to be recorded. Pointer 3 gives the location in the table which contains the location in core of the next block to be written to drum. These are used to insure that the first block read into the core will be the first block written on the drum, the second in will be the second written, etc. (See figure 2)

For each channel, 16 such blocks (128 words each) are accumulated on the drum in consecutive locations to make up a "lead" to be read back into the core, and then transferred to the CDC Computer. There is a "Lead Ready Table" used to keep a record of all the "leads" ready to be transferred. This table has 64 entries because that is the number of "leads" that can be stored on the drum, a maximum of eight "leads" from each channel. The entries in this table are of the form YYFO, where YYFO is the starting location on the drum of the "lead" and F is 0, F is set to 4 to avoid having 0000 when "lead" starts at location 0000 field 0. When the field is 1, F is set to 1. When a "lead" has been read in to PDP-8 core and transferred, that entry in the "Lead Ready Table" is set to 0000. There are two pointers associated with this table too. Pointer 0 is the location in the table where the next "lead" completed is to be recorded. Pointer 1 is the location in the table which contains the location on drum of the next "lead" to be read and transferred. These are used to make sure that the "leads" are transferred in the same order they are completed. (See figure 2)

## The Program

### Initialization

The program starts by clearing the flags of the peripheral equipment.

Next, the usual initialization, such as clearing tables, setting counters and initial exits, etc. is done. Then the drum flag is set by writing a sector on the drum. The clock is set to interrupt 500 times/sec. The program comes to a halt, and when everything is ready, "Continuous" is pressed. The clock is started, the interrupt turned on, and the program is sent to the write routine, where the first interrupt will occur. (See figure 3)

Interrupt Service

The interrupt is off during interrupt servicing.

Answer Interrupt - The clock interrupt flag is cleared, the contents of the accumulator are saved, and the return from interrupt is set up. (See figure 4)

A/D Service Routines - (one for each channel 0-7) A value from the signal is digitized (0-10 bits) and stored right adjusted. When all the channels are serviced, the program goes on to test the eight words just read in. (See figure 4)

Test Routines or Path Selector - (one for each channel 0-7)

Exit 1 - No data Path - Originally set to come to this path. As soon as a non-zero value is received on this channel, Exit 2 is set. (See figure 5)

Exit 2 - Code Recognition Path - The data values while in this path are examined point by point, and the BCD digits represented by the square waves are recognized. (This program was translated to a PDP-8 program for CDC 160-A program). After the required number of digits have been found and stored in the first block for a "lead" Exit 3 is set. (See figure 5)

Exit 3 - Data Store Path - The first block of data has 2 words for channel identification; the next 13 words are the BCD digits just recognized; the remaining 113 words are filled with data points from signal. When these 16 blocks (a "lead") have been input, Exit 2 is set to wait for the next BCD Code. As each block is filled, it is recorded in the "Block Ready Table," pointer 2 is incremented, and the functions of Block 1 and Block 2 are switched. (See figure 5)

Exit from Interrupt Service - When each of the eight words has been processed, the interrupt is turned on and the program returns to where it was when it was previously interrupted. (See figure 5)

I/O Routines

Drum Write - By using pointer 3, the program checks "Block Ready Table" to see if there are any blocks to write. If not, the program goes to the Drum Read Routine. If so, the block is written from the starting core location XX00 given in the table to a location on drum found indirectly by using the channel number C, which is also found in the table entry. When written, the entry in the "Block Ready Table" is set to 0000 and pointer 3 is incremented. The drum location for this channel is advanced properly. When 16 blocks have been written, a "lead", has been stored on the drum. It is recorded in the "Lead Ready Table" in the location indicated by pointer 0, then pointer 0 is incremented. The program then goes back to Drum Write Routine, to see if there are any more blocks to write. (See figure 6)

Drum Read - By using pointer 1, the program checks "Lead Ready Table" to see if there are any "leads" to read and transfer. If not, the program goes to the Drum Write Routine. If a "lead" is ready, and the last transfer completed, the "lead" is read in from location and field YYF0 of the drum into the space reserved for a "lead" in core. Then, this entry in the "Lead Ready Table" is set to 0000 and pointer 1 is incremented. (See figure 6)

Transfer to CDC - The program sets up and initiates the transfer of the "lead" from the PDP-8 core to the CDC computer core, and then goes to the Drum Write Routine. (See figure 6)

## Testing the Program

This program has been tested by using as input the same signal split 8 ways. We feel that this is one of the most severe tests, in that longest paths are encountered at the same time on all 8 channels; all the input blocks are full at the same time, and 8 leads are completed at the same time. This test included transferring the data to the CDC Computer where a program accepted the data and wrote it on magnetic tape in a format that can be used by the analysis program.

## Conclusions

A simple program for the PDP-8 has been written and tested. It embodies a straightforward method in which each channel has its own routines, thus simplifying changes, additions, and deletions to any one or all of the channels. We expect to use it as a basis for increasingly powerful programs.


## ACKNOWLEDGEMENTS

Dr. J. Whiteman* wrote the Code Recognition program for the 160-A.

Mrs. Anna Lea Weihrer** and Dr. Whiteman suggested use of the lead rather than the patient approach. Miss Doyle Darragh** wrote the CDC Program to receive the data from the PDP-8.

Mr. F. D. McLernon, who has been working with me from beginning, deserves equal credit for the accomplishment of this project, especially in the area of hardware and timing.

* Formerly of MSDL, now U.S.P.H.S. Hospital, San Francisco, California

** MSDL employees


TIMING TABLE

| Block Size (# words) | Drum Transfer Time Core → Drum (mas. access time=17.3ms): 1 Channel | Drum Transfer Time Core → Drum (mas. access time=17.3ms): 8 Channels | Data Input Time A/D → Core: 1-8 Channels |
|---|---|---|---|
| 16 | 17.3 + 0.25 = 17.55 ms | 140.4 ms | 32 ms |
| 64 | 17.3 + 1.00 = 18.3 ms | 146.4 ms | 128 ms |
| * 128 | 17.3 + 2.00 = 19.3 ms | 154.4 ms | 256 ms |
| 256 | 17.3 + 4.00 = 21.3 ms | 170.4 ms | 512 ms |
| 512 | 17.3 + 8.00 = 25.3 ms | 202.4 ms | 1024 ms |
| 1024 | 17.3 + 16.00= 33.3 ms | 266.4 ms | 2048 ms |

* The input block size we use (8 blocks can be written on the drum faster than 8 blocks can be read into core.)

**Figure 1 TIMING TABLE**

EXAMPLES OF TABLES

| LOC | CONTENTS | |
|---|---|---|
| 0042 | 0165 | (Pointer 2) |
| 0043 | 0161 | (Pointer 3) |

"Block Ready Table"

(XX0C)

| LOC | CONTENTS |
|---|---|
| 0160 | 0000 |
| 0161 | 4201 |
| 0162 | 4402 |
| 0163 | 5406 |
| 0164 | 5607 |
| 0165 | 0000 |
| 0166 | 0000 |
| 0167 | 0000 |

Pointer 2 - Where to record next block ready

Pointer 3 - The block to be written on drum next

XX00 = Starting location in core of the block (128 words)

C = Channel number 0-7 (used indirectly to find starting location on drum.)

| LOC | CONTENTS | |
|---|---|---|
| 0040 | 3006 | (Pointer 0) |
| 0041 | 3002 | (Pointer 1) |

"Lead Ready Table"

(YYF0)

| LOC | CONTENTS |
|---|---|
| 3000 | 0000 |
| 3001 | 0000 |
| 3002 | 0040 |
| 3003 | 0010 |
| 3004 | 2210 |
| 3005 | 4040 |
| 3006 | 0000 |
| 3007 | |
| ⋮ | |
| 3072 | 0000 |
| 3073 | 0000 |
| 3074 | 0000 |
| 3075 | 0000 |
| 3076 | 0000 |
| 3077 | 0000 |

Pointer 0 - Where to record next "lead" ready

Pointer 1 - The "lead" to be read and transfered next

YY00 = Starting location on drum of "lead" (2048 words)

F = Drum Field 0(=4) or 1.

Figure 2 EXAMPLES OF TABLES

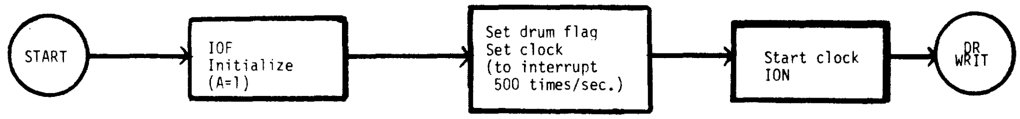



Figure 3 INITIALIZATION

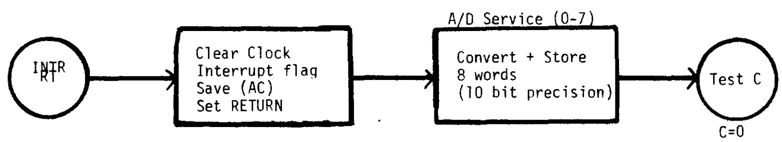



Figure 4 INTERRUPT SERVICE

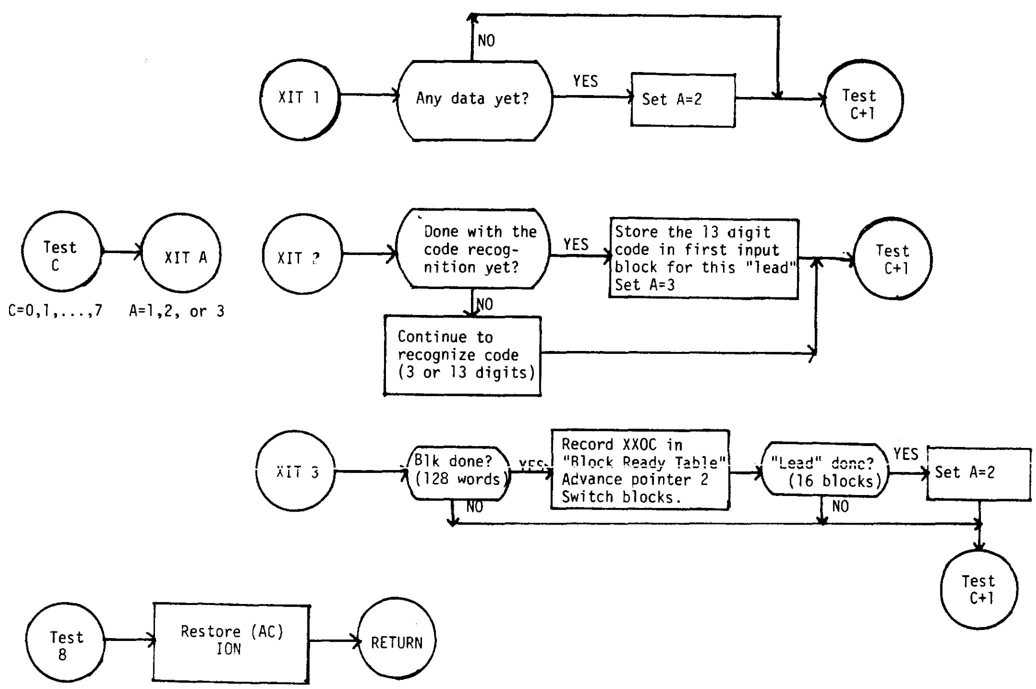



Figure 5 INTERRUPT SERVICE (cont.)

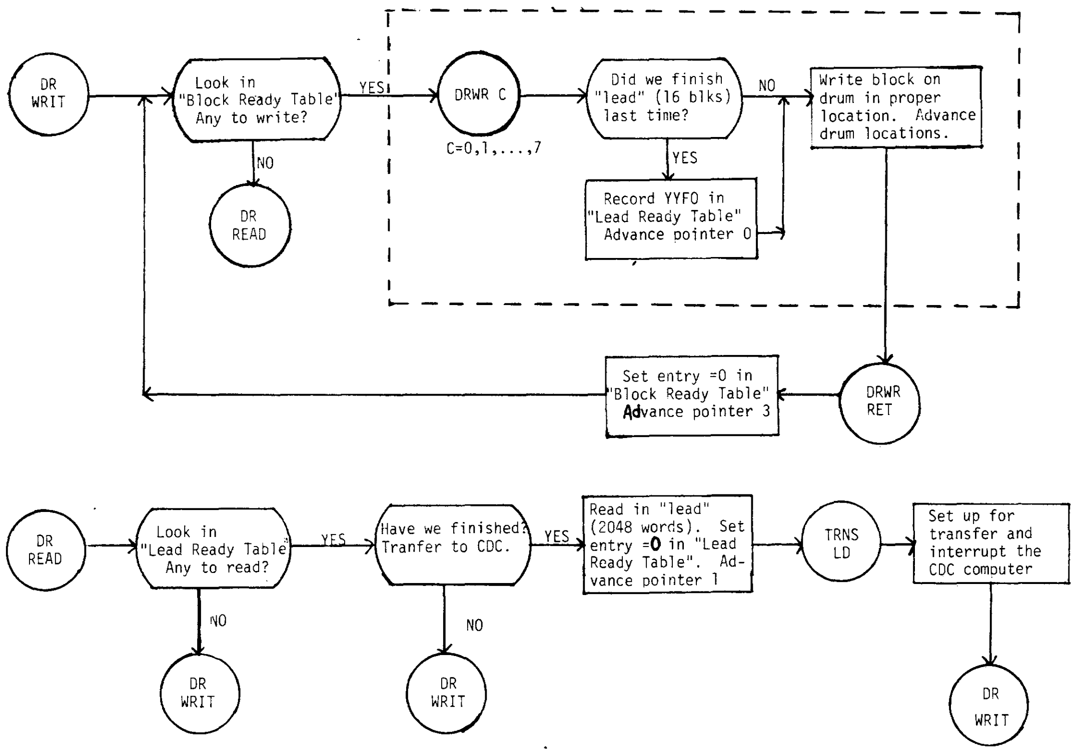



Figure 6 I/O ROUTINES (INTERRUPT IS ON)

# DEVELOPMENT OF CARDIOVASCULAR PULMONARY PATIENT CARE TECHNIQUES*

Jerome A. G. Russell
Research Data Facility
San Francisco, California

## Abstract

The Research Data Facility employs a DEC PDP-7 to develop physiological monitoring and modeling techniques in their cardiovascular and cardiopulmonary clinically-oriented research activities. The PDP-7 is connected to several transducers which monitor patients during the course of open-heart surgery. Once these techniques have been developed, they are included in a standard library of programs which monitor the recovering patient on a 24-hour-a-day basis. Many of the computer techniques employ a combination of analog pre-processing under control of the digital computer. Several of these patient care development efforts will be described.

* This paper was not received for publication.

# A PROGRAMMED DISTRIBUTION GENERATOR*

David N. Samsky
Booz, Allen Applied Research Inc.
Albuquerque, New Mexico

## Abstract

Hardware additions to a PDP-8 analyzer system have made it possible to form a Poisson-weighted sum of the n-fold convolutions of any distribution. The associated program, consisting of a generator and a monitor, uses as an input any initial spectrum and pulses from a random pulse generator. The hardware and program have been used successfully to generate Poisson statistics and spectra from the stochastic process of dark current from a photomultiplier tube. In the field of radiation dosimetry, it can be used to generate the distribution of expected dose in a cell, knowing only the single-event energy deposition spectrum and the average number of events per cell.

## Introduction

With the advent of small computers, it has become possible for individual research groups to buy and use them for their specific research task. A PDP-8 computer was purchased with analogue-to-digital converters and special interfaces so that it could perform multiparameter analysis or pulse-height analysis for two different sets of signals. It soon became evident that the computer had many more uses. For example, the computer has been used to do time-of-flight neutron spectroscopy, to perform life-time measurement of excited states, as a sampling scope to examine very fast repetitive pulses, and as a Programmed Distribution Generator.

The Programmed Distribution Generator (PDG) was conceived to predict the final distribution of the energy deposited in a biological cell if the probability of more than one ionizing event was non-zero. Using hardware and software, the PDG is able to form the distribution resulting from the Poisson weighting sum of the n-fold convolutions of any original distribution. In addition to the existing computer facility, it was necessary to construct an interface to perform probability measurements. The software consists of two parts, a generator and a monitor. The generator takes the initial input spectrum and translates it into the generator distribution that is used by the monitor. The monitor controls the functions of the hardware and sorts the resulting information into the final distribution.

The terms used in the description of the Programmed Distribution Generator are based upon microdosimetry. The definition of sub-event, event, and run are summarized in Table 1.

## Theory

The pulse-height spectrum is a measure of the probability that an event will occur, in terms of its measured parameter. In

* This article is based upon work performed in the radiological physics group, Battelle Northwest Laboratory.

microdosimetry, it is the probability that an ionizing particle will deposit an energy $\Delta Z$ in a single cell. Figure 1 shows the probability as various values for fixed energy intervals. An equally satisfactory, although not as common, representation is shown in Figure 2; it has equal heights with various energy intervals. Here the probability of a sub-event happening with energy deposition between $\Delta Z_i$ and $\Delta Z_{i+1}$ is the same, but the intervals between the $\Delta Z_i$'s are not equal. This condition is simulated by assigning to the fixed interval between counts of a counter a fixed probability of a sub-event happening. After a predetermined number of counts, the sum of the interval-probabilities is exactly equal to one, indicating that an event with more energy cannot occur. The generator function is defined as the distribution of energy as a function of interval-probabilities. Figure 3 is the representation of the generator function of the previous distribution. If we randomly choose one of the intervals many times, the resulting distribution of number versus energy will be identical to the first distribution with equal energy intervals. This is the desired result. Note that for the middle values of energy, the curve has a minimum slope; hence these values will be chosen in preference to the energy values, where the slope is greater.

In the above, it was required that one, and only one, interval was chosen many times. If we allow any number of intervals to be chosen in a random manner, the final distribution will be the Poisson-weighted sum of the n-folded convolutions of the original distribution. That is, the final distribution P(Z) given will be:

$$P(Z) = \sum_{n=o}^{\infty} e^{-m} m^{n} (n!)^{-1} P_n(Z)$$

where

$$P_n(Z) = \int_{o}^{Z} P(\Delta Z') P_{n-1}(Z - \Delta Z') d\Delta Z'$$

and

$$P_o(Z) = \delta(Z) \quad .$$

The first terms of the sum are the Poisson weighting factors and $P_n(Z)$'s are the $n^{th}$ folded convolution of the original distribution $P(\Delta Z)$. $P_o(Z)$ is the Dirac delta function.

## Hardware

The machine consists of a PDP-8 computer, an interface, and a Nuclear Data 160 dual-parameter analogue-to-digital converter (ADC). The ADC has been modified so that under program control, the two halves can be used independently as single-parameter ADC's. The computer includes the automatic multiply and divide option, so that on-line computation time is kept to a minimum. Part of the interface includes the data break facility, which allows the rapid transfer of data into and out of the computer. When operated as a single-dimension pulse-height analyzer, a signal is digitized by the ADC and the program is interrupted by an incremental data break. The regions into which the data can be stored are determined by external selection switches. The number of channels can be set to 1024, 512, or 256.

A special interface, shown schematically in Figure 4, was built between a random pulse generator and the computer. When the system is operating, a signal from the random pulse generator, corresponding to a sub-event, causes the machine to enter a one-cycle break and the inhibit circuit to stop the clock. The computer reads the counter register and stores its value in memory, as specified by the location register. The value-accepted pulse from the computer increments the location register and resets the inhibit circuit. The clock restarts and continues to scale into the counter register until the next random pulse occurs or until the counter register reaches 256 counts. When the latter happens, an event-complete signal stops the clock and signals the computer to perform an analysis of the event. After analysis, under program control, the various registers are reset, the clock is started, and another event is performed.

The additional complications involved in the inhibit circuit and the location register were made to assure an equal probability of transferring to the computer the 256 possible values of the counter register. The timing of the inhibit circuit makes it possible to transfer all 256 values, providing the reciprocal of the repetition rate of the external pulser is less than the average time between clock pulses (roughly 1 μ sec). The incrementation of the location register provides storage of the counter values in consecutive memory locations. Thus, the program functions can be performed independently of the interface. A one-cycle data break was used because it is three times faster than the three-cycle break.

The external pulse should be shaped so that it is at least 1/2 μ second wide and 150 millivolts negative. The average rates can be from a few to $10^5$ per second, corresponding to an average number of sub-events per event of from $10^{-3}$ to 30. Care should be taken to minimize dead time after a pulse has passed. Usually 2 μ seconds are sufficient; but for higher counting rates, dead times of 1 μ second or less should be used.

## Software

The generator and monitor programs are based upon the dual one-dimensional analysis and display program (DODAD). This program performs all the functions of two single-parameter analyzers. It collects data and displays the resulting spectrum. It can integrate the data, write or punch all or part of them, and compare and/or subtract them from another spectrum. DODAD operates either with 1024 channels and a 12-bit word, or with 400 channels and a 24-bit word. The Programmed Distribution Generator uses the latter.

Figure 5 is the block diagram of the software addition to DODAD. The "P" command calls the generator and the "X" and "C" commands call the monitor. The generator involves three separate operations: an integration, a normalization, and an inversion. The first two steps are straightforward; they leave the 400 channels of data integrated and normalized to 256 counts. The inversion process exchanges the ordinate and the abscissa so that there are 256 channels of data normalized to 400 counts. If we recall a previous slide where the probabilities were all equal, but the energy increments were not, I think we can see how the above steps have given us the desired generator function.

Since the action of the interface is to choose randomly a number between 1 and 256, there is an equal probability of choosing any of the generator channels. Because of the integration and inversion, a channel represents the probability of a sub-event occurring with an energy between $\Delta Z_i$ and $\Delta Z_{i+1}$, where the energy intervals are not equal. This result provides a generator function that will reconstruct the original distribution for one, and only one, sub-event per event.

The monitor is used after the generator has calculated the proper generator function. If there are many sub-events per event, it is obvious that the final scale of the generated function can be several times the original. The "X" command determines a new scale factor which can compress the final distribution by a factor up to $2^9$. To continue with the old compression factor, the "C" command is used. After initialization, the keyboard is interrogated so that after each event, it is possible to break out of the monitor loop. The main routine can be described as a test, test, and return operation. Since the occurrence of a sub-event places a non-zero value in memory, the program makes this test on the first appropriate memory location. After finding a value, it looks up the corresponding energy value and goes on to the next memory location. The summation of the energy values continues until the program encounters a zero location. After each such encounter, it tests to see if the event has been completed. If it has, ordinary pulse-height analysis, modulo 1024, is performed. After completing a predetermined number of events (normally about 65,000), the program prints the average number of sub-events per event and returns to the DODAD program.

## Applications

The results of this program are the Poisson-weighted sums of the n-folded convolutions of an input distribution. To use the program, an input distribution is read into memory; the generator is used to form the generator function; and after appropriate clearing of the memory, the random pulses and the monitor form the final distribution. A near-delta function distribution was used to demonstrate the Poisson-weighting feature of this program. In Figure 6 are representations of the original function and the resulting generator function. You will note that any choice of generator number between 1 and 256 will yield an energy between $\Delta Z_1$ and $\Delta Z_2$. Table 2 is a comparison of the experiment and the theory for an average number of sub-events per event of 1.15. They agree within the experimental error of one over the root of events, i.e., 4/10 of 1 percent. Figure 7 shows several experimental distributions for various average values. Two points are noteworthy. First, the peaks corresponding to integral values in the Poisson distribution are distinct and the average value corresponds to the peak of highest probability. Second, as the average value increases, the distribution begins to fall within a Gaussian envelope.

The exponential distribution was used to test the regeneration features of the generator. Figures 8 and 9 show semi-log plots of the exponential distribution for several average n values. In Figure 8, the values are one or less and the resulting distribution still approximates an exponential decay, but with a greater "decay constant". In Figure 9, the plots show definite peaks at or near the average values; they are definitely not exponential in form. Roesch has shown that for large average n values, the folded exponential distribution approaches a Gaussian with a mean value of $\bar{n}$ and a standard deviation of $2\bar{n}$.[1] Figure 10 compares Roesch's theoretical Gaussian distribution with the generator results. It can be seen that, even though the initial distribution was exponential, the final result is nearly Gaussian.

An application of this last distribution can be found in the probability distribution of the energy deposited in a nuclear detector. For example, if a charged particle produced, on the average, 25 detected photons in a scintillator, the output of a photomultiplier tube might look like Figure 9. Since the photo cathode and the first few diodes of a photomultiplier tube produce noise electrons which are exponentially distributed, the final result is a mixture of signal and noise. Using the distribution generator, it should be possible to estimate the average number of electrons produced by noise and to subtract their effect from the final distribution, thereby getting a truer measure of the energy lost by the charged particle.

The original intent of this device was to calculate the final distribution of energy deposited in a cell due to depositions of more than one ionizing event. By starting with a single ionizing event spectrum obtained experimentally, say from a spherical proportional counter, it is possible to calculate the distribution of energy that would be due to more than one ionizing event. Since this work is in its initial stages, definitive results are not available at this time.

Reference

1. W.C. Roesch, Private communication.

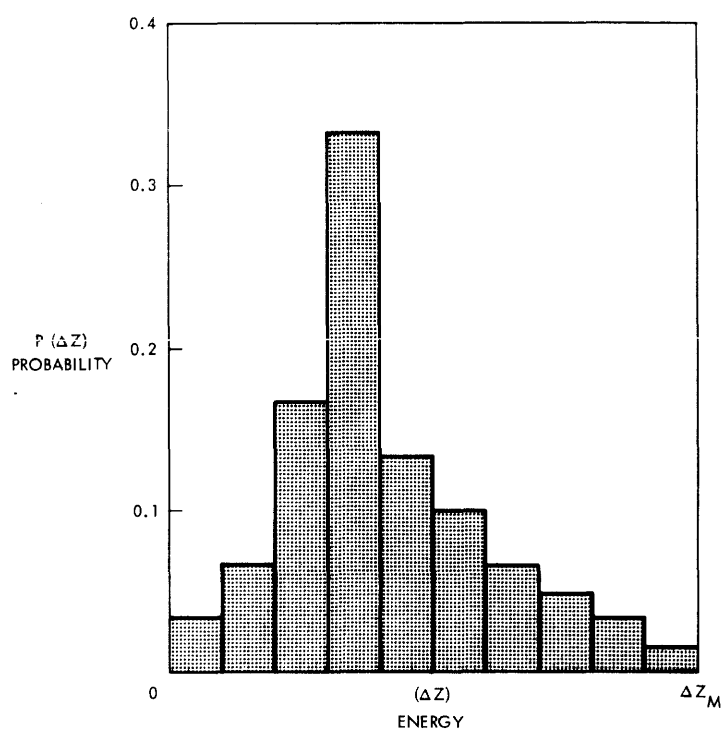



Figure 1 A Probability Histogram with Equal Energy Increments

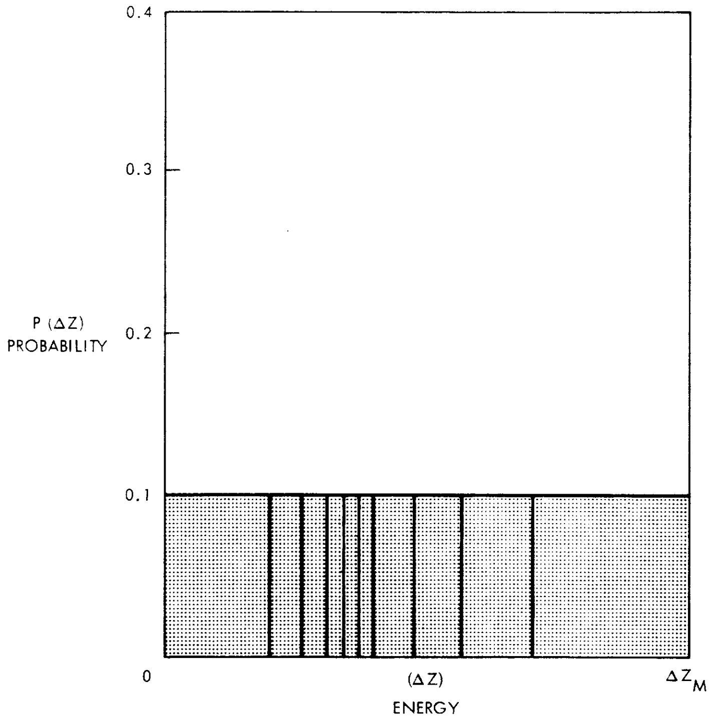



Figure 2 A Probability Histogram with an Equal Probability for Unequal Energy Increments

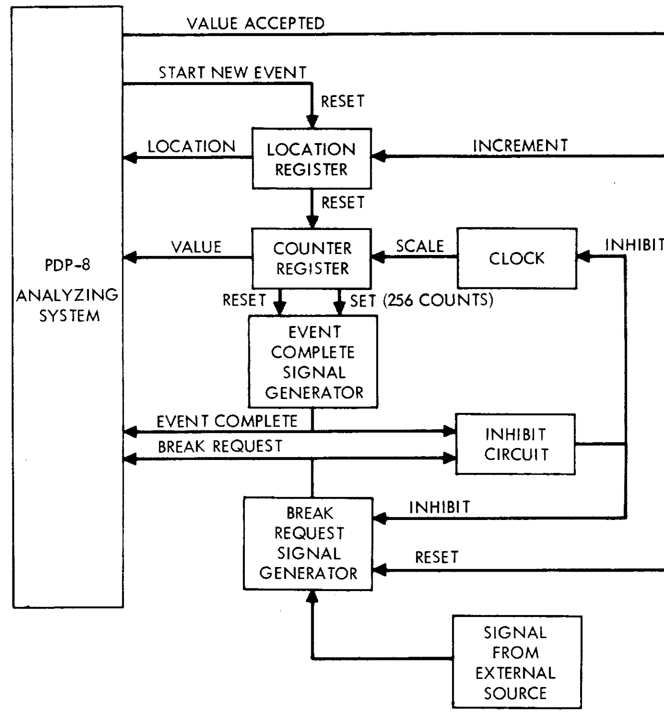



Figure 4 A Block Diagram of the Programmed Distribution Generator Interface

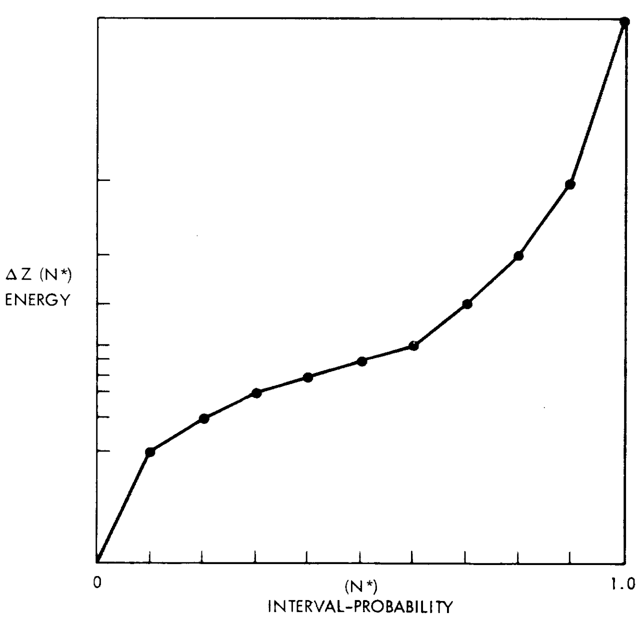



Figure 3 The Generator Function of the Previous Distribution; Energy versus Interval-Probability

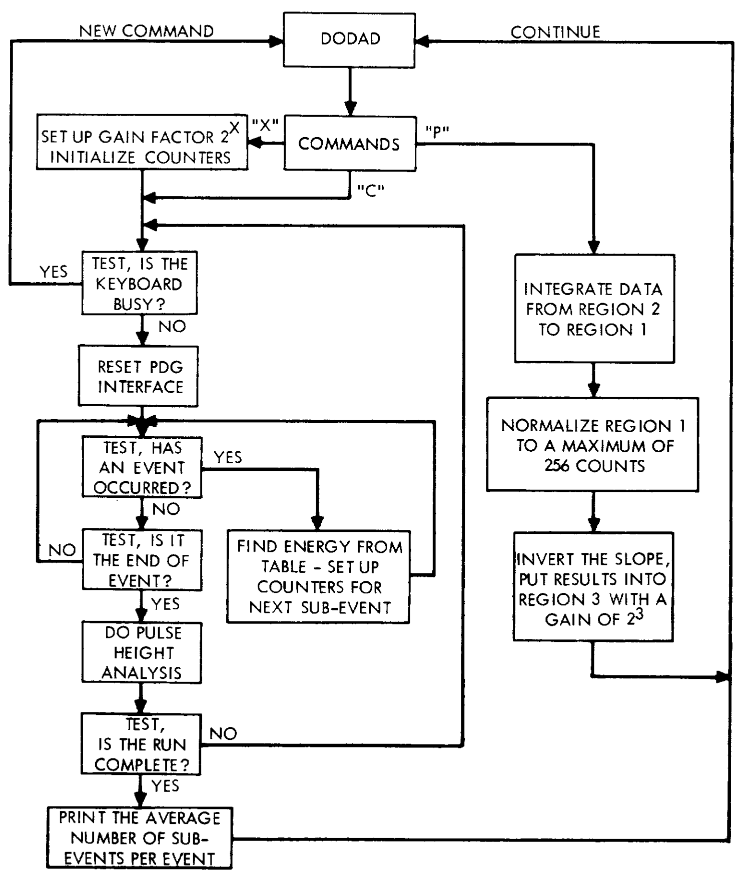



Figure 5 A Block Diagram of the Programmed Distribution Generator Code

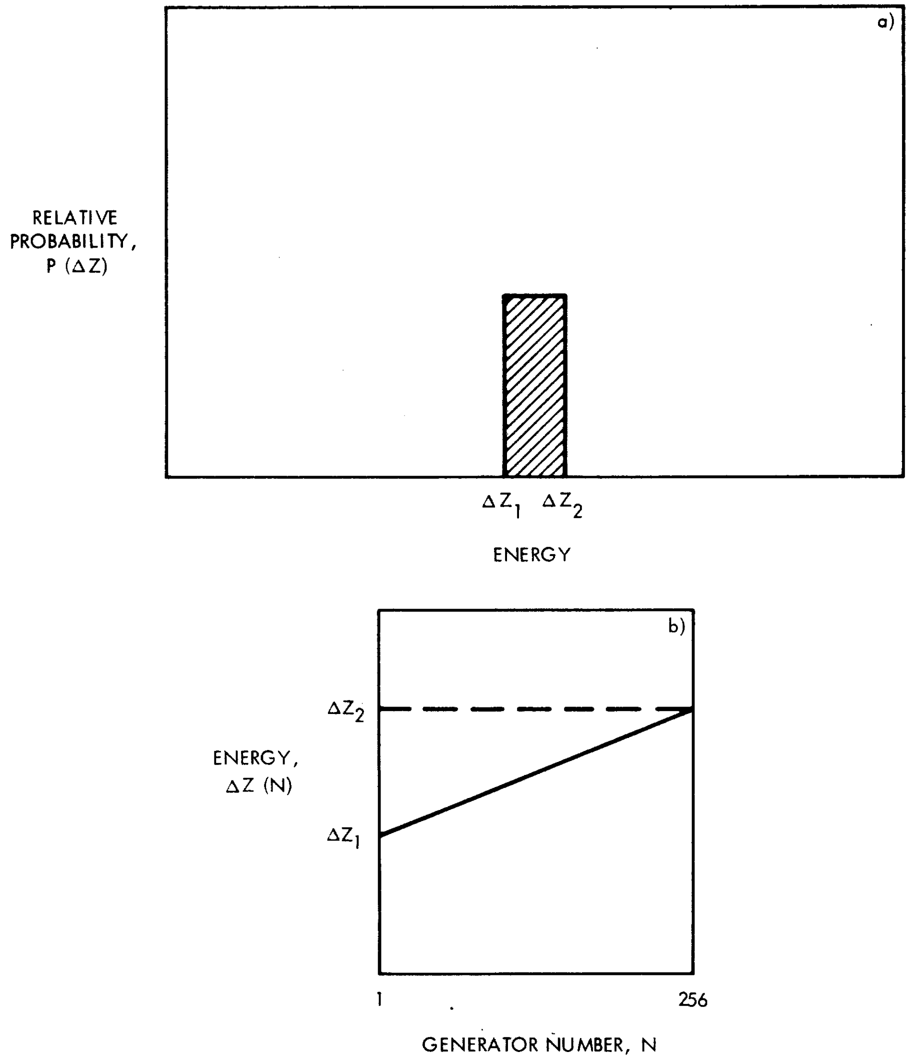



Figure 6 A Near-Delta Function and its Corresponding Generator Function

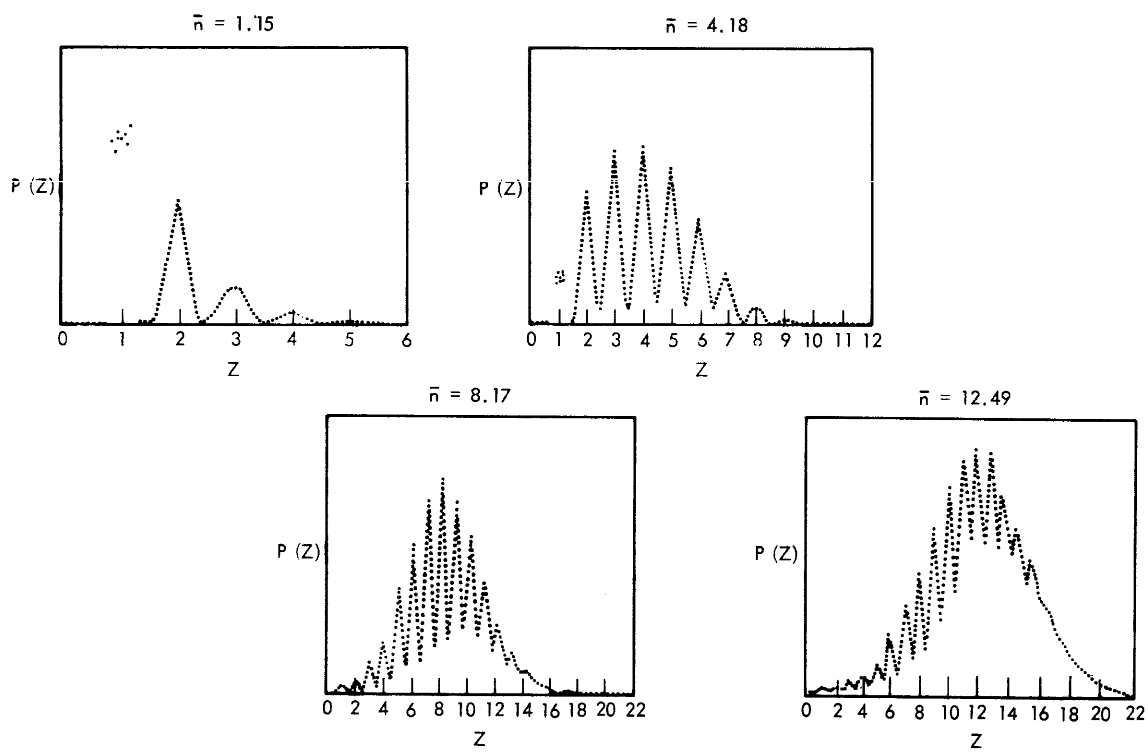



Figure 7 The N-Folded Convolutions of a Near-Delta Function for $\bar{n}$ = 1.15, 4.18, 8.17, and 12.49

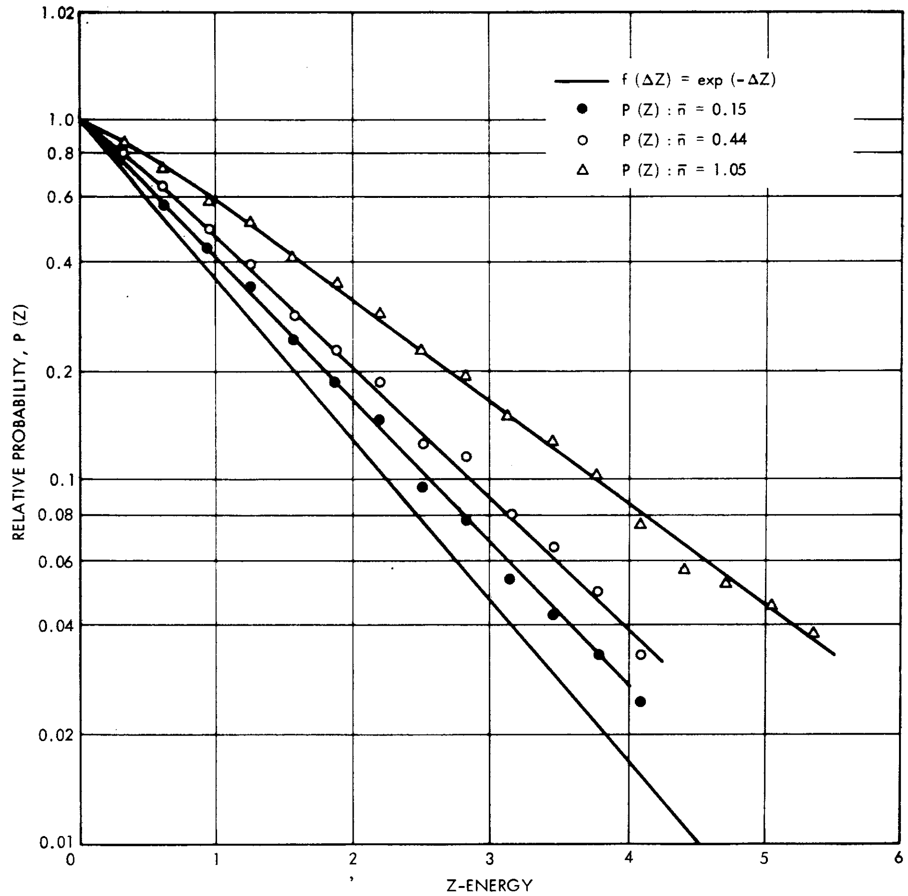



Figure 8 An Exponential Function and its N-Folded Convolutions for $\bar{n}$ = 0.15, 0.44 and 1.05 (semi-log plot)

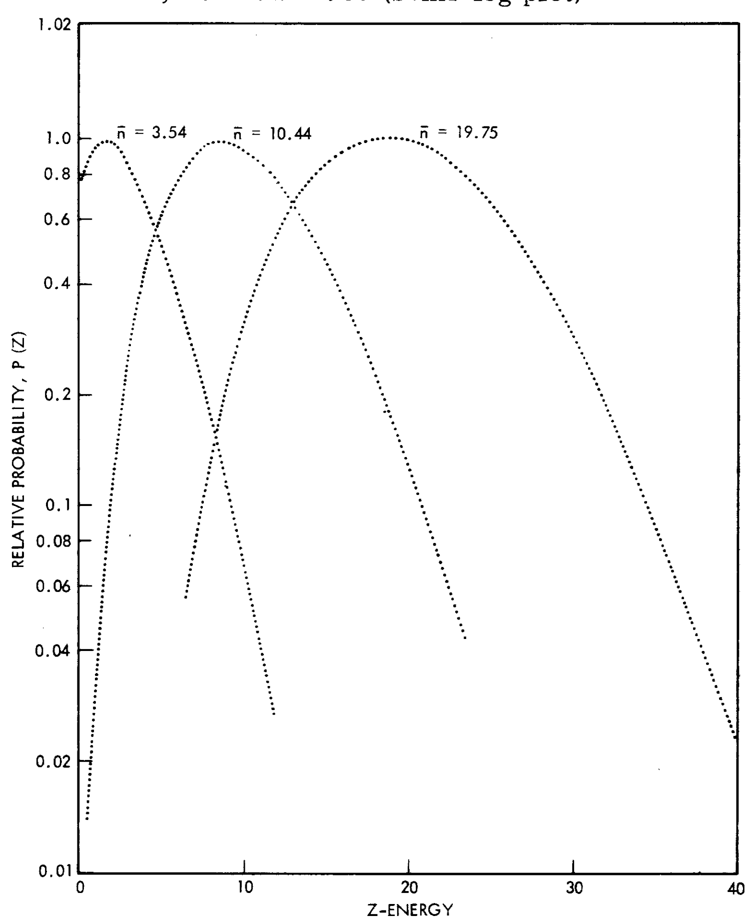



Figure 9 The N-Folded Convolutions of an Exponential Function for $\bar{n}$ = 3.54, 10.44 and 19.75 (semi-log plot)

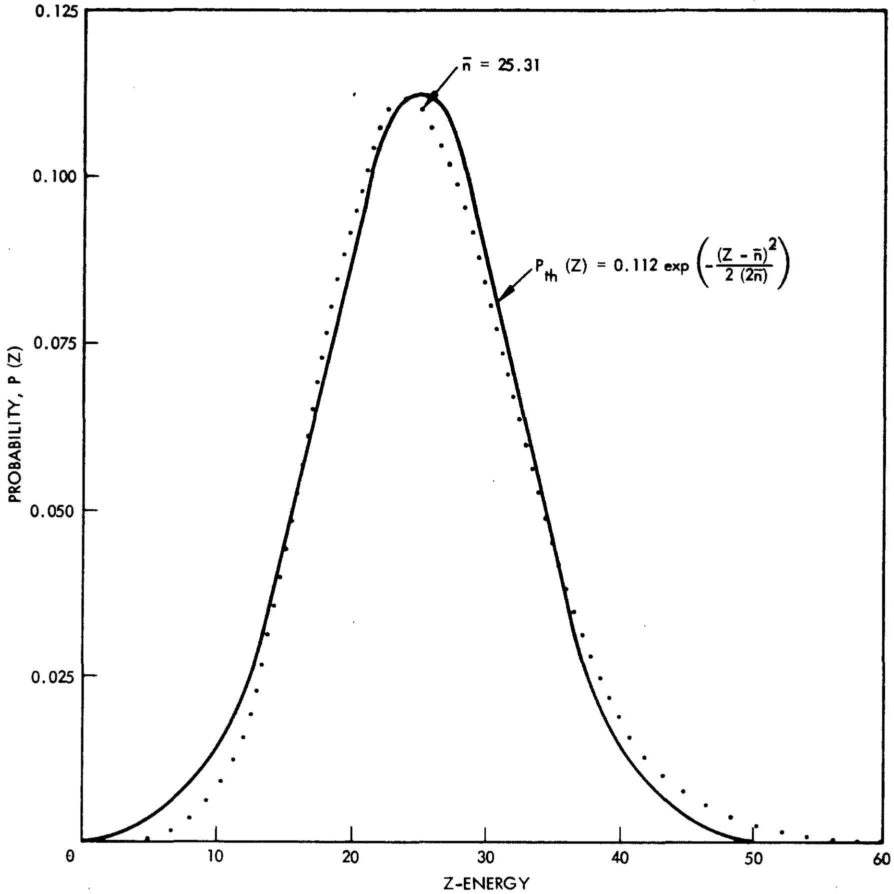



Figure 10 A Comparison of a Gaussian Distribution with $\bar{z} = \bar{n}$ and $\sigma^2 = 2\bar{n}$ and the N-Folded Convolution of an Exponential Function for $\bar{n}$ = 25.31

Table 1 Summary of Programmed Distribution Generator Terms and Their Relation to Microdosimetry

| NAME | SYMBOL | MICRODOSIMETRY | PROGRAMMED DISTRIBUTION GENERATOR |
|---|---|---|---|
| SUB-EVENT | ΔZ | ONE IONIZING EVENT | ONE TRANSFER OF A VALUE TO THE PDP-8 VIA DATA BREAK |
| EVENT | Z | A SUM OF IONIZING EVENTS | A SET OF SUB-EVENTS |
| RUN | P (Z) | A PULSE HEIGHT SPECTRUM | A SET OF EVENTS, TERMINATED BY THE PRINTING OF THE AVERAGE NUMBER OF SUB-EVENTS PER EVENT |

Table 2 A Comparison of Poisson $(e^{-m} m^n (n!)^{-1})$ Distribution with $m = \bar{n}$ and the N-Folded Convolution of a Near-Delta Function for $\bar{n}$ = 1.15

| NUMBER (n) | EXPERIMENT | THEORY |
|---|---|---|
| 0 | 0.321 | 0.317 |
| 1 | 0.366 | 0.365 |
| 2 | 0.210 | 0.209 |
| 3 | 0.078 | 0.080 |
| 4 | 0.022 | 0.023 |
| 5 | 0.004 | 0.005 |

# ANALYSIS OF MULTICHANNEL ANALYZER DATA WITH LIGHT PEN AND FUNCTION BOX TECHNIQUES†

C. Wendell Richardson
Phillips Petroleum Company
Idaho Falls, Idaho

Abstract

A software program package has been developed for a 4K PDP-8 computer using a stepping recorder and DECtape to readily access and store 4096 channel data from a multichannel analyzer. An inexpensive 21-inch display console, light pen, and function box are used to study the data, make fast preliminary calculations to determine parameters, and obtain final results. The need for listings and plots has been eliminated and an analysis can be completed immediately after the data are taken.

