NASA TECHNICAL NOTE

NASA TN D-5623

c. 1

NASA TN D-5623

THE DATA REDUCTION LABORATORY: AN AID TO THE SPACE SCIENTIST

by Barbara A. Walton, Frank A. Keipert, and John J. Quann

Goddard Space Flight Center
Greenbelt, Md.

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • FEBRUARY 1970

1. Report No. NASA TN D-5623	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle THE DATA REDUCTION LABORATORY: AN AID TO THE SPACE SCIENTIST		5. Report Date February 1970
		6. Performing Organization Code
7. Author(s) Barbara A. Walton, Frank A. Keipert, and John J. Quann		8. Performing Organization Report No.
9. Performing Organization Name and Address Goddard Space Flight Center Greenbelt, Maryland 20771		10. Work Unit No.
		11. Contract or Grant No.
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, D. C. 20546		13. Type of Report and Period Covered Technical Note
		14. Sponsoring Agency Code
15. Supplementary Notes		

16. Abstract

The Data Reduction Laboratory was designed as a system that can be used easily by the space scientist. This system allows him to interface with his data; it not only processes data in real-time, but is also capable of being 'programmed' rapidly. The key to this system is the use of a combination of dialog and SYNTAX. Dialog is used where possible–in those areas that can be preconceived, where the possibilities are finite and either limited or subject to limitations—SYNTAX where not. The input and output, therefore, are defined through dialog, experiment computation by SYNTAX.

Relatively complicated processors can be written, checked out, and operating in terms of hours rather than months. Anomalies in the data can be observed and compensated for; data correlations are facilitated. The time lag between launch and analysis is considerably shortened.

17. Key Words Suggested by Author Data-Reduction Laboratory Dialog SYNTAX User Terminal		18. Distribution Statement Unclassified—Unlimited	
19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 13	22. Price* $3.00

*For sale by the Clearinghouse for Federal Scientific and Technical Information
Springfield, Virginia 22151

CONTENTS

INTRODUCTION 1

SYSTEM DESCRIPTION 3

Dialog 7

SYNTAX. 7

Tree 8

EXAMPLE—ISOTOPIC ABUNDANCE/GALACTIC COSMIC RAY EXPERIMENT. 8

FUTURE EXPANSION 10

ACKNOWLEDGMENTS 10

References 11

THE DATA REDUCTION LABORATORY: AN AID TO THE SPACE SCIENTIST*

by
Barbara A. Walton, Frank A. Keipert, and John J. Quann
Goddard Space Flight Center

INTRODUCTION

The Data Reduction Laboratory (DRL) is a facility developed by the Goddard Space Flight Center to aid experimenters in the rapid analysis of their flight data telemetered from spacecraft. Included in this primary goal were many functional requirements which had to be achieved for the system to be operationally effective. These requirements include complete generalization of spacecraft telemetry as input; generalization of the type and format of data display; ability to develop a processor for any experiment regardless of form or complexity; capability of processing multiple experiments simultaneously, all without requiring the user to perform any machine language coding.

It is apparent that as a system the Data Reduction Laboratory has many facets in both hardware and software, which cannot be fully understood without the benefit of some background information about spacecraft telemetry processing.

The first U. S. satellite to transmit scientific data successfully was the Explorer I, launched in January 1958. The first NASA satellite to successfully return scientific data was the Vanguard II, launched in February 1959. A multitude of satellites has been launched since then and a large effort has been made in the processing and analysis of the data from experiments on board these spacecraft.

Figure 1 shows a generalized space information system for a typical satellite-borne experiment. Since the data source is a number of sensors on the spacecraft, the sensor outputs are routed through signal and data processing circuits to the central data system, where data from all sources are combined. The telemetry system transmits the data to the central data processing facility on the ground, where the data are extracted from the noise and reduced to a standard "raw" form, together with time, spacecraft orbit, and attitude information. The data are further reduced for analysis by the experimenter, who may:

check all data to choose the most interesting portions;

perform a detailed analysis of portions containing data of special interest;

map a data field in either time or space;

analyze in detail all data when such action is required to provide statistical significance or a sufficiently fine-scaled mapping grid.