## Introduction

The precision measurement of gamma-ray energies will be used to illustrate the feasibility of using a 4K PDP-8 computer in the analysis of 4096 channel multichannel analyzer data. An obvious problem must be considered. The data consists of 4096 18- to 20-bit words whereas the memory of the computer is only 4096 12-bit words which must be used for both program and data storage. To be useful the programs must be capable of generating meaningful displays of the data from which the user can determine parameters for extensive calculations. This study will demonstrate that even though the data and programs must be divided into sections both display and computation can be handled with ease.

The program package was divided into the following four parts to solve the storage problem:

1. Input program - 4096 18-bit words are taken from the conventional multichannel analyzer and made available to the computer.

2. Display program - One quadrant of the data is stored in memory at one time for display and computation with access time of about 3 seconds for the other quadrants.

3. Editing program - Results from the display program are available for study and editing.

4. Energy calculation program - The floating point package is used for extensive computation using information from the editing program and final results are printed.

## Equipment

The configuration consists of a 4K PDP-8 computer and teletype with a digital stepping recorder (DSR), three DECtape units, a 21" oscilloscope, a light pen, and a function box. Paper tape input of data allows use of the package without the stepping recorder.

The function box consists of six push buttons which set a register that can be read by the computer. When it has been determined that a button is down the corresponding function is executed.

## Energy Measurements

The output pulses from a lithium-drifted germanium [Ge(Li)] detector are digitized to form a channel number by the multichannel analyzer. The analyzer memory locations corresponding to the channel numbers are incremented to record the pulses. Fig. 1 is a plot of counts versus channel number. The counts in a particular channel indicate the number of times the corresponding memory location was incremented. Peaks on the graph indicate the presence of gamma rays whose energies are a function of the peak positions. Scattering effects, detector resolution, etc. determine the shape and width of the peaks.

† Work performed under the auspices of the U. S. Atomic Energy Commission.

To measure the energy of an unknown gamma ray, it is included in a spectrum containing peaks of well known energies. If $X_i$ is the channel location of the $i^{th}$ peak then the known energies can be used to determine the constant G (gain in keV/channel) such that

$$E_i = GX_i \tag{1}$$

where $E_i$ is the energy of the $i^{th}$ peak. However, this expression for $E_i$ assumes the use of a perfectly linear system with a correctly adjusted "zero intercept" which in reality is never the case. To correct for error in the zero intercept, equation (1) becomes

$$E_i = G(X_i + A) \tag{2}$$

where A is the intercept of the multichannel analyzer's analog-to-digital converter (ADC) in channels. The correction for nonlinearity is given by

$$\sum_{K=0}^{N} \alpha_K (A + X_i)^K \tag{3}$$

where the $\alpha_K$ are measured linearity coefficients for the analyzer system. For the derivation of expression (3) see ref. 1. Equation (2) now becomes

$$E_i = G\left(X_i + A + \sum_{K=0}^{N} \alpha_K (A + X_i)^K\right) . \tag{4}$$

Thus G and A can be determined by least squares method from the gamma rays of known energies and the unknown gamma-ray energies may then in turn be determined from these two parameters.

## Analyzer-Computer Communication

A magnetic tape digital stepping recorder (DSR) was interfaced to both a 4096 channel analyzer and the PDP-8 computer. The memory of the analyzer is sequentially transferred to the recorder and written in 18-bit word column binary on the tape. The magnetic tape can then be sent to a main computing center or can be read directly by the PDP08. The computer-stepping recorder connection is via the PDP-8/DSR controller so that the recorder can be operated under program control. The controller allows interrogating the status of the unit and the following operations: Read, Write, Write END of File, Rewind, and Fast Forward. Under program control the user can elect to write either 200 or 556 bpi and binary or BCD codes.

## Permanent Mass Data Storage

DECtape is used as the mass storage medium for the data. Up to twenty-five 4096 channel spectra can be placed on one reel. The random access feature of DECtape is very useful for this particular application since all parts of a spectrum must be available for repeated read in. The largest section that can be handled by the display program is 1024 words. Thus the DECtapes are formatted with 2049 words per block allowing two DECtape words (12 bit) for each analyzer word (18 bit). The complete spectrum is placed in four successive blocks.

The first block of the tape is used for an index of the contents of the tape and the remaining blocks are divided into twenty-five groups such that each group contains the four quadrants of one spectrum. To facilitate the computation required in the display program each 18-bit word is normalized and stored in two computer words in a special floating point format. An exponent, derived from the normalization, is placed in the low order five bits of the low order mantissa. Due to the nature of the data neither the mantissa nor the exponent is signed. In order to allow numbers of magnitude of $2^{20}$ the high order bit of the mantissa is not saved and always assumed to be a one.

The spectra on the DECtape are numbered from one to twenty-five. Associated with each spectrum number is an identification stored in block zero of the tape. The identification is supplied through the teletype at the time the spectrum is placed on the tape. To access quadrants of spectrum number X the following relationships are used:

1st Quadrant —— DECtape Block Number = 4X-3

2nd Quadrant —— DECtape Block Number = 4X-2

3rd Quadrant —— DECtape Block Number = 4X-1

4th Quadrant —— DECtape Block Number = 4X

## Determination of Fitting Parameters Using Display, Light Pen, and Function Box

With the spectral data on DECtape the next step is to use the display program to determine peak positions. The spectrum to be analyzed is located in the index and the first quadrant is loaded into the memory. The number of channels displayed is determined from the switch register setting and ranges from 8 channels to 1024 channels.

A semi-log display is required to handle the range of counts; thus logarithms of the channel contents must be calculated. If the display were the only consideration the data words could be replaced by their corresponding logarithm and the display refreshed from the logarithms; however, the original data are needed for peak position calculations. Hence, a data point must be converted from the two-word floating point format to a 10-bit logarithm each time it is displayed. To keep refresh rates at a reasonable speed a fast logarithm conversion algorithm is required. To construct a logarithm from a two-word floating point format only the exponent and 8 high order bits of the mantissa are used since the scope digital-to-analog converter has a 10-bit resolution.

The distance between decade lines is obtained by taking the maximum exponent allowed to be $24_8$ and the maximum logarithm to be $7777_8$ even though it will be divided to 10 bits for the scope. Since

$$\frac{7777_8}{24_8} = 315_8$$

the logarithm can be expressed as

$$\text{Log} = \text{EXPONENT} \times 315_8 + \alpha$$

where $\alpha$ is determined by the 8 high order bits as follows. A precomputed list of $40_8$ number is used and expressed as:

$$\text{List element \#1} = \log_2 (1 + \frac{0}{40_8}) \times 315_8$$

$$\text{List element \#2} = \log_2 (1 \frac{1}{40_8}) \times 315_8$$

$$\text{List element \#3} = \log_2 (1 \frac{2}{40_8}) \times 315_8$$

.

.

.

.

$$\text{List element \#}41_8 = \log_2 (1 \frac{40_8}{40_8})^* \times 315_8$$

The 5 high order floating point mantissa bits are used to select a list element. The value of $\alpha$ is the selected list element plus bits 6, 7, and 8 of the high order mantissa. Any error due to the linear addition of bits 6, 7, and 8 is insignificant when the 12-bit logarithm is divided to 10 bits for the scope. Computing logarithms in this way requires only 17 computer instructions which can be included in the display loop without making the refresh rate unacceptable.

When displaying less than 1024 channels there is a need to move the display window so as to be able to look at any portion of the spectrum in detail. A rotate function is used for this purpose. Fig. 2 is a sequence indicating the effect of rotating the data through a window. The window width is the same as the number of channels currently being displayed and determined by the switch register setting. Push buttons on a function box are used to control the rotation direction, the rotation speed, and starting positions for peak location calculation and storage of results.

The first versions of the program used function dots on the display screen and a light pen in place of the function box. However, controlling the functions can be handled with much greater ease using the function box. The light pen is now used solely for information retrieval. When the pen is pointed at the data the channel number and contents of the point are printed on the teletype and displayed on the scope.

To assist user-computer communication the point at the center of the screen is displayed at a higher intensity that are the other points. The spectrum can be rotated so as to place any point in the bright dot position.

When a request is given to find a peak location the bright dot is used as a reference point in the calculations. The procedure is to look in either direction from the bright dot for evidence of a peak. The contents of the bright dot channel are compared with the contents of the channels on either side. Assuming that the bright dot has been positioned on the side of the peak, the comparison will show the contents of one adjacent channel greater than the bright dot channel and the contents of the other adjacent channel lower than the bright dot channel. A search is made in the direction of increasing counts until a channel is found with lower counts than the bright dot channel indicating the opposite side of the peak has been located.

For example in Fig. 3 i is the bright dot channel number. Since channel i + 1 has greater counts than i the peak is assumed to be above channel i. Channel i + 4 is the first channel above channel i with lower

* Note than $\log_2 (1 \frac{40_8}{40_8}) = 1$.

counts. Thus channels i and i + 4 are on opposite sides of the peak and will be used to determine the peak location given by

$$\text{PEAK LOCATION} = i + \frac{3 + \dfrac{C_{i+3} - C_{i+4}}{C_{i+3} - C_i}}{2} \text{ channels}$$

where $C_i$ is the contents of channel i. The expression is obtained by using similar triangles. The peak position is defined as being the midpoint of Line A in Fig. 3A. This method assumes the curve of the peak to be linear between channels i + 3 and i + 4. This assumption is improved by using the logs of the counts; hence, the peak location becomes:

$$\text{PEAK LOCATION} = i + \frac{3 + \dfrac{\log \dfrac{C_{i+3}}{C_{i+4}}}{\log \dfrac{C_{i+3}}{C_i}}}{2} \text{ channels.}$$

## Calculation of Energies

As the peak positions are calculated the results can be saved in either a calibration list or an unknown list. The lists are preserved in memory as the editor program is loaded. The editor arranges the list entries in ascending magnitude and allows listing, deleting, inserting and averaging. The averaging function is used when the position of a peak has been calculated from more than one starting point and an average of the results is desired.

When the list editing is complete the entries are converted to three word floating point numbers to be used with the standard floating point package in the energy calculation program. Again the lists are stored in memory as the final program is loaded. In the energy calculation program straight forward use of the floating point package allows reading the linearity coefficients for the system, reading energies of calibration peaks, and evaluating equation (3) to obtain values for the unknown energies.

The shape of the peak as well as the statistics of the data determine the accuracy of the results. If the peak is symmetric with good statistics the peak position calculation is within 0.05 of a channel of being the same as results from gaussian fitting methods.

Table I shows output from a large FORTRAN program using gaussian fitting techniques and output from this package respectively. Comparison of the results indicates good agreement in the calculated values of the energies.

Table I

| Gaussian Method | Present Method | Difference |
|---|---|---|
| 159.590 | 159.568 | - 0.022 |
| 209.550 | 209.551 | + 0.001 |
| 413.541 | 413.537 | - 0.004 |
| 512.510 | 512.511 | + 0.001 |
| 570.900 | 570.865 | - 0.035 |
| 662.398 | 662.381 | - 0.017 |
| 834.562 | 834.556 | - 0.006 |
| 897.238 | 897.207 | - 0.030 |

## Conclusions

The above description of the determination of gamma-ray energies should indicate the potential of the small computer in the analysis of experimental data. Several advantages can be noted: 1) An inexpensive oscilloscope can be used to display the data in a variety of different ways; 2) Using the light pen for information retrieval, plots and listings are not necessary; 3) Parameters can be estimated and tested with immediate results for evaluation and retrial. In the case of energy calculation a complete analysis can be completed in an hour compared to approximately two weeks to format the data, obtain listings and plots, make enough trial runs to determine parameters, and obtain final results by more conventional methods.

Even though the memory of the computer is much smaller than the required amount of memory for program instructions, the use of a bulk storage peripheral eliminates nearly all disadvantages of handling both data and programming in sections.

## Acknowledgments

The author would like to thank W. W. Black for his assistance in developing the programs, D. D. Metcalf and L. C. Walton for their work in developing the initial programs used in the final package, and R. L. Heath for his continued interest and support.

## References

1. W. W. Black, USAEC Report IDO-17140 (1965).

Figure 1 Oscilloscope display of a portion of a 4096 channel gamma-ray spectrum and a number indicating channel position of bright dot.

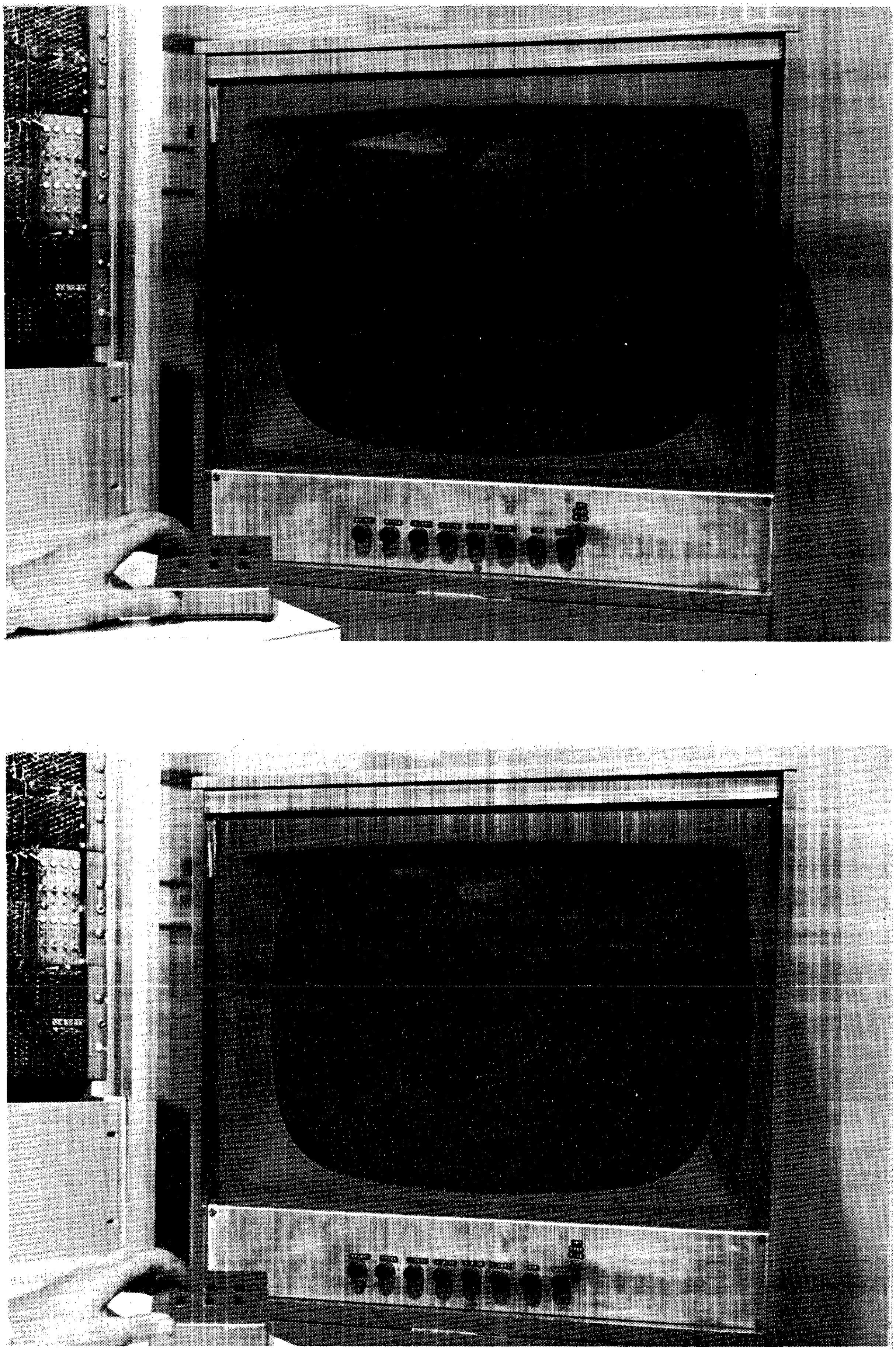


Figure 2 A function box is being used to control the rotation of a spectrum on an oscilloscope. In the bottom picture the spectrum has been rotated 14 channels to the left from its position in the top picture.

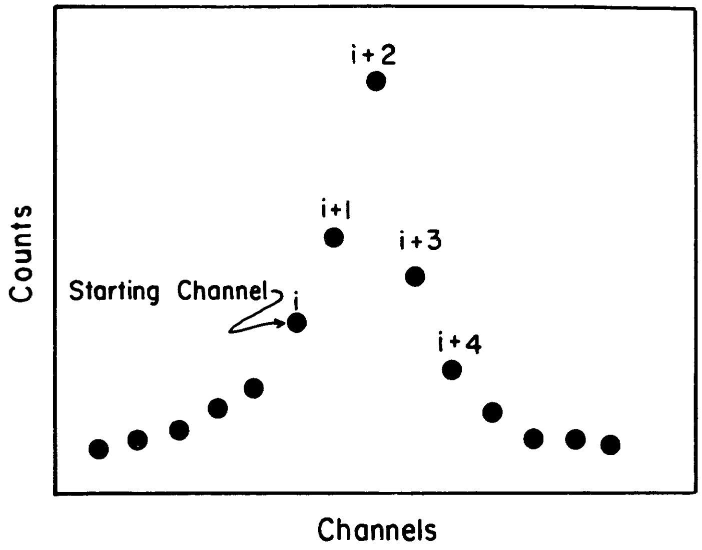


Fig. 3

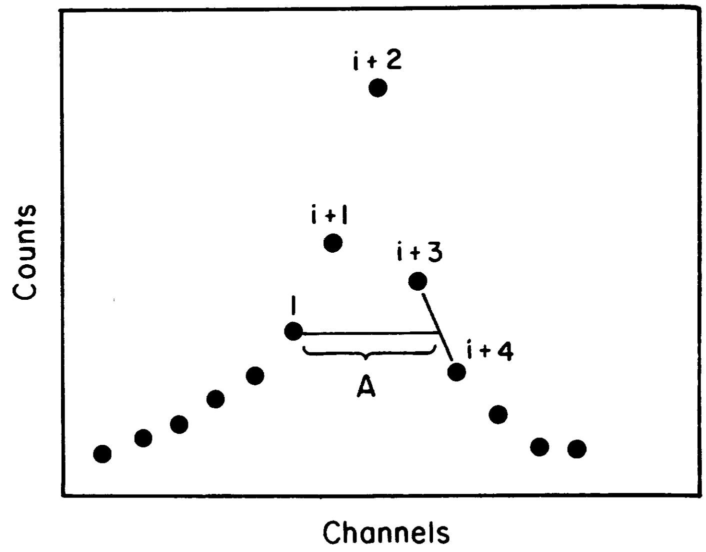


Fig. 3A

Figure 3 Diagram showing method of determining peak position.

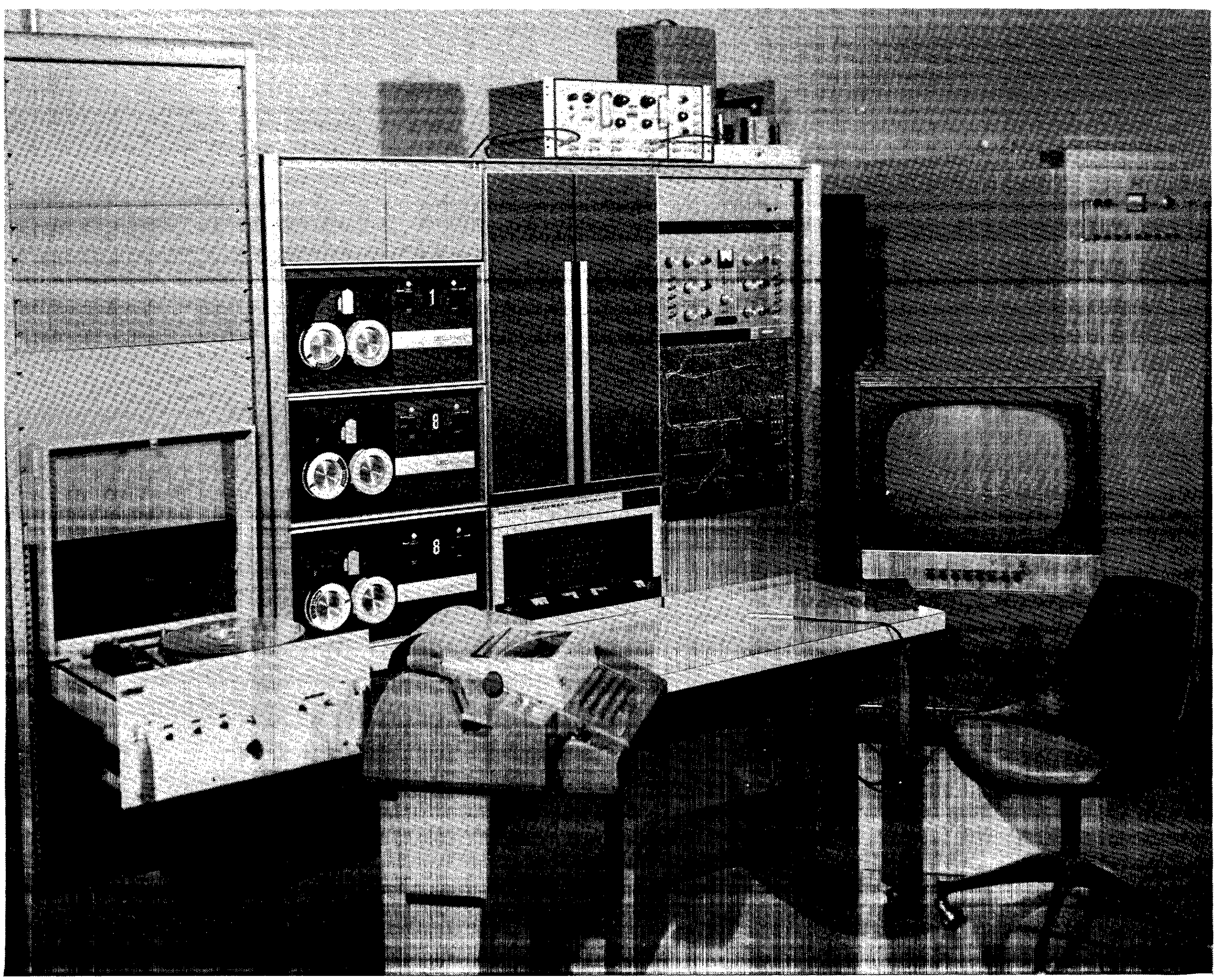


Figure 4 Picture showing system configuration. From left to right is the stepping recorder, teletype, DECtape units, PDP-8 computer, light pen, function box, and 21" oscilloscope.

# AUTOMATIC CALIBRATION AND EVALUATION OF MULTICHANNEL ANALYZERS USING A PDP-8†

W. Wayne Black
Idaho Nuclear Corporation
Idaho Falls, Idaho

## Abstract

A complete package of PDP-8 programs have been developed for automatic evaluation and calibration of multichannel analyzer systems. These programs require a basic PDP-8, a specially designed computer-controlled pulser, and a DEC type 160B interface for dual analog-to-digital converters (ADC's). The programs permit the following parameters to be measured: stability of the ADC zero reference level, stability of the system gain, deviation of the system from a linear response, and ADC channel profiles. These parameters can be measured for any ADC ramp length up to 4096 channels with an accuracy of 5 parts in 100 thousand.

## Introduction

This paper will discuss equipment and techniques that have been developed for the calibration and evaluation of scintillation spectrometers. When this work was originally begun, 1024-channel Analog-to-Digital Converters (ADC's) were the largest available. These ADC's could still be calibrated and evaluated by manual methods, but at the expense of a great deal of labor. When 4096-channel ADC's became available, some manual measurements were still possible, but very time consuming, and certain other measurements became impossible to perform manually. These considerations led to a series of exploratory investigations to determine the feasibility of calibrating and evaluating spectrometer systems using computers.

The techniques were developed on a system centered around a PDP-8 computer. (Similar work has been initiated on a PDP-9.) The system to be discussed includes a 4K PDP-8 interfaced to a dual nuclear ADC unit, a mercury-relay pulser, and a 21" oscilloscope display.

## Parameter Isolation

Fig. 1 illustrates the major features of a scintillation spectrometer. Some of the electromagnetic radiation emitted by the radioactive source interacts with the detector. Although the physical processes differ depending on whether the detector is NaI, Ge, or Si, the end result is the same. In each case a pulse appears at the output of the detector which is related in amplitude to the electromagnetic energy deposited in the detecting material. This pulse is then amplified and fed to an ADC to be digitized, and finally stored as a channel number.

If the spectrometer were ideal, the energy, E, of the electromagnetic radiation deposited in the detector would be related to the resulting channel number, C, by the relation:

$$E = \beta C \quad . \qquad (1)$$

However, the system will be nonlinear, and because of DC voltage levels at certain points in the system all pulse amplitudes will be displaced by some constant amount. Taking these facts into consideration the electromagnetic energy can be more accurately related to the resulting channel number by the relation[1]:

$$E = \beta[\alpha + C + \sum_{n=0}^{N} a_n(\alpha + C)^n] \quad , \qquad (2)$$

where β is the gain of the system in keV/channel, α is the voltage displacement in channels, and the $a_n$ are a set of coefficients that describe the deviations from

† Work performed under the auspices of the U. S. Atomic Energy Commission.

linearity of the system excluding the detector. Since α and β may be different for every experiment, they must be determined from the experimental data itself using Eq. (2). However, in evaluating an ADC it is necessary to determine how stable these two parameters are with time. The remaining parameters are the $a_n$, which must be determined prior to analyzing any experimental data. One would expect, and experience has proven, that these coefficients vary slowly with time.

With the advent of new high resolution detectors one other ADC characteristic becomes important; sharpness of channel edges. Although there is no explicit parameter that can be used as a measure of this characteristic, channel profiles will exhibit it clearly for visual judgement. The concept of channel profiles will be discussed below.

Thus the parameters of interest are α, β, and the $a_n$. Along with these parameters it is necessary to obtain data relating to the channel edges.

## Parameter Determination

Two methods have been used for the determination of the parameters α, β, and the $a_n$. The first method utilizes many radioactive isotopes emitting gamma-ray energies that have been precisely measured with other techniques. The second method utilizes a precision pulser, and is the one this work will describe. The latter method has several advantages and the one major disadvantage that it does not include the effects of the detector. However, all evidence to date indicates that solid-state detectors are linear.

The measurement of spectrometer parameters by computer was made possible by the development of a precision mercury-relay pulser which is completely under program control. Control is exercised by the computer via a relay-operated resistance divider. Complete details on the pulser may be found in reference 2.

Measurement of all the parameters requires an accurate determination of pulser amplitude versus channel number. More specifically, of accurately determining what channel a pulser peak has fallen in. This cannot be done with more conventional methods of peak determination, because the pulser peak will normally only occupy one or two channels. The method we have arrived at is what might be described as the channel splitting method. Assume that a pulser is adjusted so that all events fall in channels N and N+1. Now if the events stored in channel N are equal to those in channel N+1, the centroid of the pulser peak is at N+0.5. By using this channel splitting method there is now available a straight forward way of placing a pulser peak in an accurately known channel position. The only remaining consideration is to determine how fast the accuracy of the channel position changes when the counts in channels N and N+1 are not exactly equal. It has been found from examining the channel profiles of good ADC's that the split between channels N and N+1 can be as bad as 70% and 30%, and still have the pulser peak remain within 0.05 channels of N+0.50. The programs presently in use require the channels to be split within 12.5%, i.e., if C(N) represents the events in channel N then

$$\left|\frac{C(N)-C(N+1)}{C(N)+C(N+1)}\right| < 0.125 \quad .$$

Two program subroutines have been written which split a specified pair of channel numbers under program control; they are called CRUDE and FINE. From a determination made at the beginning of the program, there is always a constant available to these subroutines that represents the conversion factor from channel number to pulser units. This constant is accurate to a few percent. Using this conversion factor CRUDE moves the pulser peak by integer channel units until events are being stored in the proper two channels. At this point the subroutine FINE is called to adjust the pulser until the events in the two channels are split within 12.5%. This is accomplished by a ping-pong technique. FINE first determines which direction the pulser must be moved, and then using the conversion factor, moves the pulser in the appropriate direction by 0.25 channels. Again the direction is determined. If it is the same, the pulser is moved by 0.25 channels once more. If not, the pulser is moved in the opposite direction by .125 channels. Thus the procedure is to hold the increment constant until a change of direction is required and then the increment is halved. These iterations continue until the channels have been split within 12.5%.

Both CRUDE and FINE have live display showing each iteration. Fig. 2 shows photographs of two iterations under the control of FINE. Only the 48 channels in the region of the pulser peak are stored and displayed. One of the channels of the pulser peak is displayed with a brighter intensity and is identified in the upper left-hand corner of the oscilloscope screen. When testing spectrometers with solid state detectors the detector will normally be tied into the system. This means that events arising from normal background radioactivities will be interspersed with pulser events. The present program can differentiate the pulser events from the natural events to find the correct region of channels to be displayed and stored.

One series of measurements will supply information on all of the desired parameters; the $a_n$, and the time variations of $\alpha$ and $\beta$. Before making the measurements themselves some conventions must be established. Fig. 3 illustrates the present choices. The straight line represents the response of a linear spectrometer system and the other curve is exemplar of an actual spectrometer. Two channel numbers are selected which along with their corresponding pulser amplitudes make up two pairs of coordinates which establish the straight line. These two channel numbers are normally referred to as the tie points, and in Fig. 3 are represented by $C_3$ and $C_4$. The y-axis intercept, $C_1$, is the parameter $\alpha$ and the slope of the straight line is the parameter $\beta$ within a multiplicative constant.

Let the function f(P) be the straight line in Fig. 3 and the function R(P) be the actual spectrometer response function. Then the spectrometer system deviates from linearity by $R(P_i)-f(P_i)$ channels at pulser amplitude $P_i$. Thus if a systematic set of measurements of these deviations are made all along the range of the ADC, data are obtained which represent the function R(P). This representation is found by using the method of least squares on a power series, i.e.,

$$R(P) = \sum_{n=0}^{N} a_n C^n \quad . \qquad (3)$$

Through application of the method of least squares to Eq. (3) the $a_n$ are determined. Referring back to Eq. (2) it can be seen that the summation there represents the nonlinearity of the spectrometer system. The only difference in Eq. (2) being that $\alpha$ has been inserted to account for the fact that the value of $C_1$ may have been different when the spectral data were taken from that at the time the nonlinearity was measured.

To illustrate how the parameter determinations described above are actually carried out refer to Fig. 4. This is print out from an actual run on a 1024-channel spectrometer system. Typically for a 1024-channel measurement the deviation from linearity would be measured for at least 20 different channel positions, but in this case the computer was instructed to make the measurement at only two channels to reduce the volume of output for this example. All the material entered by the user is underlined, all other was printed by the computer.

The first question asks which of the dual ADC's is to be used. The user has designated the F ADC. (The computer interface was originally designed for an ADC manufacturer who designated his two ADC's by F and M. However, at various times the computer has been used with four different manufacturers' ADC's.) The next request is for the number of channels in the ADC. This may be anything up to 4096 channels. Once the user has entered the number of channels, the pulser is set at 3/4 of its maximum value and a pulser peak is stored for two seconds and its channel position is printed out; channel 771 in the example. The user is then asked if he wants to continue with the set up or repeat the running of the pulser peak. This is to allow the gain of the amplifiers to be adjusted in order that the amplifier output overlaps the ADC input. That is, the pulser peak should be in a channel approximately 3/4 of the maximum channel number. In the example of Fig. 4 the user has adjusted the gain of the amplifier then struck an "R" on the teletype keyboard to indicate he wants to repeat. Again the pulser is set to 3/4 of its value and the location of the resulting pulser peak printed out. At this point the user has struck the "C" to indicate he wants to continue. The length of time the pulser is to be run per iteration is now requested and the user has responded with two seconds. The next request is for the tie points and the response was channels 63.5 and 831.5. The next request is for the maximum drift in the high tie point. This is the amount of drift in the system that will be tolerated between successive positionings of the pulser at the high tie point. The user responded with 0.025 channel. When this entry is made the program goes through the iterations shown in Fig. 5. Once the three tie point positions have been successfully run without excessive drift of the high tie point, the values of $\alpha$ and $\beta$ are calculated and the value of $\alpha$ is printed out. This is followed by asking the user if he wants to adjust and repeat the measurement or continue. In this case the user has adjusted the zero reference level control on the ADC and asked for a repeat measurement of $\alpha$. Once the second $\alpha$ measurement was completed the user struck a "C" to continue. This was followed by a request for the channel increment, and the response was to measure the deviation from linearity every 500 channels. This is followed by a request for the first channel at which measurements are to be begun: channel 500. Next the user is asked for the number of measurements, i.e., how many complete passes through the range of the ADC are to be made. Finally, the last request asks the user whether he wants the maximum or minimum output. If yes, the response is a Y; if not, an N.

At this point the measurements are begun and the user no longer need remain at the computer. Fig. 6 shows how the measurements are made. The first channel at which a measurement is to occur is set and then the tie points are run. This allows $\alpha$ and $\beta$ to be calculated. Then the current two channels at which the measurement is to be made are split and the high tie point run to assure that no drift has occurred. If no drift has occurred then the deviation from linearity is calculated and printed out along with $\alpha$ and $\beta$. It can be seen that although the values of $\alpha$ and $\beta$ may be varying with time this should have no effect on the linearity measurements because of the continual checking for drift. That is, the calculation of deviation from linearity is made only after no drift has occurred within the length of time of one linearity measurement. Once all specified measurements have been made, the program averages the linearity measurements and finds the root-mean-square deviation from this average.

Returning now to Fig. 4, it can be seen that all the parameter data are available. By scanning the columns labeled ZERO SHIFT ($\alpha$) and GAIN ($\beta$), one can determine the magnitude and the rate of drift of these quantities during the course of the run. At the bottom of the print out are the averages of the deviations from linearity (CORRECTION) from which the $a_n$ may be calculated, and the RMS deviations to show how much these quantities varied during the course of the run.

Fig. 7 shows a typical linearity plot of a 4096-channel ADC. Channels are plotted along the X axis and correction in channels is plotted along the Y axis. The correction is equal in magnitude, but opposite in sign, to the deviation from linearity. To illustrate the accuracy and the repeatability with which these measurements can be made the data of Fig. 7 were taken in the following way. First a complete run was made measuring the deviations from linearity at channels 100, 300, 500, etc. Then a second run was made measuring the deviation from linearity at channels 200, 400, 600, etc. It can be seen that one can expect repeatability within several hundredths of a channel on a 4096-channel spectrometer system.

## Channel Profiles

As indicated above if the channel edges of an ADC are not sharp the spectral resolution of a scintillation spectrometer will be degraded. It was also mentioned that the best way of judging the sharpness of channel edges is from channel profiles. Fig. 8 was prepared to illustrate the concepts of a channel profile. Fig. 8a represents a series of channel profiles that would result from an ideal ADC. In this instance if one were to slowly increase the output from a pulser starting at channel N and record the percentage of events that fall in adjoining channels, it would be found that 100% of the events fall in channel N until the boundary of channel N+1 is reached at which point 100% of the counts would fall in channel N+1. Fig. 8b shows the stylized response of a poor ADC. In this case 100% of the counts only fall in one channel when the pulser corresponds to the exact center of a channel. From the depiction of channel edges in Figures 8a and 8b it can be seen that the more nearly the slope of the channel edge approaches 90 degrees, the better the ADC.

A program has been written to measure channel profiles using the computer-controlled pulser. The user specifies the channel edge at which the measurement is to start, the spacing between pulser peaks, and the length of time each pulser peak is to be run. Fig. 9 is an example of channel profiles measured on a 4096-channel ADC. The pulser was run for one second at each position and the positions are approximately 1/10 of a channel apart, or approximately 1/40000 of the ADC range. Thus for the measurements to be meaningful the spectrometer system must be stable to a few parts per million. Since spectrometers are not inherently this stable, the measurements can only be made if the data are taken in a very short time, such as the data of Fig. 9 which were taken in 30 seconds. It is obvious that this data could not be taken manually.

It should be added that if the stability of scintillation spectrometer systems are ever significantly increased the methods discussed here would allow an entirely new way of measuring differential linearity of these systems. By running a series of channel profiles at intervals throughout the range of the ADC variance in channel widths could be measured directly from the profile plots.

## Control Panels

One of the major disadvantages of teletype conversational input to programs involving ADC's or any type of experimental equipment is that data must be repeatedly entered in the set-up phase of an experiment. Commonly a digital control panel has been used to overcome this type of difficulty. Most control panels are constructed in such a way that there is a one-for-one correspondence between a bit and some control function. This type approach requires large numbers of bits and is fairly difficult to write programs for because of the bit handling required.

It is our contention that a panel becomes much more versatile if all the control functions are encoded into binary numbers. This also makes the programming much easier because bit manipulation is eliminated and only binary numbers have to be dealt with. This also has considerable impact when applied to a short word length machine like the PDP-8. Using the older method the panel functions quickly exceed 12 bits and the programmer is forced to write routines for utilizing multiple word formats. By carrying the concept of encoding to its ultimate end, a 12-bit word would be sufficient to handle 4096 control panel actions. For example, there could be 64 knobs with 64 positions each. In short, the proposal is to make the control panel an extension of the function box concept.

To illustrate how well a panel can be used for setting the control functions of an experiment, a conceptual panel design is shown in Fig. 10. This is a panel that could be used for the control of functions involved in the program described above to measure deviations from linearity of spectrometer systems. It should be obvious how the panel works by comparing Figures 4 and 10.

## Conclusions

The interfacing of nuclear ADC's and a precision pulser to a computer has proven to be very successful. The evaluation and calibration of scintillation spectrometers can now be done with significantly more accuracy and considerably less labor. Furthermore, the speed with which the measurements can be made has made it possible to make some measurements that were impossible manually. It has also been concluded that the addition of a control panel either for data acquisition or calibration greatly facilitates the ease with which experiments can be performed.

## References


1. For a detailed derivation of this expression see "Precision Energy Measurements Using Lithium-Drifted Germanium Detectors", W. W. Black, USAEC Report IDO-17140, 43 (1965).

2. "A Precision Computer-Controlled Pulse Generator and Its Application", W. W. Black, Nucl. Instr. and Meth. 53, 249 (1967).

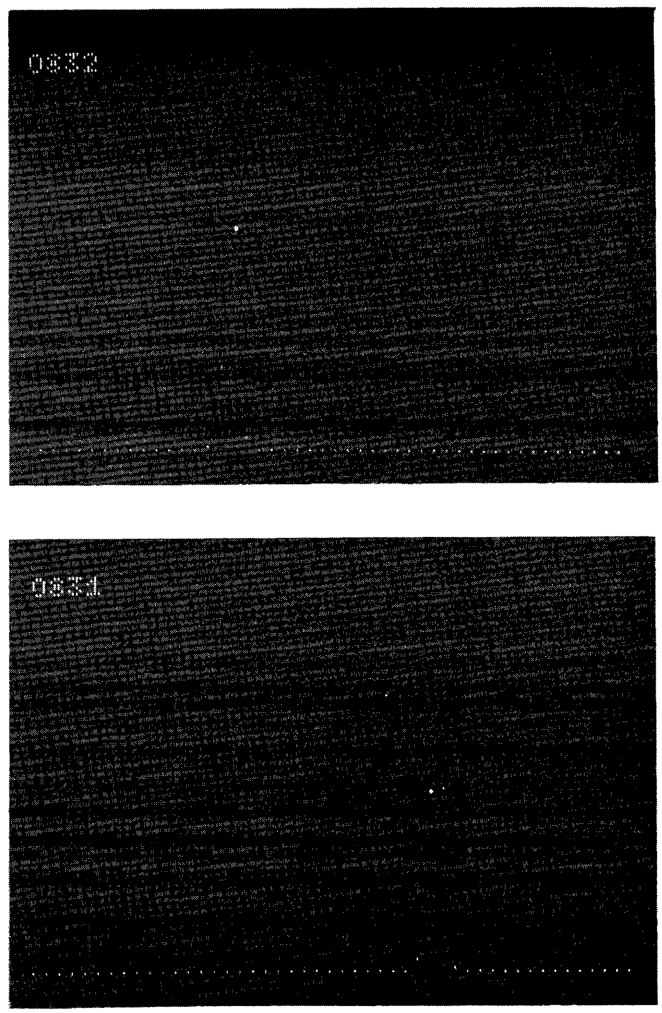



Figure 2.

Photographs of oscilloscope showing two pulser iterations in the course of splitting a pair of channels.

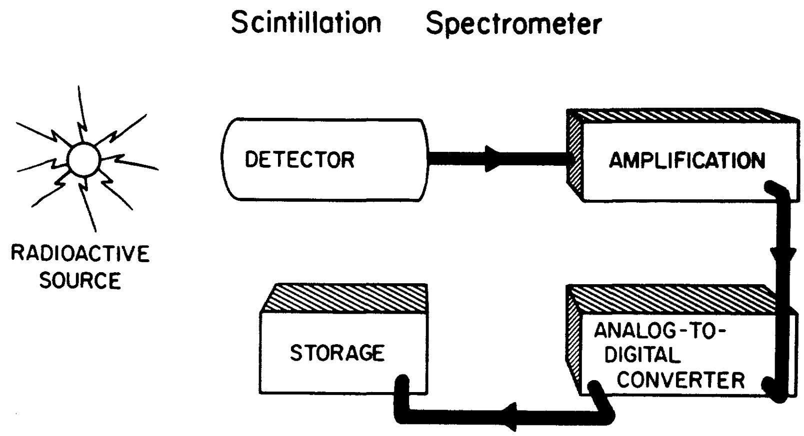



Figure 1. Experimental arrangement for a scintillation spectrometer.

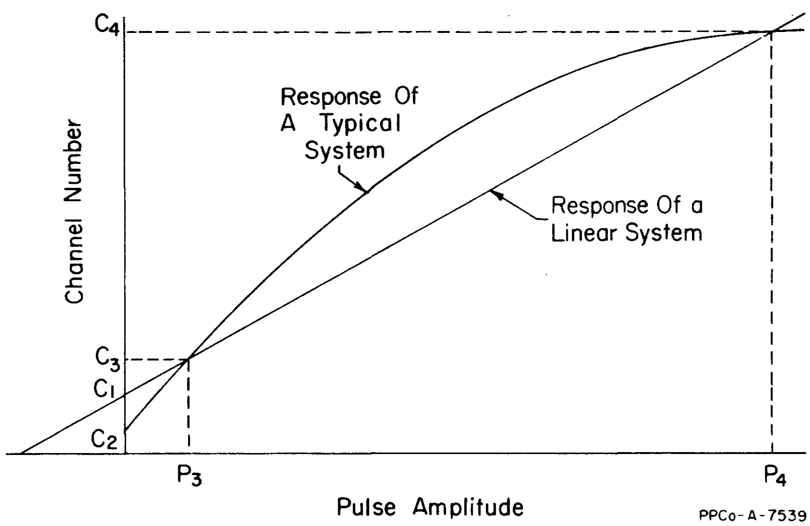



Figure 3. Comparison of the response of a typical spectrometer system to that of an ideally linear spectrometer system. Channels $C_3$ and $C_4$ are the lower and upper tie points, respectively. The channel intercept $C_1$ is equal to the magnitude of $\alpha$.

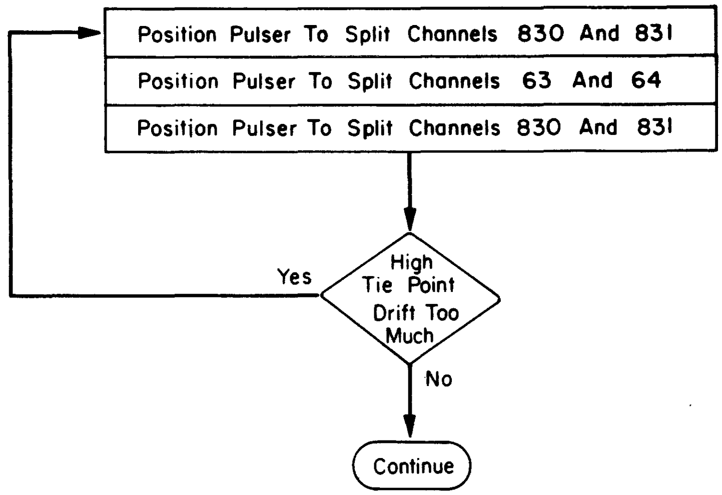
PRINT OUT FROM LINEARITY PROGRAM

```
F OR M ADC
F
HOW MANY CHANNELS IN ADC
1Ø24

Ø771
CONTINUE OR REPEAT
R
Ø759
CONTINUE OR REPEAT
C
ENTER RUNNING TIME
2

ENTER TI PTS, 1 THEN 2
63.5
831.5

MAXIMUM DRIFT IN HI TI POINT
.Ø25

+Ø.7689478E+ØØ
CONTINUE OR REPEAT
R
-Ø.449Ø7Ø7E-Ø1
CONTINUE OR REPEAT
C
ENTER CHANNEL INCREMENT
5ØØ

NUMBER OF MEASUREMENTS
3

MAXIMUM OUTPUT
Y
```

| CHANNEL | CORRECTION | ZERO SHIFT | GAIN |
|---|---|---|---|
| Ø5ØØ.5 | -Ø.1455749E+Ø1 | -Ø.449Ø7Ø7E-Ø1 | +Ø.1298398E+Ø3 |
| 1ØØØ.5 | +Ø.14Ø1367E+Ø1 | -Ø.1155315E-Ø1 | +Ø.1298346E+Ø3 |
| Ø5ØØ.5 | -Ø.1455749E+Ø1 | -Ø.449Ø7Ø7E-Ø1 | +Ø.1298398E+Ø3 |
| 1ØØØ.5 | +Ø.1394653E+Ø1 | -Ø.449Ø7Ø7E-Ø1 | +Ø.1298398E+Ø3 |
| Ø5ØØ.5 | -Ø.1469Ø55E+Ø1 | -Ø.1155315E-Ø1 | +Ø.1298346E+Ø3 |
| 1ØØØ.5 | +Ø.1298461E+Ø1 | -Ø.3472381E-Ø1 | +Ø.129819ØE+Ø3 |

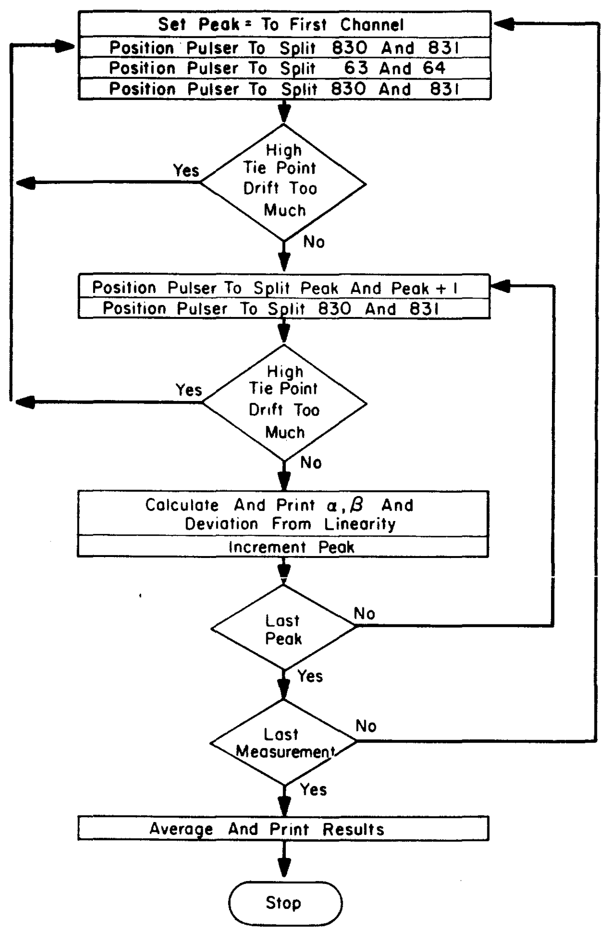
| CHANNEL | AVERAGE CORR. | RMS DEVIATION |
|---|---|---|
| Ø5ØØ.5 | -Ø.146Ø184E+Ø1 | +Ø.6227579E-Ø2 |
| 1ØØØ.5 | +Ø.1364827E+Ø1 | +Ø.47ØØ366E-Ø1 |

Figure 4. An illustrative print out from a measurement of the linearity deviations of a 1024-channel spectrometer system. The underlined text was entered by the user, and all other text was printed by the computer.




Pulser Iterations To Determine α And β

Figure 5. Flow chart of that portion of the linearity program showing the pulser iterations used to establish the pulser amplitudes corresponding to the two tie points. These pulser amplitudes plus the channel numbers representing the tie points make up two coordinate pairs that establish the straight line of Figure 3.




Pulser Iterations For Measuring Linearity

Figure 6. Flow chart of that portion of the linearity program showing the pulser iterations used to measure deviations from linearity for a scintillation spectrometer system. This set of deviations is then used to determine the coefficients $a_n$.

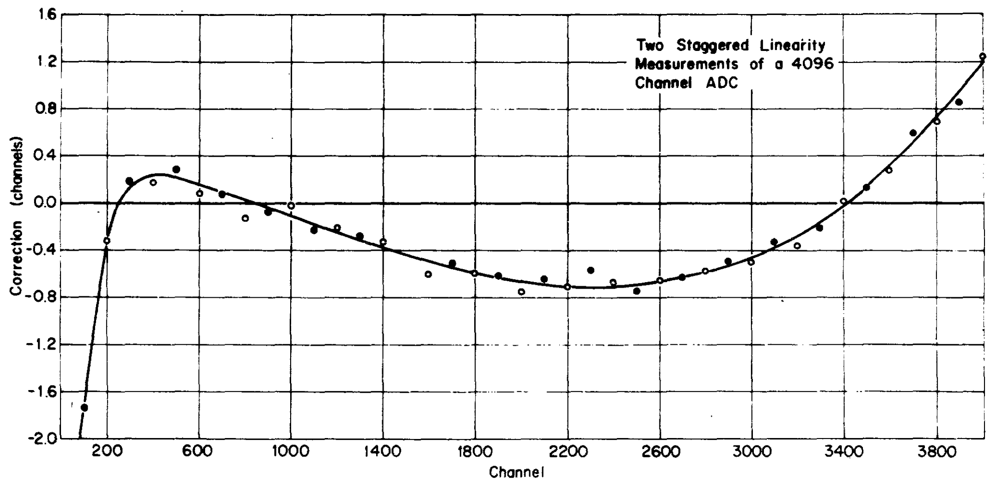
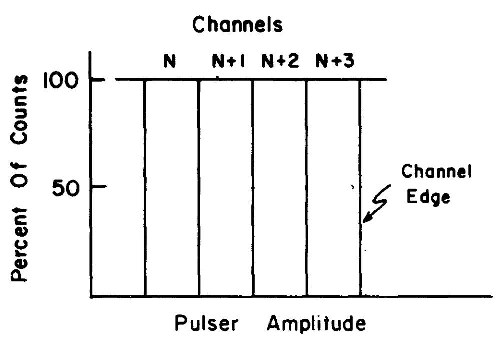
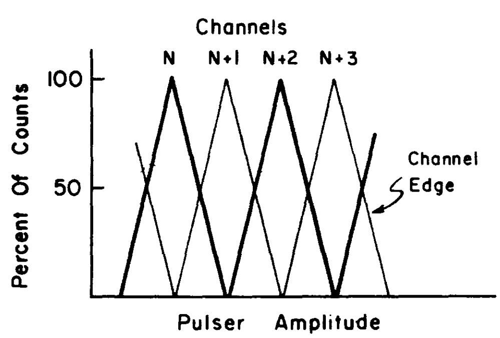



Figure 7. Actual plot of the deviations from linearity of a 4096-channel ADC. The plot represents two measurements: one which measured the deviations at channels 100, 300, 500, etc.; and one which measured the deviations at channels 200, 400, 600, etc.

## Example Channel Profiles For An Ideal ADC And A Very Poor ADC




(a) Ideal ADC Channel Profile




(b) Very poor ADC Channel Profile

Figure 8. Comparison of the channel profiles of an ideal ADC and a very poor ADC to show the significance of the channel edges.

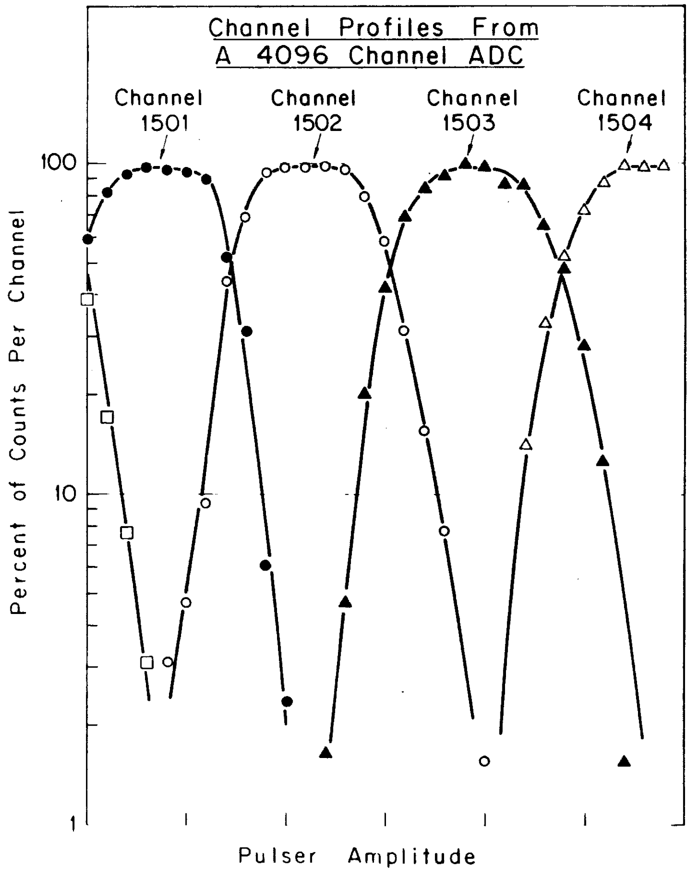
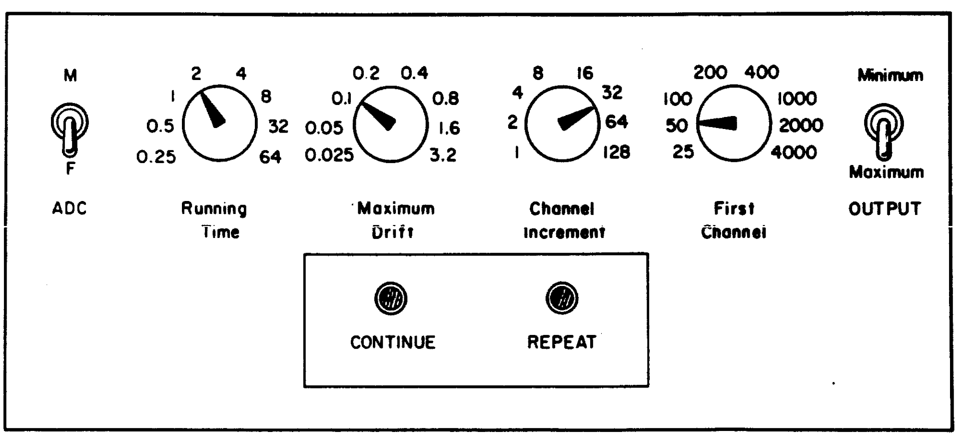



Figure 9. Actual channel profiles in the region of channel 1500 as measured on a 4096-channel ADC.




Conceptual Design For A Panel
To Control Linearity Experiments

Figure 10. The conceptual design of a digital control panel that could be used to control the computer and ADC's for the measurement of deviations from linearity of a scintillation spectrometer system.

# TAPE RECORDER I/O OPERATION IN A PDP-8 CONTROLLED BETA-RAY SPECTROMETER SYSTEM

J.J.H. Park & J. Ohkuma*

Division of Applied Physics, National Research Council, Ottawa, Canada

Abstract

A simple high speed I/O system, utilizing an inexpensive tape recorder, has been developed for use in conjunction with PDP-8 or PDP-8/S machines. The I/O speed of up to 500 (12 bit) words per second has been achieved on PDP-8 and about 250 on PDP-8/S. With the I/O routines modified, the Symbolic Tape Editor and PAL III assembler are being used extensively in this system.

## Introduction

A PDP-8 computer is on-line with a beta-ray spectrometer for data collection and control of the system.

The straightforward data handling facilities which include interfacing of three scalers and an external timer have already been completed and the programs debugged but the spectrometer system monitoring portion is not yet complete.

This paper will concentrate on a high speed tape recorder I/O device which was developed to improve the I/O facilities of the basic PDP-8 without too much cost. The outcome of it turned out to be quite versatile and a very simple addition to the system. It is frequently used while the spectrometer proper is down between the different experiments, when it is undergoing modifications or during construction of various new interfaces.

It has come down to such a simple level that the module cost of less than two hundred dollars and a home tape recorder will enable the addition to be made.

The early stage of this work was described at the Canadian Decus Symposium in April at which time the system was restricted to the use of a stereo tape recorder.

A brief review of the earlier paper[1] might be in order.

The basic philosophy of the

*Postdoctorate Fellow, Present Address: Osaka University, The Institute of Scientific and Industrial Research, Sakai-Shi, Osaka, Japan.

study was, and still is, not having to modify the tape recorder and make the system adaptable to any tape recorder. Thus all the modification was done outside the tape recorder; at the interface and onto the pulses rather than on the tape recording/playback circuitry.

One channel of stereo tape recorder was used as the data channel and the other as the timing or sync. channel. The standard twelve bit word was serialized by the program and unloaded from the ACO. The sync. pulses from the clock controlled the speed of unloading as well as being recorded on timing channel for use at the loading time. At the loading time the pulses were reconstructed into twelve bit words, again by the program under the control of timing pulse rate.

The new system uses a single channel tape recorder. The advantage of the new system is manifold; wider scope of application, increased reliability, less cost, overall simplicity, etc..

## General Outline

The major difference of the present system is based on recording the timing pulses on the same channel as the data and let the program sort out the two. In addition, frequency of timing pulses may be reduced to a fraction of that of data provided that the variation in the tape speed is kept below a few percent.

The standard twelve bit word is serialized by the program and recorded on the tape following one bit of timing pulse with a fixed amount of delay between each pulse. A word recorded on tape will have a series of up to thirteen pulses, the leading one being the reference timing pulse.

On play back, as soon as the leading pulse is detected the program searches for the subsequent pulses after each predetermined lapse time. The words are thus reconstructed serially (Fig. 1) and deposited in proper location. Both the half delay and the full delay can be adjusted to give best results. They are normally set such that IOT-1 pulses will fall midway between the succeeding data pulses.

## Hardware

The schematic diagram of the interface is shown in Fig. 2. The recording portion is simply an IOT-4 pulse lengthened by the first half of R302 to suit the frequency response of the tape recorder. The width of the pulse is typically 75 microseconds. Before going into the tape recorder these pulses have to be converted to bipolar ones. As the simplicity was the main theme of this system, an elementary shorted delay line as shown in Fig. 3 was tried and found adequate for the purpose.

When the tape recorder is played back the pulses from the tape trigger the one shot multivibrator, W501, the output of which sets a flip-flop (Fig. 2). An IOT-1 pulse will either skip or not skip the next instruction depending on the state of the flip-flop indicating whether there was a pulse at the corresponding bit.

An additional feature is incorporated so that the tape transport can be set on/off by the program. An IOT-1 pulse lengthened by the remaining half of R302 actuates the relay during the time the program calls for the device and stops when the program executes other functions or ends.

## Programs

Record on Tape - The flow chart of the recording routine is shown in Fig. 4.

The program identification, initial address and final address are deposited via S.R. and the program waits until the tape recorder reaches the proper speed which may be a few tenths of a second. The program identification, a duplicate initial address and a duplicate final address are recorded at the beginning and contents of registers from IA to FA follows with the checksum at the end. Actual program is listed in Fig. 5. It should be noted that both record and read routines are compressed into the last memory page and thus could be made permanently resident.

Read from Tape - The reading portion of the program is illustrated in Fig. 6. The program searches for the first word and compares it with the given identification. If it is correct, then it reads and compares the following two IA's and FA's. Any failure during these tests will send it back to the beginning until all are correctly read. Following the tests the actual loading takes place until final address is reached and the computer halts with the checksum error displayed in AC. This portion of the program is in Fig. 7.

Symbolic Tape Editor Modifications - In addition to the read/record programs the more frequently used existing programs have been modified for this system. Fig. 8 lists the modification on the symbolic tape editor. The teletype I/O routines are branched to the location 1400 where the subroutine "PEDITOR" channels the output to either the teletype or the tape recorder depending on the state of bit 11 of S.R. Another subroutine "REDITOR" will scan between the tape recorder and the keyboard until either of the flag is switched on. Thus it is possible to dump out the editor buffer onto the tape and use it for reloading into the editor or feed it into PAL III to be assembled. In fact, because of this easy I/O operation it was possible to draft the text of this manuscript on the editor, although the editor itself in the present form is not very flexible.

PAL III Modifications - The PAL III modification for this purpose is shown in Fig. 9. It utilizes the portion of the existing high speed reader routine plus some more of the memory space at the end of the PAL III program proper. Because of this extra requirement of the memory, the symbol table has been shifted in this particular example. Although this is not the best solution it will have to remain in this form temporarily until a better format of record/read is developed.

## Results

Three different models of tape recorder have been used in this arrangement. Grundig TK 45, Tanberg model 821 and a battery operated portable Uher 4000 Report L. The tape speed was normally set at 7½ i.p.s. However, 3 3/4 i.p.s. did the job almost as well. In fact, 3 3/4 i.p.s. was the highest on the Tanberg tape recorder. In most cases it is possible to adjust the volume and tone controls such that the same settings could be used for both record and playback. Once the proper settings are found, all the analog control knobs can be left untouched or place a protective cover over the knobs.

The pulse repetition rate is normally 3.2 kc/s corresponding to approximately 250 computer words or 500 paper tape characters per second. Up to twice of this speed is possible with the change of two constant, HDELAY to 20 and MDELAY to 7770. Even higher speed may be possible with narrower pulse width. However, the reliability of the system may depend more heavily on the quality of the tape recorder at higher speeds.

Various kinds of tapes were also tried and all were found to be usable, but acetate base tapes became less reliable after a number of runs due to the stretching of the tapes. The Mylar based ones showed no such effect.

Finally, the system was successfully implemented on a PDP-8/S with a minor modification to the program. For the 8/S version the delay loop in the program has been removed and the delay time is taken up by the execution of instructions in the routine, as shown in Fig. 10. The NOP codes in locations 7755 through 7761 correspond to the half delay of the PDP-8 version and can be adjusted by replacing with other non-operative codes of different execution times, e.g. TAD SPARE1 for 36 μseconds, DCA SPARE1 for 46 μseconds, etc..

Summary

It has been shown that an inexpensive home tape recorder can be utilized to act as a highspeed I/O device, with absolutely no modification to the tape recorder and a minimum of interfacing modules.

Acknowledgement

The authors wish to thank Mr. D.J. Doyle and his staff of the Digital Equipment of Canada Limited at Carleton Place, for the use of their PDP-8/S to try out that version of the program.

Reference

1. J.J.H. Park "Utilization of a Tape Recorder as a High Speed I/O Device". DECUS Canadian Symposium Proceedings p.p. 43-54, April, 1967.

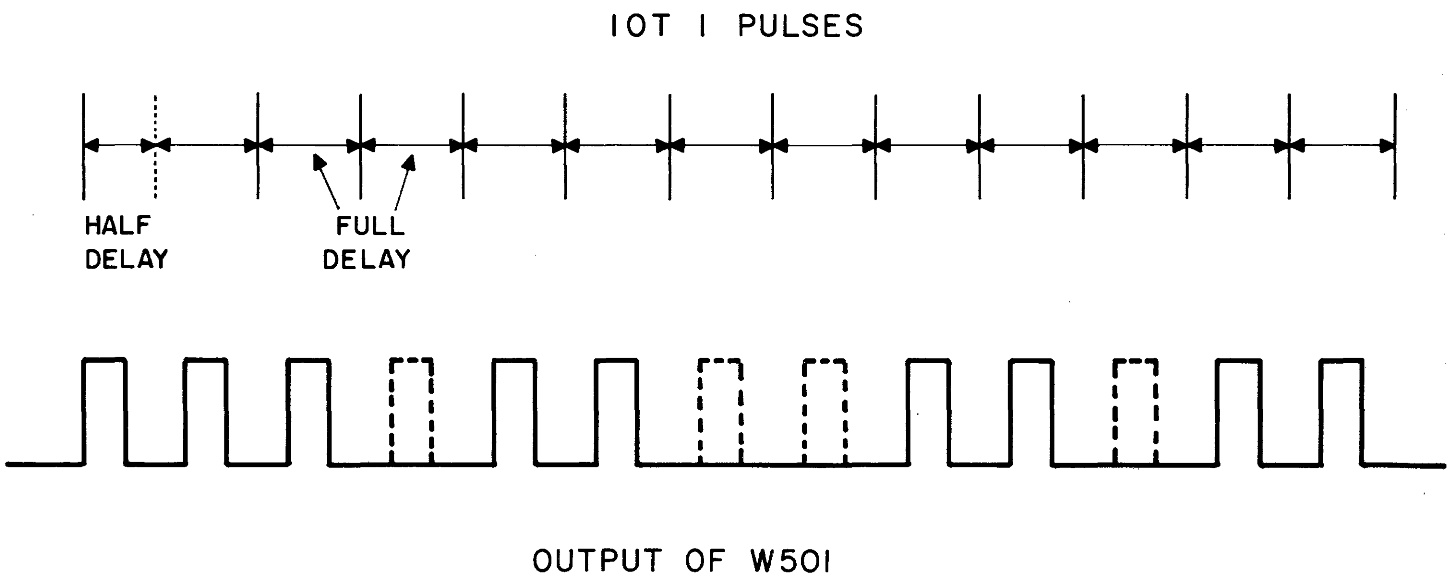



Fig. 1 Illustration of reading process. The first pulse is the timing pulse and the remaining twelve pulses constitute a word.

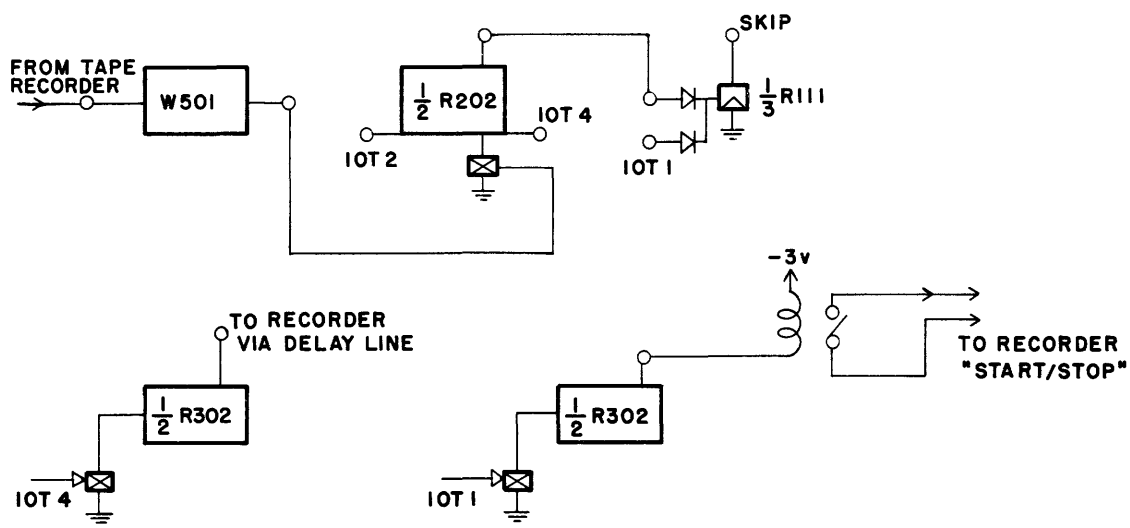



Fig. 2 Schematic diagram of the interface.

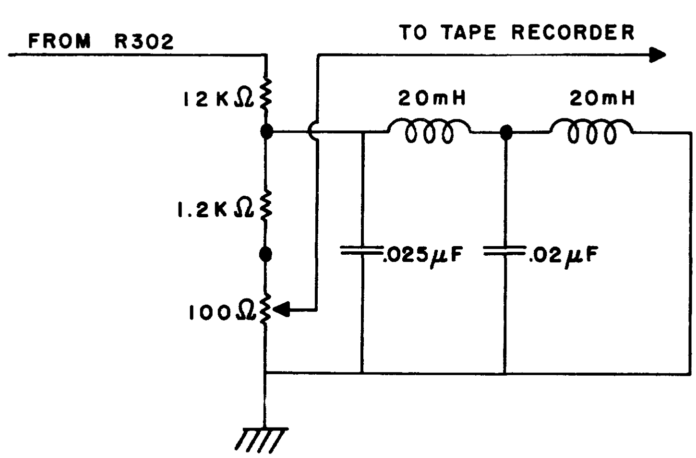



Fig. 3 Pulse shaping element.

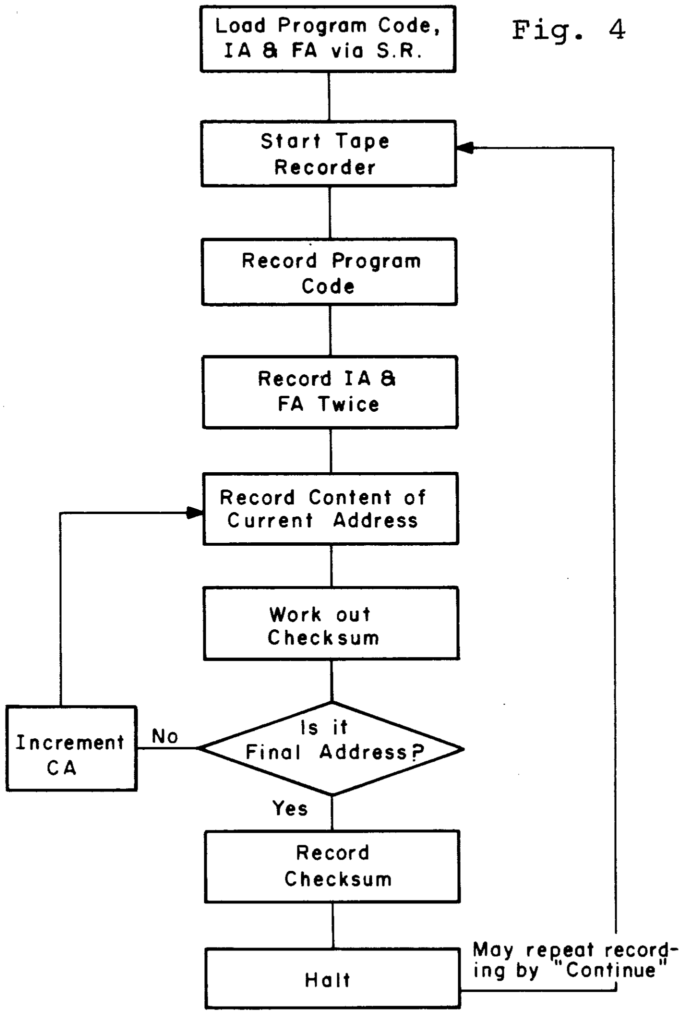



Fig. 4 Flow chart of "Record on Tape".

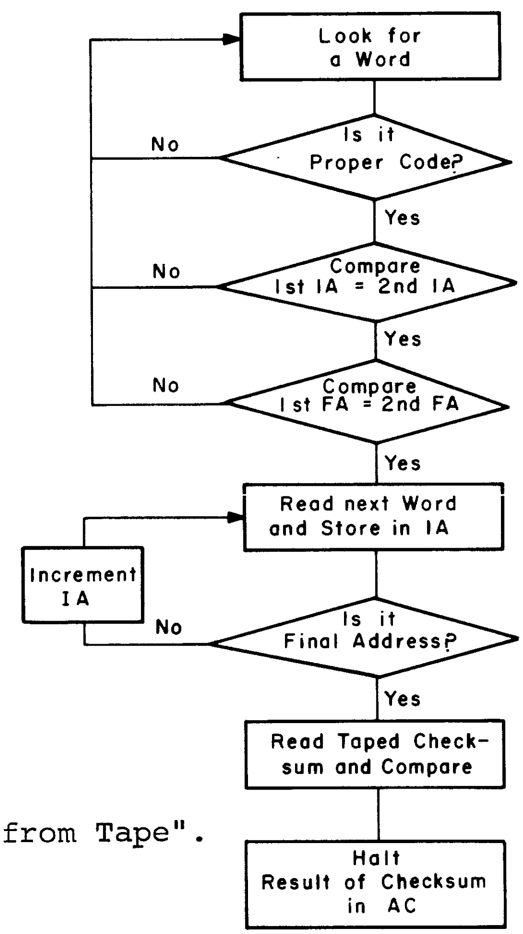



Fig. 6 Flow chart of "Read from Tape".

```
                *7600     /RECORD ON TAPE         /*131
7600  0000  CODER,    0
7601  0000  IA,       0
7602  0000  FA,       0
7603  4372  REPEAT,   JMS DL      /WAIT FOR NORMAL TAPE SPEED
7604  4372            JMS DL                  /
7605  6173            6173                    /
7606  2274            ISZ COUNTER             /
7607  5203            JMP REPEAT              /
7610  1200            TAD CODER   /RECORD IDENTIFICATION
7611  4247            JMS REC                 /DITTO
7612  3273            DCA CKSM    /CLEAR CKSM
7613  1201            TAD IA      /RECORD IA
7614  4247            JMS REC                 /DITTO
7615  1201            TAD IA      /RECORD IA
7616  4247            JMS REC                 /DITTO
7617  1202            TAD FA      /RECORD FA
7620  4247            JMS REC                 /DITTO
7621  1202            TAD FA      /RECORD FA
7622  4247            JMS REC                 /DITTO
7623  1201            TAD IA      /RESET CURRENT ADDRESS TO IA
7624  3272            DCA CA                  /DITTO
7625  1672  CONTINUE, TAD I CA    /RECORD CONTENT
7626  4247            JMS REC                 /DITTO
7627  1672            TAD I CA    /WORK OUT CHECKSUM
7630  1273            TAD CKSM                /DITTO
7631  7420            SNL                     /DITTO
7632  7101            CLL IAC                 /DITTO
7633  3273            DCA CKSM                /DITTO
7634  1272            TAD CA      /IS CA EQUAL TO FINAL ADDRESS?
7635  7041            CIA                     /DITTO
7636  1202            TAD FA                  /DITTO
7637  7650            SNA CLA                 /DITTO
7640  5243            JMP END     /YES, JUMP TO END
7641  2272            ISZ CA      /NO, INCREMENT CA
7642  5225            JMP CONTINUE/CONTINUE
7643  1273  END,      TAD CKSM    /RECORD CKSM
7644  4247            JMS REC                 /DITTO
7645  7402            HLT         /END OF RECORDING
7646  5203            JMP REPEAT /REPEAT RECORDING IF DESIRED
7647  0000  REC,      0           /SUBROUTINE RECORD
7650  6174            6174        /RECORD TIMING BIT
7651  3275            DCA STORE   /STORE THE WORD
7652  1271            TAD MDOZEN /RESET COUNTER
7653  3274            DCA COUNTER             /
7654  4372            JMS DL      /GIVE PROPER INTERVAL BETWEEN BITS
7655  1275  BIT,      TAD STORE   /BRING BACK REMAINING BITS OF WORD
7656  7500            SMA         /IS THERE A BIT?
7657  6171            6171        /NO, SKIP NEXT INSTRUCTION
7660  6174            6174        /YES, RECORD A BIT
7661  7104            CLL RAL     /ROTATE THE WORD FOR NEXT BIT
7662  3275            DCA STORE   /AND STORE
7663  4372            JMS DL      /GIVE INTERVAL BETWEEN BITS
7664  2274            ISZ COUNTER/INCREMENT COUNTER.  IS IT LAST BIT?
7665  5255            JMP BIT     /NO, RECORD MORE BITS
7666  5647            JMP I REC   /YES, RETURN TO MAIN PROGRAM
7667  0030  HDELAY,   30
7670  7700  MDELAY,   -100
7671  7764  MDOZEN,   7764
7672  0000  CA,       0
7673  0000  CKSM,     0
7674  0000  COUNTER,  0
7675  0000  STORE,    0
7676  0000  SPARE1,   0
7677  0000  SPARE2,   0
```

Fig. 5    Program "Record on Tape". This program and the one in Fig. 7 share some common registers.

```
                *7700       /READ IN FROM TAPE
7700    0000    CODE,       0
7701    4350    BEGIN,      JMS READ    /FIND A WORD
7702    7041                CIA         /CHECK IF IT IS RIGHT PROGRAM
7703    1300                TAD CODE                /DITTO
7704    7640                SZA CLA                 /DITTO
7705    5301                JMP BEGIN   /WRONG PROGRAM. TRY AGAIN
7706    4350                JMS READ    /READ AND CHECK IF TWO IA'S ARE SAME
7707    3201                DCA IA                  /DITTO
7710    3273                DCA CKSM                /DITTO
7711    4350                JMS READ                /DITTO
7712    7041                CIA                     /DITTO
7713    1201                TAD IA                  /DITTO
7714    7640                SZA CLA                 /DITTO
7715    5301                JMP BEGIN   /WRONG IA.  TRY AGAIN
7716    4350                JMS READ    /READ AND CHECK IF TWO FA'S ARE SAME
7717    3202                DCA FA                  /DITTO
7720    4350                JMS READ                /DITTO
7721    7041                CIA                     /DITTO
7722    1202                TAD FA                  /DITTO
7723    7640                SZA CLA                 /DITTO
7724    5301                JMP BEGIN   /WRONG FA.  TRY AGAIN
7725    4350    CONT,       JMS READ    /READ AND DEPOSIT IN PROPER REGISTER
7726    3601                DCA I IA                /DITTO
7727    1601                TAD I IA                /DITTO
7730    1273                TAD CKSM    /WORK OUT CHECKSUM
7731    7420                SNL                     /DITTO
7732    7101                CLL IAC                 /DITTO
7733    3273                DCA CKSM                /DITTO
7734    1201                TAD IA      /CHECK IF IT IS END OF JOB
7735    7041                CIA                     /DITTO
7736    1202                TAD FA                  /DITTO
7737    7650                SNA CLA                 /DITTO
7740    5343                JMP ENDS    /YES
7741    2201                ISZ IA      /NO, CONTINUE
7742    5325                JMP CONT                /DITTO
7743    4350    ENDS,       JMS READ    /READ TAPED CHECKSUM
7744    7041                CIA         /AND COMPARE
7745    1273                TAD CKSM    /WITH COMPUTED ONE
7746    7402                HLT         /HALT WITH CHECKSUM ERROR IN AC
7747    5301                JMP BEGIN   /TRY AGAIN BY PRESSING "START"
7750    0000    READ,       0           /SUBROUTINE TO READ 12 BIT WORD
7751    6171                6171
7752    7751                JMP .-1
7753    1271                TAD MDOZEN
7754    3274                DCA COUNTER
7755    1267                TAD HDELAY
7756    4372                JMS DL
7757    6172                6172
7760    7104    BITS,       CLL RAL     /ROTATE AC TO LEFT AND PREPARE FOR NEXT
7761    3275                DCA STORE               /DITTO
7762    4372                JMS DL      /PROVIDE DELAY FOR TIME JITTER
7763    1275                TAD STORE   /BRING BACK CONTENT OF STORE
7764    6173                6173        /WAS THERE A BIT?
7765    7410                SKP         /NO
7766    7001                IAC         /YES, ADD ONE BIT
7767    2274                ISZ COUNTER/INCREMENT COUNTER.  IS IT LAST BIT?
7770    5360                JMP BITS    /NO, GET MORE BITS
7771    5750                JMP I READ /YES, RETURN TO CALLING PROGRAM
7772    0000    DL,         0           /SUBROUTINE DELAY LINE
7773    1270                TAD MDELAY /LOAD MINUS DELAY AND COUNT
7774    7101                CLL IAC                 /
7775    7440                SZA                     /
7776    5374                JMP.-2                  /DITTO
7777    5772                JMP I DL    /ENOUGH DELAY, BACK TO CALLING PROGRAM
```

Fig. 7    Program "Read from Tape".

```
          /SYMBOLIC TAPE EDITOR MODIFICATIONS

                *1034
1034  4636                JMS I 1036 /WAS 6031
1035  7410                SKP        /WAS JMP.-1
1036  1455                REDITOR    /WAS 6036
                *1151
1151  4753                JMS I 1153 /WAS 6046
1152  7410                SKP        /WAS 6041
1153  1400                PEDITOR    /WAS JMP .-1
                *1400
1400  0000      PEDITOR,  0          /"PUNCH" ROUTINE MODIFICATION
1401  3332                DCA STORE  /SAVE WORD TO BE PUNCHED
1402  7604                CLA OSR    /IS IT PAPER TAPE OUTPUT?
1403  7010                RAR                  /
1404  7630                SZL CLA              /
1405  5213                JMP TAPE   /NO, OUTPUT TO TAPE RECORDER
1406  1332                TAD STORE  /YES, FOLLOW USUAL ROUTINE
1407  6046                6046                 /
1410  6041                6041                 /
1411  5210                JMP .-1              /
1412  5600                JMP I PEDITOR/RETURN TO CALLING PROGRAM
1413  2336      TAPE,     ISZ SWITCH /IS IT BEGINNING OF RECORDING
1414  5225                JMP CONTINUE/NO, CONTINUE OUTPUTTING WITHOUT INTE
1415  1330      INT,      TAD EXTN   /YES, DELAY OUTPUTTING FOR START UP TI
1416  3333                DCA ST2    /THIS ROUTINE KEEPS RELAY HOLD
1417  4313                JMS DL               /
1420  6173                6173                 /
1421  2333                ISZ ST2              /
1422  5217                JMP .-3              /
1423  1327                TAD PBUF             /
1424  3335                DCA COUNT            /
1425  6174      CONTINUE, 6174       /START DUMPING
1426  1326                TAD MDOZEN /FOLLOWING ROUTINE IS STANDARD
1427  3331                DCA COUNTER/PROCEDURE FOR THIS SYSTEM
1430  4313                JMS DL               /
1431  1332                TAD STORE            /
1432  7106                CLL RTL              /
1433  7006                RTL                  /
1434  7500      BIT,      SMA                  /
1435  6171                6171                 /
1436  6174                6174                 /
1437  7104                CLL RAL              /
1440  3333                DCA ST2              /
1441  4313                JMS DL               /
1442  1333                TAD ST2              /
1443  2331                ISZ COUNTER          /
1444  5234                JMP BIT              /
1445  7200                CLA        /ALL BITS ARE RECORDED
1446  6172                6172       /RESET FLIP-FLOP
1447  2335                ISZ COUNT  /KEEP COUNT OF BUFFER CONTENT
1450  5253                JMP .+3              /
1451  7240                CLA CMA    /BUFFER FULL
1452  3336                DCA SWITCH /RESET SWITCH
1453  1332                TAD STORE  /THIS INSTRUCTION IS OPTIONAL
1454  5600                JMP I PEDITOR/RETURN TO CALLING PROGRAM
```

Fig. 8a   Symbolic Tape Editor Modifications.

```
1455  0000  REDITOR,  0           /"READ" ROUTINE MODIFICATION
1456  7240            CLA CMA     /SET SWITCH
1457  3336            DCA SWITCH  /TO ALLOW START UP TIME
1460  6172            6172        /STANDARD PROCEDURE
1461  6031  KEY,      6031                    /
1462  5306            JMP OUT                 /
1463  6036            6036                    /
1464  5655            JMP I REDITOR           /
1465  6171  BACK,     6171
1466  5261            JMP KEY
1467  1325            TAD MOCTAL
1470  3331            DCA COUNTER
1471  1323            TAD HDELAY
1472  4313            JMS DL
1473  6172            6172                    /
1474  7104  BITS,     CLL RAL                 /
1475  3332            DCA STORE               /
1476  4313            JMS DL                  /
1477  1332            TAD STORE
1500  6173            6173                    /
1501  7410            SKP                     /
1502  7001            IAC                     /
1503  2331            ISZ COUNTER             /
1504  5274            JMP BITS                /
1505  5255            JMP REDITOR
1506  1721  OUT,      TAD I READER/THIS ROUTINE ALLOWS TAPE-
1507  1322            TAD MADDRESS/RECORDER TO BE SWITCHED ON
1510  7640            SZA CLA     /ONLY WHEN INPUTTING REQUIRED
1511  5261            JMP KEY                 /
1512  5265            JMP BACK                /
1513  0000  DL,       0           /SUBROUTINE DELAY
1514  1324            TAD MDELAY
1515  7101            CLL IAC
1516  7440            SZA
1517  5315            JMP.-2
1520  5713            JMP I DL
1521  1033  READER,   1033
1522  6661  MADDRESS, -1117
1523  0030  HDELAY,   30
1524  7700  MDELAY,   -100
1525  7770  MOCTAL,   -10
1526  7766  MDOZEN,   -12
1527  7210  PBUF,     -570
1530  0000  EXTN,     0
1531  0000  COUNTER,  0
1532  0000  STORE,    0
1533  0000  ST2,      0
1534  0000  ST3,      0
1535  0000  COUNT,    0
1536  0000  SWITCH,   0
```

Fig. 8b Continuation of Figure 8a.

```
                *115        /PAL III MODIFICATIONS
0115    3047                3047        /WAS 2735   /SYSTA-1
                *200
0200    7604                CLA OSR     /DETERMINE INPUT MODE VIA SR
0201    7012                RTR                     /DITTO
0202    7630                SZL CLA                 /DITTO
0203    5243                JMP 243     /TAPE RECORDER INPUT
0204    5210                JMP 210     /PAPER TAPE INPUT
                *1123
1123    5334    OPCDP,      5334        /WAS 5446   /-OPCD
                *1571
1571    2444    MCRI,       2444        /WAS 2332   /OPCD
                *2130
2130    2377    SATAB,      2377        /WAS 2265   /OPTABL-1
                /THE ORIGINAL SUBROUTINE "HIREAD" WILL LOOK LIKE FOLLOWING
                /AFTER THE CHANGE.  FOR CLARITY ALL THE UNALTERED
                /INSTRUCTIONS HAVE BEEN INCLUDED IN COMMENT FORMAT.
                /*1426
                /HIREAD,    /0
                            /ISZ RCNT
                            /JMP GETCHR
                            /ISZ SWITCH
                            /SKP
                *1433
1433    7000    RESET,      NOP         /WAS 5302   /JMP ENDBF
1434    3703                DCA I IEXAM/WAS 3156    /DCA SWITCH
                            /TAD RBGN
                            /DCA AUTOA
                            /TAD RKON
                            /DCA RCNT
                *1441
1441    4702    READIN,     JMS I READ /WAS 6014    /RFC
1442    5251                JMP 1451    /WAS 3113   /DCA TEM3
1443    7240    CRLF,       CLA CMA     /WAS 2113   /ISZ TEM3
1444    1010                TAD AUTOA   /WAS 7410   /SKP
1445    3010                DCA AUTOA   /WAS 5241   /JMP FULL1
1446    1070                TAD C215    /WAS 6011   /RSF
1447    3410                DCA I AUTOA/WAS 5243    /JMP .-4
1450    5267                JMP FULL    /WAS 6012   /RRB
                            /SNA
                            /JMP READIN
                            /TAD C200
                            /DCA I AUTOA
                            /ISZ RCNT
                            /JMP READIN
                *1462
1462    5243                JMP CRLF    /WAS 5267   /JMP FULL
                /FULL1,     /CLA CMA
                            /DCA SWITCH
                            /TAD RCNT
                            /CMA IAC
                /FULL,      /TAD RKON
                            /SNA
                *1471
1471    5233                JMP RESET   /WAS 5302   /JMP ENDBF
                            /DCA RCNT
                            /TAD RBGN
                            /DCA AUTOB
                /GETCHR,    /TAD I AUTOB
                            /DCA CHAR
                            /JMS I CHEKI
                            /JMP HIREAD+1
                            /JMP I HIREAD
                *1502
1502    2266    READ,       RPAL        /WAS 7240   /ENDBF,CLA CMA
1503    2323    IEXAM,      EXAM        /WAS 3157   /DCA RCNT
```

Fig. 9a   PAL III Modification.

```
                AUTOA=10
                C215=70
                RBGN=125
                RKON=126
                RCNT=157
                FULL1=1463
                FULL=1467

                /EXTENSION OF READING SUBROUTINE
                *2266
2266    0000    RPAL,       0
2267    6172    PLAY,       6172
2270    1342                TAD MOCTAL
2271    3344                DCA COUNTER
2272    1343                TAD WAIT
2273    3347                DCA CT3
2274    6171    BACK,       6171
2275    5323                JMP EXAM
2276    1336                TAD ISZC
2277    3323                DCA EXAM
2300    1340                TAD HDELAY
2301    4330                JMS DL
2302    6172                6172
2303    7104    BITS,       CLL RAL
2304    3345                DCA STORE
2305    4330                JMS DL
2306    1345                TAD STORE
2307    6173                6173
2310    7410                SKP
2311    7001                IAC
2312    2344                ISZ COUNTER
2313    5303                JMP BITS
2314    5666                JMP I RPAL
2315    6031    KEY,        6031
2316    5274                JMP BACK
2317    6036                6036
2320    3410                DCA I AUTOA
2321    2157                ISZ RCNT
2322    5737                JMP I ENDER
2323    2346    EXAM,       ISZ CT2
2324    5315                JMP KEY
2325    2347                ISZ CT3
2326    5315                JMP KEY
2327    5737                JMP I ENDER
2330    0000    DL,         0
2331    1341                TAD MDELAY
2332    7101                CLL IAC
2333    7440                SZA
2334    5332                JMP .-2
2335    5730                JMP I DL
2336    2346    ISZC,       ISZ CT2
2337    1463    ENDER,      FULL1
2340    0030    HDELAY,     30
2341    7700    MDELAY,     -100
2342    7770    MOCTAL,     -8
2343    7774    WAIT,       -4
2344    0000    COUNTER,    0
2345    0000    STORE,      0
2346    0000    CT2,        0
2347    0000    CT3,        0
```

Fig. 9b   Continuation of Figure 9a.

```
                *7600       /RECORD ON TAPE  /FOR PDP-8/S
7600   0000   CODER,      0
7601   0000   IA,         0
7602   0000   FA,         0
7603   3274   REPEAT,     DCA DUMP    /WAIT FOR NORMAL TAPE SPEED
7604   2274               ISZ DUMP              /DITTO
7605   6173               6173                  /DITTO
7606   2272               ISZ COUNTER           /DITTO
7607   5203               JMP REPEAT            /DITTO
7610   1200               TAD CODER   /RECORD IDENTIFICATION
7611   4247               JMS REC               /DITTO
7612   3271               DCA CKSM    /CLEAR CKSM
7613   1201               TAD IA      /RECORD IA
7614   4247               JMS REC               /DITTO
7615   1201               TAD IA      /RECORD IA
7616   4247               JMS REC               /DITTO
7617   1202               TAD FA      /RECORD FA
7620   4247               JMS REC               /DITTO
7621   1202               TAD FA      /RECORD FA
7622   4247               JMS REC               /DITTO
7623   1201               TAD IA      /RESET CURRENT ADDRESS TO IA
7624   3270               DCA CA                /DITTO
7625   1670   CONTINUE,   TAD I CA    /RECORD CONTENT
7626   4247               JMS REC               /DITTO
7627   1670               TAD I CA    /WORK OUT CHECKSUM
7630   1271               TAD CKSM              /DITTO
7631   7420               SNL                   /DITTO
7632   7101               CLL IAC               /DITTO
7633   3271               DCA CKSM              /DITTO
7634   1270               TAD CA      /IS CA EQUAL TO FINAL ADDRESS?
7635   7041               CIA                   /DITTO
7636   1202               TAD FA                /DITTO
7637   7650               SNA CLA               /DITTO
7640   5243               JMP END     /YES, JUMP TO END
7641   2270               ISZ CA      /NO, INCREMENT CA
7642   5225               JMP CONTINUE/CONTINUE
7643   1271   END,        TAD CKSM    /RECORD CKSM
7644   4247               JMS REC               /DITTO
7645   7402               HLT         /END OF RECORDING
7646   5203               JMP REPEAT /REPEAT RECORDING IF DESIRED
7647   0000   REC,        0           /SUBROUTINE RECORD
7650   6174               6174
7651   7000               NOP
7652   3273               DCA STORE
7653   1266               TAD MEXTRA
7654   3272               DCA COUNTER
7655   1273   BIT,        TAD STORE
7656   7500               SMA
7657   6171               6171
7660   6174               6174
7661   7104               CLL RAL
7662   3273               DCA STORE
7663   2272               ISZ COUNTER
7664   5255               JMP BIT
7665   5647               JMP I REC   /RETURN TO MAIN PROGRAM
7666   7760   MEXTRA,     -20
7667   7764   MDOZEN,     7764
7670   0000   CA,         0
7671   0000   CKSM,       0
7672   0000   COUNTER,    0
7673   0000   STORE,      0
7674   0000   DUMP,       0
7675   0001   ONE,        1
7676   0000   SPARE1,     0
7677   0000   SPARE2,     0
```

Fig. 10a  PDP-8/S Version of "Read/Record" Programs.

```
                *7700       /READ IN FROM TAPE
7700    0000    CODE,       0
7701    4350    BEGIN,      JMS READ    /FIND A WORD
7702    7041                CIA         /CHECK IF IT IS RIGHT PROGRAM
7703    1300                TAD CODE                /DITTO
7704    7640                SZA CLA                 /DITTO
7705    5301                JMP BEGIN   /WRONG PROGRAM. TRY AGAIN
7706    4350                JMS READ    /READ AND CHECK IF TWO IA'S ARE SAME
7707    3201                DCA IA                  /DITTO
7710    3271                DCA CKSM                /DITTO
7711    4350                JMS READ                /DITTO
7712    7041                CIA                     /DITTO
7713    1201                TAD IA                  /DITTO
7714    7640                SZA CLA                 /DITTO
7715    5301                JMP BEGIN   /WRONG IA, TRY AGAIN
7716    4350                JMS READ    /READ AND CHECK IF TWO FA'S ARE SAME
7717    3202                DCA FA                  /DITTO
7720    4350                JMS READ                /DITTO
7721    7041                CIA                     /DITTO
7722    1202                TAD FA                  /DITTO
7723    7640                SZA CLA                 /DITTO
7724    5301                JMP BEGIN   /WRONG FA.  TRY AGAIN
7725    4350    CONT,       JMS READ    /READ AND DEPOSIT IN PROPER REGISTER
7726    3601                DCA I IA                /DITTO
7727    1601                TAD I IA                /DITTO
7730    1271                TAD CKSM    /WORK OUT CHECKSUM
7731    7420                SNL                     /DITTO
7732    7101                CLL IAC                 /DITTO
7733    3271                DCA CKSM                /DITTO
7734    1201                TAD IA      /CHECK IF IT IS END OF JOB
7735    7041                CIA                     /DITTO
7736    1202                TAD FA                  /DITTO
7737    7650                SNA CLA                 /DITTO
7740    5343                JMP ENDS    /YES
7741    2201                ISZ IA      /NO, CONTINUE
7742    5325                JMP CONT                /DITTO
7743    4350    ENDS,       JMS READ
7744    7041                CIA
7745    1271                TAD CKSM
7746    7402                HLT
7747    5301                JMP BEGIN
7750    0000    READ,       0           /SUBROUTINE TO READ 12 BIT WORD
7751    6171                6171
7752    5351                JMP .-1
7753    1267                TAD MDOZEN
7754    3272                DCA COUNTER
7755    7000                NOP
7756    5362                JMP .+4
7757    7000                NOP
7760    7000                NOP
7761    7000                NOP
7762    6172                6172
7763    7104    BITS,       CLL RAL     /ROTATE AC TO LEFT AND PREPARE FOR NEXT
7764    3273                DCA STORE               /DITTO
7765    1273                TAD STORE   /BRING BACK CONTENT OF STORE
7766    6173                6173
7767    7410                SKP
7770    1275                TAD ONE
7771    2272                ISZ COUNTER
7772    5363                JMP BITS
7773    5750                JMP I READ
7774    0000    SPARE3,     0
7775    0000    SPARE4,     0
7776    0000    SPARE5,     0
7777    0000    SPARE6,     0
```

Fig. 10b  Continuation of Figure 10a.