*Presented at the August 28, 1969 24th National Conference of the ACM and published in the "Proceedings of the 24th National Conference," ACM Publication P-69.

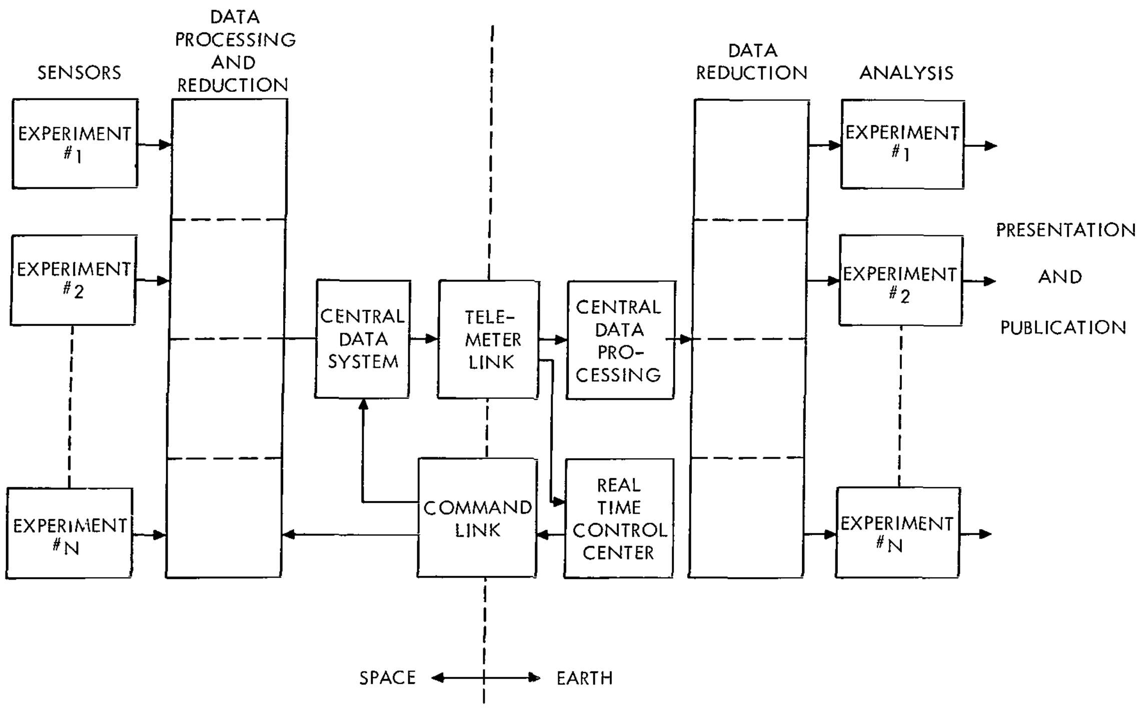


Figure 1—Generalized space information system (Reference 2).

The volume of flight data generated for most experiments is so great that it requires electronic data processing (References 1 and 3). Since the data processing cannot begin until the associated computer program has been written, analysis of the data and publication of experiment results may be delayed by as much as 1 or 2 years after the date of launch (Reference 2). Although completion of the program before launch should reduce the delay between the collection of data and the dissemination of results, such is rarely the case with new experiments because spacecraft preparation takes precedence over software development: also, radically new experiments may require some flight data before the program can be completed. The development of analysis programs before launch does not always preclude delays in obtaining results: conditions which often delay results include anomalous behavior of the spacecraft or experiment, interference among experiments or spacecraft systems, or unexpected characteristics of the phenomemon under observation.

The processing of data for final analysis requires a great deal of computer programming. Preliminary programs are usually completed from 3 months before launch to 9 months after launch. The relationship between the date of completion of preliminary programs and the launch date is determined by the novelty and complexity of the program. For an experiment similar to that flown on a previous mission, the preliminary program may be completed well before launch. Final programs, incorporating changes found necessary during the preliminary processing, are usually completed from 6 months to 2 years after launch.

The Data Reduction Laboratory provides a means for rapid presentation of processed data to experimenters and other users, either in real time or from stored data. The laboratory uses a computer with a large storage capacity and associated display and output devices (Figure 2). The laboratory allows the rapid generation, checkout, and modification of processing programs to meet the individual experimenter's needs and allows the presentation of data to meet certain operational needs in close cooperation with the project Operational Control Centers. The laboratory provides data to experimenters with delays measured in minutes rather than in days. The Data Reduction Laboratory is designed to service remote terminals so that experimenters can receive critical data at their laboratories.

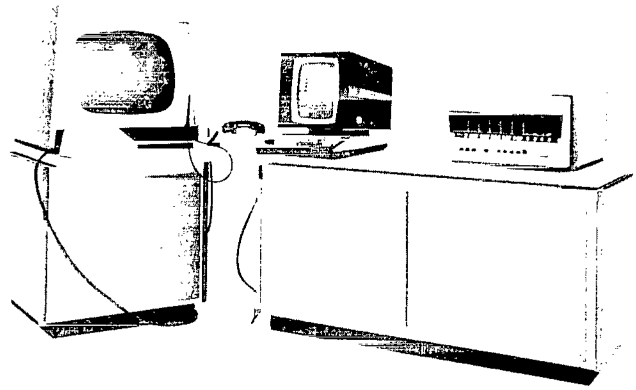


Figure 2—Data reduction laboratory user terminal.

The aim of the DRL is for the space scientist (user) to communicate his requirements directly to the computer. Not all space scientists are computer experts; indeed, the majority have no experience in programming. The burden of communication has therefore been placed upon the computer. In those areas in which choices can be pre-conceived, it is possible to construct a dialog presenting these choices to the user for selection.

Data structure is unique to each user and must therefore be described through an application oriented SYNTAX.

The programs developed by the Data Reduction Laboratory will be incorporated into a program library. As more experience is obtained with the methods of handling and presenting data, the library of laboratory functions available to the user will expand. Concurrent with this progress, the capabilities of the laboratory will increase, giving the user more flexibility in the processing of data.

The DRL enables the user to select both the data to be displayed and the type of presentation. The user may defer making a decision as to which data are to be output as hard copy. The user may view data plots on a cathode ray tube display and selectively obtain a hard copy of data of interest.

SYSTEM DESCRIPTION

The DRL (Figure 3) is designed around a Univac 1108 multiprocessor equipped with a full complement of peripheral equipment.

Real-time data, analog tape-recorded data, or simulated telemetry data is input to the computer by a stored-program telemetry processor. This processor performs data synchronization and reformatting and will time-tag the frames of data transferred to the computer. The computer processes the frames of data and converts them as requested by the user. The output data is displayed on cathode ray tubes on the communication and data display consoles. If desired by the