# ADAPTATIONS OF PDP-8 FORTRAN FOR LABORATORY COMPUTING*

Russell B. Ham and Christopher B. Nelson
U.S. Public Health Service
N.E. Radiological Health Laboratory
Winchester, Massachusetts

Abstract

The PDP-8 FORTRAN system provides a higher-level language for short but moderately complex computations. In addition, the system programs are sufficiently modular that it is possible to alter specific routines to accomplish a desired objective much more easily than writing an entire assembly language program.

We have made several modifications to the FORTRAN system in order to assist our basic project of X-ray and gamma-ray spectroscopy. Most significant of these modifications are DECtape routines for the TCØ1 control, incremental DECtape reading and writing in order to access entire blocks, a mechanism for appending an argument list to a PAUSE-type subroutine call, and graphical output.

* This paper was not received for publication.

# AN INTEGRATED DISK-TAPE OPERATING SYSTEM FOR THE 338 BUFFERED DISPLAY COMPUTER*

Jerrold M. Grochow and Thomas P. Skinner
Project MAC, Massachusetts Institute of Technology
Cambridge, Massachusetts

Abstract

A user oriented operating system allowing both the convenience of temporary mass storage and the availability of permanent secondary storage is described. The entire operating system is resident on the disk and accessible through bootstrap routines stored in a single core page. The user's program may use all 8K of core memory (except for the bootstrap page) and yet have immediate access to the "invisible" operating system. Programs may utilize system primitives for I/O buffering and creation of binary and symbolic tape files. System primitives also handle such activities as updating file directories, saving core images, loading binary-image tape files, and resuming user programs.

## Introduction

In designing an operating system for a general purpose computer, the system programmers are faced with many decisions based on the size of the system, the load of the shop, and the general nature of the jobs to be run. Many system programmers would in fact argue that for certain answers to these questions no operating system at all is the best choice. For a basic DEC 338 (PDP-8 with buffered display) the programmer may indeed be better off if he can communicate directly with all facilities of the machine. As additions are made to the computer, however, especially I/O devices, the programmer will rapidly find himself wrapped up in handling data management details in which he has no interest. It is here that the advantages of an operating system become obvious.

The DEC-338 installation at Project MAC is certainly an atypical one, but the operating system is designed to be usable on a much smaller configuration. At Project MAC, the system is equipped with 8K of core memory, an extended arithmetic unit, a 32K DEC fixed head disc, a two drive DECtape, a 2 speed real-time clock, and other special purpose I/O devices such as a digital-analog converter for a sound system, a low-speed dataphone interface to the Project MAC Compatible Time-Sharing System's IBM 7094, and a high-speed full duplex direct-linked dataphone to the Project MAC GE-645 (see Figure 1).

The current disc-tape operating system (DOS/8) makes use of the following special features:

- extended arithmetic
- 8K of core memory
- the display unit
- DECtape
- DEC disc

With the elimination of only one system function (LIST, see below) the system will run on a PDP-8 without display. The basic file system will run without a disc, and with slight modification will run in only 4K of memory. The use of the extended arithmetic unit can also be eliminated by the modification of several of the programs.

When the DECtape unit was first installed on the then PDP-8, it was found that the company provided software was totally inadequate for the proper utilization of the system. At that time, the only functions that were provided were the saving and restoring of core images. The system was not easily expanded and not particularly well documented.[1] It was also found (after only a few minutes of operation) that it was extremely easy to destroy, or at least

*Work reported herein was supported by Project MAC, an M.I.T. research program sponsored by the Advanced Research Projects Agency, Department of Defense, under Office of Naval Research Contract Number Nonr-4102(01).

put in an undefined form, the entire contents of any tape. It was for these and other reasons that work was started on the tape file system (FS) and the associated operating system (OS/8).

Since the programmers designing and implementing the system were very familiar with the CTSS[2] system it was decided to attempt implementation of a number of the ideas of its file system and to base the commands around a limited subset of CTSS commands.

In OS/8, most of the command programs are stored on tape and brought into core as requested, destroying the previous contents of the overlayed core areas. Restrictions were thus placed on the programmer to limit the areas of his program to allow for the basic file system and the SAVE command program at least. The DEC supplied tape software did not place this restriction on the programmer as all overlayed areas were first written on the tape. This system took additional time but, more importantly, opened the possibility of destroying not only the contents of the core image but other sections of the tape as well. If proper initialization and finalization procedures were not followed, the swapping and bootstrapping operations left the tape in an unrecoverable form. It was for this reason that we made the basic decision that OS/8 was not to write on a user's tape unless requested to. It was now possible for the user to insure that at least his tape would be preserved under user error.

With the addition of the disc and the implementation of DOS/8, all restrictions on the user were removed. The system is resident in its entirety on the disc and a complete core image is saved (at the option of the user) on the disc when the system is called in. Communication is via the last page of core field zero, as in the DEC system and OS/8.

The remainder of this paper will discuss in detail the file system, the operating system, and several aspects of the modification of such programs as MACRO, the symbolic editor, and FORTRAN for use in the system. A complete listing of the current system programs available, along with a short description of their functions will be found in the appendix.

## Overview of the File System

The file system consists of a package of subroutines that allow the user to reference existing tape files and to create new ones. Data transfers can be made to either core memory field. All communication with the file system is on a per file name basis. The actual file management on the tape is performed by the file system.

A user or system program can call the file system from either core by indirect references to page zero in core zero. A full interrupt handler is included in the file system that allows the user to multiprogram if he so desires. The time to search a tape may thus be utilized to do other tasks.

### The Subroutines

There are four basic subroutines to the file system:

1. OPEN
2. CLOSE
3. READ
4. WRITE

OPEN is used to instruct the file system that a particular file on a particular tape is to be used by one of the other subroutines. This enables the file system to keep in core only information about the particular files the user is working with. Information is also given to open about the type of operations that are going to be performed on the file. If a file is opened for reading and it does not exist on tape, OPEN will signify this to the user. The companion routine, CLOSE, is used to instruct the file system that we are no longer interested in a particular file and that it should close out any references it has to that file. In most cases this involves some tape movement.

READ and WRITE are used to transfer information from or to the magnetic tape from core memory. This information can be anything the user desires. The file system merely dumps the contents of the prescribed core locations onto the tape. Both READ and WRITE can be called more than once. There is no limitation on the sequence of calls to READ and WRITE. As long as the particular file is opened correctly, READ and WRITE will work regardless of what files may also be opened.

DELETE is the remaining subroutine and its operation is quite obvious. There is, however, one major difference between DELETE and READ or WRITE. A call to open is not required for DELETE and, in fact, is not correct and will cause an error.

The MFD - The MFD or Master File Directory, is a block or blocks on each individual tape that gives information as to what and where the files for that tape are located. The following information is kept for each file entry in the MFD:

1. File Name (three words)
2. Type and Mode
3. Starting Block
4. Number of Blocks
5. Number or Words Written in the Last Block

The MFD length is now set to a maximum of three blocks on the tape. The MFD is read by the file system each time a file is opened or closed. The MFD is re-written each time a file is created or deleted.

The AFD - The AFD, or Active File Directory, is a directory kept in core as part of the file system. Any file that is correctly opened will appear in the AFD. The information kept here is more elaborate than that kept in the MFD. Each entry contains the following information:

1. File Name (three words)
2. Type and Mode (Mode includes mode of opening)
3. Starting Block
4. Number of Blocks
5. Number of Words in Last Block
6. Block to be Processed Next
7. Tape Unit Number
8. FBL Pointer

All the other subroutines with the exception of CLOSE refer to the AFD for all information about a file. The user may also refer to the AFD for information if he so desires. The exact structure of the AFD will hold four different entries with several locations available for the user if he so desires. The user can store information about opened files, and the file system will ignore these locations.

The FBL - The FBL or Free Block List is a single block on each tape used with the file system. This block contains a BIT Table representing the total number of blocks on the tape. The operation of this is quite simple. Each word in the block represents 12 consecutive blocks on the tape. If the BIT is a one, this indicates that this block is free to be written on. If the BIT is a zero, the block contains information used by one of the files on the tape. Some quick arithmetic will reveal that indeed all the $2671_8$ blocks can be represented by the $128_{10}$ twelve BIT words of the FBL.

The file system reads this FBL into core whenever a file is opened to be written on. The FBL is then available for rapid reference by the file system while writing a tape. When the file is closed, the FBL is put back on the tape with the correct bits set to zero. At present there is room for two FBL's to be kept in core. Since there are currently only two tape drives, this is the maximum number of tapes that could be concurrently written on. The opening of a file for reading does not cause a free block list to be read in.

File deletion naturally causes the Free Blocks to be returned to the FBL and the FBL rewritten on the tape.

Structure of Files on the Tape - The structure of files placed on a file system created tape is a linked structure. The 129th word of each block points to the next block in the file. The last block of the file points to 7777. This is not really a pointer but an end fence that is detected by the file system.

Files are initially written on blocks as determined by the free block allocating subroutine within the file system. This subroutine reads the FBL and obtains free blocks in sequential order. Assuming that all blocks are available, this subroutine will obtain every consecutive block (see Figure 2).

## The Operating System

The operating system consists of a series of programs designed to accomplish certain frequently desired functions relating to program composition, storage, and execution within the context of the supervisory control of the file system. All commands of DOS/8 are initiated via a line typed on the teletype when the supervisor is in core. The commands may be divided into several categories:

1. Tape editing: listing file directory, deleting files, renaming files, etc.

2. File creation: saving core images, creating symbolic and binary files.

3. Program execution: restoring core image, loading binary files.

4. Processors: editor, MACRO, FORTRAN, etc.

In OS/8, only the commands associated with program execution were included in the operating system proper (that is, the core resident part). In DOS/8, with the expanded core space available to the operating system and with the possibility of storing other programs on parts of the disc, the attempt has been made to include the most frequently used commands from all categories in fast access storage. Other command programs are still kept on tape files and called in as needed.

DOS/8 is entirely open ended in that any user program may classify as a command program: the user can cause it to be disc resident, he can access the command line for arguments, and he can communicate with the file system subroutines if he so desires.

In order to implement the command structure, DOS/8 has an internal dictionary of those commands that are resident within its own 8K image (SAVE, RESUME, LOAD, LOADGO, START - explanations of these and other commands are given in the appendix). If the current command line argument is not one of these, then the file directory of the current "system tape" (the user may specify which tape he wishes to be called the "system tape"; it may be any user tape), and searches it for a SAVE'd file with the corresponding name. If one is found then it is loaded onto a disc "core image" (the disc may be thought of as four 8K core images), swapped into core, and started in the appropriate location. All command programs are stored on tape in this format.

The user may cause programs to be loaded into any one of three core images (the fourth contains DOS/8) and be saved there for future access. For instance, a user sitting down to compose, assemble, and edit a program may immediately load MACRO and TECO (symbolic editor using the display) onto the disc. He can then refer to them directly by just issuing the "DR" command followed by the appropriate image number. This is especially convenient when several "edit-assemble" functions are anticipated.

Additional functions that the operating system provides are the ability to zero out any core image, change "system tapes" by simply typing a new tape unit number, and the chaining of commands (a series of commands to be executed one after the other without destroying the core image left behind).

A typical operating session using DOS/8 would be as follows: The second disc image is loaded from tape (using OS/8) and the bootstrap routines are put in the top page of core zero. The user manually starts DOS/8 by going to the bootstrap routine. Disc image 2 is now read into core. The user might now want to zero out image 1 (types a "Z") and load MACRO for assembly of a previously prepared symbolic file. "MACRO" is typed. This causes MACRO to be put in image 1. If the user now wants to save MACRO in one of the other disc images (for recalling later without reference to the tape file), he might type "DS 3" which will transfer the contents of image 1 to image 3 (via a core buffering process). MACRO may now be referenced by typing "DR 3" followed by a starting address in which case the contents of image 3 will be transferred to image 1 and resumed in the normal manner. After the file is assembled, he may run it by typing "LOAD filename" and "START starting-addr" (or "LOADGO filename starting-addr"). The file will be loaded into image 1, swapped into core and started. After editing (the editor might have been loaded onto the other disc image), another assembly can be started as above.

## The Input-Output System

Character information is stored in special tape files using the ASCII character set and a special packing scheme. Five seven-bit ASCII characters are packed in every three twelve-bit PDP-8 words. A user wishing to perform character I/O with the file system would have to have a buffering routine to read "blocks" from the tape as well as a character unpacking algorithm. It was felt that the system should provide a standard package to perform these operations and that the user should appear to be communicating with a teletype when using them. This group of routines is the present I/O system (IOS). A user is now able to merely call the IOS to obtain the next character in the currently opened file. If no file is opened, the IOS will request a file name. Error checking and handling is also performed by the IOS. Character output is handled in a similar manner with the IOS writing in the current file opened for writing (a file cannot be opened for both reading and writing at the same time). When the user has completed reading or writing a file he must make another call to the IOS to "close" the appropriate file and remove its name from the AFD.

The IOS provides a user with the ability to first write a program using the teletype for I/O and then, after debugging is complete, change the teletype I/O instructions to calls to the IOS. Existing programs can also be modified quite easily, as has been shown with the DEC-supplied MACRO assembler. Several other current system and user programs also use the IOS.

## Plans and Conclusions

Almost a year of use of OS/8 and approximately a month with DOS/8 have proved to be very fruitful at the Project MAC installation. The ease of access of DECtape that the file system allows has speeded the various tasks associated with program development by several orders of magnitude. With the addition of disc buffering schemes for input-output, these tasks will be speeded up even more. The "foolproof" system of tape management (it is almost impossible to put a tape in unsalvageable form using the operating system - except when experimental changes have been made!) has made the use of paper tape backup systems virtually unnecessary. And with the major programming effort nearly complete, more time can be devoted to application programs using the facilities of the systems.

The next step in system development is further integration of the display into operating system communication. Eventually we envision a system where it is only necessary to point to a program name displayed on the CRT to cause it to be loaded and executed or assembled and listed, depending on the type of file. Integration with the various other computers at Project MAC will also allow us the facilities they provide and the possibility of direct transmission of files over telephone circuits. We can then look to the efficient use of a system with small owned computers coupled, for only a few seconds at a time, to the more expensive but more powerful time sharing installation.

## Acknowledgements

Thanks are due to Professor Edward L. Glaser without whose constant support this work could not have been completed; and to Daniel Edwards for his suggestions during the initial planning and for the development of TECO, the one program that most aided the completion of this project.

## References

1. PDP-8 DECTAPE Programming Manual, Digital Equipment Corporation, Maynard, Mass., 8-27-U (1965)

2. Crisman, P. A., Editor, The Compatible Time-Sharing System Programmer's Guide, Second Edition, MIT Press, Cambridge, Mass. (1965)

## Appendix

Current system programs:

L - incorporates the file system functions of displaying (or listing) file directories and free block lists, deleting and renaming files.

LOAD, LOADGO, START - using some combination of these commands it is possible to load a binary-image file (produced by MACRO) and start its execution at a specified location.

SAVE - will save a core image in a special format file. Other information such as starting address may also be saved with the file for use by RESUME.

RESUME (R) - program to reload core image files created by SAVE and to take appropriate action as to execution as specified in the SAVE'd file.

COPY - will copy a file from one tape to another.

MACRO - the PDP-8 macro assembler adapted for tape I/O with the file system. Creates a binary image file that can be LOAD'ed.

TECO - a text editor using the display. Symbolic files may be edited character-by character according to context or line. Creates files suitable for input to MACRO or other translators using the I/O simulator system.

CTSS - a basic program used for connecting the PDP-8 to other computers or console units via the computer telephone lines. Performs the various timing actions and code conversions.

Under modification are DDT, FORTRAN and several other programs.

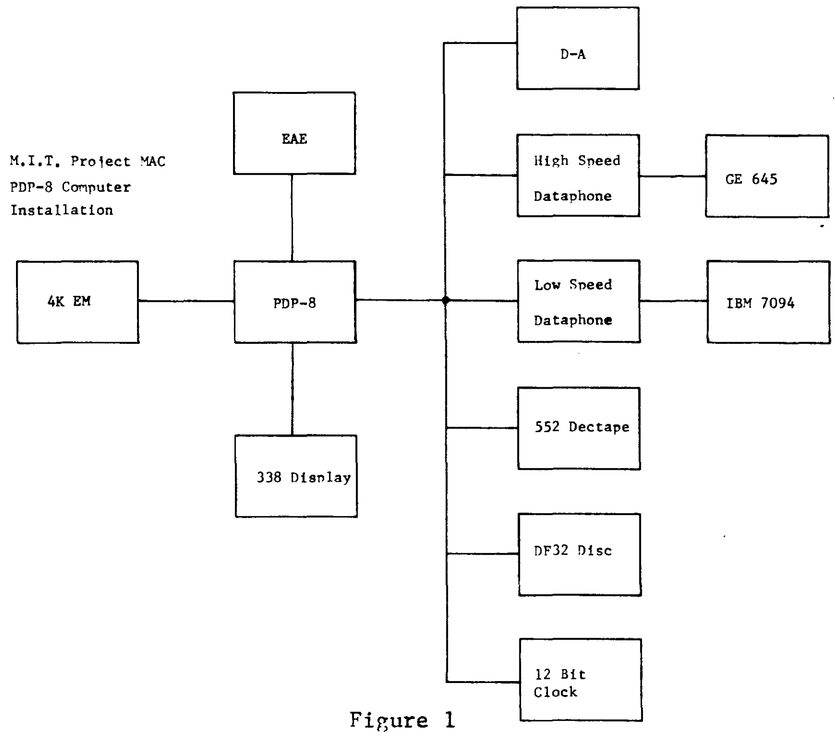
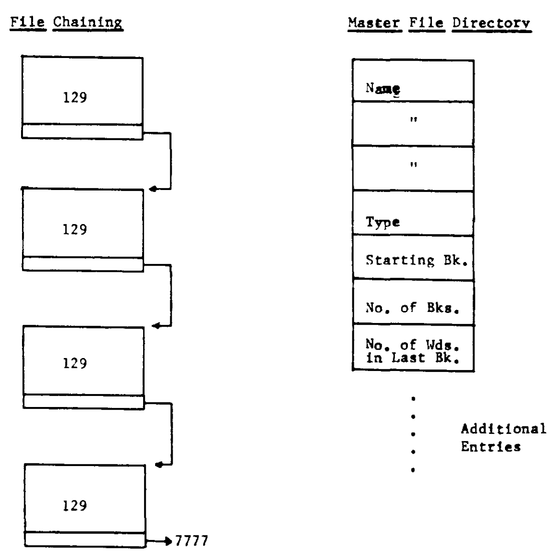



Figure 1




Project MAC PDP-8 operating system file structure

Figure 2

# COMPUTERS IN THE LABORATORY: EDUCATION

Ronald G. Ragsdale
The Ontario Institute for Studies in Education
Toronto, Ontario.

Abstract.

The PDP-9 facility of the Department of Computer Applications is designed to serve all eight departments of the Ontario Institute for Studies in Education. In addition to the research and development work of the eight departments, the PDP-9 will also be a part of the Regional Data Processing Center (RDPC), through a link to the RDPC's 360/40. The RDPC is a prototype educational data center which at some time may be linked to school districts through computers like the PDP-9.

This paper describes the PDP-9 and 360/40 configurations and the applications of the PDP-9 system.

The Ontario Institute for Studies in Education (OISE) was created by the Ontario Legislature in July, 1965. It brings together, for the first time, the three separate activities of research, development, and graduate studies. The OISE functions as a research institute devoted to the scientific study of matters and problems relating to education, as a center for curriculum development, and as a college which conducts graduate degree courses in education.

There are eight departments in the OISE: Adult Education, Applied Psychology, Computer Applications, Curriculum, Educational Administration, Educational Planning, History and Philosophy, and Measurement and Evaluation. These departments have their own specialized research programs and are relatively autonomous in their operation. The Office of the Coordinator of Research is concerned with Institute-wide aspects of these programs.

The Coordinator of Development is responsible for all development programs, whether these are conducted within the Office of Development or in cooperation with the Institute's various departments.

Specialized programs of graduate study are carried on in each of the Institute's departments, and are coordinated by the Office of Graduate Studies. Through an Agreement of Affiliation with the University of Toronto, the University's Graduate Department of Educational Theory is located primarily in the Institute, and students proceed to University of Toronto degrees of Master of Education, Master of Arts, and Doctor of Philosophy. There are now more than 90 members of the OISE faculty.

The Regional Data Processing Center (RDPC) is a pilot project being jointly carried out by the Ontario Department of Education and OISE. The purpose of this project is to determine the best way of making computer services available to educational organizations in Ontario. The Center is currently involved in a feasibility study to determine how such services can be planned, managed, and financed for the Province.

The RDPC currently offers data processing services to the Metropolitan Toronto area utilizing an IBM 360/40 which will soon have 250 million bytes of disc storage. Currently being implemented is a student accounting and information system which involves the records for 26,000 students. Data processing for the OISE is also being done by the 360/40 on a batch basis. The RDPC is further linked to the OISE by a telephone line from the 360/40 to the PDP-9 of the department of Computer Applications.

The department of Computer Applications includes five full time academic staff members plus one joint appointment. Its graduate program is just developing, with several students now proceeding to M.A. degrees. The department's primary research tool is the PDP-9 computer with two DECtapes, automatic priority interrupt, extended arithmetic element, analog/digital converter, extra teletype interface, type 34H CRT interface, and card reader.

There are many departmental projects in various stages of development, most of them involving the PDP-9. One of the most important projects has to do with the dissemination of information about the role of computers in education to the Province's teachers. Without the facilitating affect of these seminars and demonstrations, the seeds of the other projects may fall on barren soil.

Another information project has to do with the evaluation of systems for computer-assisted instruction. A report has been completed on one system and made available to a number of educational institutions. Additional reports will be completed when other systems are more fully defined.

Somewhat similar to the systems evaluation project is the comparison of languages for computer-assisted instruction. This will be carried out by accessing other installations by remote teletype or by implementing the languages on the PDP-9. One possibility is that TRAC might be implemented initially and other languages could then be imbedded in TRAC.

Two closely related projects involve computer control of experiments and on-line data analysis. These might overlap when the data from behavioral experiments is analyzed during the course of the experiment and fed back to alter the experimental procedure, with the analysis procedures being altered, on-line, by the experimenter.

Since the on-line analysis may require some larger processor than the PDP-9, the telecommunications project is required. The nature of telecommunications seems to be such that if one only wishes to become aware of what is available, it is a considerable project. The PDP-9 will telecommunicate with remote teletypes as well as other computers.

One of the uses of the link to the 360/40 of the RDPC will be in studying the general area of the use of satellite computers in education. An example would be the application of the Student Information System. This will be most relevant to counselling in a setting such as the ISVD system at Harvard or similar work at SDC.

The interfacing versatility of the PDP-9 will be exploited in the investigation of experimental teaching terminals. In a rather important reversal of the norm, multimedia course materials will be generated first and devices to present the material will then be attempted. The demonstration device should hold more promise for instruction than the demonstration course.

In a subject matter area that has received little attemtion thus far, non-verbal training under computer control will be investigated. This would include training in perceptual-motor skills, which has implications for industrial training as well as perceptually-based learning disabilities. Another non-verbal area involves training procedures for the use of prosthetic sensory devices. The use of these devices generally requires a lengthy training period.

By investigating the areas outlined above, the department of Computer Applications of the OISE hopes to serve as a source of ideas in the educational community concerning the role of computers in the educational process.

# A COMPUTER DATA LINK FOR HIGH-ENERGY PHYSICS EXPERIMENTS*

Sypko W. Andreae

Lawrence Radiation Laboratory
University of California
Berkeley, California

## Abstract

At LRL small computers are used on-line with high-energy physics experiments. When the experimenter desires to estimate the relative value of the experimental data, the necessary analysis can only be performed on a large computer. Hence the need arose for a fast two-way Data Link between one or more small computers and a large computer.

The lack of interrupt facilities in the available large computer and the necessity of using twisted-pair telephone lines both posed unusual design problems.

The main components of the Data Link are interfaces for the two computers, a buffering and error-correcting system, and a transceiver system using 1-mile-long telephone lines. A unique demand and response system both maintains synchronization and supports a highly accurate error-correction system.

Control words communicate word count, transfer direction, etc., between the computer programs. Special high-reliability pulse-train signals transmit critical information, such as "error" and "end of record" between the interfaces.

## Introduction

At the Lawrence Radiation Laboratory in Berkeley, small computers, mostly PDP5's and 8's, are used on-line to gather data from high-energy physics experiments. The data is typically stored on tape by the small computer and later analyzed off-line by a CDC 6600. The small computer can provide the experimenter with some very simple checks of the incoming data--for example, whether the experimental equipment is working at all-but does not have the size or power for performing sufficient analysis to indicate whether the experiment is producing the desired data in terms of physics. Thus the experimenter must wait for the hand-carrying of the data tape from the experimental area to the 6600, and for the normal batch processing of the job, before he can make any necessary corrections in his experiment. This feedback loop might occupy hours or even days.

From this situation there arose the need for a Data Link between the small computers and a powerful machine capable of complete data analysis (see Fig. 1). The relatively small throughput which it was anticipated the link would carry precluded the use of a medium-size time-sharing computer, or a small computer to interface the link with the 6600. Instead, the link was to provide a direct connection, via telephone lines up to several miles long, between the small experimental computer and the CDC 6600. The link was to incorporate the following design objectives:

1. Modifications to the 6600 were to be minimized. Since the link would constitute a very small part of 6600 throughput, interference or serious degradation of the normal batch processing could not be tolerated. Also, modifications to the 6600's Chippewa Operating System were to be kept as simple as possible.
2. A reasonably high data rate was required, preferably exceeding that of the high speed tape units already used by the 6600.
3. Error-correction facilities were to be kept as much as possible within the data link itself, to avoid complicated software checking by both the small computer and the 6600. Also, since the data was to be carried

*This work was done under the auspices of the U. S. Atomic Energy Commission.

by twisted-pair phone lines, the error-detection system needed to be quite powerful.

4. The link would need to be used only occasionally, no more often than every half hour or so, for the transmission of 20 to 30 000 words of data.

5. Rapid response by the 6600 to the link was unnecessary: a lapse of several minutes from the time the link requested service from the 6600 until the 6600 was able to respond was acceptable.

6. The link was to be kept as general-purpose as possible to enable it to be used with other types of computers if the need arose.

## Experimental Environment

At this point it may be helpful, in order to clarify the intended function of the link, to review some typical experimental techniques employed by the physicist and see how the data generated is used.

Let us compare two much-used approaches to the recording of data from high-energy nuclear events, the bubble-chamber technique and the counting technique.

The bubble-chamber approach uses primarily a photographic process to record an event. The photographs are later analyzed by elaborate man-machine scanning systems which yield data in the appropriate digitized form. This data can be fed into a large computer, where the analysis is performed. The bubble-chamber film frames contain in general much more information than is abstracted from them during the first analysis. It is therefore possible to go over the same film frames again and again to analyze different phenomena from the same events.

By contrast, the counting approach uses a technique by which the experimental data is immediately translated into a digitized form. Many years ago counters were the main devices used to register the data. Today we see many other digital data-producing devices, for instance, spark chambers and photomultipliers. The success of a spark-chamber experiment depends largely on how correct the physicist was initially in assuming which phenomena were to be expected. His experiment is carefully aimed at one or perhaps a few phenomena, and if well aimed, his data will be rich in information on only those phenomena. Selecting phenomena from an event and digitizing them is here done on-line, in contrast to the approach taken with bubble-chamber film frames, where selecting and digitizing takes place during the off-line scanning process. Since the physicist involved in such counting physics experiments can never be completely sure about his assumptions, he needs to be able to adjust his experiment when he perceives that his aim was poor.

The limited capabilities of a small computer like the PDP-5 permit only the most rudimentary analysis of the data. However, simple checks can be performed to see if the experimental equipment works as expected and if the data is valid in a very general sense. The results of these simple checks can be displayed on an oscilloscope, and a program can be constructed to provide several displays which can be requested via teletype or the console of the small computer.

These displays may show the experimenter that his equipment is working correctly, but what it does not show him is the quality of the data in terms of physics: Does all this data bring him nearer to his experimental goal? It would be desirable to perform "analysis in depth" on-line.

The link system was therefore conceived to provide a reliable high-speed data-transfer medium using private telephone lines over distances of a few miles. The data is transmitted in records of a certain number of 12-bit words per record; each word is transmitted in parallel. In addition to data transmission a conversation is carried on between the input-output (I/O) programs of the small and the large computer to exchange relevant information on the data records before and after each data record is sent.

## Aspects of the CDC 6600

Several aspects of the CDC 6600 system are considered in order to illustrate some of the problems encountered in the design of the Data Link (Fig. 2). The CDC 6600 system is constructed to protect the central processing unit (CPU) as much as possible from I/O interference. As Fig. 2 shows, the CPU can be thought of as being surrounded by a protective layer of a 131K 60-bit word core memory, around which ten peripheral processors (PP) are situated. The only way the CPU can communicate with the PP's is via the 131K core memory. Nearly all PP's are involved in I/O communications. Their tasks are assigned by a controlling I/O program on the basis of availability, which makes for a very efficient use of the PP's. Thus it is possible to keep all PP's busy for most of the time, alternating between a number of assignments that is far higher than the number of PP's. In this sense there exists a true time-sharing within the system. This same time-sharing approach is taken in many other areas of the CDC 6600, and given the correct software, this indeed results in a very fast computer.

Still this computer system lacks the essential (hardware) mechanisms to make efficient time-sharing of the CDC 6600 possible between devices outside the CDC 6600. For example, there is no way in which the PP's can be interrupted by a signal from the outside world, in the way this is possible with interrupt hardware in many other machines. Of course, one can to a certain extent make up for that by software simulation of the desired mechanisms. In our situation this was virtually impossible. The systems software used (Chippewa) is made for batch processing of jobs and would need major alterations to implement such simulation. Another important possibility was to dedicate to the Data Link one PP which could perpetually test for conditions indicating whether the Data Link would be alive or not (simulation of an interrupt). However, dedication of a PP for one single I/O task obviously violates the PP time-sharing principle and therefore decreases the throughput of the system, especially when the job mix tends to be I/O-limited. It was therefore decided that a PP would not be dedicated but only temporarily assigned to the Data Link, just as with any other I/O equipment.

## Use of the Data Link

The only way a conversation can start between programs on both sides of the Data Link is to give the CDC 6600 system the initiative. However, it is up to the experimenter or the program of the small computer to decide whether enough of the right data is gathered to be sent over to the analyzing Fortran program. To reconcile these conflicting requirements, a procedure is followed in which the human operator of the CDC 6600 is included in the interrupt system. In fact, the operator is the only part of the CDC 6600 system that can be interrupted from the outside world, using the current operating system. Near his console a light signal indicates that the small computer is making a service request. As soon as he can reserve a control point and enough space in core, he will load the Fortran Analysis program in central core memory, just as he would a batch-process job. However, this Fortran program is placed into the job stream from the side, avoiding the normal priority mechanisms, and slowing the normal job flow slightly. It should be noted that although operator intervention at this point will be time-consuming in terms of the computer time scale, it is still well within the acceptable overall response time of several minutes specified in the design objectives.

After the Fortran program is compiled it soon requires input or output. At this point control is transferred to a Chippewa subroutine which calls for a special I/O routine to be loaded into some PP. This I/O routine is made to deal with the Data Link for the duration of at least one physical record and its surrounding control communications. Depending on wait times and system needs, the originally assigned PP may quite possibly be withdrawn by the system to be used for some other assignment, and a different PP assigned back to the Data Link for the next physical record.

It is mentioned above that the large computer would preferably be the same computer already used for off-line processing of the experimental data. One reason for this preference is the existence of Fortran analysis programs and subroutines within the computer system, which it would be desirable to use unmodified for both on-line and off-line analysis.

Using the link system, these programs need not have their READ and WRITE statements changed at all. Instead, several control cards tell the Chippewa system during compilation to replace all READ TAPE statements with READ Data Link statements.

## Higher-Level Program Interaction

The Data Link can be thought of as merely a vehicle for data and special instructions between the two programs. Since this vehicle is available, communications on a higher level are possible: Many kinds of interaction between the two programs can be invented to make the Data Link more versatile. For example, the small computer may send a record of data. This record, either by prearrangement, or by means of the accompanying command words, is to be interpreted by the Fortran program as a large set of instructions to itself. Branches can be modified, switches set, the different available analysis approaches chosen, and quantity, kind, and format of output results selected. Thus the experimenter has available to him, in a limited way, an extended on-line processing capability which includes some control over the analyzing process.

## Design Aspects for the Data Link

The Data Link is designed as an asynchronous logic device. Basically it consists of a string of buffers, arranged more or less as 12 parallel shift registers (Fig. 3). The transmitters, telephone lines and receivers are located halfway along this buffer string. When data transfer starts, the first buffer is loaded from one of the computers with a 12-bit word, and that buffer is then declared FULL. If the control logic senses that the next buffer in line is EMPTY, data is transferred from the first buffer into the next buffer. This process repeats itself for all the buffers in the string. When for some reason that last buffer is not emptied at the receiving end, it is possible to fill up all the buffers of the Data Link, at which point the transfer of data stops. As described here the control logic only needs to sense the condition of pairs of neighboring buffers in order to decide whether the transfer between the buffers should take place or not. A difficulty arises where two neighboring buffers are located on different sides of the telephone lines. It is desirable to avoid dividing the control logic for such a pair of buffers between the two parts of the Data Link on each side of the telephone line, because of the high number of control signals that would have to run back and forth between the two locations. In the Data Link the solution chosen is the use of a response signal. Each word transmitted over the telephone lines to the other location is received and echoed back (retransmitted) as a response signal. Echoing occurs only when a new word is welcome, that is, when the first receiver buffer is EMPTY. Thus, this response signal is used in the same way as are the FULL and EMPTY signals from buffers on the same side of the telephone lines.

In addition the echoed word is compared with the originally transmitted word. If the echoed word is in error, the same original word is transmitted again. This may be repeated many times until finally the echoed word returns successfully and the next word can be transmitted.

A more detailed description of this error-checking scheme is given in the paper by Robert W. Lafore for this symposium. The individual elements of information as processed by the Data Link are 12-bit words. All buffers in the Data Link therefore can contain at least 12 bits, and the data is transmitted as words of 12 bits in parallel over the telephone line. A 13th bit is added to indicate whether the word is a data or a nondata word (Fig. 4). Nondata words are used to transmit commands and their responses.

## Speed of Data Transfer

From the previous paragraph it now becomes obvious that the travel time of the words along the telephone lines is one of the speed-limiting factors of the Data Link. Other factors are the channel performance of the two computers on each end of the Data Link, and the propagation, detection, and deskewing delays in the buffer logic of the Data Link itself. The delays in the Data Link are insignificant and will not be discussed here. Most modern small computers can transfer a record of data at the rate of one word per 2 μs. The PP of the CDC 6600 System is capable of transferring one 12-bit word per microsecond for the duration of one physical record.

If words are to be transmitted every 2 μs, the propagation delay of the telephone cables between the Data Link stations should not exceed 1 μs. Since this represents a cable length of only 600 ft, the Data Link system tends to be limited by the telephone line. The lines will be from 1/2 to 3 miles long, which corresponds with maximum transfer rates between one word per 8 μs and one word per 50 μs. From a system's point of view, these rates are quite acceptable, being of the same order as the data rates used in the existing magnetic tape units of the CDC 6600 system, as discussed above.

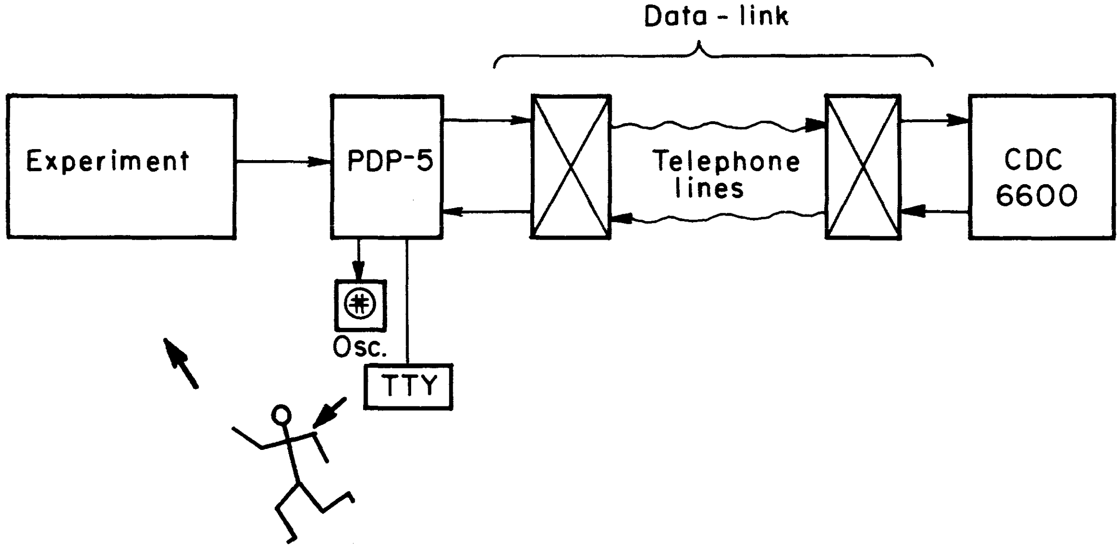



Fig. 1. The Data Link consists of the Device Synchronizer (PDP-5 side) and the Channel Synchronizer (CDC 6600 side).

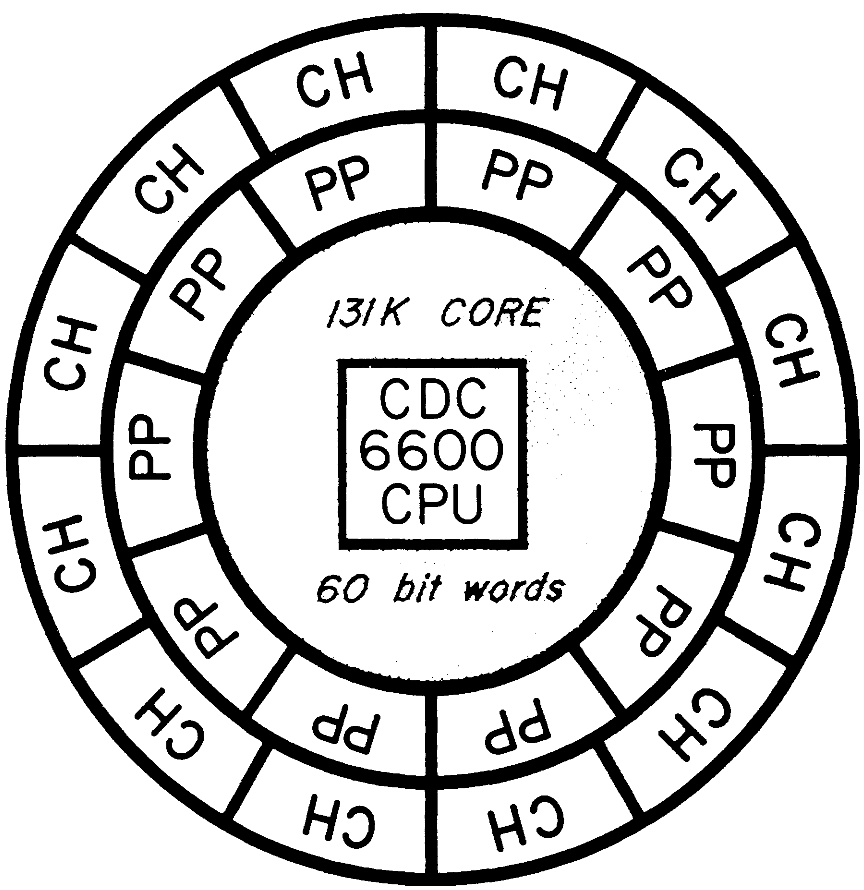



Fig. 2. CDC 6600 system.

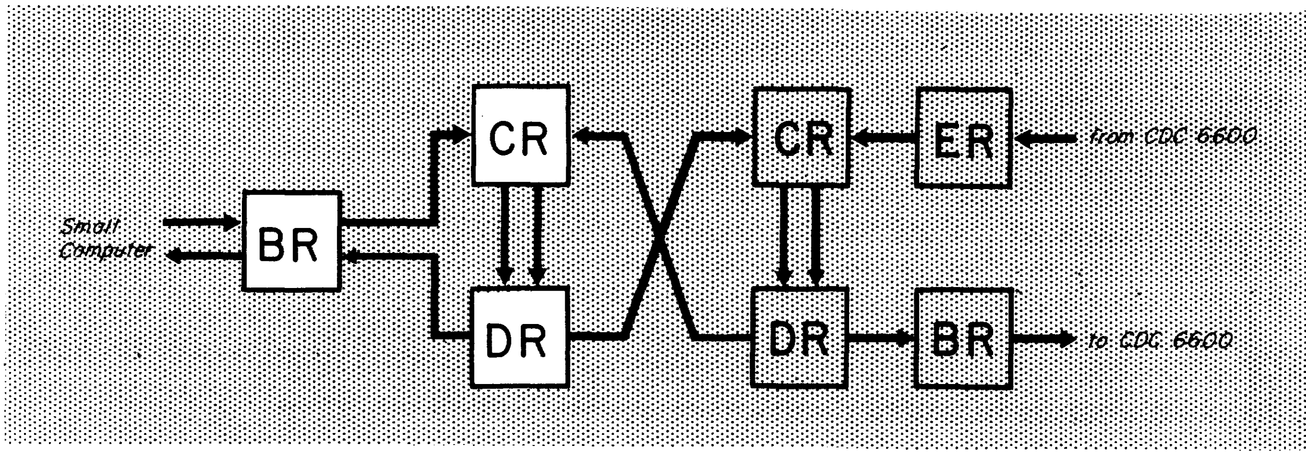
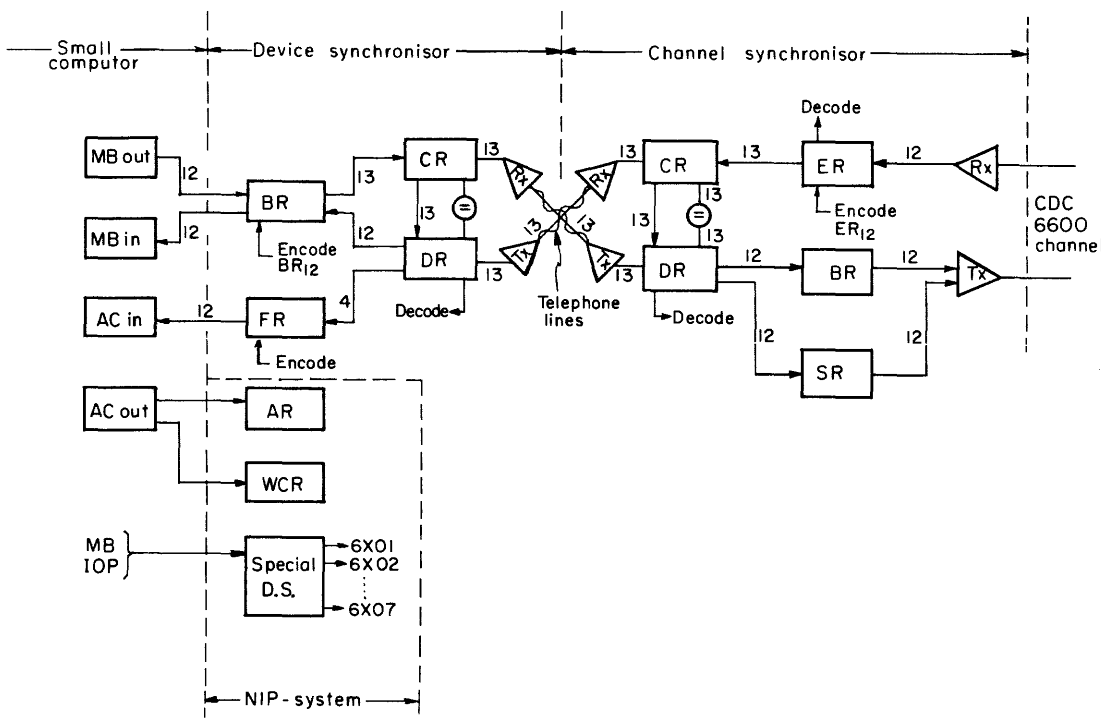



Fig. 3. Simplified diagram of the Data Link.




XBL6710-5378

Fig. 4. Detailed diagram of the Data Link, used with a PDP-5.

# ERROR CHECKING AND OTHER ASPECTS OF DATA-LINK ORGANIZATION*

Robert W. Lafore

Lawrence Radiation Laboratory
University of California
Berkeley, California

## Abstract

The arrangement of registers in the data link permits error detection and correction and solves the problem of synchronizing the computers at the ends of the line. Preliminary information must be exchanged between the two synchronizers before a data record may be sent. The approach to the logic design of the system is also briefly described.

## Introduction

The organization of the Data-Link was conceived with two primary requirements in mind. First, a powerful error correction and detection system is necessary, and second, a means for the computer programs to communicate information concerning the data record to be sent must be provided. This paper describes the methods used to achieve these goals.

## Link "Conversation"

Before the data can be transferred via the link, the I/O routines in the computers at each end of the telephone lines must exchange information regarding the format of the data to be transmitted. This is done by nondata words, which, borowing from CDC terminology, are called "function words" if they go from the 6600 to the small computer, and "status words" if they are initiated by the small computer.

An initial status word from the small computer is used to signal the operator of a service request, via the console light, and a function word carries back his push-button response (see Fig. 1). Later, when the operator has loaded the Analysis program, the program itself initiates the sending of a function word to the small computer to inform it that the data may now be transmitted.

Since the Fortran Analysis program in the 6600 requests only logical records of a certain length, the peripheral processor (PP) I/O routine must obtain the physical record length from the small computer: This is sent as a single data word. To read in this word (as to read in any status or data words) the PP first sends a function word, in this case "GO-WORD-COUNT", whose only purpose is to cause the status word to be transmitted from the channel synchronizer (CS) to the PP.

Following the transfer of each physical record, the small computer informs the PP that either (a) more physical records must follow to complete the logical record, in which case an "end of physical record" status word is transmitted, or (b) the logical record has been completed, in which case an "end of logical record" is sent.

Data output (from the 6600) is analogous to data input except that the PP must inform the small computer at the beginning of each record whether the record will be composed of data or only an end of file.

Since the status and function words are to a large extent initiated and interpreted by software, the "elements of the conversation" described above can be altered to meet future changes in the 6600 operating system or even to provide the interface for entirely different computers at either end of the link.

As noted in the previous paper, data is transferred in words of 12 parallel bits. Function and status words are distinguished from data by the presence of an extra bit which is transmitted as the 13th bit

*This work was done under the auspices of the U. S. Atomic Energy Commission.

in parallel with each word. This 13th bit has the logical value "1" when a data word is sent, and "0" when a nondata word is sent. Since every word must contain at least one bit to be recognized by the receiver, the 13th bit also provides a means of recognizing an all-zero data word.

## The Carrousel

The four registers connected directly to the long-line receivers and transmitters (see Fig. 2) are called the "carrousel" and constitute the heart of the error checking and correcting system.

During normal data transmission (for example, during input to the 6600), a word flows out of the small computer memory buffer into the B register (BR) in the device synchronizer (DS), then into the CR, and finally the DR, where it is both stored and transmitted to the CS. In the CS it is received in the CR, shifted to the DR, and retransmitted to the DS. Arriving in the CR of the DS, this echo is compared with the original word still in the DR. If the comparison succeeds, both CR and DR are cleared. During the round trip of this first word from DS to CS and back, a second word will have shifted from the small computer memory to the BR, where it awaits a successful comparison of the previous word. When both CR and DR are cleared, this word is free to shift into the CR, and the sequence is repeated again. A similar process takes place for data output from the 6600.

Since I/O operations between the computers and the link are concurrent with the transmission and echoing of the previous word, the cycle times of the computers cease to effect the data rate of the system (assuming line length is the limiting factor). Also, if the computer on the receiving end is not able to accept the data fast enough, the process simply pauses until the received word is finally read in, thus preventing loss of data or the necessity for complex synchronization and timing devices.

## Error Correction

If the comparison check on the echoed word should fail, only the CR is cleared, and a special signal called a "fuzzy signal" is sent to the receiving end to warn that the previously received word was in error. This clears the DR in the receiver and permits the same word to be retransmitted as before. The same word may be sent many times until a correct echo is finally received. However, if this echoing process should continue too long, the link assumes that a serious fault has occurred and sends function and status words to this effect to the two I/O programs.

During the periods when the link is waiting for the computers or the 6600 operator to give it further instructions, it arranges that both the synchronizers will be in transmit mode. This ensures that any noise word received at either end will be compared with the zeros in the empty DR and erased, as with an ordinary error. (If the synchronizers were left in receive mode they would retransmit any noise word arriving in them and eventually fill up all registers with replicas of the noise word.)

The "fuzzy signal" referred to above for signalling errors is one of several signals that, because of their critical role in the operation of the link, must be as nearly as possible immune from noise. This is accomplished by sending a pulse train instead of a single pulse as with data or function/status words. Because these signals are very rarely sent in comparison with the number of data words, the additional time needed to provide a pulse train does not increase significantly the time necessary to transmit each record. Other fuzzy signals indicate the beginning and end of a physical record and provide a general reset.

## State Diagrams

In the design of the data link considerable use was made of state diagrams to clarify the operation of the system (see Fig. 3). The position of various flip-flops (for example, those indicating whether a particular register is full or empty), constitute "conditions," a particular combination of which generates a "state." The state in turn alters the conditions by performing various operations, e.g., shifting a word from one register to another. These altered conditions then generate a new state, and so forth.

Since each synchronizer contains more than a dozen flip-flops, plus many external signals which also provide conditions, it becomes extremely difficult to keep in mind the exact sequences and timing. Therefore, to completely specify the logical operation of the machine, both for the initial design and for future debugging, a computer program was written to simulate the operation of the logic. This proved invaluable in tracking down some of the more obscure design and wiring errors.

## Debug Boards

Debugging of the link required the development of another feature of the system: debug interrupt boards. In normal operation the transitions from one state to another are made very quickly and are frequently nonrepetitive: e.g., a single status word is sent, then the link waits for some response from the outside world. This situation would create extreme difficulties in debugging unless some method of slowing down the transition from state to state could be found. (If this were a clocked system, this effect could of course be achieved by merely slowing down the clock.) The interrupt boards fulfill this function by inserting a switch and a light into the output line from each state. Thus, with the switch open, the conditions for a particular state cause the corresponding light to appear; but the effects of that state will not be enacted until the switch is closed, whereupon the light for the next state in the sequence should appear. Since lights indicating the position of the major flip-flops and the contents of each register appear on the front of the synchronizer control panel, the debug interrupt boards provide a means of seeing the status of all active elements in slow motion. Used in conjunction with the computer printout of the sequence of conditions and states, this provides an easy and effective debugging method.

## Conclusions

The data-link system provides a reliable and satisfactory solution in its present context to the problem of interfacing small data-gathering computers to a large computer for the analysis of a comparatively small quantity of data at infrequent intervals. Should this type of service become more popular in the future, the link can be readily modified to accept changes in the 6600 operating system (such as using a dedicated PP for all link data) or interfacing to a time-sharing computer devoted to link data handling.

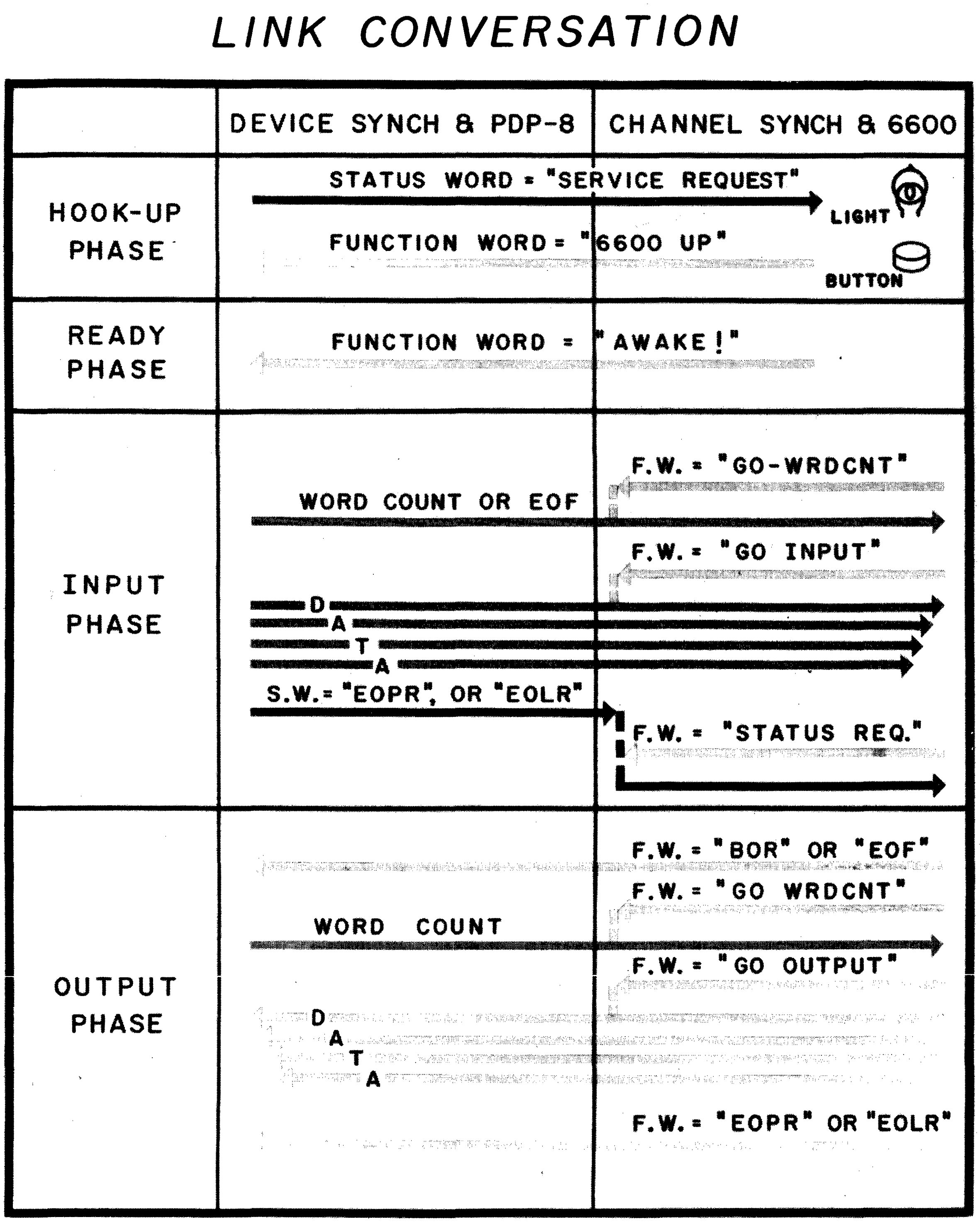



Fig. 1. Link conversation.

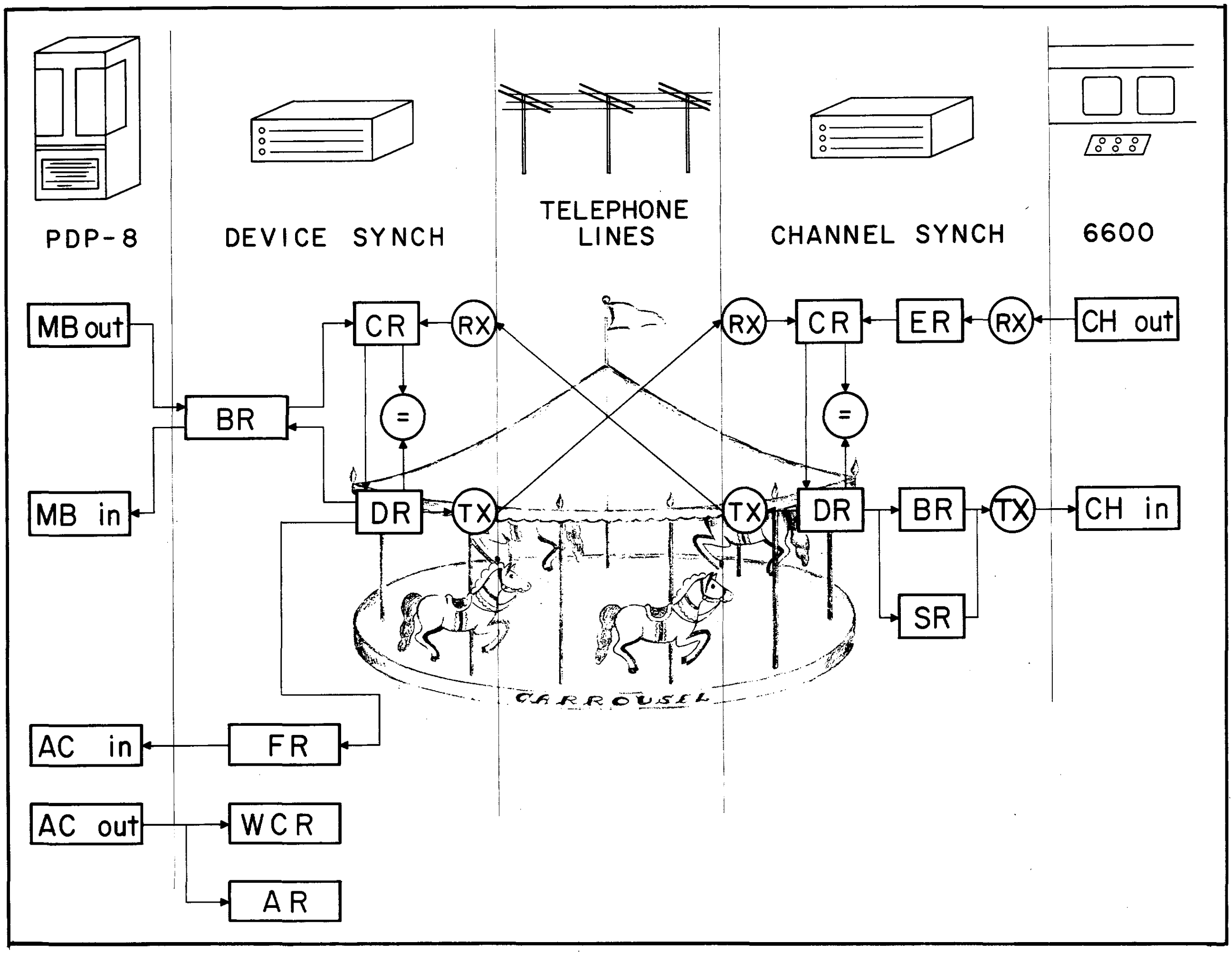



Fig. 2. Organization of the link registers.

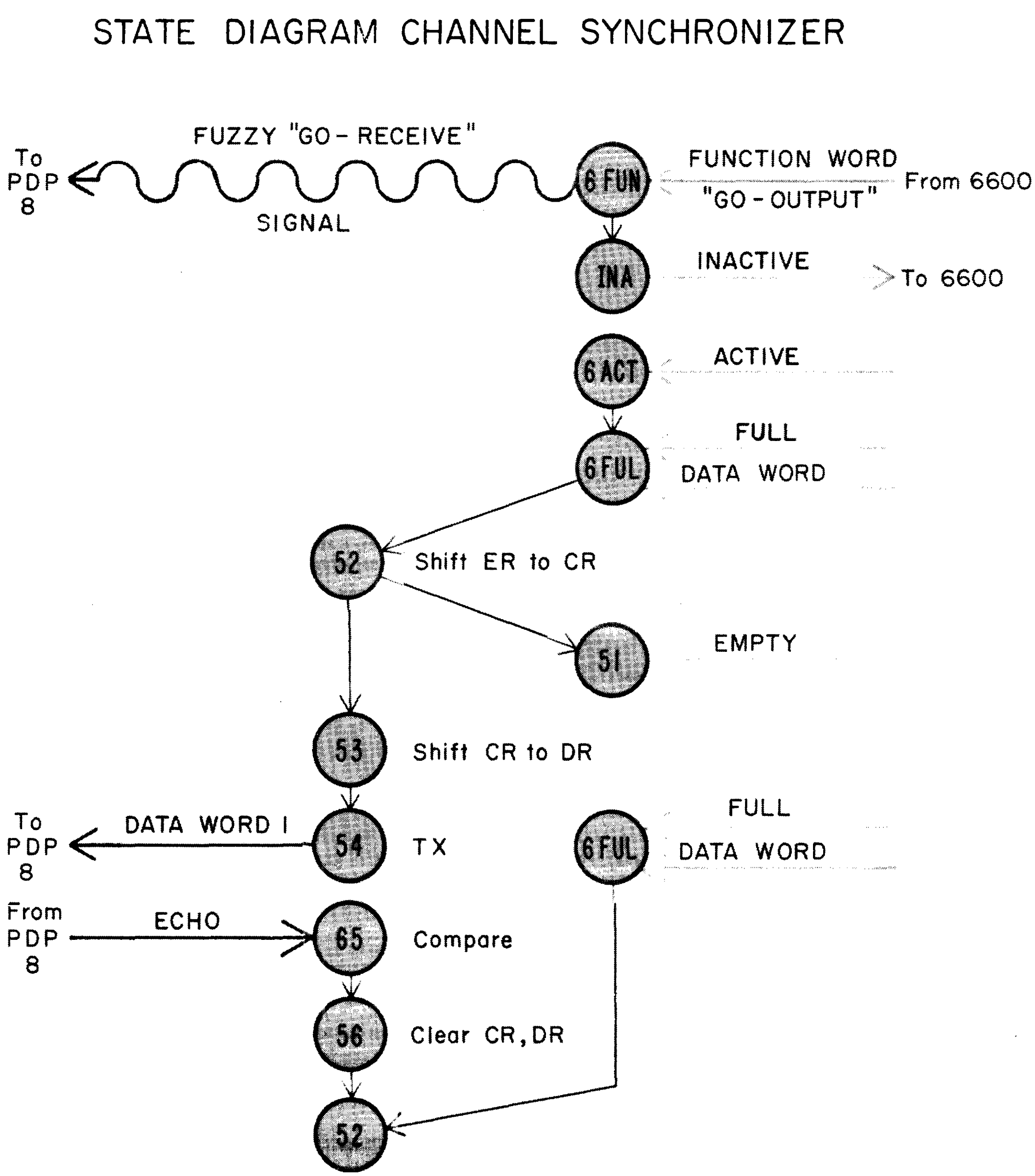



Fig. 3. Typical state diagram (channel synchronizer, data output).

# TRANSMISSION OF LINK DATA OVER TELEPHONE LINES

Alan E. Oakes

Lawrence Radiation Laboratory
University of California
Berkeley, California

## Abstract

A technique is described for transmitting computer data over twisted-pair telephone lines at a rate limited only by the round-trip propagation delay.

The LINK method of signal generation and detection is presented along with a discussion of the effects of telephone line distortion, attenuation, and noise.

## Introduction

The Data Link system has been developed to transmit data from building to building at LRL Berkeley over telephone lines on the order of 1 mile long. The Data Link should not be confused with systems that use telephone lines to transmit data over long distances. Long telephone lines have a bandwidth of about 3kHz, which limits the rate of transmission to about $1.2\times 10^3$ bits per second. The noise on long telephone lines causes about one error per $10^5$ bits.

The Data Link will transmit data between any of the several experimental areas at LRL and the computer center in Building 50B. The first line to be used runs from the 184-inch cyclotron to Building 50B, a distance of 4,000 feet. I will hereafter refer to this line as the cyclotron line. On the cyclotron line the rate of transmission is $8\times 10^4$ words per second or about $10^6$ bits per second. The present raw error rate is about one error per $10^7$ words, but even this low rate should be reduced several orders of magnitude by the Data Link error-detection and correction system described by Sypko W. Andreae and Robert W. Lafore in previous papers.[1]

The Data Link at present is a full-duplex system using 34 "voice quality" twisted-pair telephone lines. Twenty-six lines are used for transmission of 13-bit words[2] and eight lines are used for transmission of control signals. At this writing, the hardware for the system has been built and debugging is just beginning. The first experiment to use the Data Link is scheduled to receive beam in the 184-inch cyclotron in January 1968.

## Transmission

Data are transmitted in the form of bipolar pulses. The arrival of a pulse indicates a logic "1" and the absence of a pulse indicates a logic "0." Since the Data Link uses complete handshaking, the period between pulses is equal to the round-trip propagation time. The 4,000-foot cyclotron line has a period of 12.5 microseconds.[3] A 3-μsec bipolar pulse is used on this line. The attenuation factor of 4 allows the use of a simple integrated circuit transmitter. The 9.5 μsec between the end of one pulse and the start of another is just long enough to allow base-line recovery at the receiver. A preliminary study of telephone lines both longer and shorter than the cyclotron line indicates that a bipolar pulse can always be found which is wide enough to keep the attenuation factor near 4 and yet narrow enough to allow base-line recovery between pulses. The correct ratio between pulse width and period between pulses appears to be approximately 1/4.

Two monostable multivibrators control the width of the positive and negative areas of the bipolar pulses for all the 13 transmitters. Each transmitter consists of two power nand integrated circuit gates which drive two transistors connected in a push-pull configuration to produce a bipolar pulse in the secondary of the output transformer. An oscilloscope picture of a transmitter output pulse is shown in Figure 1.

The signal is filtered as it travels along the phone line. The high-frequency components are reduced more than the low-frequency components. The resultant waveform at the receiver is made up of only the lowest-frequency components and closely resembles one period of a sine wave. See Figure 2.

### Detection

Our telephone lines lie in underground ducts paralleling other lines for thousands of feet. These other lines are constantly producing noise. Most of the noise is of low frequency and is of no particular concern; but some high-frequency noise (e. g., switching transients) is produced. The cyclotron line, after entering the cyclotron building, runs in a wire-trough several hundred feet to the experimental area. Here the line is subjected to noise produced by spark chambers and other apparatus as well as the rf field of the cyclotron itself.

The noise on the cyclotron line occurs rarely enough, however, to make direct observation and photography of oscilloscope displays of the errors extremely difficult. No reliable photographs of telephone line noise were taken before the signal detection system had been designed. As a result we were overcautious. With the impression that the background might include noise of about the same frequency and amplitude as the data pulses, we at first envisioned using detectors with multiple height and time windows.

For purposes of testing and gathering data on noise, we designed several simpler detectors, one of which required the positive-going portion of the bipolar pulse to pass through a height window much like a single-channel analyzer. A detector of this type was constructed. The first hour it was connected to one of the cyclotron lines, it detected 300 counts due to noise. After a few days of tuning and debugging, the noise count was down to less than 100 per hour.

This low error rate encouraged us to make a full-scale test of the transmitter-receiver system using the height-window method on the cyclotron phone lines. The technique used for testing required all 26 transmitters and 26 receivers of the actual system. It is shown schematically in Figure 3, and is described more fully in the Test Program section.

This test demonstrated that many more error words were caused by the failure to detect the desired signals then by the detection of unwanted noise. Even with our already weakened detection requirements, we were still being too strict. The present system, which requires only that the positive amplitude be greater than a given threshold, was then designed. The round-trip error rate when this system is used on the cyclotron line is about one error per $10^7$ words.

### Control Signals

In a previous paper by Robert W. Lafore, control or "fuzzy"[4] signals were mentioned in connection with conversation format, general reset, and the error detection and correction system. In all, eight fuzzy signals are used--five by the Device Synchronizer and three by the Channel Synchro-

The failure of any of the many control signals sent in each conversation would cause a system abort. The control signals must therefore never fail. This is, perhaps surprisingly, not difficult to approach in practice. Our technique at present is to send a train of square waves 16 cycles long. (The same types of transmitters and receivers are used for both the data and control signals.) The output of each control signal receiver goes to an analog device which essentially counts the number of cycles. Approximately 5 out of the 16 cycles must arrive for the control signal to be detected.

More than $10^{11}$ control signals have been transmitted without failure (i. e., both without failing to detect a signal and without triggering on noise). Since this figure is several orders of magnitude larger than the total number of control signals likely to be sent during the lifetime of the Data Link, the possibility of even one failure appears remote.

### Test Program

A block diagram of the test system is shown in Figure 3. A 12-bit word is generated by the computer, sent to the test rig, and transmitted over phone lines to the cyclotron. There the word is detected and retransmitted back to the test rig in Building 50B. After detection, the word received is sent to the computer for comparison with the original word. Words may be chosen by the PDP-5 on any basis: random numbers, all sevens, all zeros, all but one bit up, all but one bit down, etc.

The test rig was very effective in locating problems during development. The transmitters, receivers, and associated registers and logic went through considerable evolution before the present design was adopted. Each change was dictated by test results and then checked for effect. The PDP-5 had one drawback, however; it was not fast enough to send words at the maximum rate the Data Link could sustain (12.5 μsec). The fastest PDP-5 program we used sent words every 150 μsec.

In order to investigate the behavior of the Data Link at the maximum rate, the PDP-5 was replaced with a hardware device. The word to be sent was chosen by a switch register. The word returning after a round trip was stored in a register and compared with the switch register with exclusive-or gates. Regardless of the outcome of the comparison, the switch register word was immediately retransmitted, thus maintaining the 12.5-μsec rate. Each time the comparison failed, a count was stored in a scaler. At a fixed period of time after an error was detected, the contents of the scaler were printed out on a typewriter and the scaler was reset. By varying the time from error detection to scaler readout, a crude idea of the noise pattern could be obtained.

## Error Rate

The following is a summary of the findings of the test program:

(a) The errors usually occur in bursts with a particular time pattern (i. e., one or two errors followed by eight to ten correct transmissions followed by one or two errors, etc.).

(b) As might be expected, errors are more probable during the day than at night, and are more probable during the week than on weekends.

(c) During an 8-hour working day, the raw error rate averages about one error in $10^7$ words.

It should be emphasized that the above describes the error rate of the transmission system only. The error rate of the entire Data Link system (after automatic error correction) is expected to be reduced several orders of magnitude below one error per $10^7$ words.

## Footnotes and References

*Work done under the auspices of the U. S. Atomic Energy Commission.

1. Sypko W. Andreae, A Computer Data Link for High-Energy Physics Experiments, this Symposium;
   Robert W. Lafore, Error Checking and Other Aspects of Data-Link Organization, this Symposium.
2. Data Link accepts and delivers 12-bit words, but within the system a 13th bit is transmitted with data words to distinguish them from status and function words. Status and function words are described in a previous paper by Robert W. Lafore.
3. Because the telephone lines twist not only on themselves but also about other wires in the cable to form double helixes, more than one foot of line is contained in a linear foot of cable. We find the propagation velocity to be nearly 1.6 nsec per linear foot of cable, or about 12.5 μsec for the cyclotron line.
4. Many of the bugs of the Data Link system were discovered and ironed out by using a large "gameboard"--a block diagram representation of the buffers and their interconnections drawn on a piece of cardboard. Data, function, and status words were represented by small cards and were moved from register to register to simulate the action of the actual system. When control signals first made their appearance on the gameboard, they were represented by ordinary signals. In our haste to see how the system would work with control signals, the wave trains were drawn on the cards before the background was completely dry. The black ink leached into the yellow. The wave train looked fuzzy. The name stuck.

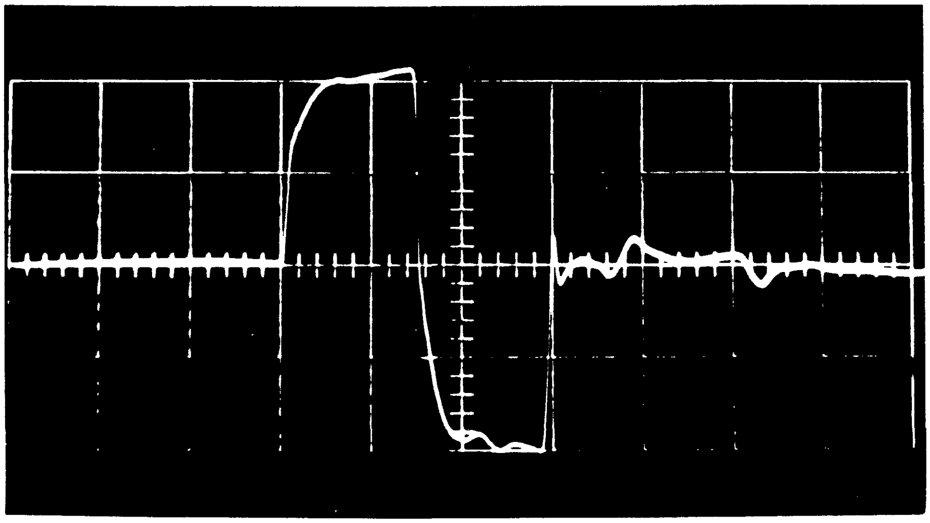


Figure 1. **Transmitted Pulse**
4 V/cm 1 microsec/cm

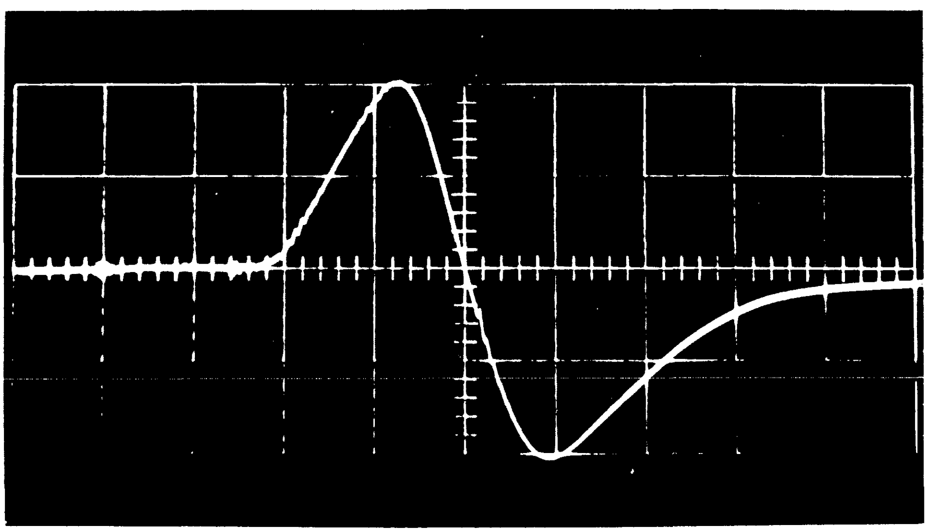


Figure 2. **Received Pulse**
500 mV/cm 1 microsec/cm

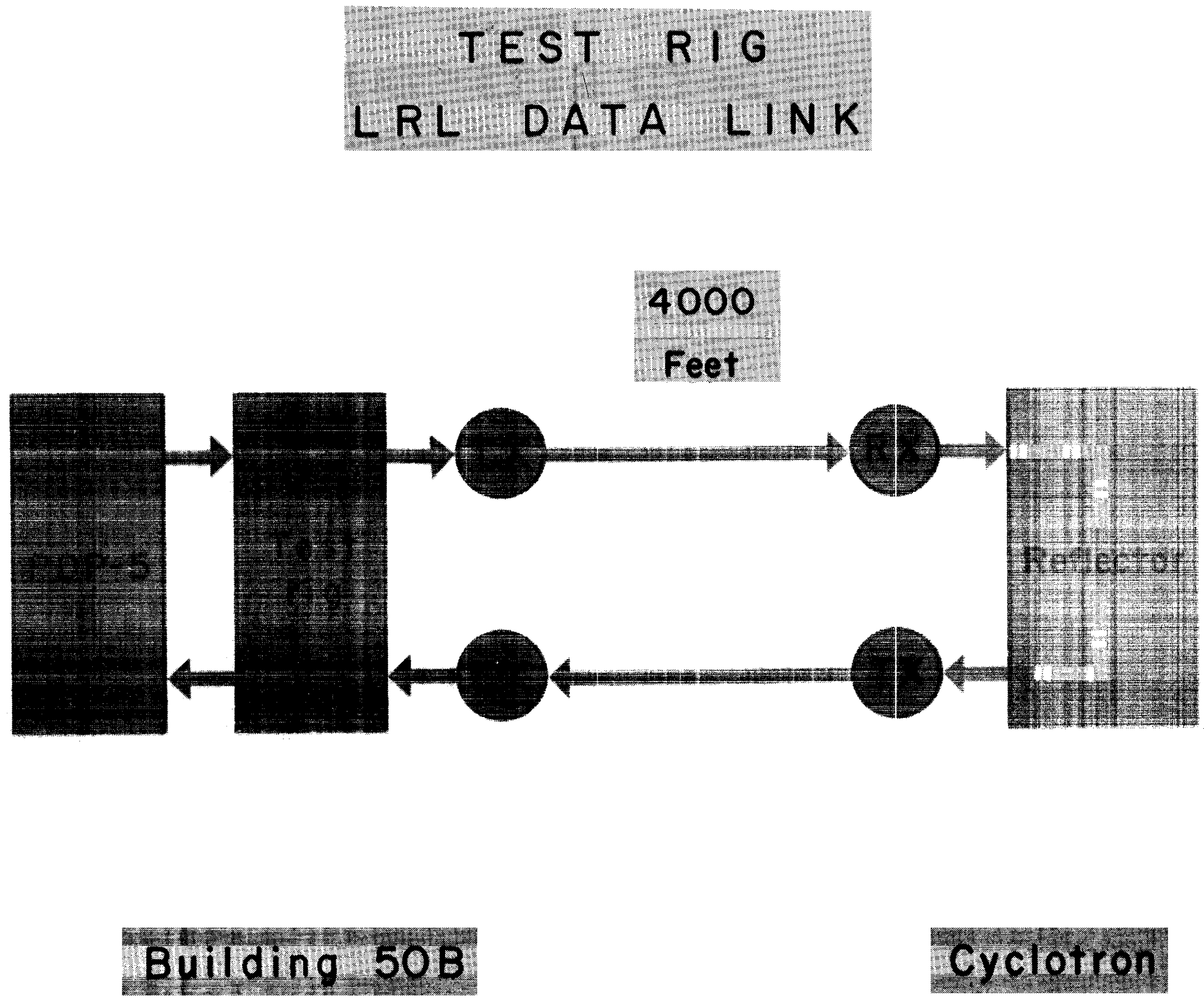



Figure 3. Test rig.

# QOLFIM - AN ON-LINE FILM MEASURING SYSTEM (PDP-5/8) FOR THE PROCESSING OF BUBBLE CHAMBER PHOTOGRAPHS---MARK II † *

B. B. Culwick, E. Glazer, B. Lebowitz, and T. Meehan
Brookhaven National Laboratories
Upton, Long Island, New York

## Abstract

The Bubble Chamber Group at BNL has recently revised an on-line, manual, measuring system originally installed in 1965. The purpose of the system, consisting of a time-shared PDP-5 and several semi-automatic, measuring microscopes with associated ASR-33 Teletypes, is to collect the coordinate pair measurements of high energy particle interactions, recorded on film. The data collected is put on magnetic tape (with sufficient redundancy to reduce transmitted tape errors to a very small number) and read by a CDC 6600. The system of programs (locally dubbed QLOD) that reconstructs this data into three dimensional interactions and then performs a test of appropriate kinematical hypothesises for event identification and then does a myriad of other bookkeeping and display tasks, represents an enormous number of programming man-hours by physicists and mathematicians. The guiding of the initial measuring tasks by QOLFIM is intended to maximize the quality of the data entering the system. This latter goal has been accomplished with a minimum cost in hardware and a reasonable software effort.

† Work performed at Brookhaven National Laboratory under the auspices of the U.S. Atomic Energy Commission.

* This paper was not received for publication.

GAMMA RAY SPECTRUM STRIPPING

F. Riess - Stanford University
Stanford, California

Abstract

A FORTRAN program is described which analyzes multi-channel pulse-height spectra of gamma rays by a least-squares fitting method. Up to five gamma ray lines can be analyzed simultaneously. A background of arbitrary shape can be included in the fit.

A considerable variety of gamma ray spectrum stripping and gamma ray unfolding programs by least-squares fitting methods have been reported in the literature.[1] Whereas the limitations of stripping programs are obvious when dealing with complex gamma ray spectra, unfolding procedures involve large matrices and hence are restricted to big computers. In order to be able to analyze the gamma ray spectra taken with the Stanford 10 inch NaI(Tl) assembly[2] on a PDP-7, we designed a program which is a combination of a least-squares fitting method and a variation procedure. The program is written in FORTRAN II and makes use of the SCANS system as described in a paper by A. Anderson.[3]

Figure 1 shows one of numerous spectra taken with the Stanford 10 inch NaI(Tl) assembly. The gamma ray energies in this spectrum vary between 4 and 18 MeV. One of the most striking features is the weak interaction between the shapes of the lines and their energies. Hence, consideration for available space of the 8k memory of the PDP-7 as well as for a reasonable computation time brought us to a rather unusual approach to the generation of the lineshapes.

Two reference lineshapes are matched at the peak and linearly interpolated point by point. Simultaneously they are normalized to the correct energy calibration. Missing channels are obtained by a linear interpolation. The disadvantage of this method lies in the fact that the interpolated lineshape is not necessarily a true picture of the two reference lineshapes. This can be understood by an example: The sum of two Gaussians is only a Gaussian if they have the same width. However, the width of our lineshapes do not vary strongly as a function of the energy. Moreover by keeping the interpolation range small, we can keep this error small. As a matter of fact we still can obtain reasonably good fits even if the interpolation range extends over a region of 5 to 10 MeV centered, around a $\gamma$-ray energy of 15 MeV. Any extrapolation of the lineshapes, however, should be strictly avoided. A similar, although more refined, approach of generating lineshapes was reported by A. H. Wapstra and J. Oberski.[1] We feel that the simplicity of our lineshapes justify this method well enough and that the approximations involved are very much compensated by the gain in computation speed and decrease in required program space.

The interpolated lineshapes are shifted by fractions of a channel via a quadratic interpolation and normalized in their total area.

In order to obtain the amplitudes of the different lines we minimize

$$\chi^2 = \sum_{j=min}^{max} \left( SPEC(j) - \sum_{k=1}^{N} A(k)\ LINE(k,j) \right)^2 / SPEC(j) \qquad (1)$$

as a function of the amplitudes A(k). The array SPEC(j) represents the experimental spectrum, the arrays LINE(k,j) N lineshape spectra and A(k) their amplitudes. The sum extends over an appropriate region in the spectrum. The problem can be solved algebraicly and gives a set of linear equations of rank N in the form:

$$\sum_{k=1}^{N} A(k)\, M(i,k) = C(i) \qquad (2)$$

The method used to solve these equations is an iteration procedure described by N. E. Scofield.[1] The exact solution can be formally written as

$$A(k) = \sum_{i=1}^{N} \left(M(i,k)\right)^{-1} C(i)$$

$$= \sum_{i=1}^{N} D(i,k)\, C(i) \qquad (3)$$

The diagonal matrix D serves as a substitute for the inverse of the matrix M and is obviously defined by

$$D(i,k) = A(k)/C(i)\ \delta(i,k) \qquad (4)$$

with $\delta(i,k)=1$ for $i=k$ and $\delta(i,k)=0$ for $i \neq k$. It should be noted that the matrix D is not the diagonal matrix of $M^{-1}$ but is a function of the vector C.

Assume we have a starting vector $A^{(o)}$. We can use equation (2) to evaluate the corresponding constant $C^{(o)}$ and hence their deviations from the actual vector C. Equation (4) defines a diagonal matrix $D^{(o)} = A^{(o)}/C^{(o)}$ which we can use to compute a correction for the vector $A^{(o)}$ with the help of the deviations of $C^{(o)}$ from C. Hence we get a new vector $A^{(1)} = D^{(o)}\, C$ which can be inserted again in equation (2). Thus we get an iteration procedure for which the convergence is not at all established. Without giving a mathematical proof we note that the strength of the diagonal elements of the matrix M with respect to the non diagonal elements is the governing factor in the convergence of the iteration. Inspecting the definition of the matrix M as a result of the minimization of equation (1), we find that the diagonal matrix elements are always dominant as long as the gamma ray lines are fairly well seperated.

This iteration procedure has several advantages with respect to an exact solution of equation (2): non-zero positive starting values $A(k)^{(o)}$ generate always non-zero positive amplitudes $A(k)^{(n)}$. If one of the amplitudes is set to zero, it always stays zero providing an easy way to check the contributions of different lines in a fit spectrum. The iteration procedure does not occupy much space in the program with reasonable computing time.

The iteration is stopped if the length of the amplitude vector $A(k)^{(n)}$ does not change more than a given percentage in one step of iteration or if the number of iterations exceeds a fixed value. In a very unfavorable case of four poorly resolved gamma rays, a maximum of about 100 iterations were needed in order to generate less than .02% change in the length of the amplitude vector. The starting parameters were arbitrarily chosen.

A search for the best energy for each line is made by going one step lower and one step higher from the initial energy and calculating the minimum of the parabola in the $\chi^2$ -plane given by these three points. In order to assure convergence even if the curvature of the parabola is zero or negative, the maximum allowed change in the energy is limited and only the magnitude of the curvature is used. By changing the energy of all lines simultaneously a new fit spectrum is obtained, the $\chi^2$ -value of which is compared with the $\chi^2$ -value of the first fit. The variation continues until the change in $\chi^2$ is less than a given percentage.

However since $\chi^2$ is not a smooth function of the energy of a line, it is possible that the fit gets worse again near the minimum. In this case one more set of variations is calculated in an attempt to come closer to the minimum again.

Reference lineshapes and interpolated lineshapes are stored in the same two-dimensional array. Hence it is very easy to increase the number of reference lineshapes by decreasing the number of interpolated lines. This has been utilized to include a background into the fit: a third lineshape when inserted into the program is regarded as a background spectrum. It can be easily seen that the maximum number of fit lines decreases by two if this facility is used.

The program is controlled via the sense switches of the computer console and the IDIOT panel. The fits can be monitored at an oscilloscope. Typical control parameters are: the step width in the variation, variation of the background energy, printout of intermediate $\chi^2$ -values or fit data and automatic servicing. The program allows smoothing of spectra as well as stripping of single gamma ray lines or groups of gamma ray lines. The number of lines which can be fitted simultaneously is essential limited to five by the available space. The allowed length for each lineshape array is 128 channels, whereas the spectrum may have up to 256 channels. Real time operation as well as batch processing is possible.

The program starts with equal amplitudes for all lines when starting from scratch, and with the amplitudes of the previous fit spectrum when analyzing several spectra in turn. Typical computing times for a 3-line fit as shown in Figure 2 are about 15-20 minutes: most of the time is used for the search for the best position of each line. This figure shows a spectrum from the reaction $^{11}B(p,\gamma)^{12}C$. The two strong lines have energies of 13.57 and 18.0 MeV and the weak line an energy of 15.11 MeV. The tail of the lineshapes is an extrapolation from the defined portion of the reference lineshape. The slope of this straight line extrapolation can be changed.

Figure 3 shows the use of a background spectrum. The fit to a gamma ray spectrum from the reaction $^{27}Al(p,\gamma)^{28}Si$ contains three lines on an exponentially decreasing background. Contributions from gamma rays with higher energies are approximately corrected for by the background. Figure 4 shows a somewhat different use of a background spectrum: in the reaction $^{7}Li(p,\gamma)^{8}Be$ the transition of the first excited state is broadened by about 1.45 MeV. A reference lineshape was broadened by folding with a Breit-Wigner function of 1.35MeV and is used as a background spectrum.

The program will be available in the DECUS library.

I would like to thank P. R. Bevington for helpful discussions and N. G. Puttaswamy for making his GAMMA RAY UNPEELING program available to me.

1 "Applications of Computers to Nuclear and Radiochemistry" Proceedings of a Symposium, Gatlinburg, Tennessee, Oct. 1962 NAS-NS 3107

2 Martin Suffert, W. Feldman, J. Mahieux and S. S. Hanna, to be published

3 Albert Anderson, to be published in "Proceedings of the Fall 1967 DECUS Symposium." DECUS, Maynard, Mass.

This research was supported in part under the National Science Foundation.

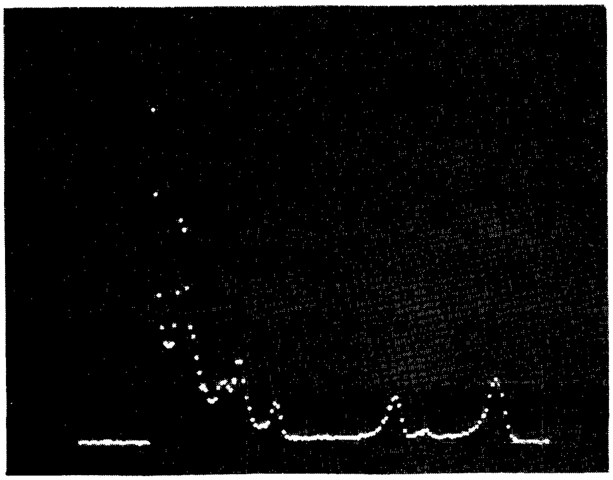


Figure 1

Gamma ray spectrum from the reaction $^{11}B(p,\gamma)^{12}C$; gamma ray energies vary between 4.4 and 18 MeV.

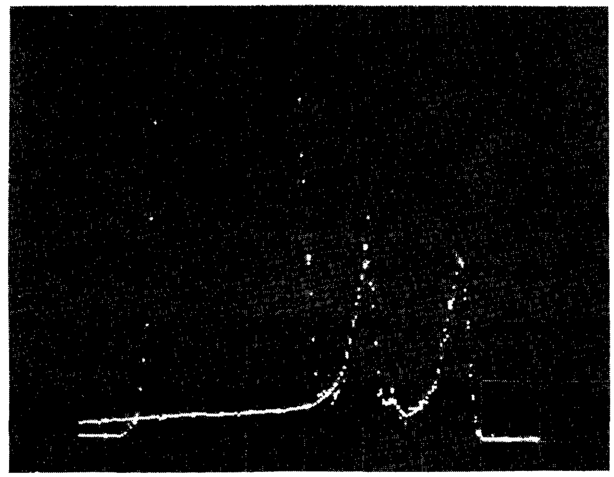


Figure 2

Gamma ray spectrum from the reaction $^{11}B(p,\gamma)^{12}C$; the computer fit is also shown.

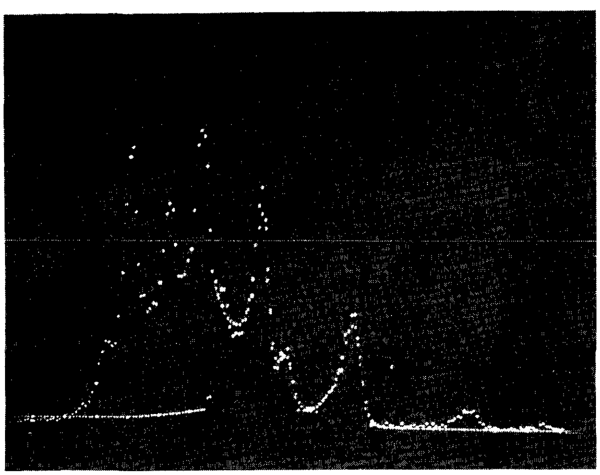


Figure 3

Gamma ray spectrum from the reaction $^{27}Al(p,\gamma)^{28}Si$; a computer fit to three lines with a background is also shown.

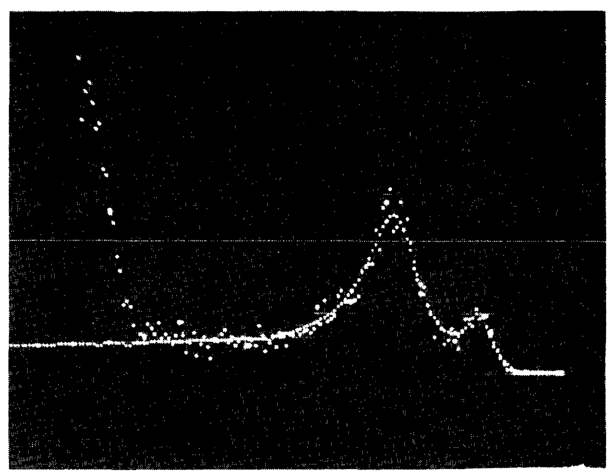


Figure 4

Gamma ray spectrum from the reaction $^{7}Li(p,\gamma)^{8}Be$; the computer fit is also shown.

# REAL-TIME DATA ANALYSIS WITH FORTRAN

Albert Anderson - Stanford University
Stanford, California

Abstract

The effectiveness of FORTRAN programs to implement real time reduction and analysis of nuclear physics data is discussed, and two typical programs for the PDP-7/9 are described. The Gaussian Reduction and Analysis of Spectrum Peaks (GRASP) program makes Gaussian fits to experimental data peaks in multichannel pulse-height spectra, with provision for background fitting and subtraction. Fitting the peaks analytically provides a consistent method for extracting meaningful parameters (area, centroid) from the data. The DIrect REaction Cross Sections (DIRECS) program calculates theoretical angular distribution curves that may be compared with their experimental counterparts. Even under FORTRAN, these programs provide optimum interaction in real time between the experimenter and the analysis. The experimenter may monitor and control both programs with the Teletype, IDIOT panel switches, scope display, and light pen by means of general-purpose, FORTRAN-compatible, symbolic subroutines.

## Introduction

Extensive use of the small computer in the laboratory has made real-time and on-line data reduction and analysis a wide-spread phenomenon. Not so wide spread, it seems, is the use of FORTRAN as the source language for these programs. This is a report on the success we have had in the Stanford Nuclear Structure Laboratory with real-time programs written in FORTRAN.

FORTRAN is almost invariably used by scientists and engineers in non-real-time applications because of its simplicity, similarity to ordinary mathematical expression, and pre-existing universality. The standard FORTRAN instruction set, however, makes no provision for real-time control and display because of the complex and singular requirements of individual installations. Even allowing interleaved symbolic statements, as in the PDP-7/9 FORTRAN II compiler, does not help the researcher/programmer who does not have time to learn machine language and to deal with mode compatibility, timing and other problems associated with symbolic coding. Either such a person must depend on the installation's systems programmers to deliver his reduction and analysis package, or a system must be designed to enable the purely FORTRAN programmer to write his own real-time programs.

At Stanford, we have chosen the latter course. Our solution is to combine all of our real-time processes (which must of necessity be partially in assembly language) into symbolic subroutines that may be easily called by FORTRAN programs. This means that all of the mathematics of the data analysis is couched in terms of FORTRAN, leaving only the control and I/O functions in the hands of non-FORTRAN subprograms. This simple system avoids the necessity of modifying the compiler or assembler for specific system requirements.

## Hardware

The Stanford Computer for the Analysis of Nuclear Structure (SCANS) and its on-line applications were described in detail at a previous DECUS conference.[1] Briefly, it consists of an 8K PDP-7 and an 8K PDP-9. Both have a type 34 display feeding a Tectronix 503 oscilloscope, a type 370 light pen, and a system of controlling switches housed in an IDIOT (Indicating Digitizer for Input/Output Transformations) box. The IDIOT box provides 36 bits (two words) of information, some of which can be sampled by a FORTRAN-compatible subroutine. Its purpose in real-time use is to provide the operator greater convenience in controlling display scales, program branching, etc., than by manipulating accumulator switches.

In addition, the PDP-7 has an NCR card reader, a Moseley X-Y plotter and two dual Analog-to-Digital Converters (ADC's) attached. Also, a number of Nuclear Data analyzers (an ADC with a storage unit) may dump the contents of their memories into the PDP-7 memory. The PDP-9 generally stands alone as a separate processor, but its memory can be used to extend that of the PDP-7, even under FORTRAN control. This is accomplished by having instructions which reference arrays declared in EXTEND MODE statements trap out to a routine using the interprocessor link to read the PDP-9 memory.

## Software

Appendix 1 exhibits the system of FORTRAN compatible display and control routines. The first column indicates the subroutine's use. The second column lists the subroutines whose arguments are predominantly fixed point, while the third column lists those for floating point. Oscilloscope displays are generated by the -SCOPE series of routines and the light pen employs the -PEN series. The Moseley plotter is controlled by the -PLOT set of routines. The LISTEN subroutine provides FORTRAN with the facility to skip on the teletype flag and is used in conjunction with the scope displays. ISENSE is the subroutine for reading the status of the IDIOT box switches. The INPUT- series of subroutines reads paper tapes punched by Nuclear Data or Victoreen analyzers into FORTRAN arrays named in the argument list. Error checking and overflow analysis are also performed. The DUMP- series functions similarly except that the data are read via cable directly from the analyzers' memories. The OUTFLT subroutine punches floating arrays onto paper tape in the fixed point Nuclear Data format. ACTION is a FORTRAN program tying all of the above subroutines together.

These subroutines provide a complete link between FORTRAN and all of the real-time I/O equipment except the card reader. The CREASE program is that link. Under this program, conventional IBM FORTRAN II programs punched in cards are translated into DEC FORTRAN II programs punched into paper tape. This tape may be read into the compiler without further editing.

## The Program GRASP

Gaussian Reduction and Analysis of Spectrum Peaks (GRASP) is a typical FORTRAN program designed to operate on experimental data obtained in the lab. Spectrum peaks can be fit sequentially with a Gaussian function, providing a consistent method for extracting parameters from the data.