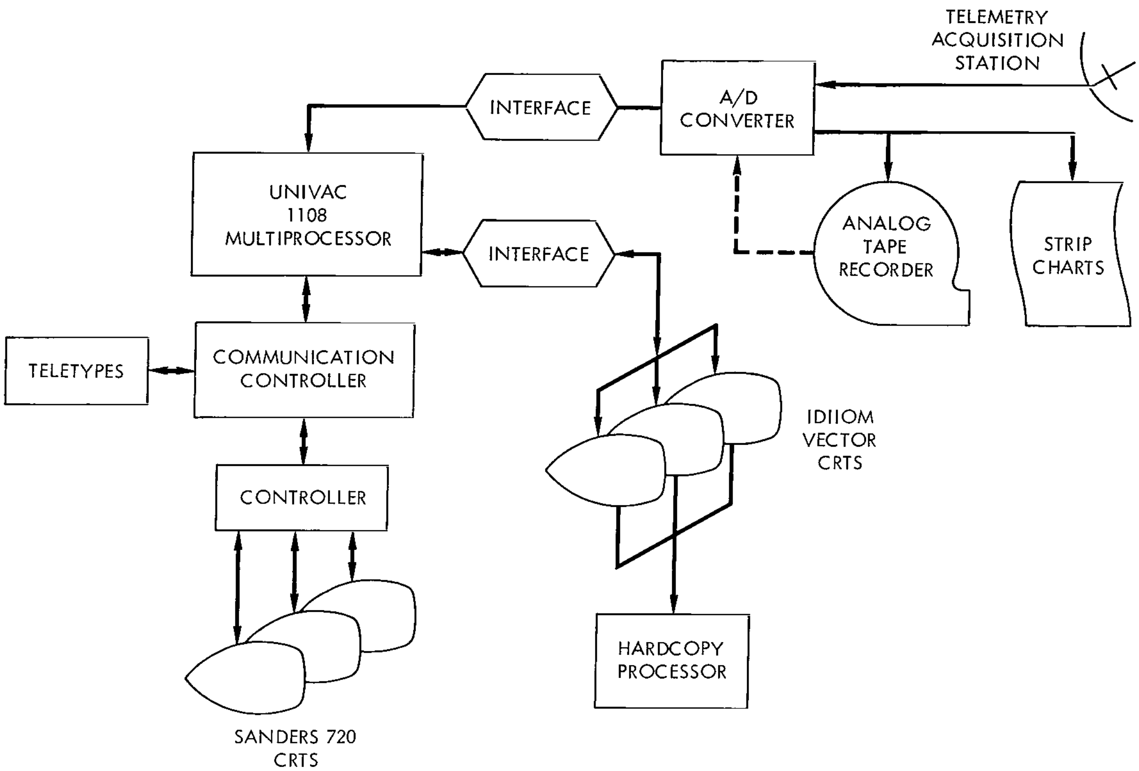


Figure 3—DRL system diagram.

users, permanent records of specified data may be produced using the hard copy processor or printer.

The communication consoles (Sanders 720) provide the user with a means of communicating with the computer. Computer output data are displayed on a cathode ray tube capable of flicker-free display of approximately 1000 alphanumeric characters. The user exercises control over the computer by means of an alphanumeric keyboard, edit-function keys, and special-function switches. During on-line program design, assembly, and checkout, the communication consoles provide a medium for dialog between the user and the computer. The communication consoles are also used for on-line presentation of selected data words and for initiating data requests from remote sites.

The data display consoles (IDIIOM) provide the user with alphanumeric or graphic data presentations. Each console display has an addressable raster of 1024 $\times$ 1024 positions and a set of 64 alphanumeric characters. Each console is capable of flicker-free display of a mix of approximately 1000 randomly positioned characters, lines and points, with all consoles operating simultaneously but using independent data. Format for the various presentations is determined by dialog before program assembly, but the selection for display of a particular presentation will be controlled by the function switches on the consoles. The user effects data manipulation through communication with the computer by means of an alphanumeric keyboard and a light pen.

The Data Reduction Laboratory was designed to operate in a multiprogramming environment with multiple users. The DRL software is considered one program by the multiprogramming executive (UNIVAC 1108 EXEC 8). This program has its own resource allocation facilities to enable it to handle multiple users. Each user has his own unique data but shares reentrant routines with the other users. In addition to being reentrant, all DRL routines are self-locating so that they can be moved easily in and out of the core space allocated to the DRL program. Mass storage (drum) is used for routines not in current use, for large amounts of data and for the "library" of user responses.

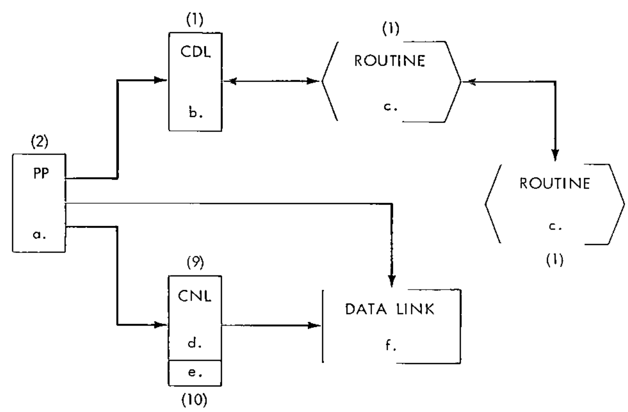
The allocation of resources among the various DRL users is performed by a central control program (DRL nucleus). The DRL nucleus program exists as the resident portion of the system, where system here refers to the DRL program. Its two major functions are:

1. To initialize the system to the point that several users may sign on

2. To selectively switch between activities within DRL while maintaining the integrity of all data tables and index registers.

The UNIVAC EXEC 8 system is in charge of the bulk symbiont traffic and the allocation of machine resources and compute-time above that set aside for DRL.

A run being performed for a single DRL user is called a *process*. A process may include a number of *activities* which perform specific functions of the process. An activity (Figure 4) may be thought of as an "instruction-thread" which is started and stopped (at specified points) by the nucleus. (While the instruction-thread of one activity is stopped, that of another is underway.) All of the activities of a given user-process operate from the same process-pointer table (PP, address in register 2). Each activity consists of a code link (CDL, address in register 1) and a control link (CNL, address in register 9). The control link is a user-peculiar data area which may contain pointers to other user-peculiar data links. The code link, as reentrant code, may be simultaneously in use by several activities and may dynamically call other reentrant routine code. As reentrant routines are dynamically referenced from the code link or other routines, registers are saved/restored in a save area at the bottom of the control link.



The above illustrated code/data links are:

a. A pointer-vector of all data links and code links of the process One-word entries contain either the current drum address or current core address of each link

b. The main code link of one of the activities of the process

c. A reentrant routine dynamically called by the CDL or by another routine

d. The activity data area of the CNL

e. The register save area of the CNL

f. A data link used by the activity, identified in (d)

Figure 4—A DRL activity.

The first operation performed by or for every DRL routine is to "load the address of itself into register 1." This permits every routine to be dynamically positioned/repositioned in core wherever core-space is available. All jumps within routines are thus "relative base register 1." Note that if linkage is "direct," the current value of register 1 (base of the calling routine) must be saved before register 1 is loaded with the base of the called routine.

All DRL code is written so that it can be time-shared by any number of activities. The nucleus switches activities simply by changing base register constants. This means that the instruction thread of two or more activities will from time to time go through the same routine or code link.

In order for the instruction-thread of one activity not to conflict with that of another, none of the commonly used code can be modified. However, if necessary, an activity-peculiar instruction may be performed by placing it on the user's control link and executing remote; also an activity peculiar routine may be placed in memory on a data link and referenced through an address placed on the control link.

In order for each reentrant routine to dynamically call another without restriction, there must be a register-save area peculiar to each activity where the routine can save/restore registers upon entry/return. This is accomplished by means of a "B10 Save Stack" at the bottom of the code link. An activity begins with register B10 containing the address of the top of the save-area. Each time a register is saved, the address in B10 is incremented by 1. This saving and incrementing takes place each time a routine is called. Each time a register is restored (upon return from a routine), the address in B10 is decremented.

When an activity first becomes active, it is represented in core by a user-peculiar control link and a user-common code link, and control initially passes to the first instruction of the code link. The sequence of instructions on the code link may include calls to subroutines. If the subroutine is internal, it is coded relative to the same base register 1 address as the code link. However, note that the code link may not exceed a size of 2047 words (the maximum for all DRL data links). All calls to external routines are dynamic in the sense that if the routine is not in core at the time the call is made, it is brought in.

While almost all code of DRL is user-common, all data of DRL is essentially user-unique. However, all data links are dynamically positioned into core (similar to code links) by means of the Data Link Package. Using this package of routines, a given array or table is not opened until it is actually needed by the code, and is closed as soon as it is no longer needed.

In general, code links and data links are always kept as small as possible, and cannot exceed 2047 words. There is one access-word (w-word) on the PP-table for each data link of a user-process. When the link is in core, the w-word contains a core address. When the link is on drum the w-word is a drum address.

Data links are formatted in two basic ways; Single-link and Multi-link. A single-link is pointed to directly by a PP w-word. A multi-link is one of several sequential links of a pool, and the address of each link of the pool is carried on a link-pointer (LP) table (i.e., the LP carries w-words for each link of the pool). In this case, the PP w-word points to the LP which in turn points to the current in-core link.