Appendix 2 is a control map of GRASP. This program is composed of three sections (called LISTEN LOOPS), each section providing a display and branching structure. The first section provides for reading, writing and editing the data. The second section uses the light pen to identify a peak to be fit. Up to ten sets of light pen location values identifying peaks may be stored at once. The fitting process and parameter extraction for any particular peak is performed in the third section.

Appendix 3 lists the first few lines of the source program. Note the ease with which the displays are generated by the FORTRAN program. The process is entirely between statements 102 and 103. PTSCOPE is the display routine. IDATA is a vector containing the fixed point data to be displayed, and NUMBER is the length of the vector. The verticle scale, MAXY, is determined by ISENSE as the current setting of the IDIOT box scale knob. The remaining numbers control the position of the display on the scope face. PTSCOPE displays the data once, then the program proceeds to LISTEN. If the teletype keyboard is struck, L is non-zero and the program branches accordingly. Otherwise L = 0 so that the display is repeated. Flicker free operation is possible for NUMBER less than 1000. The PTPEN subroutine displays a cross at position LTPENX and LTPENY which may be moved with the pen.

Figure 1 shows a typical spectrum in real-time oscilloscope display. The data represent the deuterons (d) that were detected in our lab when thin phosphorus ($^{31}P$) targets were bombarded with 16 MeV protons (p). Along the horizontal axis is the channel number, which in this case is directly proportional to the energy of the detected particles. The vertical scale is the number of particles recorded in the particular channel. In theory, for each of these groups there should be only one channel displaced off the axis, representing the discrete energy levels in the residual phosphorus isotope. In practice, because of experimental uncertainties in the detection process, the spectral groups assume a Gaussian shape, and closely neighboring levels may not even be resolved. The task of the program is to extract information regarding the strength of the reaction (area of peaks) or the precise energy of the levels (centroid of the peak in terms of channel number).

We start this process by indicating a selected region around a peak with the light pen. Within the selected region, the light pen also marks the area containing only background (on which the peak is superimposed) as well as the peak to be fit, since there may be more than one in a selected region. For a simple illustration of the process, Figure 2 shows the selected portion containing the peak to be fit, and Figure 3 shows the fit. The baseline is a polynomial fit to the background, on which is superimposed the Gaussian line shape. In another selected region, shown in Figure 4, there are five states. This is not immediately apparent, but smoothing shows them clearly (Figure 5). The peaks can be fit in succession (Figure 6 shows the fit to the leftmost peak) and subtracted from the remainder until finally all reaction strength has been measured and subtracted away. Figure 7 shows the result after two peaks have been removed, and Figure 8 the result after four peaks have been subtracted. Output from the program includes the goodness of fit criterion $\chi^2$, the area of the peak after subtracting the background, and the peak's width and location, as well as the background fitting parameters.

## The Program DIRECS

After reducing and analyzing the data, we must compare the results with theory. The program DIrect REaction Cross Sections (DIRECS) is designed for this purpose when the data result from nuclear stripping or pickup reactions. It computes Plane Wave Born Approximation (PWBA) cross sections, that is, the strength of a reaction as a function of the angle between the outgoing particle's direction and that of the incident beam. The PWBA is a particularly simple approximation of the actual reaction but it predicts remarkably well the significant feature of the cross sections: the location of the peaks. DIRECS provides for the display of read-in empirical data along with the calculated cross section. Typical data are shown in Figure 9 as the large dots. These data represent the cross section for deuterons leaving the .229 MeV state in $^{26}$Al. The peaks are the characteristic feature of direct reactions; their location in PWBA depends on the orbital angular momentum ($\ell$) transferred and a fitting parameter (R) which corresponds loosely to the nuclear radius. Along with the data, Figure 9 shows a theoretical calculation. This calculation has $\ell = 1$ and $R = 2.8$ Fermi. Figure 10 shows another calculation with $\ell = 2$ and $R = 5.2$ Fermi. The object is to choose the correct value of $\ell$ for the reaction. An automatic search would choose $\ell = 1$ as resulting in the best fit. However, more physical insight than just curve fitting is involved in the selection process because in this approximation the relative magnitude of succeeding peaks is not correct. Also, the depths of the minima are exaggerated in the theoretical curve. However, the location of successive minima and maxima are correctly predicted. On this basis we may choose unambiguously $\ell = 2$.

Appendix 4 is an excerpt of the DIRECS source program containing the real-time display and control statements. This section is considerably more complex than that in GRASP. The arrays here are in floating point mode. The sense switches of the IDIOT box are tested before a display is executed. FXYSCOPE (statement 11) provides for the experimental data display. The points are generated as a Y vector plotted against an X vector. The length of both vectors is NUMBER, and the order of the data points in the vectors is immaterial.

The subroutine FPTPLOT below statement 30 provides for output on the X-Y plotter. If its arguments are the same as those of FPTSCOPE, the same output will be produced.

## Discussion

In conclusion we feel that the only disadvantage to this system has been that our data storage arrays are somewhat limited in size, but in practice this has meant only that the operator has to analyze his data in sequential segments rather than all at once. Generally, a small data array is even desirable so that detail may be visible in the display.

The primary advantage of the system is that our installation is open shop. This is vital because out of 24 users, one third have doctorates, the rest are Ph. D. degree candidates and all are actively engaged in research. We have no systems engineers or programmers. Everyone has a different research problem to solve and each needs his own special considerations incorporated into the program he uses. A FORTRAN-based system makes it possible for a user to easily modify an existing program, adopt a program from the outside world, or write his own in the shortest possible time. Also, ideas for system improvements are continually generated when all of its users are involved in programming.

Another advantage is the ease with which our programs may be adapted to other installations. This helps not only the user who leave our lab and wishes to run his program elsewhere, but also the user who desires a larger processor for increased speed and storage when real-time processing is no longer required. This occurs occassionaly when a researcher, after perhaps months of real-time work, is able to develop the appropriate algorithm to process his data authomatically. In such a case, all that is required is that he delete the calls to the display routines and substitute the appropriate FORTRAN logic for the real-time controls.

All of the programs and subroutines discussed here are available from the DECUS library.


Acknowledgement

I would like to acknowledge that the success of our system has in large measure been due to Professor Philip Bevington, who conceived and wrote the FORTRAN-compatible system of display and control subroutines.


1 Philip R. Bevington, "On-line Reduction of Nuclear Physics Data with the PDP-7", Proceedings of the DECUS 1965 Fall Symposium at Stanford University.


**This research was supported in part under the National Science Foundation.**

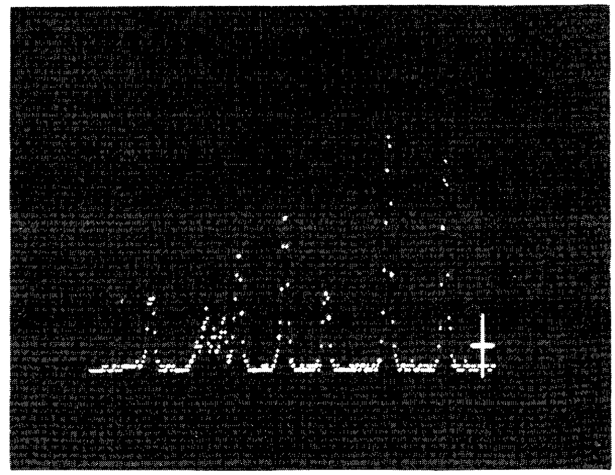



Figure 1

Oscilloscope display showing spectrum of deuteron groups from $^{31}P(p,d)$ nuclear reaction. Light pen cross is also visible.

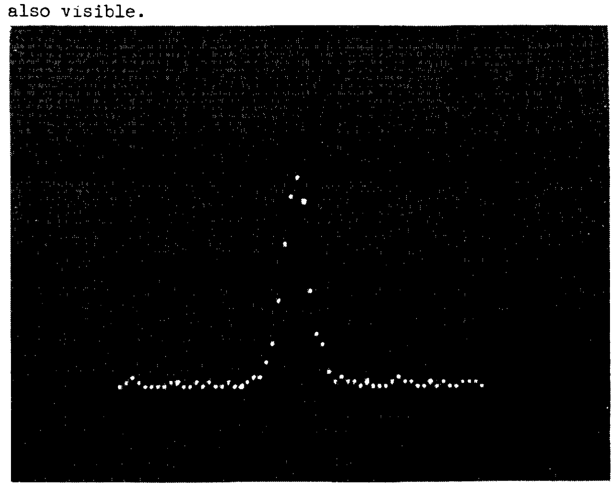


Figure 2

Selected portion of spectrum in Figure 1 showing rightmost peak.

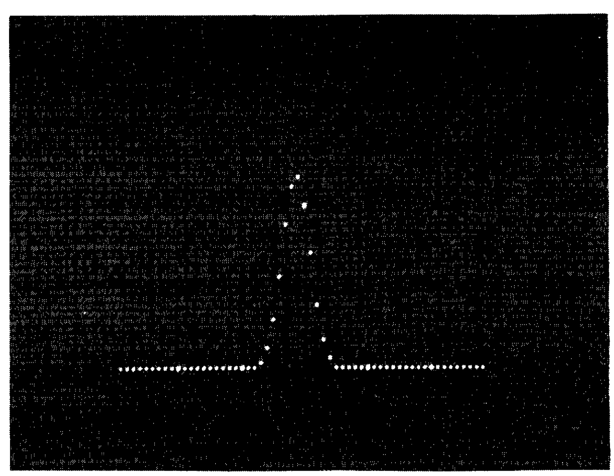


Figure 3

Calculated least-squares fit to peak appearing in Figure 2.

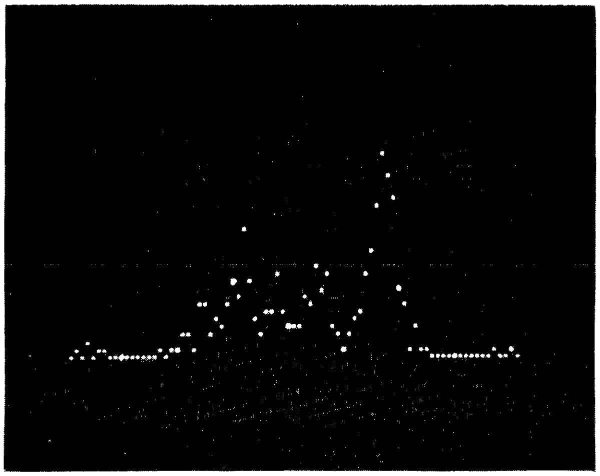


Figure 4

Selected portion of spectrum in Figure 1 showing five unresolved deuteron groups.

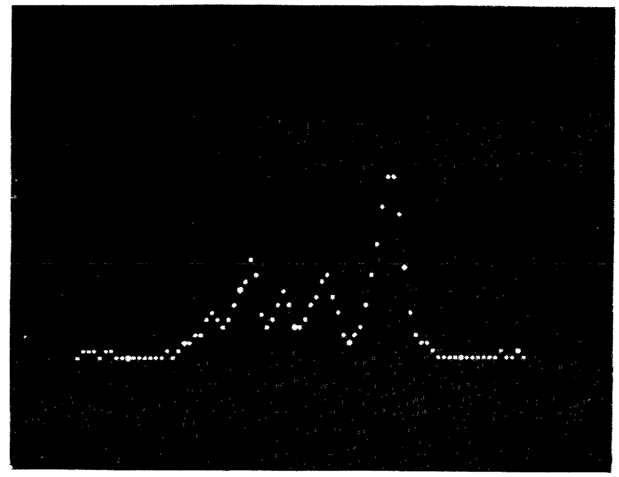


Figure 5

The data of Figure 4 after smoothing process.

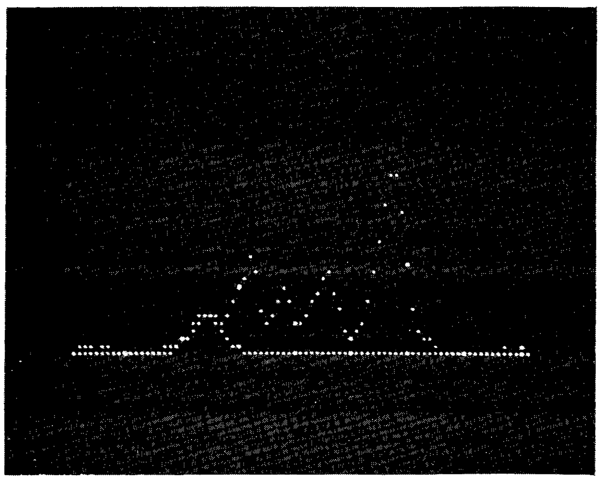


Figure 6

The data of Figure 5 with a least-squares fit to the left-most peak.

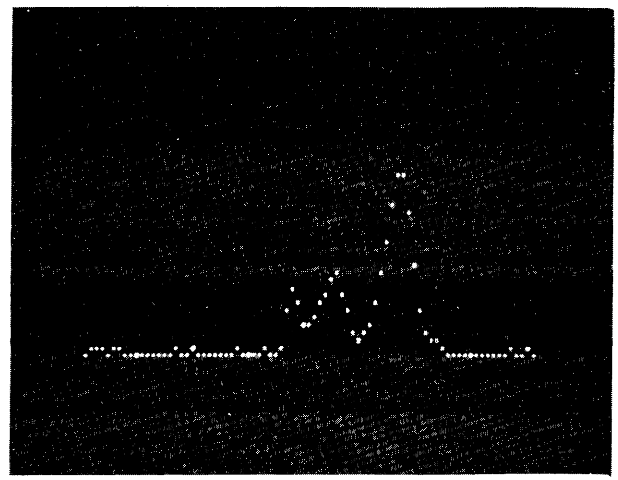


Figure 7

The data of Figure 5 after the fits to the first and second peaks have been subtracted away.

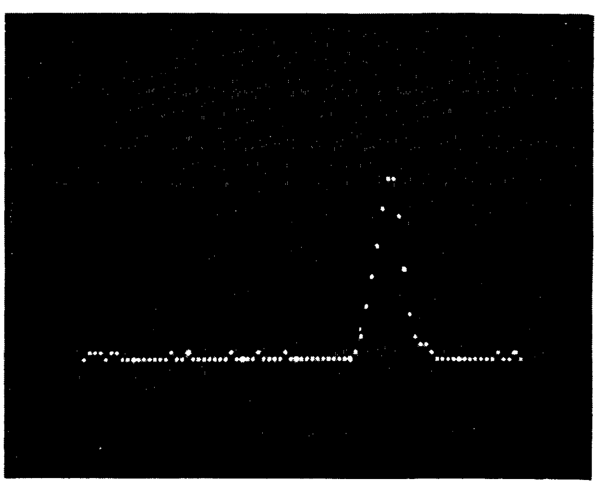


Figure 8

The data of Figure 7 after the fits to the third and fourth peaks have been subtracted away.

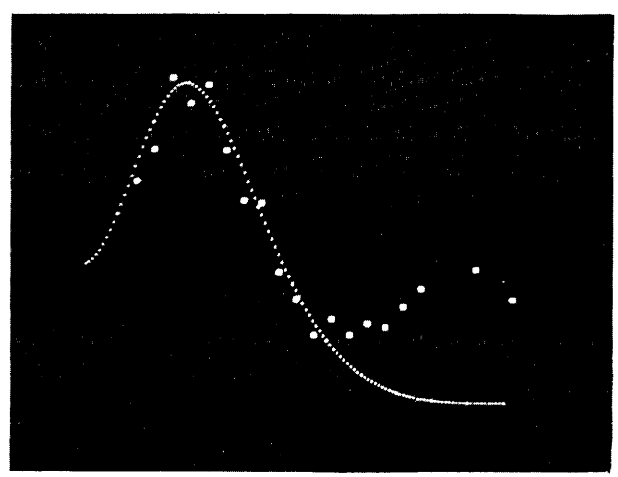


Figure 9

Experimental cross section data for the .229 MeV $^{26}Al$ energy level (heavy dots) with a theoretical computation using $\ell = 1$.

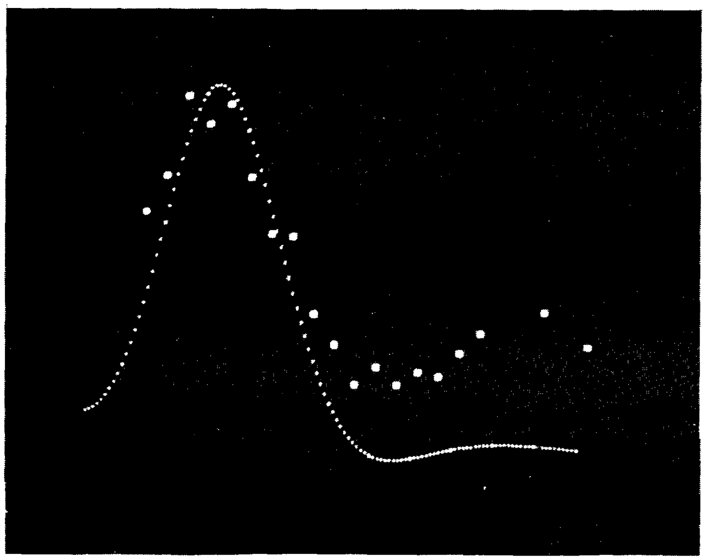


Figure 10 Theoretical computation with $\ell = 2$ compared with the data of Figure 9.

APPENDIX 1

FORTRAN COMPATIBLE SUBROUTINES
FOR REAL-TIME DATA ANALYSIS

| PURPOSE | FIXED ARGUMENTS | FLOATING ARGUMENTS |
|---|---|---|
| DISPLAY | PTSCOPE<br>PTPEN | FPTSCOPE<br>FPTPEN<br>FXYSCOPE |
| PLOTTING | CALIBRATE<br>PTPLOT | FPTPLOT |
| CONTROL | LISTEN<br>ISENSE FUNCTION | |
| INPUT | INPUT ND 180<br>INPUT VICTOREEN<br>DUMP ND 180 | INFLT ND 180<br>INFLT VICTOREEN<br>FDUMP ND 180 |
| OUTPUT | | OUTFLT |

UTILITY PROGRAMS

CREASE (CARD READER FOR EASE IN ASCII SOURCE EDITING)
ACTION (ANALYZER-COMPUTER TRANSFER OF I/O FOR NUCLEAR DATA)

APPENDIX 2

OPERATOR'S MAP FOR GRASP
GAUSSIAN REDUCTION AND ANALYSIS OF SPECTRUM PEAKS - 10/24/67

```
C  LISTEN LOOP 1   'I/O'
      GO TO LISTEN LOOP 2
0     DISPLAY STORED DATA AND LIGHT PEN
1     TYPE IN FIRST CHANNEL AND MATRIX SIZE (UP TO 512) FOR READING DATA
2     READ DATA FROM PAPER TAPE
3     SMOOTH DATA
4     PUNCH RESULTS ONTO PAPER TAPE
5     READ DATA FROM NEXT FILE ON DECTAPE
6     TYPE IN FILE NUMBER THEN READ DATA FROM DECTAPE
7     TYPE OUT CONTENTS AND NUMBER OF CHANNEL MARKED BY LIGHT PEN
8     CHANGE CONTENTS OF A CHANNEL
9     TYPE OUT CURRENT PEAK NUMBER AND BOUNDARIES 1-6


C  LISTEN LOOP 2   'PEAK'   (TYPE IN PEAK NUMBER)
      GO TO LISTEN LOOP 3
0     DISPLAY STORED DATA AND LIGHT PEN
1     SELECT LOWER BOUNDARY OF FITTING SPECTRUM)
2     SELECT UPPER BOUNDARY OF LOWER BACKGROUND)
3     SELECT LOWER BOUNDARY OF PEAK            )FITTING SPECTRUM MUST BE
4     SELECT UPPER BOUNDARY OF PEAK            ) LESS THAN 115 CHANNELS
5     SELECT LOWER BOUNDARY OF UPPER BACKGROUND)
6     SELECT UPPER BOUNDARY OF FITTING SPECTRUM)
7,I   TYPE IN BOUNDARY I (I = 1,6)
8,I   TYPE IN VALUE OF PARAMETER P(I)  (I = 1,6)
9,I   TYPE IN INCREMENTAL STEP FOR PARAMETER P(I)  (I = 1,3)

      FITTING PARAMETERS ARE
      P(1) PEAK AREA
      P(2) PEAK POSITION
      P(3) PEAK WIDTH  (FWHM)
      P(4) BACKGROUND AVERAGE
      P(5) BACKGROUND SLOPE
      P(6) BACKGROUND CURVATURE


C  LISTEN LOOP 3  'FIT'  (SET SENSE SWITCHES 1-7 FOR VARYING PARAMETERS)
      GO TO LISTEN LOOP 1
0     DISPLAY FITTING SPECTRUM AND FIT
1     START FIT THEN DISPLAY
2     CONTINUE FIT AND TYPE OUT CHI SQUARE AND P(1)
3     TYPE OUT CHI SQUARE AND P(1) THROUGH P(6)
4     DISPLAY SPECTRUM ONLY
5     DISPLAY FIT ONLY
6     DISPLAY BOTH SPECTRUM AND FIT
7     SUBTRACT FIT FROM SPECTRUM THEN DISPLAY RESULT
8     ADD FIT TO SPECTRUM THEN DISPLAY RESULT
9     GO TO LISTEN LOOP 2
```

APPENDIX 3

```
GRASP  10/24/67
C      GAUSSIAN REDUCTION AND ANALYSIS OF SPECTRUM PEAKS
C      ALBERT ANDERSON, STANFORD UNIVERSITY
       2WORD
       DIMENSION IDATA(513), NAME(15), P(6), DELIBE(3), DELSAV(3),
$       PEAK(115),IDSEL(115),IFIT(115),LTPEN(10,6)
       COMMON IDSEL, IFIT, PEAK, NBKD, LPCON, LPSEL2, LPSEL3, LPSEL4,
$       LPSEL5, LPSEL6, CHI, P
       LT = 1
       NFILE = 0
       READ 3005
       WRITE 2, 1
1      FORMAT (/6H GRASP)
2      FORMAT (I10)
3      FORMAT (F10.0)
100    WRITE 2, 101
101    FORMAT (/4H I/O)
102    MAXY = ISENSE(-0)
       CALL PTSCOPE (NUMBER,0.,1.,.1,MAXY,IDATA)
       CALL PTPEN (NUMBER,LTPENX,MAXY,LTPENY)
       CALL LISTEN(L)
       IF (L) 200, 102, 103
103    GO TO (110,120,130,140,150,160,170,180,190), L
```

APPENDIX

DIRECS 10/18/67
REAL TIME CONTROL AND DISPLAY SECTION

```
6      YMAX = SCALE * ISENSE(-0)
       CALL FPTSCOPE (NTHETA, 0.0, 1.0, 0.0, YMAX, SIGMA)
       IF (ISENSE(-1))  11, 200, 11
11     CALL FXYSCO (NPTS, X, 0., XMAX, Y, 0., 0.)
200    IF (ISENSE(-2))  202, 8, 202
202    CALL FPTSCOPE (NTHETA, 0.0, 1.0, 0.0, YMAX, SIGSAV)
8      CALL LISTEN (MOVE)
       IF (MOVE)  99,6, 1
1      JUMP = MOVE
       GO TO (91, 20, 30, 40, 50, 60, 70, 80, 90), JUMP
20     YMAX = SCALE * ISENSE(-0)
       CALL FPTPEN (NTHETA, LTPENX, YMAX, YLTPEN)
       CALL FPTSCOPE (NTHETA, 0.0, 1.0, 0.0, YMAX, SIGMA)
       CALL LISTEN (MOVE)
       IF (MOVE)  25, 20, 25
25     TEMP = LTPENX - 1
       THETAL = THETA0 + TEMP*DTHETA
       WRITE 2, 150, THETAL, LTPENX, SIGMA(LTPENX)
       GO TO 6
30     YMAX = SCALE * ISENSE(-0)
       CALL CALIBRATE (NTHETA)
       CALL FPTPLOT (NTHETA, 0.0, 1.0, 0.0, YMAX, SIGMA)
       GO TO 6
```

# A NON-LINEAR LEAST-SQUARES SEARCH ROUTINE FOR A SMALL COMPUTER

Thornton R. Fisher - Stanford University

Stanford, California

Abstract

The Newtonian method of non-linear regression is shown to provide a simple and effective technique for performing many types of least-squares searches, particularly when the analysis is performed in real time.

Least-squares fitting is a special case of the more general problem of finding the minimum (or maximum) value of a function. If $\Phi(x_1, \ldots x_m)$ is a function of the $m$ variables $x_j, \ldots x_m$, the condition for a minimum is expressed by the set of equations

$$\frac{\partial \Phi}{\partial x_j} = 0, \quad j = 1, \ldots m \qquad (1)$$

When these equations are linear in the variables $x_j$, they may be solved directly. Otherwise, techniques of non-linear regression must generally be employed. In these techniques, the minimum value of the function is reached by a search or a series of successive approximations.

Techniques of non-linear regression fall into two basic categories: the gradient method and the Newtonian method. In the gradient method, one calculates the first partial derivatives of $\Phi$ and moves in the direction in which $\Phi$ is decreasing most rapidly. There are numerous prescriptions for deciding how large a step to take before recalculating a new direction, but even in its most refined forms the gradient method may converge very slowly. In the Newtonian method, $\Phi$ is expanded in a second order Taylor series about the starting point $\Phi_o$

$$\Phi = \Phi_o + \sum_{j=1}^{m} \frac{\partial \Phi}{\partial x_j}\bigg|_{\Phi=\Phi_o} \Delta_j + \frac{1}{2} \sum_{j=1}^{m} \sum_{k=1}^{m} \frac{\partial^2 \Phi}{\partial x_j \partial x_k}\bigg|_{\Phi=\Phi_o} \Delta_j \Delta_k \qquad (2)$$

Setting the partial derivative of $\Phi$ with respect to each parameter equal to zero leads to the set of equations

$$\left.\begin{aligned} &P2 \cdot \Delta = -P1 \\ &P2(J,K) = \frac{\partial^2 \Phi}{\partial x_j \partial x_k}\bigg|_{\Phi=\Phi_o} \\ &P1(J) = \frac{\partial \Phi}{\partial x_j}\bigg|_{\Phi=\Phi_o} \\ &\Delta(K) = \Delta_k \end{aligned}\right\} \qquad (3)$$

which can be solved for the $\Delta_k$'s. The variables $x_j$ are incremented by the corresponding $\Delta_j$'s and the process repeated until convergence is attained. The Newtonian method converges extremely rapidly in the region close to the minimum in $\Phi$ since equation (2) is a very good approximation in this region. It would appear to be superior to the gradient technique for this reason, but has not received wide application since the time required to compute all the second partial derivatives of $\Phi$ is often excessive. However, in performing least-squares fits to physical data, one encounters a large number of cases in which the Newtonian method is quite feasible. The present paper illustrates how a least-squares search utilizing the Newtonian method can be programmed for a small computer such as the PDP-7 or PDP-9. Full advantage is taken of the "real time" controls available to the operator to reduce the actual computation time involved.

In a least-squares fit, one obtains the set of parameters which give the optimum fit between a theoretical function and a set of data points by minimizing $\chi^2$ where $\chi^2$ is defined by

$$\chi^2 = \sum_{i=1}^{n} \frac{(f_i(x_1, \ldots x_m) - N_i)^2}{\sigma_i^2} \quad (4)$$

In equation (4) $f_i$ is a function of m parameters $x_j$ and $N_i$ is the set of data points with associated standard deviations $\sigma_i$. We apply the Newtonian method to the problem by calculating the partial derivatives of $\chi^2$ up to second order and solving equation (3) to obtain new estimates of the parameters.

To calculate the partial derivatives of $\chi^2$ we choose a set of increments $\delta_j$ and calculate an array of values of $\chi^{2j}$ as follows:

(5)

$$\mathrm{CHI}(J,K,1) = \begin{cases} \chi^2(x_j - \delta_j,\ x_k - \delta_k),\ j \neq k \\ \chi^2(x_j - 2\delta_j),\ j=k \end{cases}$$

$$\mathrm{CHI}(J,K,2) = \begin{cases} \chi^2(x_j - \delta_j,\ x_k + \delta_k),\ j \neq k \\ \chi^2(x_j),\ j=k \end{cases}$$

$$\mathrm{CHI}(J,K,3) = \begin{cases} \chi^2(x_j + \delta_j,\ x_k + \delta_k),\ j \neq k \\ \chi^2(x_j + 2\delta_j),\ j=k \end{cases}$$

The CHI array is dimensioned (m,m,3) and has the following symmetries which may be utilized to reduce the time required per iteration

$$\begin{aligned} \mathrm{CHI}(J,K,1) &= \mathrm{CHI}(K,J,1) \\ \mathrm{CHI}(J,K,3) &= \mathrm{CHI}(K,J,3) \\ \mathrm{CHI}(J,J,2) &= \mathrm{CHI}(K,K,2),\ \text{etc.} \end{aligned} \quad (6)$$

The expressions for the partial derivatives are

$$\mathrm{P2}(J,K) = \frac{\mathrm{CHI}(J,K,3)-\mathrm{CHI}(J,K,2)-\mathrm{CHI}(K,J,2)+\mathrm{CHI}(J,K,1)}{4\delta_j\delta_k}$$

$$\mathrm{P1}(J) = \frac{\mathrm{CHI}(J,J,3) - \mathrm{CHI}(J,J,1)}{4\delta_j} \quad (7)$$

If the expressions in (7) are to represent good approximations to the true partial derivatives, the choice of the $\delta_j$'s is not completely arbitrary. However, in practice this choice is not at all critical. Once the P2 and P1 arrays have been calculated, equation (3) may be solved by the application of a simple Gauss-Jordan reduction. The application of sophisticated matrix inversion techniques is unnecessary since the equations to be solved are approximate in any case. The jth parameter may be omitted from the search by setting

$$\begin{aligned} \mathrm{P2}(J,K) &= \begin{cases} 0, & K \neq J \\ 1, & K=J \end{cases} \\ \mathrm{P1}(J) &= 0 \end{aligned} \quad (8)$$

before solving equation (3). When the search has converged, the standard deviations in the final values of the parameters are given to a good approximation by

$$\sigma_j^2 = 2\ (\mathrm{P2}^{(-1)})_{jj} \quad (9)$$

where $\mathrm{P2}^{(-1)}$ is the inverse of the P2 matrix.

A listing for a subroutine which utilizes the preceding expressions is included at the end of the paper. The significance of the variables and arrays is as follows:

| | |
|---|---|
| CTS | array of data ($N_i$) |
| TH | theoretical function ($f_i$) |
| Q,X | array of parameters ($x_1$, .. $x_m$) |
| D | array of increments for computing partial derivatives ($\delta_1$, .. $\delta_m$) |
| CHI | chi square array |
| P1 | array of first partial derivatives |
| P2 | array of second partial derivatives |
| DELTA | computed changes in parameters |
| KL | number of parameters |
| IMIN, IMAX | initial and final members of CTS array which define the region to be fitted |
| DELM | limits maximum step per iteration |

A listing for the matrix inversion subroutine, SOLVE, is also included. Total storage required for both subroutines is 1500 octal. The subroutine SESAME performs one iteration of the search and writes out the new value, the change computed, and three values of the CHI array for each parameter. When $CHI(I,I,1) \approx CHI(I,I,3) > CHI(I,I,2)$ the search has converged. The function ISENSE (I) is either 0 or 1 depending on whether the corresponding control switch is down or up. Therefore, if the jth control switch is up, the jth parameter is omitted from the search. If control switches 1,...m and -1 are all up, the program does not perform an actual iteration but computes and writes out the values of the CHI (I,J,K) array for which I = J. It is thus possible to tell whether the minimum has been reached without taking the time to perform a full additional iteration.

Two features have been included in an attempt to prevent the search from diverging if the starting point is at too great a distance from the true minimum. The parameter DELM limits the maximum step which can be taken in one iteration. Also, the diagonal elements of the P2 matrix are arbitrarily forced to be positive. When one of these elements is negative, the curvature of the $\chi^2$ surface is negative with respect to the corresponding parameter, and the program will move in a direction of increasing $\chi^2$. Forcing the curvature to be positive hopefully causes the program to move in the right direction, although the estimates computed for the changes in the parameters may be expected to be poor.

As has been pointed out, the convergence of the search is very rapid if the starting point is close to the true minimum, but may be very bad if the starting point is at too great a distance from the minimum. When the analysis is being performed in real time, the operator can speed the progress of the search considerably by inspecting the quality of the fit frequently. This often enables him to make obvious adjustments in the values of the parameters, or to omit certain parameters from the first few iterations.

Figure 1 illustrates the application of the program to the problem of fitting the line-shapes of gamma-rays observed with a Ge(Li) detector. The characteristic line-shapes result from the Doppler effect, and differ depending upon whether the nuclei emitting the gamma ray recoil into vacuum or into a dense medium. The fits to the vacuum line-shapes involve eight free parameters. The fits to the stopped line-shapes involve four free parameters, of which one is the lifetime of the nuclei emitting the gamma rays. The principal object of the fitting procedure is to extract the best estimate for the lifetime parameter.

The program has also been successfully applied to the problem of fitting a neutron spectrum taken with a stilbene detector with a Fermi shape, and to the problem of fitting two interfering Gaussian peaks superimposed on a background. Its potential range of application appears quite large. We conclude that the Newtonian method of nonlinear regression may be applied quite successfully to the problems of least-squares fitting provided: 1) the theoretical function is sufficiently simple that the time required to compute all partial derivatives up to second order is not excessive, and 2) the operator has the benefit of real time analysis.

**This research was supported in part under the National Science Foundation.**

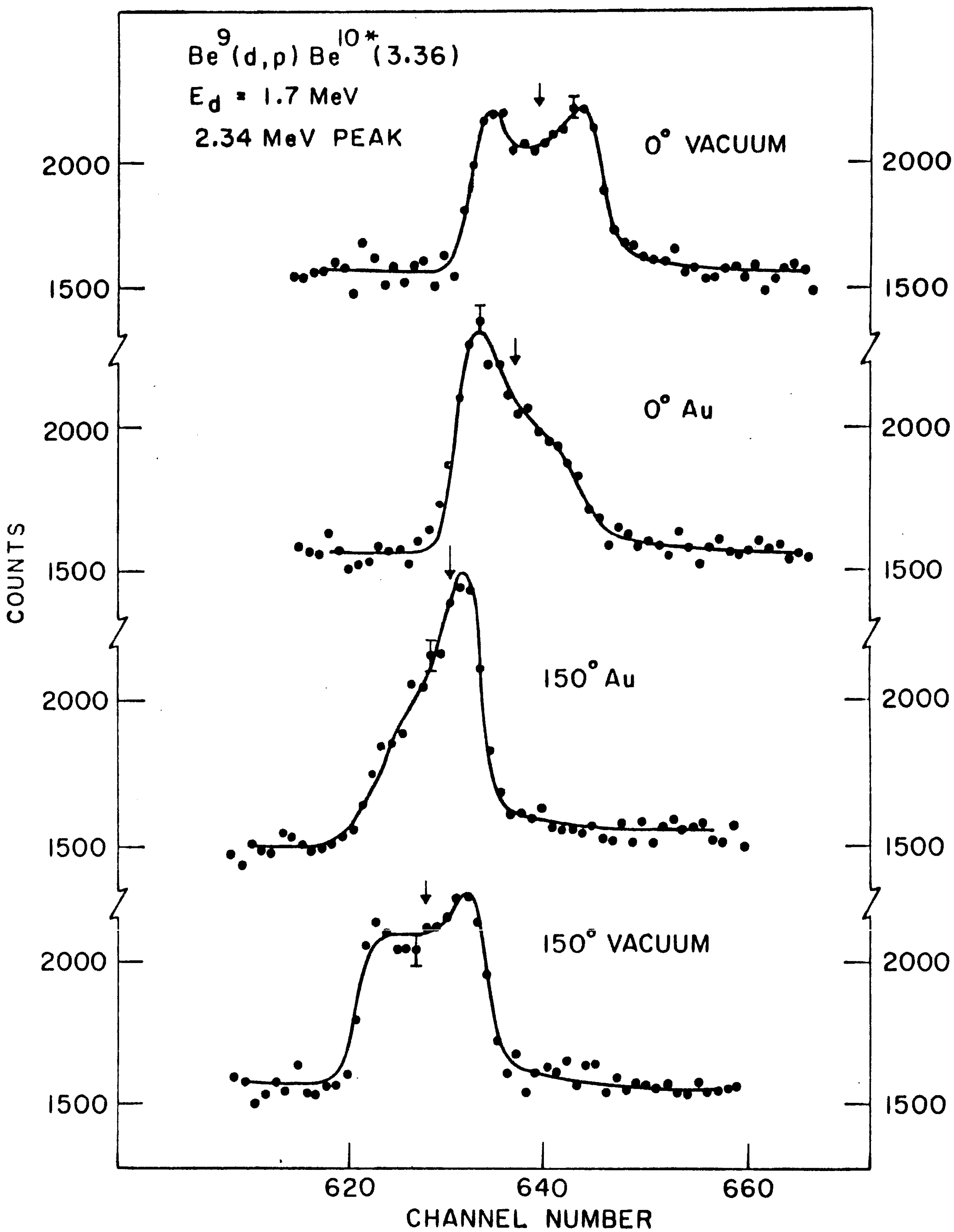



FIGURE CAPTIONS

Figure 1 Line-shape fits to $Be^{10*}$(3.36 MeV) Gamma Rays Observed with Ge(Li) Detector

APPENDIX I

SPECTRUM EVALUATION BY A SEARCH ANALYSIS WITH MAXIMUM EFFICIENCY

```
      SUBROUTINE SESAME (CTS,D,Q,X,CHI,P1,P2,DELTA,TH,KL,DELM)
      2WORD
      DIMENSION CTS,D,Q,X,CHI,P1,P2,DELTA,TH
      COMMON IMIN,IMAX
C     COMPUTE CHI SQUARE ARRAY
305   DO 312 I=1,KL
312   X(I)=Q(I)
      DO 410 I=1,KL
      DO 400 J=1,KL
      DO 380 K=1,3
      CHI(I,J,K)=0.
C     CHECK TO SEE IF ITH OR JTH PARAMETER IS OMITTED FROM SEARCH
      IF (ISENSE(I)) 1050,1050,1070
1050  IF (ISENSE(J)) 315,315,490
1070  IF (J-I) 490,1080,490
1080  IF (ISENSE(-1)) 455,455,315
315   IF (K-2) 460,450,460
450   IF (J-I) 480,455,480
455   IF (I-1) 320,320,490
460   X(I)=X(I)+(K-2)*D(I)
470   X(J)=X(J)+(K-2)*D(J)
      IF (J-I) 500,320,320
480   X(I)=Q(I)-D(I)
      X(J)=Q(J)+D(J)
320   CALL THEORY (TH,X)
      DO 370 L=IMIN,IMAX
      M=L-IMIN+1
      CHI(I,J,K)=CHI(I,J,K)+(CTS(L)-TH(M))*(CTS(L)-TH(M))/CTS(L)
370   CONTINUE
      GO TO 510
490   CHI(I,J,K)=CHI(1,1,2)
      GO TO 510
500   CHI(I,J,K)=CHI(J,I,K)
510   X(I)=Q(I)
      X(J)=Q(J)
380   CONTINUE
400   CONTINUE
410   CONTINUE
      IF (ISENSE(-1)) 560,560,650
C     COMPUTE FIRST AND SECOND PARTIAL DERIVATIVES
560   DO 620 I=1,KL
      P1(I)=(CHI(I,I,3)-CHI(I,I,1))/4.
      DO 615 J=1,KL
      P2(I,J)=(CHI(I,J,3)-CHI(I,J,2)-CHI(J,I,2)+CHI(I,J,1))/4.
615   CONTINUE
      P2(I,I)=ABSF(P2(I,I))
      IF (ISENSE(I)) 620,620,580
580   P2(I,I)=1.
```

```
620     CONTINUE
C       SOLVE FOR CHANGES IN PARAMETERS
        CALL SOLVE(P1,P2,DELTA,KL)
C       CHECK TO SEE IF MAXIMUM ALLOWABLE STEP EXCEEDED
        SNORM=0.
        DO 630 I=1,KL
        SNORM=SNORM+DELTA(I)*DELTA(I)
        DELTA(I)=DELTA(I)*D(I)
630     CONTINUE
        SNORM=SQRTF(SNORM)
        IF (SNORM-DELM) 640,640,637
C       IF CALCULATED STEP EXCEEDS MAXIMUM ALLOWABLE STEP TAKE
C       MAXIMUM ALLOWABLE STEP IN SAME DIRECTION
637     DO 640 I=1,KL
        DELTA(I)=DELTA(I)*DELM/SNORM
640     CONTINUE
        DO 740 I=1,KL
        Q(I)=Q(I)+DELTA(I)
740     CONTINUE
        IF (ISENSE(-5)) 680,680,650
650     WRITE 2,660
660     FORMAT(3X,9HPARAMETER,6X,6HCHANGE,8X,4HCHI1,8X,4HCHI2,
S       8X,4HCHI3//)
        DO 680 I=1,KL
        WRITE 2,70, Q(I),DELTA(I),CHI(I,I,1),CHI(I,I,2),CHI(I,I,3)
70      FORMAT (5F12.3)
680     CONTINUE
        RETURN
        END
```

## APPENDIX II

```
SUBROUTINE SOLVE
        SUBROUTINE SOLVE (P1,P2,DELTA,KL)
        2WORD
        DIMENSION P1,P2,DELTA
        DO 300 I=1,KL
        EPIV=P2(I,I)
        P1(I)=P1(I)/EPIV
        DO 100 J=1,KL
        P2(I,J)=P2(I,J)/EPIV
100     CONTINUE
        DO 250 K=1,KL
        IF (K-I)200,250,200
200     ELIM=P2(K,I)
        P1(K)=P1(K)-P1(I)*ELIM
        DO 150 L=1,KL
        P2(K,L)=P2(K,L)-P2(I,L)*ELIM
150     CONTINUE
250     CONTINUE
300     CONTINUE
        DO 400 I=1,KL
        DELTA(I)=-P1(I)
400     CONTINUE
        RETURN
        END
```

# USE OF A PDP-9 FOR REAL TIME OFF LINE ANALYSIS OF SPECTRA FROM AN AERIAL SURVEY FOR RADIOACTIVE MINERALS

C. J. Thompson

Atomic Energy of Canada Limited,
Commercial Products,
Ottawa, Canada.

## ABSTRACT

A PDP-9 is being used here to process data in real time from an airborne gamma ray spectrometer being used to detect deposits of radioactive minerals. A pulse height to width converter in the helicopter records pulse widths proportional to gamma ray energies on magnetic tape. The pulse widths are digitized into 1024 channels by an interface to the PDP-9. A program examines the spectrum as a function of real time and records the ratios of the counts due uranium and thorium to natural potassium. The updated results are displayed using an interface designed to make full use of a variable persistence oscilloscope.

## INTRODUCTION

During the past six months a system has been developed by Atomic Energy of Canada Limited, Commercial Products, for real time off-line analysis of gamma ray spectra.

The system has been used in the development of an aerial surveying technique[1,2], for detecting naturally radioactive elements as an aid to geological mapping. The work was carried out for Geological Survey of Canada. A gamma ray spectrum is recorded on magnetic tape while flying, in a helicopter, over regions where the activities of potassium, uranium and thorium had been measured on the ground. The tape is later played into a PDP-9 computer which analyzes the spectrum in real time. The system has been field tested near Bancroft and Elliot Lake, Ontario.

The project as a whole was aimed at finding what size detectors would be necessary and what accuracy of mapping one could achieve, by flying a "developed system" in an aircraft at about 120 m.p.h. The "developed system" will be a detector-spectrum analysis package designed to make regular flights in a fixed wing aircraft.

This paper deals with the trial system instrumentation in the helicopter, and the subsequent analysis of data by the PDP-9 which together form the real time, off-line data reduction system.

## THE DATA ACQUISITION SYSTEM

The trial system was required to measure the count-rate in the gamma ray photopeaks associated with the radioactive decay of potassium, uranium and thorium. These count-rates were to be corrected for Compton scattering from gamma rays of higher energy, the height of the helicopter above the ground, and the natural background activity from cosmic rays, etc. The count-rates were to be measured as a function of distance travelled by the helicopter. This could then be correlated with the helicopter position using aerial photography.

The trial system used three 5" x 5" NaI (Tl) crystals with 5" photomultiplier tubes, as gamma ray detectors. The signals from these were added in a summing amplifier. The next step in most spectrum analysis of this kind, is to sort out the various gamma ray energies with a multi-channel analyzer. In this case, however, since it is the changes in photopeak count-rate which are to be measured, a multi-channel analyzer is unsuitable.

The data acquisition system which we used was developed around a Sanborn type 3917B eight-channel analogue magnetic tape recorder, and a PDP-9 computer, both of which were already in use in our laboratories. Because of the cost of the helicopter flights it was decided to record, on magnetic tape, each pulse from the detector, live in the helicopter. The tape could then be played back into the computer and the spectrum analysis done off-line. By recording each pulse live it was possible to preserve a "real time" system so that count-rates could be measured from the tape with reference to the computer's clock.

Since the recorder's bandwidth is only 300 KHz and the pulses from the sodium iodide detectors have frequency components much higher than this, the pulses could not be recorded directly on magnetic tape. To overcome this difficulty the ramp type pulse height encoder, which was to digitize the pulses for presentation to the computer, was split into two sections. The section containing the linear gate, pulse stretcher, ramp generator, and control logic was flown with the detectors in the helicopter. The output from this section is a pulse width, whose duration is proportional to the energy of the gamma ray and this pulse width is recorded on magnetic tape in such a way as to saturate it. In this way excellent linearity is preserved and the effective bandwidth of the recorder is increased. This pulse width, at playback time, is used to gate on a crystal clock in the second section of the encoder. This section is interfaced to the computer, and it is the number of "ticks" of the clock which determine the channel into which the incident gamma ray is encoded.

PULSE HEIGHT ENCODER

The pulse height encoder used in this project is a development of one built a year previously for use with a four window-integrating analyzer. Its use in this project was made possible by modifying it to work in two sections, and interfacing it to the PDP-9. The two sections work equally well together, and are used in our laboratory for routine multi-channel analysis. The encoder has 1024 channels and digitizing rates of 2 or 10 MHz. In the pulse height to time section of the encoder [4], the incoming pulse is used to charge up a capacitor to the potential of the pulse peak. This capacitor is then discharged by a constant current, so that the potential difference across it is a linear ramp. The current is drained from the capacitor until a reference potential is reached. A flip-flop in the output circuit is kept on only during the period of the ramp. The output of this flip-flop is a pulse width proportional to the original pulse height, and is recorded on the tape. This section also contains upper and lower level discriminators. During the conversion a linear gate is closed to prevent pulses entering. This is re-opened 25 μsec. after the ramp is finished, or by the computer if both sections are working together.

The second stage of the encoder consists of the digitizing section and interface to the PDP-9. It takes pulse widths either directly from the first stage or from tape, and converts them to binary numbers. The pulse width is used to gate on a 10 MHz crystal oscillator. Ten MHz is the normal digitizing rate. The bandwidth of the tape recorder limits the minimum width of the pulse to be digitized to 10 μsec. For this reason the analysis time is lengthened and a 2 MHz digitizing rate is used for taped spectra. The number of clock pulses is divided by five, using part of a divide by ten TTL integrated circuit. The gating is done on the 10 MHz signal to help synchronize the oscillator and the gating pulse.

The pulse widths are digitized by counting the gated clock pulses with a 10 bit ripple counter, consisting of five dual JK TTL integrated circuit flip-flops. The trailing edge of the gating pulse sets the "program interrupt" flag in the encoder. The PDP-9 issues a "read the encoder buffer" IOT command which puts the digital representation of the pulse, together with some high order bits specifying the absolute memory address to be incremented, into the accumulator. The ripple counter is cleared and the linear gate opened, if possible automatically, after this IOT and the encoder is ready to accept the next pulse.

The digitizing section, which also includes a live time clock, and the interface to the computer, is built on one 8" by 4" circuit board. The analogue section is on two similar size boards. The encoder is built into a 4-width standard "Nuclear Instrument Module".

## DISPLAYING OF RESULTS

Another interface to the PDP-9 which has been very useful in this project is the CPD 233 "Spectrum Display Unit". This display, constructed almost entirely from integrated circuits was built especially for displaying multi-channel analysis spectra. The program written for its operation selects both the X and Y gain settings so that the region of the spectrum specified by the operator fills the entire CRT screen. When displaying spectra live from the encoder the display auto-ranges to a lower gain setting lest the peak of the spectrum go over the top of the screen. The program provides for linear or logarithmic displays. The display operates with a Hewlett-Packard 141A variable persistence oscilloscope. The persistence is varied to prevent the display flickering when displaying large numbers of channels. The screen is also erased under program control.