Data links are accessed in two basic ways: serial-access and parallel-access. A serial-access link may be used by only one activity at a time. As link numbers are passed from one activity to the next, the first activity must have terminated before the next activity can open and use the links. A parallel-access link (single or multi) is a type of data link which may be simultaneously in use by several activities. In this case, a given link cannot be closed until all activities have closed it, and the current drum/core address is known to all activities which use the link. Parallel-access link usage requires that the activities using the link do not interfere with each other.

At certain places on its instruction-thread an activity may be in position to share time, even though it is not in position to share core. An activity *must* make a time-sharing call before elapse of its allocated timeout interval or it will be aborted looping.

At certain places on its instruction-thread an activity may be in position to share core, even though it is not yet finished. An activity *must* make a core-sharing call before elapse of its allocated core residence time.

Dialog

The DRL dialog facility is one of the major original and continuing objectives of the DRL development. The activity switching environment introduced above was developed as a practical answer to the combined requirements of human-speed dialog and high-speed telemetry data reduction.

Dialog may be thought of as a user-oriented question-and-answer sequence in which a given answer (produced by "filling in the blanks" of an on-line scope image) provides the basis for the next logical question (image); and where "mistakes" (in the on-line typed-in answer) cause the proper explanation and try-again feedback.

When the on-line user is in dialog, he is actually executing an activity called DCP (Dialog Control Program). The output image data, the input response data, and instruction code required to process the response make up a dialog node. A network of dialog nodes is required to complete a dialog question/answer sequence. The entire sequence (or network) of nodes required for a given on-line DRL user support capability must be more than a fixed computer instruction sequence supported by operator instructions. The network represents a delicately balanced man/machine interface in which the requirements of the computer (node network) must be properly communicated to the on-line user and the requirements of the on-line user must be met with acceptable computer response.

When the user signs on at an initially blank on-line scope, he begins at the starting node/network. From this point, the network is designed to get the user to his required DRL function as quickly and as simply as possible.

SYNTAX

A telemetry-reduction user-oriented language called GORTRAN (Group Organization Translator) permits the user to give a free-form definition of his data reduction algorithm. DRL users must be familiar with a simplified set of syntactical rules for constructing such algorithms.

Using the proper sequence of statements, the user may identify telemetry-channel values, numerical constants (literals), execution-time parameters, etc. He may specify arithmetic combinations of these values to produce time-sequenced bursts of data to be displayed. To control data selection and computation, the user may define tests to be performed on a frame-by-frame basis or time-sequenced burst basis. Other statements are used to define selection and synchronization of telemetry data pools, data arrays, etc.

The on-line user enters into SYNTAX mode from dialog by reaching a dialog node which starts an activity called CLC (Conversational Language Control). When in SYNTAX mode, the format of the on-line scope image changes from "fill-in-the-blank" to "scratchpad" for type-in and display of free-form statements. The scratchpad is divided into a top look-only area and a bottom type-in area. The latest set of SYNTAX statements that have been processed are in the look area, and new statements are typed in below.

Each time a new (set of) statement(s) is sent by the on-line user, it is scanned and any errors are conversationally returned at the time they occur, with the sentence(s) containing the error appropriately marked. To correct these errors the on-line user enters SYNTAX-alter mode, makes the correction, and then resumes SYNTAX compile mode (the normal mode).

Tree

The DRL tree is the hierarchical library structure required for support of on-line dialog, for retrieval of dynamically linked routines, for filing of user processes and data, etc. The overall tree function is to make information produced by earlier DRL runs available to following DRL runs. For example, having defined and executed a telemetry-reduction process, the on-line user may wish to use the same process on following days without performing the definitional dialog/SYNTAX again.

EXAMPLE—ISOTOPIC ABUNDANCE/GALACTIC COSMIC RAY EXPERIMENT

A simple example will be used to demonstrate the actions of a space scientist who wishes to look at his data in a rather simple way before determining how best to display it for final analysis and publication. This example uses digital tape as input and high speed printer output (Reference 4).

This experiment occupies channels 42, 43 and 44 of the main commutator. Each word is 9 bits long where bit 1 is the least significant.

Channel 42:	Bits 9 thru 2	= DELTA E SCINTIL
	Bit 1	= GAIN
Channel 43:	Bits 9 thru 2	= E-DELTA E SCINTIL
	Bit 1	= ERROR

Channel 44: Bits 9 thru 5 = M

Bits 4 thru 1 = N

When ERROR = 1, data from channels 42 and 43 are to be ignored. When GAIN = 1, the data in channels 42 and 43 are to be multiplied by 8. Channel 44 is used to compute D as follows:

$$D = (M + 32)2^N - 32$$

At the alphanumeric scope the first node of the dialog is already displayed (Figure 5).

The user responds by choosing to create a data reduction process. At this point the user defines his data structure through the DRL application oriented syntax. The solution of this example is shown in Figure 6.

```
0101. START DIALOG

FOR LISTING PURPOSES GIVE YOUR NAME B A WALTON

PICK, BY NUMBER, FROM THE FOLLOWING LIST THAT DRL
CAPABILITY WHICH YOU WISH TO USE 3

1.    UPDATE DIALOG NETWORK (SYSTEM PROGRAMMER
      ONLY).
2.    CHANNEL DUMP FROM OGO EDIT TAPE.
3.    CREATE A DATA REDUCTION PROCESS.
4.    MODIFY AN EXISTING DATA REDUCTION PROCESS.
5.    ENTER INPUT DIALOG FOR OGO EDIT TAPE.
6.    ENTER A NEW VEHICLE DESCRIPTION (DPE ONLY).
7.    MODIFY OR DELETE AN OLD VEHICLE DESCRIPTION
      (DPE ONLY).
8.    TERMINATE USE OF THIS PROCESSING STATION.
IF SELECTION 1 WAS MADE GIVE PASSWORD______
```

Figure 5.

First bit 1 of channel 43 is tested for 0; 1 would indicate an error and control would pass to THEN (ERR) and new values for DELE and EDELE would not be computed. If bit 1 of channel 43 is 0 a value is computed for DELE: (a) bits 2 through 9 of channel 42 are used; (b) they are multiplied by 1 if bit 1 of channel 42, the GAIN, is 0; (c) otherwise they are multiplied by 7 + 1 or 8. A value is then computed for EDELE in the same manner as DELE using bits 2 thru 9 of channel 43.

D is computed according to the formula using the DRL interger exponent function EXPI and the values of DELE, EDELE and D are made available to the output program. FINI signals the end of the definition; in this case there are no errors detected, and the system responds by entering the output definition dialog (Figure 7).

At this point the user must have some idea of the format he desires for his printout and the magnitude of the results to be displayed. Through subsequent dialog the user can fully describe the output shown in Figure 8.

This description of the data processing requirements may be saved in the DRL library

```
LOOK SIZE 0
SYNTAX: COMPILE MODE

GROUP  :ISOTOP:
TEST   (ERR)   : 43(1-1).EQ.'0' :
HOLD   (DELE)  : (42(1-1)*'7'+'1')*42(9-2) :
HOLD   (EDELE): (42(1-1)*'7'+'1')*43(9-2) :
THEN   (ERR)   ::
HOLD   (D)     : (44(9-5)+'32')*EXPI<'2', 44(4-1)>-'32' :
OUTPUT::
FINI   ::
```

Figure 6.

and used another day, perhaps with different data or modifications.

FUTURE EXPANSION

Concurrent with implementation and use of the DRL, research and development in the areas of improved display and user-system communications will continue. Areas requiring improvements will also be suggested by the users' experience with the laboratory. Possible improved display techniques could include three-dimensional and color displays.

Studies of human perception and display data assimilation have been initiated. These studies will evaluate the effectiveness of various display media and will lead to the development of new display devices and equipment. Research aimed towards equipment development will be paralleled by efforts towards more meaningful presentation of data.

0230.FOPN00

YOU ARE ABOUT TO BEGIN TO DESCRIBE OUTPUT TO THE PRINTER. LISTED BELOW ARE USEFUL DEFINITIONS.