A block diagram of the display is shown in Figure 4. The Y co-ordinate is set by the computer by loading a 10 bit buffer register consisting of three Fairchild "959" integrated circuits. These and a Fairchild μA709 operational amplifier together with a resistor network form the 10 bit D/A converter for the Y axis. The X co-ordinate is set by another μA709 used as an integrator. After each point is displayed several "increment X axis" instructions in the computer move the spot to the next point, (the exact number is determined by the criterion that the display should fill the screen).

As an aid to examining features in the spectrum it is possible to identify any point in the displayed spectrum by means of the "halo generator". This turns any point on the spectrum into a small circle of light. With the point thus marked, the operator pushes the "push to locate" button on the interface. This connects the "halo" to the program interrupt facility and causes the teletype to identify the channel and give the number of counts it contains. Its use is similar to that of a "light pen" but it is rather more convenient to use. The method of operation is as follows. A Fairchild μA710 integrated comparator compares the X co-ordinate currently being displayed with the potential of the slide of the front panel potentiometer labelled "X position". When they are equal the channel is defined. This gates on a 1 MHz oscillator which is applied via a pulse splitting network to summing amplifiers for the X and Y axes. A Lissajous's figure, in this case a circle centred on the point to be identified, is displayed on the screen.

The entire display unit is constructed on three 6" x 4" printed circuit boards which are housed in a 2 width "Nuclear Instrument Module". (Figure 3)

The display unit will also drive an X-Y plotter. The point plotting speed is set by an (ISZ , JMP .-1) wait loop in the computer, giving a plotting speed of about 7 points in 10 seconds. This has been used both to plot and display most of the results obtained.

## PROGRAM FOR DATA REDUCTION

Prior to this project we were already using the PDP-9 for acquiring gamma ray spectra from both germanium and sodium iodide detectors. We use the CPD 229 Pulse Height Encoder/ Interface to bring the spectra into the computer. The multi-channel analysis program, "Spectrum" we use is similar in many respects to the DEC multi-analyser[5] program, but was written before the DEC program was available to us. A sub-routine, called "DISCOVER", was added to the main program to allow the spectra from the aerial survey to be examined as a function of time. Since every attempt was made to fly the helicopter at constant speed, we were able to obtain profiles of activity from potassium, uranium and thorium as a function of distance flown by the helicopter.

To use the "DISCOVER" program the following operations must be performed. To correct the spectrum for background gamma radiation due to cosmic rays etc., a background spectrum is read into one of the four storage areas. This spectrum was recorded while hovering in the helicopter at about 500 feet over a large lake. Each run over a test strip was timed accurately, and this time, and the number of intervals into which it must be divided to get the count-rate measured at 500 foot intervals along the strip, is typed into the computer. It is also necessary to type in the data area into which the spectrum will be placed by the encoder.

The CPD 233 display can then be used either to display any of the count-rates as a function of distance, or more commonly to help plot the height of the helicopter above the ground. This is done by feeding the output of the radio-altimeter onto the Y axis of an X-Y plotter. The X axis of the plotter is controlled by the computer, via the display interface, in such a way that the pen excursion just covers the available plotting space. It moves the same distance in plotting out the count-rate data, so that the horizontal scale is the same for all plots. The total distance moved is independent of both the number of intervals and the total time of flight over the test strip.

A flow chart of the "DISCOVER" program is shown in Figure 5. The most important steps in the program are described below. At a time corresponding to each 500 feet flown by the helicopter the real time clock in the PDP-9 causes a "program interrupt". This starts the program sequence which calculates the net counts in the potassium, uranium and thorium photopeaks since the last clock interrupt.

The sequence of events (Figure 5) which takes place, is as follows.

1. An appropriate fraction of the 5 minute "Background" spectrum is subtracted, channel for channel, from the spectrum coming into the computer from magnetic tape.

2. The counts in all channels defining photopeaks associated with gamma rays from potassium, uranium, and thorium are added together, and the value at the previous time is subtracted from this latest value.

3. Some of the counts so obtained are not due to gamma rays whose full energy is between the specified channel limits, but are due to Compton scattering of gamma rays of higher energy. To compensate for this, fractions of the counts in the high energy channels are subtracted from those in lower energy channels. These fractions, (stripping ratios) were calculated for the detector system used and are constants in the program. These stripping ratios are used by the program to give the net counts in the photopeak associated with full energy gamma rays. These counts are converted to counts per minute and the ratios uranium/ potassium, thorium/potassium and uranium/thorium are then calculated.

4. These results, together with the time after commencing the run are then typed out. In cases where there is not time to print out the results live they are stored and typed out at the completion of the run.

A typical set of results is shown in Figure 6. The printout gives the net count-rate in photo-peaks associated with the radioactive decay of potassium, uranium and thorium. In this analysis potassium is identified by the 1.46 MeV gamma ray from potassium 40. The presence of uranium and thorium is correlated with the presence of gamma rays from their daughter products[4], i.e. 1.76 MeV from bismuth 214 and 2.62 MeV from thallium 208.

The ratios of the count-rates are also calculated and included in Figure 6. The table represents a profile of the count-rates that are obtained by flying at 500 feet over a test strip. The fluctuations in the count-rates correspond to features of geological interest. These fluctuations are more pronounced when flying at low altitudes since the detector "sees" a greater area from higher altitudes. Figure 7 shows the count-rates from potassium and uranium as a function of distance along the test strip. This is an actual photograph taken from the CPD 233 display.

## REFERENCES


1. J. Berbezier, et al
Methods of Carborne & Airborne Prospecting
The Techniques of Radiation Detecting Prospects Offered by Energy Discrimination, 1958
(A/Conf 15/P/1249).

2. Thomas L. Boyle,
Airborne Radiometric Surveying Office and Field Techniques
U.S.A.E.C. RME-129 (Rev.).

3. P. Iredale,
The Gamma Ray Spectra of Uranium and Thorium in Equilibrium with Their Decay Products
United Kingdom Atomic Energy Authority, 1958
Report AERE EL/R 2696.

4. Quentin Bristow and C. J. Thompson,
A Computer P.H.A. System for Real Time Off Line Analysis of Spectra from an Aerial Survey for Radioactive Minerals.
IEEE Trans.Nuc.Sci. 1967
(to be published).

5. Digital Equipment Corp. (1967)
Multianalyzer Programs Instruction Manual (PDP-9)
Product Code DEC-9A-USAI-LA.

## GAMMA RAY SPECTROMETER - MOBILE SECTION

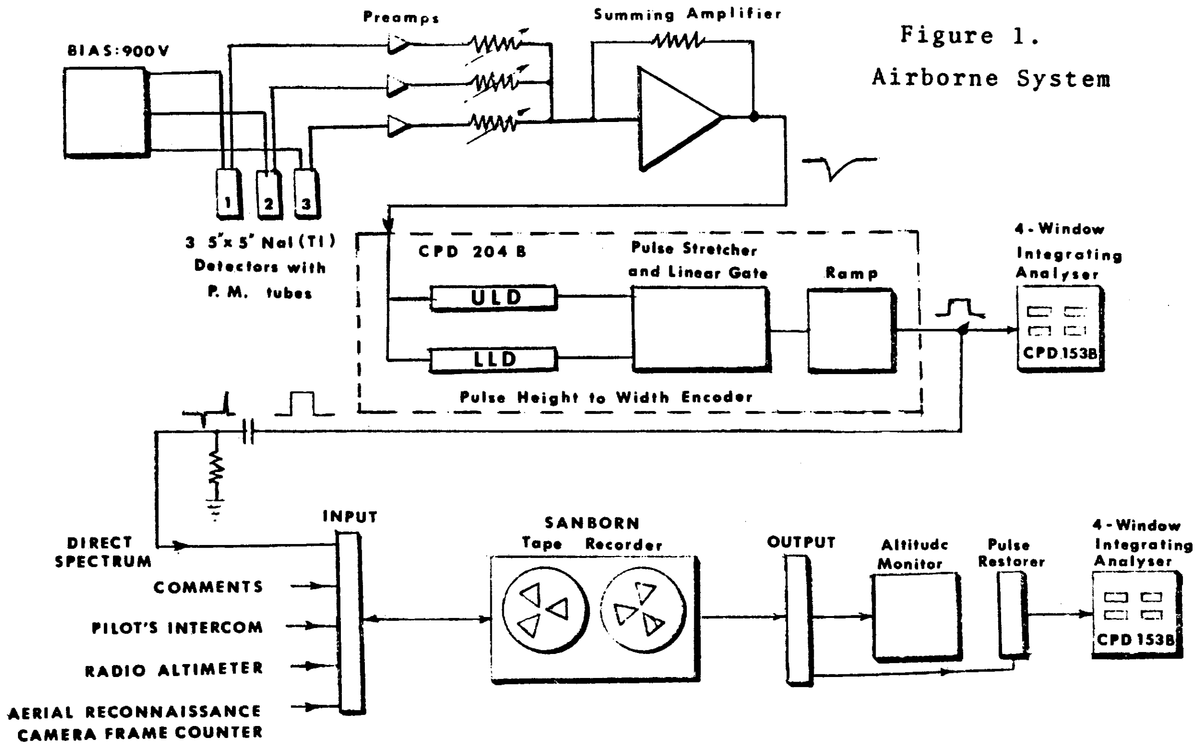



Figure 1.
Airborne System

## DATA RETRIEVING & PROCESSING INSTRUMENTATION

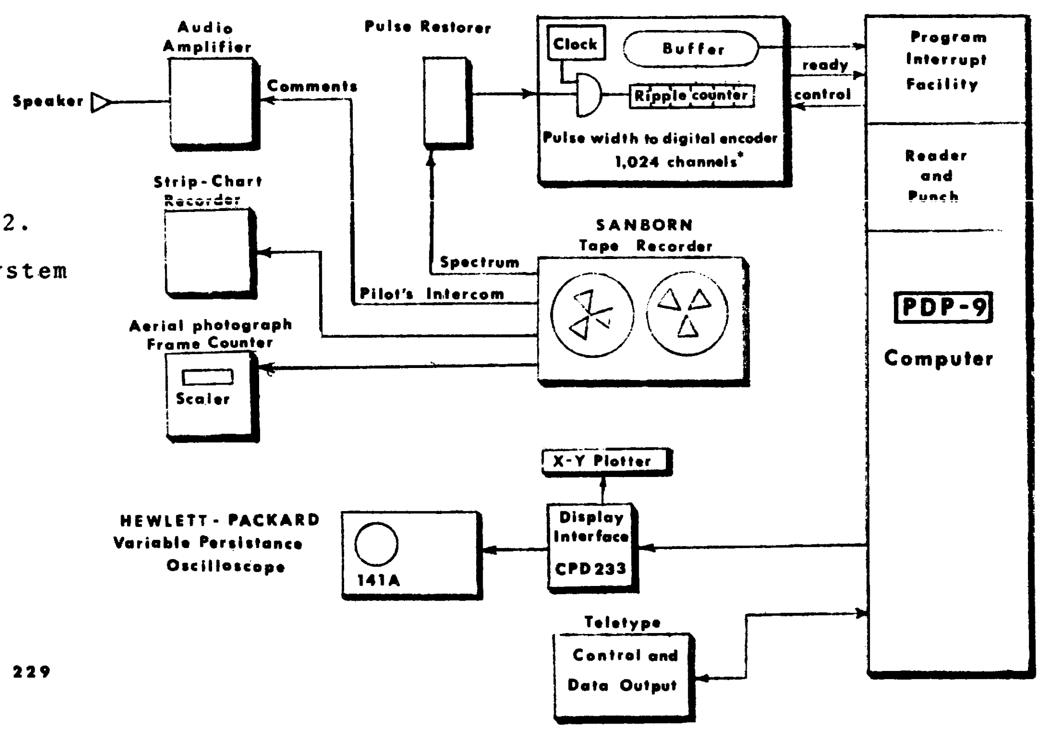



Figure 2.
Ground System

*CPD 229

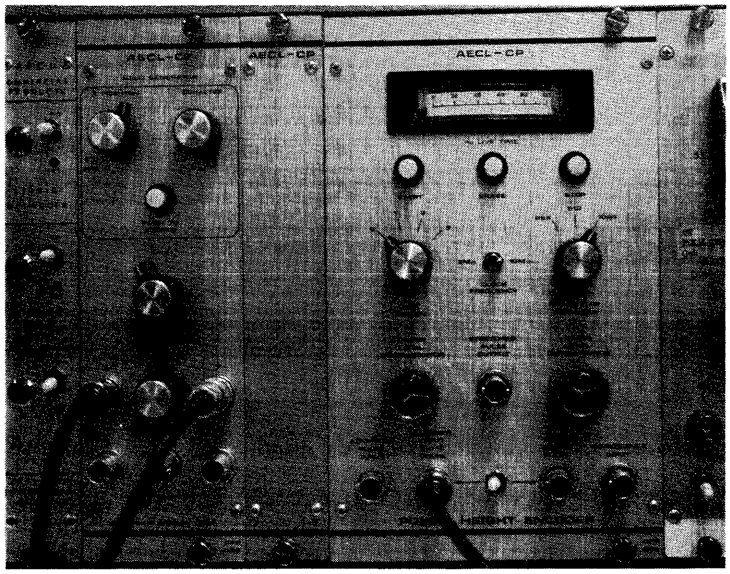
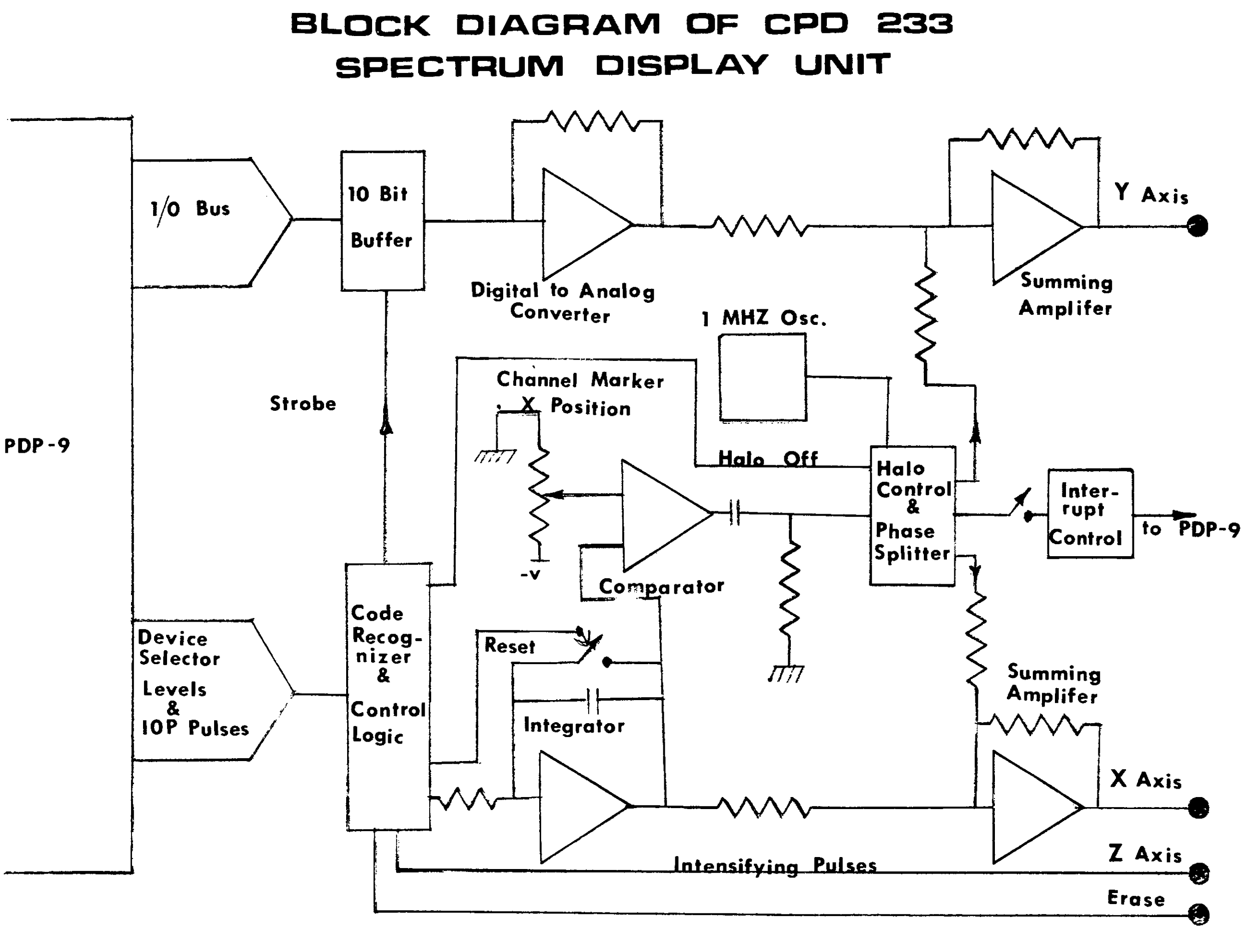


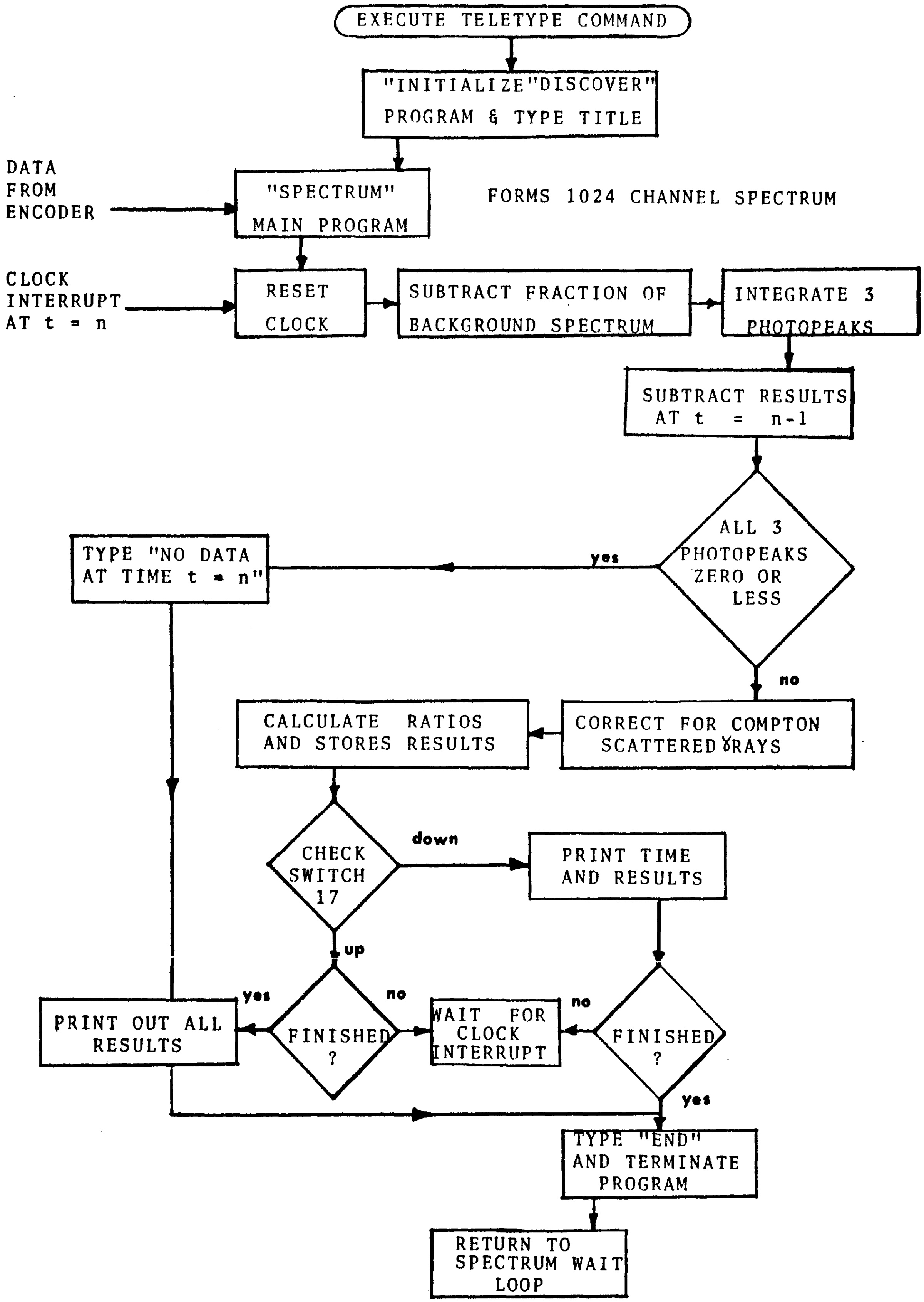
Figure 3. Photograph of Encoder and Display




IOT INSTRUCTIONS FOR CPD 233 DISPLAY

DLY = LOAD Y REGISTER AND DISPLAY
DIX = INCREMENT X REGISTER
DXC = RESET TO LEFT OF SCREEN
DER = ERASE OSCILLOSCOPE SCREEN
DMO = MARK NEXT CHANNEL
HSF = SKIP ON, AND CLEAR HALO FLAG

Figure 4. Block Diagram of Display




# FLOW CHART OF "DISCOVER" PROGRAM

**TO PROCESS DATA FROM AERIAL SURVEY FIELD TRIALS**

FIGURE 5

28,434G
1

AERIAL SURVEY TRIAL RUN NUMBER: 5
LOCATION: ELLIOT LAKE
TEST STRIP: # 9
TAPE REEL: # 27

| TIME IN SECONDS. | COUNTS PER MINUTE K | U | TH | RATIOS U:K | TH:K | U:TH | LIVE TIME | HEIGHT |
|---|---|---|---|---|---|---|---|---|
| 15: | 551 | 180 | 152 | .32 | .27 | 1.18 | 14.59 | 546 |
| 30: | 526 | 200 | 192 | .38 | .36 | 1.04 | 14.53 | 530 |
| 45: | 545 | 167 | 220 | .30 | .40 | .76 | 14.53 | 611 |
| 60: | 299 | 176 | 163 | .58 | .54 | 1.07 | 14.57 | 573 |
| 75: | 344 | 258 | 127 | .75 | .37 | 2.02 | 14.49 | 434 |
| 90: | 502 | 249 | 136 | .49 | .27 | 1.82 | 14.49 | 513 |
| 105: | 341 | 284 | 179 | .83 | .52 | 1.58 | 14.50 | 563 |
| 120: | 455 | 288 | 192 | .63 | .42 | 1.50 | 14.50 | 563 |
| 135: | 636 | 282 | 178 | .44 | .28 | 1.58 | 14.43 | 549 |
| 150: | 693 | 263 | 288 | .37 | .41 | .91 | 14.37 | 559 |
| 165: | 766 | 311 | 273 | .40 | .35 | 1.14 | 14.32 | 572 |
| 180: | 630 | 399 | 247 | .63 | .39 | 1.61 | 14.28 | 556 |
| 195: | 694 | 525 | 348 | .75 | .50 | 1.50 | 14.27 | 599 |
| 210: | 1370 | 586 | 462 | .42 | .33 | 1.26 | 13.89 | 578 |
| 225: | 1454 | 507 | 507 | .34 | .34 | 1.00 | 13.90 | [illegible] |
| 240: | 1109 | 469 | 315 | .42 | .28 | 1.48 | 14.06 | 549 |
| 255: | 893 | 314 | 376 | .35 | .42 | .83 | 14.22 | 591 |
| 270: | 609 | 397 | 235 | .65 | .38 | 1.68 | 14.41 | 646 |
| 285: | 893 | 252 | 298 | .28 | .33 | .84 | 14.38 | 636 |
| 300: | 841 | 227 | 194 | .27 | .23 | 1.16 | 14.34 | 515 |
| 315: | 621 | 205 | 238 | .33 | .38 | .86 | 14.37 | 430 |
| 330: | 810 | 263 | 217 | .32 | .26 | 1.20 | 14.37 | 451 |
| 345: | 639 | 182 | 174 | .28 | .27 | 1.04 | 14.44 | 551 |
| 360: | 580 | 249 | 227 | .42 | .39 | 1.09 | 14.50 | 571 |
| 375: | 469 | 153 | 180 | .32 | .38 | .84 | 14.44 | 374 |
| 390: | 467 | 167 | 128 | .35 | .27 | 1.30 | 14.37 | 259 |
| 405: | 605 | 218 | 146 | .36 | .24 | 1.48 | 14.42 | 418 |
| 420: | 588 | 229 | 147 | .39 | .25 | 1.55 | 14.49 | 530 |

AVERAGE HEIGHT: 528 FEET NORMALLIZED TO: 550
END.

Figure 6

Typical computer printout of results from test strip at Elliot Lake

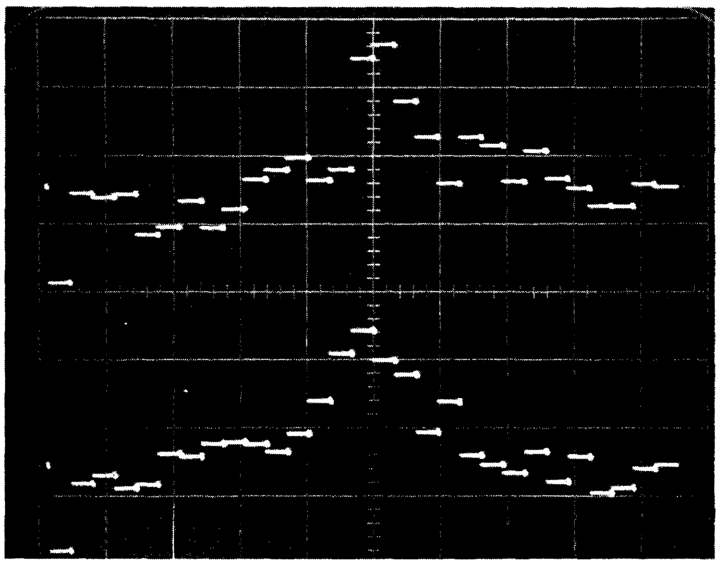


FIGURE 7

POTASSIUM (ABOVE) AND URANIUM (BELOW) COUNTRATES
AS A FUNCTION OF DISTANCE ALONG TEST STRIP AT ELLIOT LAKE

| | | |
|---|---|---|
| Scale: | 50 ft./interval | horizontal |
| | 200 counts/minute. cm. | vertical (potassium) |
| | 100 counts/minute. cm. | vertical (uranium) |

# THE PDP-9 SYSTEM AT THE UNIVERSITY OF MANITOBA CYCLOTRON

L.W. Funk, J.V. Jovanovich, R. Kawchuck, R. King, J. McKeown
C.A. Miller, D. Peterson, D. Reimer, K.G. Standing, and J.C. Thompson
Department of Physics
University of Manitoba
Winnipeg, Manitoba, Canada

## Abstract

A PDP-9 system and its on-line uses in the following experiments are described: (1) measurement of p-p bremsstrahlung cross sections using wire chambers with magnetic core readout; (2) measurement of relative abundance of certain mass isotopes using a Nier type mass spectrometer; and (3) simultaneous recording of two 64 x 64 channel pulse height spectra using one analogue to digital converter.

## Introduction

The computer system at the University of Manitoba 50 MeV cyclotron has been designed around a PDP-9 linked to an IBM 360/65 by a high speed interface. The PDP-9 is located in the control room of the cyclotron building and is interfaced to the experimental equipment situated in the cyclotron experimental room or elsewhere in the physics building; the IBM 360/65 is located at the University Computer Center about 1900 feet away. The role of the two computers is complementary. The principal role of the PDP-9 is to control the experimental equipment, to collect and pre-process data, to output the results, and/or to send them for further processing to the 360/65. The 360/65 computer will then operate on the data, output large volumes of results on a high speed printer, magnetic tape and/or disk, and/or return the results to the PDP-9. In general, the big computer will be processing data using FORTRAN programs while the PDP-9 will be using mainly assembler language programs. The types of outputs provided by the PDP-9 end of the system are basically only those which the physicist needs immediately for the control of the experiment he is performing. An oscilloscope display (DEC TYPE 34), a high precision plotter (CALCOMP), and a fast plotter (MOSELEY) are available to the physicist besides the standard PDP-9 equipment. Two DEC tape units are also available for handy storage of programs or small amounts of data.

The PDP-9 part of the system has been operational since June 1, 1967 with the experimental equipment shown in Fig.1. The high speed data link has not yet been installed. As a temporary measure the data is communicated to the 360/65 using an IBM compatible magnetic tape unit (Precision Instrument) interfaced to the PDP-9. For this reason no discussion of the high speed data link is given here. Only the PDP-9 part of the system is discussed together with three experiments, two of them involving the cyclotron.

## (A) Proton-Proton Bremsstrahlung Experiment.

In this experiment the use of both computers, PDP-9 and 360/65, is essential. The most common uses of computers in low energy nuclear physics involve storing of data in the form of a statistical sample (pulse height spectra, for instance) and operating on it.[1] In this experiment the computers analyze the individual events as soon as they have been recorded, and classify them according to certain rules, thus making computers an integral part of the experimental hardware.[1] The event rates are of the order of 10-100 per second. The PDP-9 requires about five milliseconds to process each event, rejecting most of them (90-99%) and sending the rest (1-10%) to the 360/65 for further processing. When the PDP-9 is not processing events it is plotting graphs or displaying data on the oscilloscope.

A sketch of experimental equipment, PDP-9 computer and interfaces is shown in Fig.2. A 48 MeV proton beam from the cyclotron passes through a hydrogen gas target 20 cm long. The events of interest are those for which a proton from the beam collides inelastically with a target proton yielding a gamma ray. Two outgoing protons pass through four wire chambers (WCH1-WCH4) and into two scintillation counters S1 and S2. The gamma ray is not detected. Pulses from each of the counters are fanned out and analyzed for their heights in two fast single channel differential discriminators (DISC). If the pulse heights are in a desired range and if they are in coincidence a pulse is sent out to: (1) Gate 1 which closes the fast electronics, thereby stopping the data taking. (2) A high voltage pulser driver which in turn triggers the wire chambers via a set of high voltage pulsers. (3) Gate 2 whose role is described below. (4) Interface amplifier and timing circuits.

The second pair of pulses from the phototubes is sent via 250 feet long cables to the control room where they are lengthened, their height is digitized in a pair of Nuclear Data ADC's, and read into the PDP-9 accumulator via a standard DEC interface. Gate 2 opens just before the fast pulses have arrived and closes as soon as they have passed through it, that is, before they are stretched. This is done so as to gate out noise which comes after the wire chambers are sparked.

The magnetic cores (CORE BOARDS) used in the experiment are divided into 108 groups of 18 cores each and a combination of series and parallel readout is employed.[2] The cores of each group have a common read wire which is also the output lead of a core driver circuit.[2] Through every group of 18 cores the 18 sense lines are threaded and connected to 18 sense amplifiers (modified DEC module W520). The outputs of these sense amplifiers are fed into a 18 bit buffer which is interfaced in a standard way to the PDP-9 accumulator. The selection of core drivers is performed in a conventional manner[2] but with some modifications using a 7 bit binary up counter and 3 binary to octal decoders with an 8, 4, 4 code on the 16 select lines to which the inputs of the core driver circuits are attached.

The sequence of events following a coincidence between S1 and S2 counters is as follows: - The logic pulse from the coincidence circuit is converted to a DEC logic pulse by an interface amplifier and fed into a group of timing circuits. Four μsec later, after noise generated by the high voltage spark has subsided, the core drivers are enabled[2] and a binary up counter is cleared. As 50 μsec must elapse before the core drivers can be switched on,a further delay is imposed before a signal is sent to the interface in the control room to set the device flag. This flag, when set, causes an interrupt in the PDP-9 and an IOT pulse is sent to the 7 bit binary up counter in the experimental room. The counter content is decoded to supply a unique set of three inputs to the first core driver. The sense signals appearing on the 18 sense lines are amplified and sent to the 18 bit buffer. There, under program control, these 18 bits of information are stored into computer memory and another pulse is sent to the binary counter. This sequence is repeated until all 108 groups of cores are read out and stored into memory. The computer then operates on this information from the cores using specially written software. When the computer becomes again able to accept another event, an "analysis done" pulse is sent from the PDP-9 to a 2 bit counter which opens Gate 1, provided this counter has already received a pulse signalling wire chamber recovery. (This last pulse is produced by delaying the coincidence pulse for a preselected time.) If the "wire chamber recovery" pulse arrives after the "analysis done" pulse, the former is the one which opens Gate 1.

The procedure for reading out and decoding cores used in this experiment is different from those used previously.[2] Here the raw information from the cores is stored directly into computer memory and further decoding is done by software. The gain is mainly in economy and flexibility as one does not require a relatively expensive shift-register. The disadvantage is that this procedure requires more computer time and is in general slower.

Software for the p-p bremsstrahlung experiment was written to interleave storing and processing of experimental data, displaying it on the oscilloscope, plotting on CALCOMP plotter, or outputting on the teletype. Presently, all the devices are connected to the program interrupt; further improvements will include the use of automatic priority interrupt facility.

A flow chart of the present software is shown in Fig.3. After starting and initializing for a given cyclotron run, the program turns the interrupt on and opens Gate 1 (see Fig.2) which is equivalent to turning the fast electronics on. Afterwards, the program jumps into a display routine displaying on the oscilloscope some histograms. The selection of histrograms is made by the experimenter who communicates his requests to the computer via the teletype. When a program interrupt is received a jump to a program interrupt service routine is executed. After the servicing of a particular device has been finished, the program interrupt is turned on again and the computer returns to the display routine. The order in which external devices are tested for causing an interrupt is given in Fig.3. The servicing of plotter, paper tape punch or teletype is so arranged as to allow these slow mechanical devices to be operated at essentially their full speed. The time used by these devices to physically move their mechanical parts is used by the computer to service wire chambers or to display histrograms on the oscilloscope. As only 1% of computer time is used in servicing the CALCOMP, and plotting of some experimental information may take a long time, the CALCOMP is given higher priority than the wire chambers themselves. This enables us to plot data from a preceding run while collecting new data. The oscilloscope display routines are organized differently. The histograms are displayed in their current form so the experimenter is able to follow their growth.

The details of the wire chamber service routine are not described here as they are of no general interest and are still subject to changes. The wire chamber service routine has a provision for writing raw data (data as received from ferrite cores) onto a DEC tape. This data can then be read off-line back into the computer and reanalyzed using a slightly modified program. The preprocessed and selected events are written on an IBM compatible magnetic tape so that they could be further processed on the 360/65. After the data link between PDP-9 and IBM 360/65 becomes operational, the data will be sent directly to the 360/65.

(B) Measurement of Mass Ratios.

As shown in Fig.4, the PDP-9 was used on-line in an experiment whose objective was to measure relative abundance of certain isotopes with a Nier type mass spectrometer (in particular $C^{12}O^{16}O^{18}/C^{12}O_2^{16}$, $Kr^{82}/Kr^{86}$, $Hg^{200}/Hg^{202}$). The sequence of operations is the following: The spectrometer magnet current is adjusted to a certain value so that one type of ions (say $Kr^{82}$) is transmitted through the instrument. The ions entering an electron amplifier generate pulses which are amplified and counted in a 10 megacycle scaler for a preselected time interval. At the end of the counting cycle, an RC timer raises a flag and signals the PDP-9 to stop the scaler and store its contents in memory. Then an I/O pulse is issued which sets a flip-flop which changes the magnet current via a relay so that the second kind of ions is accepted in the ion counter. After a 5 second delay allowing transients to settle, the scaler is cleared and restarted with the RC timer.

After completion of a specified series of magnet cycles the data taking is stopped. The program then calculates the quantities of interest (ratio of individual scaler readings, standard deviations and their average values, etc.) and lists them on the teletype.

The advantages of the system described above over those commonly used in mass spectroscopy experiments are: (1) Low cost (assuming the computer is available free of charge). (2) The conventional digital to analogue and then back to digital conversion is avoided. (3) Flexibility, good prospects for the measurement of extreme abundance ratios like $He^3/He^4$ which is of the order of $\sim 10^{-6}$.

## (C) Application to a A(p,2p)B Experiment

The PDP-9 is being used to control a number of peripheral devices in the real-time correlation and listing of data produced by analogue-to-digital converters and coincidence circuits. A major experimental problem when counting coincidences between two particle-identifying devices is obtaining an accurate sample of the number of chance coincidences. Use of the computer enabled the simultaneous accumulation of both real events and a sample of random coincidences, using only one pair of ADC's. Thus, the problem of exact equalization of the gains of two sets of ADC's is eliminated. Two fast coincidence circuits are used to gate the entire system input to the ADC's. One of them gates a sample of random events since its inputs are deliberately delayed with respect to one another. In order to label these random events as such, the coincidence circuit supplies a logic pulse to the computer. In the present state of implementation, this is done through the light pen interface.

Since the counting rate of the ADC output is so low in this experiment, it is possible to list all events with their real-random labels on the teletype and paper tape punch as they occur. Also, since each event is a point in a two-dimensional plot of the energy of one particle versus the energy of the other, the events are concurrently plotten on a Moseley X-Y point plotter. This plotter's interface is also improvised; its analogue signals are taken directly from a CRT scope interface, its "seek pulse" is taken from the interface to a Calcomp incremental plotter and when it is used for plotting spectra off-line, its "plot pulse" is fed into the light pen interface.

A possible disadvantage of the very low counting rate, other than the obvious one, is the risk of paralysis of some component of the electronics system feeding the computer. A facility to overcome this was provided in the software by allowing the periodic transmission of pulses through the system, simulating real events in a high energy data region unpopulated by real or chance events. When this facility is enabled, the absence of one of these periodic artificial events is signalled by the computer on the teletype.

The software for all the above was obtained through additions to the Dual Parameter AC Mode Multianalyzer Program supplied by D.E.C. In this way the interrupt handling, data manipulation and storage facilities provided by this program could be fully utilized. Additions were also made to the Single Parameter Multianalyzer Program which permitted use of the light pen, Moseley plotter and reading or punching paper tape in IBM code. Also, more versatile CRT scope display routines were provided.

## Acknowledgements

It is our pleasure to acknowledge the financial support of the Atomic Energy Control Board of Canada.

## References

1) Computer application in this experiment is similar to the applications in High Energy Physics. See for instance:L.S. Lindenbaum, Ann. Rev. Nucl. Sci. 16, 619 (1966); R.J. Spinrad, ibid, 14, 239 (1964).

2) W.A. Higinbotham, J.F. Jacobs, and H.R. Pate, IEEE Trans. Nucl. Sci. NS-12, No.1, 386 (1965).

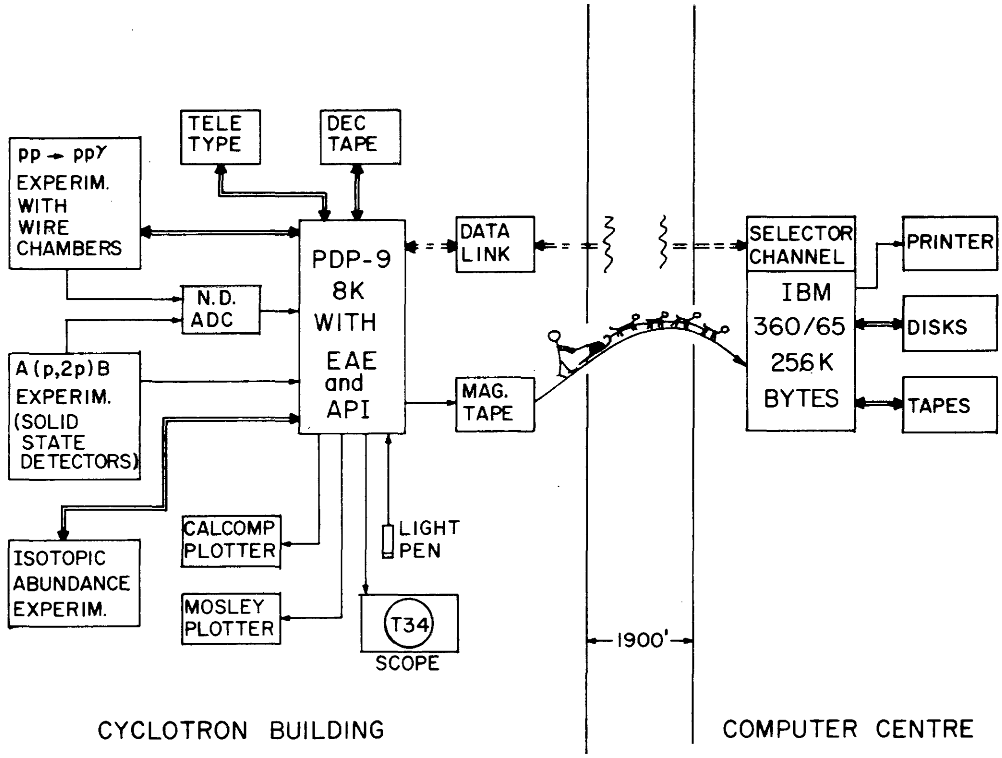



Fig.1 Manitoba Cyclotron computer system. The dashed line indicates that the fast data link is not yet operational.

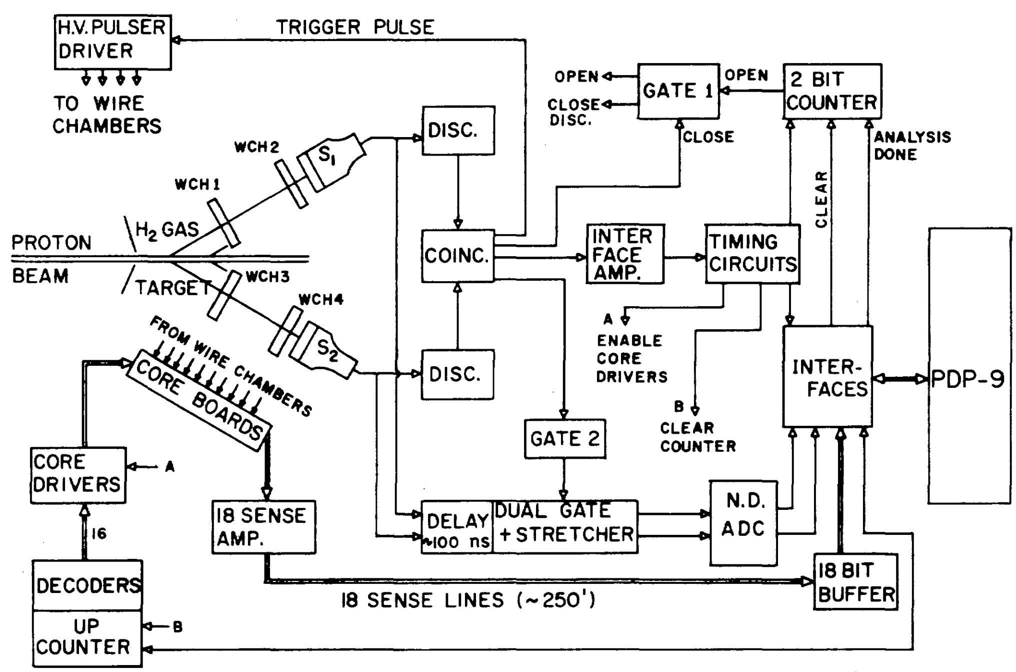



Fig.2 Proton-proton bremsstrahlung equipment and interfaces.

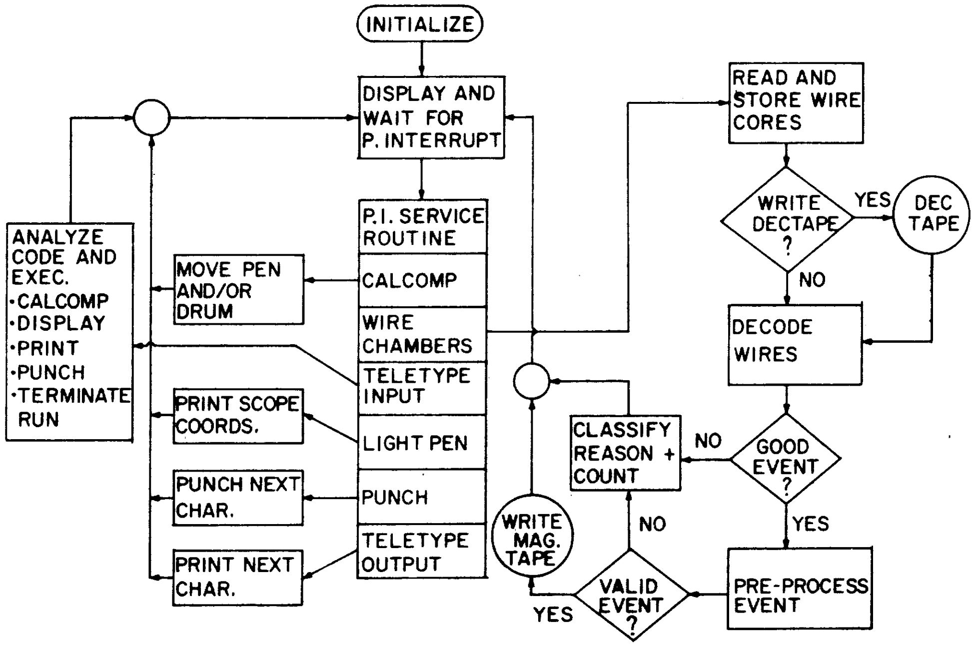



Fig.3 Software system at present in use for proton-proton bremsstrahlung experiment.

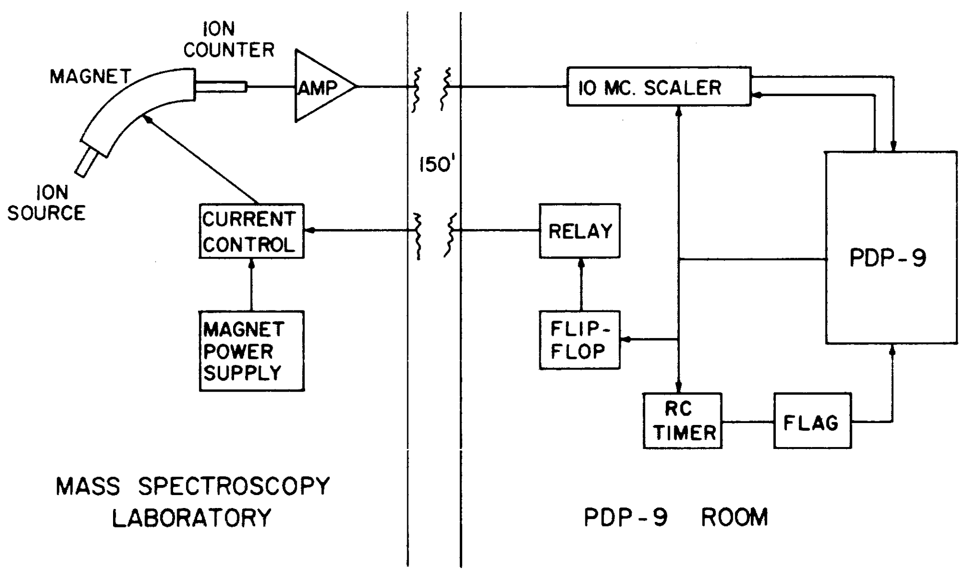



Fig.4 Mass spectrometry equipment and interface.

# TRACK FOLLOWER - A SYSTEM FOR BUBBLE CHAMBER TRACK RECOGNITION

James P. Taylor
Massachusetts Institute of Technology
Cambridge, Massachusetts

## Abstract

Track following is a major pattern recognition problem in the automatic scanning of bubble chamber film. With the PEPR device controlled by a PDP-6, track following is done in real time. Both the film and the computer are used as storage for track data. Each track is followed one view at a time and the resulting data must meet certain requirements (e.g. continuity) to be accepted. Pattern recognition problems arise from close lying tracks, and from small angle-crossing tracks.

## Introduction

The problem which project PEPR at MIT/LNS has undertaken is the automatic encoding of bubble chamber film. Our configuration consists of the PEPR hardware controlled by a PDP-6. The "6" has 16 fast AC's, 48 K, 6 DEC tapes, 3 magnetic tapes and a type display scope. The programs take 14 K of core while storage uses another 14 K. The remaining core is available and is used for I-O limited time sharing. This paper is a discussion of the methods used to filter tracks from background data. This technique is called at PEPR track following.