AN OUTPUT SET IS A SELECTION OF IMAGES WHICH MAY APPEAR INTERSPERSED IN ONE CONTIGUOUS OUTPUT LISTING. MULTIPLY OUTPUT SETS ARE PERMITTED.

IMAGES ARE:

LITERAL ELEMENTS - CONSTANT VALUES SPECIFIED BY THE USER

DATA ELEMENTS - CONTENTS OF VARIABLE DATA FIELDS

LINE - COLLECTION OF ELEMENTS WHICH MAKE UP A SINGLE PRINT LINE

BLOCK - A COLLECTION OF LINES WHICH ARE TO BE PRINTED AS A UNIT

LOGICAL PAGE - A PREDEFINED ARRANGEMENT OF BLOCKS AND LINES TO BE PRINTED AS A UNIT

HEADER LINE - A LINE OF ELEMENTS ASSOCIATED WITH A PARTICULAR LINE, BLOCK, LOGICAL PAGE, OR OUTPUT SET

DEPRESS SEND BLOCK TO ENTER OUTPUT DIALOG

Figure 7.

OGO-III		B-06	MC DONALD/LUDWIG			PAGE 1
DATE	HR	MN	SEC	R	DELE	EDELE
07 01	1	0	17.882	296	90	126
07 01	1	0	18.942	624	95	133
07 01	1	0	19.992	1312		
07 01	1	0	21.042	2656		

SAMPLE DRL PRINTOUT

Figure 8—Sample DRL printout.

ACKNOWLEDGMENTS

We wish to acknowledge the assistance of Dr. George H. Ludwig in the initiation of the Data Reduction Laboratory project in 1965 and to express our thanks for his continuing support.

We also wish to acknowledge the assistance of R. D. Cardwell, G. A. Chapman, H. L. Darling, R. L. Fitzer, A. J. Nowotny and K. D. Taylor, who participated in the DRL software development.

Goddard Space Flight Center
National Aeronautics and Space Administration
Greenbelt, Maryland, October 15, 1969
311-07-11-01-51

REFERENCES

1. Habib, E. J., F. A. Keipert, and R. C. Lee, "How to Handle 200 Million Data Points A Day," *ISA Journal,* February 1966.

2. Ludwig, George H., "Space Sciences Data Processing," *Datamation,* November, 1967.

3. Creveling, C. J., A. G. Ferris, and C. M. Stout, "Automatic Data Processing," *IRE Transactions on Space Electronics and Telemetry,* June, 1962.

4. Walton, B. A., J. J. Quann, and F. A. Keipert, "The Data Reduction Laboratory User's Manual," NASA—GSFC X-565-69-56 Document, January, 1969.

"The aeronautical and space activities of the United States shall be conducted so as to contribute . . . to the expansion of human knowledge of phenomena in the atmosphere and space. The Administration shall provide for the widest practicable and appropriate dissemination of information concerning its activities and the results thereof."

— NATIONAL AERONAUTICS AND SPACE ACT OF 1958

NASA SCIENTIFIC AND TECHNICAL PUBLICATIONS

TECHNICAL REPORTS: Scientific and technical information considered important, complete, and a lasting contribution to existing knowledge.

TECHNICAL NOTES: Information less broad in scope but nevertheless of importance as a contribution to existing knowledge.

TECHNICAL MEMORANDUMS: Information receiving limited distribution because of preliminary data, security classification, or other reasons.

CONTRACTOR REPORTS: Scientific and technical information generated under a NASA contract or grant and considered an important contribution to existing knowledge.

TECHNICAL TRANSLATIONS: Information published in a foreign language considered to merit NASA distribution in English.

SPECIAL PUBLICATIONS: Information derived from or of value to NASA activities. Publications include conference proceedings, monographs, data compilations, handbooks, sourcebooks, and special bibliographies.

TECHNOLOGY UTILIZATION PUBLICATIONS: Information on technology used by NASA that may be of particular interest in commercial and other non-aerospace applications. Publications include Tech Briefs, Technology Utilization Reports and Notes, and Technology Surveys.

Details on the availability of these publications may be obtained from:

SCIENTIFIC AND TECHNICAL INFORMATION DIVISION

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Washington, D.C. 20546