Automatic encoding requires that at some level the problem of pattern recognition of tracks be solved. This problem is solved in different ways by the users of the various automatic scanning devices in use. At PEPR we have decided, due to the nature of the PEPR device, to allow the film to act as a data store. The film can be randomly accessed by the hardware under computer control. Using appropriate techniques the data for the tracks is gathered one track at a time.

## The PEPR Device

The signal source of the PEPR device is a 4096 x 4096 precision cathode ray tube. The electron beam can be defocussed into a line segment by diquadrapole magnetics. Both the orientation ($\phi$) of the line and its length can be varied by the program. The line (or spot) can be swept in either the +x or +y direction about any one of the raster position. The maximum sweep length in the low precision pattern recognition mode (PR-10 x $10^{-6}$ m least count) is 2.5 mm. In the precision encoding mode (PE-1 x $10^{-6}$ m least count) the sweep length is .25 mm. Track follower uses the PR mode of operation.

The device enables one to encode a 35 mm frame with the line (PR mode) in 35-45 seconds depending on the density of information on the film. This x, y, $\phi$ data when displayed is a representation of the frame in question as it is seen by the hardware.

## Point Guidance

PEPR is running under a system called point guidance (PG) in which one reference fiducial, the vertex, and one point per track are pre-digitized by hand on image plane digitizers (IPD). This IPD information about a frame is transmitted to PEPR on magnetic tape with the film one view at a time. Track data from PEPR is combined off-line on an IBM 7044 where the analysis is performed.

## Track Follower

Locate the IPD Point - The first function the track following package is to use the IPD information to find the first or starting element on the track. The input data consists of the vertex of the event and one point on each track (the clear-point CP).

This data is translated, after the fiducials have been located, giving the approximate location of the event in the film plane.

The CP is used as the first scanning position. Any track elements found in this region which agree with the input x, y, $\phi$ (the $\phi$ results from a vector formed between the vertex and the CP) are assumed to be on the track. These elements (max 2) are used as starting points from which the rest of the track is to be found. The requirement for the clear point is that it be in the clear so that a scan only produces one element at the constructed angle. Due to the nature of the PEPR device confused regions exist only when there are no hits or when hits occur at close angles, angles less than four degrees, to the expected angle.

The element found about the CP that is closest to it [if more than one are found] is used as the first point on the track. From this point, using the angular information in the PEPR element an attempt is made to reach the upstream IPD vertex. If the vertex is reached the track following is considered a success. As many elements as can be found are then picked up on the downstream side of the CP.

Linear Phase - The first prediction phase is linear. The second sweep position is predicted from the first using the formula

$$x_2 = x_1 + R \cos \phi_1$$

$$y_2 = y_1 + R \sin \phi_1$$

$$\phi_2 = \phi_1$$

R has the value .25 mm on the film and is called the step.

To allow for curvature and other uncertainty in the prediction the program which determines if any of the elements found belong to the desired track (i.e. are close enough to the predicted element) will accept any element $\pm 2^\circ$ from $\phi_p$ and .2 mm from the $x_p$, $y_p$. It is essential that the CP be in the clear. If there are multiple elements found from this first prediction, the system can not pick the correct element since it does not have enough information about the desired track. This is one of the limitations of a one view device which can be overcome when more than one view is available to the machine.

Circular Prediction - Five linear predictions are performed before a more complicated prediction technique is used. A circular prediction is performed when there are five accepted track elements (5 x .25 mm 1.25 mm from the CP to the VTX). Circular parameters (radius and turning angle) are computed for the five points. The method used is to compute the parameters for each set of three points formed by the end points and one of the points in between. These parameters are then averaged to obtain a value for all the points. Using these parameters the next point on the circle is predicted, using a step of 1 mm. Since the circular predictions allows for curvature the acceptance tolerances can be narrowed to $\pm 1^\circ$ and .1 mm even though a larger prediction step is used.

If the track is in the clear, following continues until the vertex is reached. This is determined by sensing on the sign of the dot product of a master vector formed between the CP and the VTX and a running vector formed by the predicted point and the VTX. When the sign changes the predicted point has past the vertex. Track following on that track in the upstream direction ceases.

The elements found are used to predict in the downstream (from the CP) direction. Track following is resumed in this direction until there are no further hits or the fiducial volume is exceeded. The type of prediction used is determined by the number of elements available. If there are less than five it is linear. Following away from the vertex a step size of 1.25 mm is used.

Vertex Tests - Track following is considered a success if the vertex is reached. Data must come within 3 mm longitudinally and 1 mm transversely to be accepted as at the vertex. The projected trajectory of the track is used for the transverse calculation if data stops short of the vertex.

In clear regions track following is nearly always successful. Failures are caused by areas which are confused. What follows is a description of the techniques used to overcome these difficulties.

Confused Regions - There are several types of confused regions which can be handled by the track follower. The first is a region where there are no hits at the predicted point. This is caused by some hardware-film contact problem (e. g. insufficient contrast). Track follower assumes that the trajectory which is being used still holds and predicts ahead with the same step. If no elements are located within 4. 5 mm of the last found element, scanning ceases. If the following direction were toward the vertex this might cause the track to fail if no elements were found near the vertex. If following away from the vertex, the portion of the track from the CP to VTX and that already found on the downstream side of the VTX is accepted and the track is located.

Tracks which cross at less than $2^o$ and are within . 1 mm of each other are treated as unresolvable by the track follower. A gap is created by the filtering routine which is treated as a hardware gap. The only major application of this is in close lying beam tracks which are within . 1 mm of each other. These tracks can not be resolved by track follower and are rejected at the IPD level.

At some point low angle ( $< 3^o$) crossing tracks satisfy the previous discussion. This situation is solved by sensing when two or more tracks are about to intersect, but are farther than . 2 mm from each other. When two tracks are within . 5 mm of each other the point of intersection is computed (assuming the tracks to be straight lines). The distance from the last good element to this intersection is computed. Track following proceeds at this distance on the other side of the intersection. If more than one element is found here all ($<$ 4) tracks are followed for a fixed distance (10 mm). These questionable track sections are then compared to the already accepted section of the track using a least squares fitting technique. The best fitting section is saved. Track following then proceeds in the same direction having bridged the intersection.

## Summary

The PEPR system is being operated on film from the 30" hydrogen bubble chamber at A. N. L. The film has an incident beam of $\pi^+$ at 3. 9 BeV/c. Track success rates are 85%. Most failures are due to faulty input data or tracks which have no clear region within 3 mm of the vertex. Tracks of less than . 5 mm on the film also give difficulty at present. PEPR processes 15-20 events an hour. Program efficiencies in progress should, with no changes in technique, up the rate to 50-75 events per hour.

## REFERENCES

Bastien et. al., "Programming for the PEPR System" Methods in Computational Physics (1965), Academic Press.

Friesen, "A Description of the PEPR System" DECUS Spring Symposium 1965.

Yamamoto, "The Status of the PEPR Point Guidance System", Programming for Flying Spot Devices Proceedings (1967).

# A TUTORIAL ON NUMERICAL ANALYSIS WITH AN EMPHASIS ON ERROR ANALYSIS AND PREDICTION

Wayne A. Muth and Bruce C. Davis
School of Technology, Southern Illinois University
Carbondale, Illinois

Abstract

Factors of pertinence in the selection and use of certain numerical techniques are considered. Key topics include error analysis in numerical integration, matrix manipulation error analysis, and error analysis in the design of certain elementary function routines.

## Introduction

Many day-to-day computer users are somewhat casual about the treatment of certain classes of errors, e.g. truncation and rounding error. Both applications programmer/analysts and software writers can be put in this category. The cause is probably two-fold--many were never exposed to the topic in previous work; many others, however, simply find the topic distasteful (for a variety of reasons) and prefer not to think about it. This present paper is aimed at anyone who falls in either category.

## What Is Numerical Analysis?

Numerical analysis is both a science and an art. As a science, it is concerned with the processes by which problems can be solved by certain explicit arithmetic operations. Sometimes, a specific algorithm will be developed. Often, it will be necessary to replace some quantity which cannot be calculated arithmetically (e.g., an integral) by an approximation which then permits a solution to be found. In this case, one must concern himself with the errors incurred in having used the approximation.

As an art, numerical analysis is concerned with choosing some "best method" from the available problem-solving methods.

## Sources Of Error

Generally speaking, errors can be introduced into the solution of a problem during either the mathematical formulation of the problem or during the actual solving of the problem. Of particular interest are two specific errors which may be introduced during solution, viz. truncation error and rounding (or round-off) error.

### Truncation Error

This error may be caused by the chopping off of a decimal representation after some specific number of places, e.g. = 3.1415. It may also be caused by cutting off the later terms in some infinite series expansion used to approximate a function, e.g. $e^x = 1 + x + x^2/2! + x^3/3!$.

### Rounding Error

This type of error results when the less significant digits of a quantity are deleted and some rule of correction applied to the remaining part, e.g. = 3.14159265. ..., rounded to four decimals would be 3.1416. This source of error is due to the fact that arithmetic calculations can seldom be carried out with total accuracy. Most numbers have infinite decimal representations which must be rounded. Even when a number can be expressed exactly by a finite decimal representation, division may introduce numbers which must be rounded and multiplication may produce more digits than can be retained in a particular word length in the computer.

Numerical Integration

Quite often, one cannot evaluate a definite integral explicitly and use must be made of an approximation. An example would be the use of

$$f(x) = x - \frac{x^3}{3} + \frac{x^5}{5 \cdot 2!} - \frac{x^7}{7 \cdot 3!} + .. + \frac{(-1)^{n-1} x^{2n-1}}{(2n-1)(n-1)!}$$

as an approximate solution for $\int_0^x e^{-t^2} dt$.

One then faces the problem of how many terms to use.

Truncation Error

Considering only the truncation error, E, associated with the use of n terms of the alternate series just seen, reference can be made to the first term omitted: $|E| < x^{2n+1} / (2n+1)(n!)$. A quick check reveals that this sum has a non-zero digit in the first decimal place. If we assume an eight-bit word length, the eighth decimal place would be the last digit retained and we might choose to determine an n such that $1/(2n+1)(n!) < 5 \times 10^{-8}$. We could then say we "were accurate to seven places".

Rounding Error

Considering again the integral of the previous section, what happens if we allow x to take on bigger values? Defining a term r as the ratio of successive terms, we find

$$r = \left| \frac{(2n-1)\ x^2}{(2n+1)\ n} \right| \approx \frac{x^2}{n}$$

If we consider the values x = 12 and n = 143 (i.e., 143 terms in the series), the value of the last term is

$$\frac{x^{2n+1}}{(2n+1)n!} = \frac{12^{287}}{(287)(143!)} > 10^{59}$$

and we might thereby exceed the range of the sorted floating-point number. It should thus be clear that caution must be exercised when using a power series expansion for values of x some distance away from the point of expansion.

Simpson's Rule, Trapezoidal Rule, Parabolic Rule...Attributes

Simpson's rule arises from use of a polynomial approximation for the integrand. If we use just two points on the integrand and a straight line between them, we obtain

$$\int_a^b f(x)dx = \frac{h}{2}\Big[f(a)+f(b)\Big] + \frac{h^3}{12} f''(\epsilon).$$

If we get slightly fancier and use three points...

$$\int_a^c f(x)dx = \frac{h}{3}\Big[f(a) + 4f(b) + f(c)\Big] + \frac{h^5 f^{iv}(\epsilon)}{90}.$$

Thus for the typical case of h< 1.0 and f'' more or less equal to $f^{(iv)}$, the three point version will have the smaller truncation error.

The trapezoidal and parabolic rules arise by dividing the integration interval into a number of subintervals, then, respectively, applying the two-and three-point versions of Simpson's rule. The error terms are as before and the parabolic rule typically can be expected to have the lesser truncation error:

$$\text{Trapezoidal: } \int_a^b f(x)dx = \frac{h}{2}\Big[f_0 + 2f_1 + 2f_2 + .. + f_n\Big],$$

$$\text{Parabolic: } \int_a^b f(x)dx = \frac{h}{3}\Big[f_0 + 4(f_1 + f_3 + f_5 + ..) + 2(f_2 + f_4 + ...) + f_n\Big],$$

... even number of "strips", $f_j = f(a+jh)$, $h = (b-a)/n$.

A Stopping Rule

In a manner generally similar to the earlier example, one may elect to solve for n in the relation (continued )

$$\left| \frac{nh^5}{90} \max \left| f^{(iv)}(x) \right| \right| < E$$

where E is arbitrarily specified in a manner compatible with the length of the stored number. This particular relation, of course, is for the parabolic rule. When n is determined thusly, we know we are on safe ground.

However, bounds determined in this way are likely to be far too large. A prudent problem-solver is probably well-advised to make a concurrent comparison of the "present-value" against the "previous-value", step by step, and shut down the computation whenever the value stops changing. Perhaps one can stop prior to using all n terms (as just calculated in the previous relation).

## A Surprise with the Newton-Raphson Method!

Several widely used methods may be employed to solve transcendental equations or high-order polynomials. Popular methods include the half-interval search, Regula Falsi, and the Newton-Raphson method.

Typically, rounding or truncation errors are seldom encountered, per se. However, one may occasionally end up with a root other than the one sought and some caution must thereby be exercised.

Consider the pair of equations

$f_1(x, y) = e^x - y - 1 = 0$ and $f_2(x, y) = x^2 + y^2 - 4 = 0$. A quickly drawn sketch shows that one root lies near the point (1.0, 1.7), the other near (-1.8, -0.8).

Either of two pairs of relations may be used to iterate to the value of a root. The first pair is

$$x_{k+1} = x_k = \frac{e^{x_k} - y_k - 1}{e^{x_k}}$$

and $$y_{k+1} = y_k - \frac{x_k^2 + y_k^2 - 4}{2y_k} \;;$$

the second pair is

$$x_{k+1} = x_k - \frac{x_k^2 + y_k^2 - 4}{2x_k}$$

and $$y_{k+1} = y_k - \frac{e^{x_k} - y_k - 1}{-1}$$

An initial trial root of (1.0, 1.0) would appear safe in an attempt to find the root lying in the first quadrant. Using the first pair of relations, one quickly finds this root, correct to three decimal places after six interations. One finds, however, that use of the second pair of relations yields the root in the third quadrant after eight iterations -- in spite of the fact that the second and third iteration x-y values lie "right-next-to" the value of the root in the first quadrant!

## Approximations of Elementary Functions

Error analysis of a numerical result is critical to any computation. Consider the Taylor series for the Cosine:

$$\text{Cosine } x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \dots$$

$$+ (-1)^n \frac{x^{2n}}{2n!}$$

the truncation error made by stopping the summation after a finite number of terms is said to be less than the absolute value of the first term neglected. This is true if we keep an infinite number of digits for each term. The following examples show what happens when we do not carry an infinite number of digits for each term.

Cosine($60^\circ + 2N\pi$) using Taylor Series
Seven - Digit (2**23-1)Flt. Pt. 1. E-06

| Degrees | Radians | Cosine |
|---|---|---|
| 60. | 1.04719 | +0.500000E+0 |
| 420. | 7.33038 | +0.500021E+0 |
| 3660. | 63.8790 | -0.171618E+21 |

Figure 1

Cosine($60 + 2N\pi$) using Taylor Series
Nineteen - Digit Flt. Pt. 1. E-10

| Degrees | Radians | Cosine |
|---|---|---|
| 60. | 1.04719 | +0.499999E+0 |
| 420. | 7.33038 | +0.499999E+0 |
| 3660. | 63.8790 | +0.127083E+09 |

Figure 2

We see represented here the effects of truncation error and roundoff error. The truncation error is caused by truncating the infinite series. The roundoff error is caused by only carrying a finite number of digits in our calculations. As we can see by this example, roundoff errors can be quite significant.

We previously said that the truncation error committed by stopping a summation after a finite number of terms is less than the absolute value of the first term neglected. One way to reduce the truncation error is to include more terms in the summation. This of course adds to the computation time. It also brings up another problem. How do we add more terms without increasing the problem of roundoff errors? This leads us to a discussion of some of the various algorithms that have appeared for computing values of several elementary functions.

## Three Approximation Techniques

The algorithms fall into three main groups: iterative, polynomial, and rational. Sometimes a rational and an iterative method may be combined for better results. The best algorithm for evaluating a given function may be dependent on a number of factors. Some of these factors are: required accuracy, number base of the computer, word size, relative speed of arithmetic operations, and available storage space.

## Iterative Method

An example of an iterative technique is the Newton-Raphson method for finding the Nth root. DEC uses six iterations of the Newton-Raphson method to calculate the square root on the PDP-8. We find that by using the linear approximation shown in Figure 3 and two iterations of the Newton-Raphson method we can obtain a relative error of less than $2^{-31}$.

For $0.5 \leq x < 1$

$$y_0 = \frac{X + \frac{\sqrt{2}}{2}}{\sqrt{\sqrt[4]{8} + \sqrt[4]{2}}}$$

$$\approx 0.59017853\ X + 0.41731924$$

Figure 3

## Rational and Iterative Method

If more accuracy is required we can use the Pade rational approximation shown in Figure 4. Using this method with two iterations of the Newton-Raphson method we are able to obtain a maximum relative error of approximately $2^{-41}$.

Pade Rational Approximation for Square Root
(For Binary Arithmetic)

For $0.25 < X \leq 0.5$

$N = 2^{2m} \cdot X \qquad \sqrt{N} = 2^{m} \cdot \sqrt{X}$

$$\sqrt{X} \approx Y_0 = 2.533463 - \frac{4.829452}{X + 2.142858}$$

Figure 4

## Polynomial versus Rational Method

An example of a polynomial approximation is the Chebyshev polynomial (see Figure 5) used by DEC in their natural log routine for the PDP-8. Figure 6 shows how a rational approximation may be used to reduce the number of multiplications needed to obtain the desired accuracy. This reduction in the number of multiplications should result in a reduction in the roundoff error and an increase in the speed of the subroutine. (For a report on the error of certain approximations, see article by Fike.)

Function: $\ln(1 + x)$
Range: $0 \leq x \leq 1$
Approximation: $\ln(1 + x) = a_1 x + a_2 x^2 + \ldots a_8 x^8$
(for values of constants, see Hastings, pg 180)

Figure 5

Function: ln x
Range: $0.5 \leq x < 1.0$
Approximation: $\ln x \approx -\frac{\ln 2}{2} + \sum_{k=0}^{2} a_{2k+1} U^{2k+1}$

where $U = \frac{x - \sqrt{2}/2}{x + \sqrt{2}/2}$ ,

$a_1 = 2.000000815$
$a_3 = 0.666445069$
$a_5 = 0.415054254$

Figure 6

Rational and polynomial approximations for the other functions used with the PDP-8 FORTRAN System may be found in Lyusternik. In many cases the rational approximations present rather marked improvements over the existing approximations.

## Operations with Matrices

One is often concerned with the solving of a set of simultaneous linear equations or the inversion of a matrix (for one reason or another). The applicible numerical methods break into two distinct groupings: exact methods and iterative methods. In either case, errors can be introduced into the solution rather easily if care is not taken.

### Source of Error

The introduction of errors into the solution may occur as the result of the inevitable rounding of the figures in the course of the computation. This is particularly true when the product of two numbers, each having a large number of significant digits, must be either rounded or truncated to fit into the proper word length of the stored word. Concurrent with this, one may encounter the phenomenon of disappearance of significant figures during the course of the computation, as the result of the subtraction of two numbers which differ little from each other. Often, the computational scheme itself must be designed with this phenomenon explicitly in mind.

### Typical Method

The primary methods used in this class of problem are based on the idea of "elimination" -- usually the name of Gauss or Jordan in somewhere in the name of the method. Although there are several varieties, basically one is dealing with row transformations on an augmented matrix:

$$[A \mid I] \rightarrow \rightarrow \rightarrow \rightarrow [I \mid A^{-1}]$$

### Illustration

Consider the 2 by 2 matrix A wherein $a_{11} = 2.0 \times 10^{-10}$ , $a_{12} = 2.0$, $a_{21} = 4.0$, and $a_{22} = 6.0$. If we array this using the machine representation of the numbers, it might look like the following "typical" representation

$$\begin{bmatrix} 4020000000 & 5020000000 \\ 5040000000 & 5060000000 \end{bmatrix}$$

where the representation 4020000000 implies $2.0 \times 10^{40-50}$ or $2.0 \times 10^{-10}$.

A matrix of this type is said to be "ill-conditioned" in the sense that it is singular with respect to the number of significant digits carried by the machine during the course of the computations-- specifically, in this case, $a_{11}$ and $a_{12}$, for example, differ by the order of $10^{10}$ which is excessive (and is certain to cause trouble) in the assumed 8-bit word used herein.

Several steps of the Modified Jordon method depict the trouble that can ensue:

1st stage reduction:...

$$\begin{bmatrix} 506 - \frac{502 \times 504}{402} & -\frac{504}{502} \\ \frac{502}{402} & \frac{501}{402} \end{bmatrix}$$

$$= \begin{bmatrix} 506 - 604 & -602 \\ 601 & 595 \end{bmatrix}$$

...but machine only retains---

$$\begin{bmatrix} -604 & -602 \\ 601 & 595 \end{bmatrix} .$$

2nd stage reduction...

$$\begin{bmatrix} 595 - \frac{601 \times 602}{604} & \frac{601 \times 501}{604} \\ \frac{602}{604} & \frac{-501}{604} \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 4925 \\ -495 & -3925 \end{bmatrix}$$

and the resulting inverse is seen to be no good:

$$\begin{bmatrix} 402 & 502 \\ 504 & 506 \end{bmatrix} \begin{bmatrix} 0 & 4925 \\ 495 & -3925 \end{bmatrix} = \begin{bmatrix} 501 & 0 \\ 503 & 501 \end{bmatrix}$$

Generally, to avoid this type of trouble, one need only swap the position of two rows and/or two columns in order to place the largest element in the matrix into the upper left corner of the matrix. This can be done prior to any reduction step. One then "unswaps" the final resulting matrix in the reverse sequence of the swapping -- if at some step, for example, rows 1 and 3 were swapped, then later columns - 1 and 4, the unswapping sequence would be first rows 1 and 4, then columns 1 and 3.

The placing of the largest element into the corner minimizes the chance for introduction of error due to there being too large a numerical difference between minuend and subtrahend in later steps.

## References


Anon., Fortran in the Pysical Sciences, IBM Student Text.

Anon., An Introduction to Engineering Analysis for Computers, IBM Student Text.

Faddeeva, V.N., Computational Methods of Linear Algebra, Dover Publications, New York, 1959.

Fike, C. T. "Methods of Evaluating Polynomial Approximations in Function Evaluation Routines", Comm. of ACM 10(1967), 175-178.

Froberg, C. Introduction to Numerical Analysis, Addison-Wesley, Reading Mass., 1965.

Hastings, C. Approximations for Digital Computers, Princeton University Press, Princeton New Jersey, 1955.

Hemmerle, W. J. Statistical Computations on a Digital Computer, Blaisdell, Waltham Mass., 1967.

Lyusternik, L. A., Chervonenkis, O. A. and Yanpol'skii, A. R. Handbook for Computing Elementary Functions, Pergamon Press, New York, 1965.

McCracken, D. D., Fortran with Engineering Applications, Wiley, New York, 1967

Ralston, A. First Course in Numerical Analysis, McGraw-Hill, New York, 1967.

Salvadori, M. G. and Baron, M. L. Numerical Methods in Engineering, Prentice-Hall, Englewood Cliffs N. J., 1961.

Wilkes, M. V., Short Introduction to Numerical Analysis, F. R. S. University Press, Cambridge England, 1966.



Part of the presentation on error analysis is an extension of earlier work conducted by George McCauslan and John Duffy under the supervision of Keith Beyler. The work was supported in part by NSF Grant GW1903 Secondary Science Training Program.

## APPENDIX 1

## DECUS FALL SYMPOSIUM PROGRAM

Jolly Roger Inn and Anaheim Convention Center
Anaheim, California

November 10 and 11, 1967

### FRIDAY - NOVEMBER 10

#### Morning

REGISTRATION
Jolly Roger Inn - Ballroom

OPENING AND WELCOME
Prof. Donald A. Molony
DECUS Meetings Chairman

KEYNOTE ADDRESS
Kenneth H. Olsen, President
Digital Equipment Corporation, Maynard, Massachusetts

A COMPUTER CONTROLLED DIFFRACTOMETER
Howard A. Cohodas
Picker Instruments, Cleveland, Ohio

THE INTEGRATOR-COMPUTER SYSTEM FOR GAS CHROMATOGRAPHY DATA AUTOMATION
Tom Barrett
Infotronics Corporation, Houston, Texas

DATA ACQUISITION AND ANALYSIS OF HIGH RESOLUTION MASS SPECTRA IN REAL-TIME
M. L. Cramer and D. J. Waks
Applied Data Research, Inc., Princeton, New Jersey

THE PDP-9T: COMPATIBLE TIME SHARING FOR THE REAL-TIME LABORATORY
D. M. Forsyth
Harvard University, Cambridge, Massachusetts

M. M. Taylor
Defense Research Establishment, Downsview, Ontario, Canada

L. Seligman,
Digital Equipment Corporation, Maynard, Massachusetts

LUNCH
Jolly Roger

#### Afternoon

##### Session A

Jolly Roger Inn
Prof. Donald A. Molony, Chairman

A COMPUTER BASED ELECTROCHEMICAL CONTROL AND DATA ACQUISITION SYSTEM
George Lauer and R. A. Osteryoung
North American Aviation Science Center, Thousand Oaks, California

TIME-SHARED COMPUTER CONTROL IN ANALYTICAL CHEMISTRY
Jack W. Frazer
Lawrence Radiation Laboratory, Livermore, California

DEDICATED COMPUTERS FOR INSTRUMENT CONTROL
Roger E. Anderson
Lawrence Radiation Laboratory, Livermore, California

COFFEE

##### Session B

Convention Center - Garden Grove Room 3
Prof. Philip R. Bevington, Chairman

THE LINC-8 IN RESEARCH ON SPEECH
Richard Harshman and Peter Ladefoged
University of California at Los Angeles, Los Angeles, California

LINC-8 TEXT-HANDLING SOFTWARE FOR ON-LINE PSYCHOPHYSICAL EXPERIMENTS
B. Michael Wilber
Stanford Research Institute, Menlo Park, California

PRE-PROCESSING PHYSIOLOGICAL SIGNALS
(Miss) Maxine L. Paulsen
Medical Systems Development Laboratory
Washington, D.C.

DEVELOPMENT OF CARDIOVASCULAR PULMONARY PATIENT CARE TECHNIQUES
Jerome A. G. Russell
Research Data Facility, San Francisco, California

Session A (continued)

A REAL TIME AIR POLLUTION ANALYSIS SYSTEM
Carter L. Cole
Automatic Information Management, Inc.
Encino, California

AUTOMOBILE EXHAUST ANALYSIS SYSTEM
Robert Jahncke
Beckman Instruments, Inc., Fullerton, California

MODULE WORKSHOP

NEW MODULES FOR THE M SERIES
Frederick W. Macondray
Lawrence Radiation Laboratory, Berkeley, California

DISCUSSION REGARDING COMMUNICATION BETWEEN HARDWARE USERS AND DEC
T. R. Sabin
University of California, Berkeley, California

Session B (continued)

COFFEE

PDP-9 OPERATING SYSTEM WORKSHOP
James Murphy and David Leney, Chairmen
Digital Equipment Corporation, Maynard, Massachusetts

## SATURDAY - NOVEMBER 11

### Morning

Session A

Jolly Roger Inn
Prof. Donald A. Molony, Chairman

A PROGRAMMED DISTRIBUTION GENERATOR
David N. Samsky
Booz-Allen Applied Research, Inc.
Albuquerque, New Mexico

ANALYSIS OF MULTICHANNEL ANALYZER DATA WITH LIGHT PEN TECHNIQUES
C. Wendell Richardson
Phillips Petroleum Company

AUTOMATIC CALIBRATION AND EVALUATION OF MULTICHANNEL ANALYZERS USING A PDP-8
W. W. Black
Idaho Nuclear Corporation, Idaho Falls, Idaho

COFFEE

TAPE RECORDER I/O OPERATION IN A PDP-8 CONTROLLED BETA-RAY SPECTROMETER SYSTEM
J.J.H. Park and J. Ohkuma
National Research Council, Ottawa, Canada

ADAPTATIONS OF PDP-8 FORTRAN FOR LABORATORY COMPUTING
Russell B. Ham and Christopher B. Nelson
U.S. Public Health Service
N. E. Radiological Health Laboratory
Winchester, Massachusetts

Session B

Convention Center - Garden Grove Room 3
Prof. Philip R. Bevington, Chairman

GAMMA RAY SPECTRUM STRIPPING
Friedrich Riess
Stanford University, Stanford, California

REAL TIME DATA ANALYSIS WITH FORTRAN
Albert Anderson
Stanford University, Stanford, California

A NON-LINEAR LEAST-SQUARES SEARCH ROUTINE FOR SMALL COMPUTERS
Thornton R. Fisher
Stanford University, Stanford, California

COFFEE

USE OF A PDP-9 FOR REAL TIME OFF-LINE ANALYSIS OF SPECTRA FROM AN AERIAL SURVEY FOR RADIOACTIVE MINERALS
C. J. Thompson
Atomic Energy of Canada, Ltd., Ottawa, Canada

PDP-9 SYSTEM AT THE UNIVERSITY OF MANITOBA CYCLOTRON
L. W. Funk, J. V. Jovanovich, R. Kawchuck, R. King, J. McKeown, C. A. Miller, D. Peterson, and D. Reimer
University of Manitoba, Winnipeg, Manitoba, Canada

Session A (continued)

AN INTEGRATED DISC-TAPE OPERATING SYSTEM FOR THE 338 BUFFERED DISPLAY COMPUTER
Jerrold M. Grochow and Thomas P. Skinner
Project MAC, Massachusetts Institute of Technology
Cambridge, Massachusetts

LUNCH

Session B (continued)

TRACK FOLLOWER - A SYSTEM FOR BUBBLE CHAMBER TRACK RECOGNITION
James P. Taylor
Massachusetts Institute of Technology
Cambridge, Massachusetts

LUNCH

## Afternoon

### Session A

Jolly Roger Inn
Prof. Donald A. Molony, Chairman

COMPUTERS IN THE LABORATORY: EDUCATION
Dr. Ronald G. Ragsdale
The Ontario Institute for Studies in Education
Toronto, Ontario, Canada

A DATA-LINK BETWEEN SMALL COMPUTER AND A CDC 6600
Sypko W. Andreae
Lawrence Radiation Laboratory, Berkeley, California

ORGANIZATION OF THE DATA-LINK SYSTEM
Robert W. Lafore
Lawrence Radiation Laboratory, Berkeley, California

TRANSMITTER OF LINK DATA OVER TELEPHONE LINES
Alan E. Oakes
Lawrence Radiation Laboratory, Berkeley, California

COFFEE

QOLFIM - AN ON-LINE FILM MEASURING SYSTEM FOR PROCESSING OF BUBBLE CHAMBER PHOTOGRAPHS
B. B. Culwick, Eli Glazer, B. Lebowitz, and Terry Meehan
Brookhaven National Laboratory
Upton, Long Island, New York

### Session B

Convention Center - Garden Grove Room 3
Prof. Philip R. Bevington, Chairman

A TUTORIAL ON NUMERICAL ANALYSIS WITH AN EMPHASIS ON ERROR ANALYSIS AND PREDICTION
Dr. Wayne A. Muth and Bruce C. Davis
School of Technology, Southern Illinois University
Carbondale, Illinois

COFFEE

PDP-8 (DISC) OPERATING SYSTEM WORKSHOP
Roger Pyle, Chairman
Digital Equipment Corporation, Maynard, Massachusetts

### Session C

ANALYTICAL INSTRUMENTATION WORKSHOP
Sponsored by Digital's PDP-8 Marketing Group

Panel Members:

Roger E. Anderson, Chairman
Lawrence Radiation Laboratory, Livermore, California

W. W. Black
Idaho Nuclear Corporation, Idaho Falls, Idaho

T. Coburn
Stanford University School of Medicine
Palo Alto, California

Jack W. Frazer
Lawrence Radiation Laboratory, Livermore, California

George Lauer
North American Aviation Science Center
Thousand Oaks, California

APPENDIX 2

AUTHOR/SPEAKER INDEX

*Abstract Only

## APPENDIX 3

## ATTENDANCE

Mr. Donald Allen
Cornell Aeronautical Laboratory
Buffalo, New York

Mr. Ernest J. Anastasio
Educational Testing Service
Princeton, New Jersey

Mr. Albert Anderson
Physics Department
Stanford University
Stanford, California

Mr. Charles N. Anderson
Edgerton, Germeshousen, and Grier, Inc.
Las Vegas, Nevada

Mr. James J. Anderson
Minneapolis-Saint Paul Sanitary District
St. Paul, Minnesota

Mr. Roger E. Anderson
Lawrence Radiation Laboratory
Livermore, California

Mr. Sypko W. Andreae
Lawrence Radiation Laboratory
Berkeley, California

(Mrs.) Carolyn A. Bailey
Lawrence Radiation Laboratory
Livermore, California

Mr. Donald Barker
Digital Equipment Corporation
Palo Alto, California

Mr. Tom Barrett
Infotronics Corporation
Houston, Texas

Mr. A. E. Beal
Digital Equipment Corporation
Palo Alto, California

Mr. L. A. Beghtol
Beckman Instruments
Fullerton, California

Mr. Roger A. Bell
The Condon Corporation
Cambridge, Massachusetts

Mr. William S. Bergen
Mobil Research and Development Corp
Paulsboro, New Jersey

Mr. James G. Bever
Massachusetts Institute of Technology
Cambridge, Massachusetts

Prof. Philip R. Bevington
Physics Department
Stanford University
Stanford, California

Mr. W. Wayne Black
Idaho Nuclear Corporation
Idaho Falls, Idaho

Mr. E. O. Boutwell
Compata
Tarzana, California

Mr. Ronald W. Brandenburg
Argonne National Laboratory
Argonne, Illinois

Mr. J. E. Braun
California Computer Products
Anaheim, California

Mr. J. A. Brussolo
Applied Dynamics
Ann Arbor, Michigan

Mr. H. C. Buchholtz
X Ray Department
General Electric Company
Milwaukee, Wisconsin

Mr. Donald H. Bundy
Edgerton, Germeshousen,
and Grier, Inc.
Las Vegas, Nevada

Mrs. Elinor Burns
CAP Users Group
Honeywell, Inc.
Framingham, Massachusetts

Mr. Walter R. Burrus
Tennecomp, Inc.
Oak Ridge, Tennessee

Mr. R. E. Byrd
Electro-Optical Systems
Pasadena, California

Mr. Stephen F. Carlson
Minneapolis-Saint Paul
Sanitary District
St. Paul, Minnesota

Mr. Robert Carmichael
Digital Equipment Corporation
Anaheim, California

Mr. David L. Carson
Texas Institute of Rehabilitation
and Research
Houston, Texas

Mr. Ron Carter
Digital Equipment Corporation
Anaheim, California

Mr. Carton
COSEM. CSF.
France

Mr. Robert Chaminade
Centre D'Etudes Nucleaires de Saclay
France

(Miss) Margaret Child
Langley Porter Neuropsychiatric Institute
San Francisco, California

Mr. Timothy Coburn
Stanford University
Stanford, California

Mr. Howard D. Cohen
Autonetics
Anaheim, California

Mr. Howard A. Cohodas
Picker Instruments
Cleveland, Ohio

Mr. Carter L. Cole
Automatic Information Management
Encino, California

Mr. Gary Comisky
Victoreen
Cleveland, Ohio

Mr. Ronald L. Compton
Beckman Instruments, Inc.
Fullerton, California

(Mrs.) Angela J. Cossette
DECUS
Digital Equipment Corporation
Maynard, Massachusetts

Mr. Martin L. Cramer
Applied Data Research
Princeton, New Jersey

Mr. S. Crespi-Reghizzi
University of California at Los Angeles
Beverly Hills, California

Mr. Henry S. Culver
Engis Equipment Company
Sunnyvale, California

Mr. Bruce C. Davis
Southern Illinois University
Carbondale, Illinois

Mr. Donn E. Deal
Perkin-Elmer Corporation
Seattle, Washington

Mr. Richard J. DeJohn
Digital Equipment Corporation
Palo Alto, California

Mr. James R. Delorey
San Francisco Bay Naval Shipyard
Vallejo, California

Mr. Bradley Dewey
Digital Equipment Corporation
Maynard, Massachusetts

Mr. Russ Doane
Digital Equipment Corporation
Maynard, Massachusetts

Mr. C. G. Donahoe
San Francisco Bay Naval Shipyard
Vallejo, California

(Mrs.) Evelyn Dow
Digital Equipment Corporation
Maynard, Massachusetts

Mr. James L. Downs
Badger Meter Manufacturing Company
Milwaukee, Wisconsin

Mr. Roger A. Due
Naval Ammunition Depot
Crane, Indiana

Mr. William Duncan
Digital Equipment Corporation
Denver, Colorado

Mr. Glenn R. Elliott
Sandia Corporation
Albuquerque, New Mexico

Mr. Philip J. Erdelsky
California Institute of Technology
Pasadena, California

Mr. Donald R. Fanshier
Lawrence Radiation Laboratory
Berkeley, California

Mr. William G. Feeney
Chevron Research Company
Richmond, California

Mr. Gordon A. Fell
University of California
Berkeley, California

Mr. Frederick C. Fensch
Parke Davis and Company
Ann Arbor, Michigan

Mr. Eugene R. Fisher
Lawrence Radiation Laboratory
Livermore, California

Mr. Thornton R. Fisher
Stanford University
Stanford, California

Mr. D. M. Forsyth
Harvard University
Cambridge, Massachusetts

Mr. Jack W. Frazer
Lawrence Radiation Laboratory
Livermore, California

Mr. Theo Frensch
Hewlett Packard
Palo Alto, California

Mr. David Friesen
M.I.T./L.N.S.
Cambridge, Massachusetts

Mr. David G. Frutchey
Beckman instruments
Fullerton, California

Mr. Wolf Gellinek
Siemens AG
Germany

Mr. Eli Glazer
Brookhaven National Laboratories
Upton, Long Island, New York

Mr. William F. Godwin
Educational Testing Service
Princeton, New Jersey

Mr. Peter Goodeve
University of California
Berkeley, California

Dr. Peter H. Grassmann
Siemens AG
Germany

Mr. Larry Green
University of California
Santa Barbara, California

Mr. Richard P. Gruen
Digital Equipment Corporation
Palo Alto, California

Mr. Russell B. Ham
U.S. Public Health Service
Winchester, Massachusetts

Mr. Perry Harris
Digital Equipment Corporation
Maynard, Massachusetts

Mr. Richard A. Harshman
University of California
Los Angeles, California

Mr. David E. Hartig
Information Control Systems, Inc.
Ann Arbor, Michigan

Mr. R. L. Heath
Idaho Nuclear Corporation
Idaho Falls, Idaho

Dr. Frederick W. Hegge
Arizona State University
Tempe, Arizona

Mr. G. L. Helgeson
Helgeson Nuclear Services, Inc.
Pleasanton, California

Ms. Sara B. Helmick
Los Alamos Scientific Laboratory
Los Alamos, New Mexico

Mr. Jo-L Hendrickson
Lawrence Radiation Laboratory
Livermore, California

Mr. Robert Hippe
Automatic Information Management
Encino, California

Mr. G. Hunter
Digital Equipment Corporation
Albuquerque, New Mexico

Mr. Motoi Ieta
Kokusai Electric Company, Ltd.
Montebello, California

Mr. Robert Jahncke
Beckman Instruments Corporation
Fullerton, California

Mr. David W. Jenson
Lawrence Radiation Laboratory
Berkeley, California

Mr. Richard G. Johnson
Datacraft, Inc.
Gardena, California

Miss Shirley J. Johnstone
Fairchild Instrumentation
Sunnyvale, California

Mr. John Allen Jones
Digital Equipment Corporation
Maynard, Massachusetts

Mr. Richard A. Karlin
Victoreen
Wilmette, Illinois

Mr. Bruce Kennedy
Datacraft, Inc.
Gardena, California

Mr. Peter Ladefoged
University of California
Los Angeles, California

Mr. Robert Lafore
Lawrence Radiation Laboratory
Berkeley, California

Mr. Donald Larsen
Digital Equipment Corporation
Anaheim, California

Mr. Kenneth Larsen
Digital Equipment Corporation
Palo Alto, California

Mr. George Lauer
Science Center of North American Rockwell Corporation
Thousand Oaks, California

Mr. David E. Leney
Digital Equipment Corporation
Maynard, Massachusetts

Mr. Gerald Lessard
Digital Equipment Corporation
Anaheim, California

Mr. Nelson Liberman
Perkin-Elmer Corporation
Inglewood, California

Mr. P. H. Lindsay
University of California at San Diego
San Diego, California

Mr. John B. Locke
Rutgers University
New Brunswick, New Jersey

Mr. S. William Logan
Eli Lilly and Company
Indianapolis, Indiana

Dr. Carl D. Lowenstein
Scripps Institute of Oceanography
San Diego, California

Mr. F. Luccio
Politecnico of Milano
Italy

Mr. George W. Lusher
Minneapolis-Saint Paul Sanitary District
St. Paul, Minnesota

Mr. Donald E. Lusty
Electronic Associates, Inc.
El Segundo, California

Mr. J. S. MacDonald
University of British Columbia
Vancouver, British Columbia, Canada

Mr. Frederick W. Macondray
Lawrence Radiation Laboratory
Berkeley, California

Mr. Fred I. Magee
Sandia Corporation
Albuquerque, New Mexico

Miss Virginia L. Maier
Department of Psychology
University of California at San Diego
La Jolla, California

Mr. Frank Mannino
Victoreen, Inc.
Cleveland, Ohio

Mr. C. E. Mannix, Jr.
University of California
Los Angeles, California

Mr. James R. Masek
Electro-Optical Systems, Inc.
Pasadena, California

Dr. Masterpolo
Douglas Aircraft Company
Long Beach, California

Mr. Andrew J. McGill
Digital Equipment Corporation
Maynard, Massachusetts

Mr. R. L. McInturff
Digital Equipment Corporation
Palo Alto, California

Mr. A. R. McKenzie
Standard Telephones and Cables Ltd.
Cockfosters, Herts, England

Mr. Guy G. McMillan
Lawrence Radiation Laboratory
Livermore, California

Mr. Richard J. McQuillin
Inforonics, Inc.
Cambridge, Massachusetts

Mr. F. C. Mears
Mobil Oil Corporation
Torrance, California

Mr. Richard M. Merrill
Digital Equipment Corporation
Maynard, Massachusetts

Mr. Robert M. Metcalfe
M.I.T. Sloan School of Management
Cambridge, Massachusetts

Mr. S. A. Michaelson
Programming Services
Tarzana, California

Mr. Richard Mikula
Stanford Research Institute
Menlo Park, California

Prof. Donald A. Molony
Rutgers University
New Brunswick, New Jersey

Mr. Gary B. Morgan
Idaho Nuclear Corporation
Idaho Falls, Idaho

Mr. K. R. Morin
St. Paul's Hospital
Vancouver, British Columbia, Canada

Mr. James J. Murphy, Jr.
Digital Equipment Corporation
Maynard, Massachusetts

Mr. Wayne A. Muth
Southern Illinois University
Carbondale, Illinois

Mr. Paul A. Naro
Mobil Oil Corporation
Princeton, New Jersey

Mr. Charles Newman
Information Control Systems, Inc.
Ann Arbor, Michigan

Mr. James B. Niday
Lawrence Radiation Laboratory
Livermore, California

Mr. Donald A. Norman
University of California
La Jolla, California

Mrs. Carole H. O'Hare
University of California
La Jolla, California

Mr. Juzo Ohkuma
National Research Council
Ottawa, Ontario, Canada

Mr. Leif Ohlsen
Auto-chemic
Stockholm, Sweden

Mr. Kenneth H. Olsen
Digital Equipment Corporation
Maynard, Massachusetts

Mr. Gene Olson
Digital Equipment Corporation
Anaheim, California

Mr. J.J.H. Park
National Research Council
Ottawa, Ontario, Canada

Mr. Gerard Paul
Perkin-Elmer Corporation
Norwalk, Connecticut

Miss Maxine L. Paulsen
Medical Systems Development Laboratory
Washington, D. C.

Mr. Bud R. Pembroke
Computer INS Network
Salem, Oregon

Mr. R. Perelman
Victoreen, Inc.
Cleveland, Ohio

Mr. Ronald B. Perry
Argonne National Laboratory
Argonne, Illinois

Mr. Roger L. Peterson
Lawrence Radiation Laboratory
Livermore, California

Dr. Lawrence R. Pinneo
Stanford Research Institute
Menlo Park, California

Mr. Vernon G. Poulter
Digital Equipment Corporation
Washington, D.C.

Mr. Robert M. Powell
Lawrence Radiation Laboratory
Berkeley, California

Mr. Roger C. Pyle
Digital Equipment Corporation
Maynard, Massachusetts

Dr. Ronald Ragsdale
Ontario Institute for Studies in Education
Toronto, Ontario, Canada

Mr. D. R. Reimer
University of Manitoba
Winnipeg, Manitoba, Canada

Mr. Clyde E. Rettig, Jr.
M.I.T. Sloan School of Management
Cambridge, Massachusetts

Mr. William E. R. Richards
Astrodata
Anaheim, California

Mr. C. Wendell Richardson
Phillips Petroleum Company
Idaho Falls, Idaho

Mr. Freidrich Riess
Stanford University
Stanford, California

Mr. Donald W. Roland
Beckman Instruments
Fullerton, California

Mr. Jerome A. G. Russell
The Institute of Medical Sciences
San Francisco, California

Mr. Stephen Russell
Stanford University
Stanford, California

Mr. David N. Samsky
Booz-Allen Applied Research, Inc.
Albuquerque, New Mexico

Mr. E. L. Samson
Picker Nuclear
Los Angeles, California

Mr. John Santer
Stanford University
Stanford, California

Mr. Theodore R. Sarbin, Jr.
University of California
Berkeley, California

Mr. Robert A. Saunders
Stanford University
Stanford, California

Dr. Lothar Seifert
Siemens America, Inc.
New York, New York

Mr. Robert W. Shannon
Vidar Corporation
Mountain View, California

Mr. Ted Shapin
Beckman Instruments
Fullerton, California

Mr. Philip K. Shelton
Lockheed-California Company
Burbank, California

Dr. J. Gilmour Sherman
Arizona State University
Tempe, Arizona

Mr. Philip D. Siemens
Lawrence Radiation Laboratory
Livermore, California

Mr. Thomas P. Skinner
M.I.T. Project MAC
Cambridge, Massachusetts

Mr. Robert W. Skyles
Vidar Corporation
Mountain View, California

Mr. Douglas V. Smith
General Electric Tempo
Santa, Barbara, California

Mr. John S. Storck
General Foods Corporation
White Plains, New York

Dr. Michael G. Strobel
Dalhousie University
Halifax, Nova Scotia, Canada

Mr. George Sugar
ESSA
Boulder, Colorado

Mr. Dwight Sutton
Arizona State University
Tempe, Arizona

Mr. R. L. Swenson
Lawrence Radiation Laboratory
Livermore, California

Mr. Harry M. Taxin
Hughes Aircraft Company
Culver City, California

Mr. James P. Taylor
M.I.T./L.N.S.
Cambridge, Massachusetts

Dr. M. M. Taylor
Defence Research Establishment
Downsview, Ontario, Canada

Mr. Paul Tharman
X Ray Department
General Electric Company
Milwaukee, Wisconsin

Mr. Richards A. Thomas
Lawrence Radiation Laboratory
Livermore, California

Mr. C. J. Thompson
Atomic Energy of Canada Ltd.
Ottawa, Ontario, Canada

Mr. T. H. Van Gorp
Engis Equipment Company
Morton Grove, Illinois

Mr. William F. Weiher
Stanford University
Stanford, California

Mr. B. Michael Wilber
Stanford Research Institute
Menlo Park, California

Miss Barbara Williams
Mobil Research and Development
Menlo Park, California

Mr. Phil Williger
Mobil Oil Corporation
Torrance, California

Mr. Donald Wilson
University of California
Berkeley, California

Mr. William G. Witmore
Picker Nuclear
San Francisco, California

Ms. Marion E. Wolfe
Lockheed California Company
Burbank, California

Mr. Thomas Woods
Beckman Instruments, Inc.
Fullerton, California

Mr. John Wyman
Information Control Systems
Ann Arbor, Michigan

Mr. Ronald Zane
Lawrence Radiation Laboratory
Berkeley, California

Mr. Mark Zeligman
Lawrence Radiation Laboratory
Livermore, California

PRINTED IN U.S.A.