

RTLAB: A REACTION-TIME LABORATORY FOR THE DATA GENERAL
NOVA MINICOMPUTER

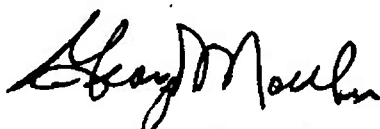
by

Gary M. Olson, George Moeller,
and
Kevin Laxar

NAVAL SUBMARINE MEDICAL RESEARCH LABORATORY
NAVAL SUBMARINE MEDICAL CENTER REPORT NUMBER 748

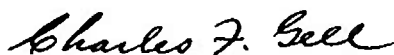
Bureau of Medicine and Surgery, Navy Department
Research Work Unit MF51.524.004-2002DX5G.03

Transmitted by:



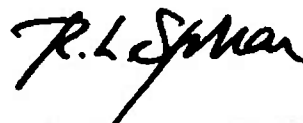
George Moeller, Ph. D.
Head, Human Factors Branch

Reviewed and Approved by:



Charles F. Gell, M.D., D.Sc. (Med)
SCIENTIFIC DIRECTOR
NavSubMedRschLab

Approved and Released by:



R. L. Sphar, CDR MC USN
OFFICER IN CHARGE
NavSubMedRschLab

Approved for public release; distribution unlimited.

SUMMARY PAGE

THE PROBLEM

To prepare a general purpose package of assembly language subroutines for the on-line design, control, and analysis of behavioral experiments.

FINDINGS

The details of the organization and use of RTLAB, a package of relocatable assembly language subroutines for the Data General NOVA family of minicomputers, is presented, and sufficient program documentation is provided to allow users to modify subroutines, as needed.

APPLICATION

RTLAB should be of use to any investigator using a NOVA minicomputer system for on-line data acquisition in discrete-trial behavioral experiments.

ADMINISTRATIVE INFORMATION

This investigation was conducted as part of Bureau of Medicine and Surgery Research Unit MF51.524.004-2002DX5G. The present report is Number 5 on this work unit. It was submitted for review on 19 July 1973, approved for publication on 7 August 1973, and designated as NavSubMedRschLab Report No. 748.

PUBLISHED BY THE NAVAL SUBMARINE MEDICAL RESEARCH LABORATORY

ABSTRACT

RTLAB is a package of relocatable assembly-language sub-routines for the design, control, and analysis of behavioral experiments on the Data General Corporation NOVA family of computers. It was originally designed for use in reaction time research but is readily adaptable to any discrete-trial task. RTLAB accepts design parameters from the experimenter for a given experiment and then produces an appropriately randomized and counterbalanced set of trials for presentation to a subject. During an experimental session RTLAB controls the presentation of stimuli, the acceptance and scoring of responses, and all of the timing and input-output constraints associated with control and data acquisition. Its current design does not allow for time-sharing among more than one subject station. A history of the experiment is saved for later analysis and transfer to storage media. User modifications of any of the details of these operations is facilitated by the organization of RTLAB into subroutine modules. The organization of RTLAB and its use are presented, along with sufficient documentation of the routines to allow user modification where appropriate.

RTLAL: A Reaction-time Laboratory for the Data General
NOVA Minicomputer

INTRODUCTION

Reaction time is one of the most basic empirical measures available to the experimental psychologist, and has never been more popular than it is today. Increased interest in reaction time paradigms has coincided with growing availability of inexpensive minicomputer systems that are ideally suited for the on-line control of such experiments. The present report describes the general organization and use of one such on-line reaction-time laboratory developed in the Human Factors Branch of the Submarine Medical Research Laboratory on a Data General Corporation NOVA 1220 minicomputer. This set of routines, called RTLAL, is a complete research package, allowing the investigator to design and run experiments and analyze data in most reaction time paradigms. RTLAL is of course totally compatible with any of the Data General Corporation family of minicomputers, requiring only minor modifications to mate it with the user's specific configuration of I/O devices. RTLAL could probably be adapted to other 16-bit machines with little difficulty, and its general logic could be used for setting up a similar set of routines on any other on-line laboratory system.

OVERVIEW

RTLAL does the following tasks. Upon entry (assuming it is being used for the first time) it queries the user

for a number of design parameters. These are saved so that subsequent runs of the same experiment do not require re-entering these parameters. It then asks for a subject number and some other information specific to the running of a given sequence of trials. Once all general and specific parameters are entered RTLAL sets up a randomized and counterbalanced sequence of trials for the current experimental session and halts, awaiting the experimenter's command to start the experimental sequence. When this sequence is started RTLAL controls the timing and presentation of stimuli from trial to trial, records responses and reaction times, provides feedback for the subject, and stores all data for later analysis and output. Once the experimental sequence is over the program again halts, providing messages to the experimenter and to the subject. The experimenter then has several options: (a) do a brief analysis of the data on the spot, obtaining overall means and means broken down by correct-incorrect, stimuli, foreperiods, and, if desired, blocks of trials; (b) list the trial-by-trial data on the teletype; (c) write the trial-by-trial data on magnetic tape for off-line analysis on a large computer. Option (c), in the version of RTLAL presented here, was dictated by our own hardware constraints and typical uses, and could be altered by another user to allow data storage on disk, paper tape, or some other appropriate medium. At the completion of the analysis and listing options exercised by the user,

control returns to the beginning of RTLAB. By entering the same experiment identification number the user can run another subject in the same design, with RTLAB automatically providing a new counterbalanced, randomized sequence of trials. Or, if the computer will be used for other purposes before another subject will be run, the user can punch RTLAB or the parameter addresses (for use as an overlay) on paper tape for later use. Finally, by entering a new experiment identification number the user can alter the design in any way he desires.

RTLAB is designed to be maximally adaptable by a user. The program is organized into subroutine modules which can be altered, or exchanged, using the NOVA extended assembler software options for independent assembly of relocatable subprograms. Thus, for instance, if the user wanted to change the magnetic tape routines to match his own configuration, only subroutines MTAPE and EOF would have to be altered.

This modular organization is important in mating RTLAB with the user's input and output devices. The version of RTLAB presented as an example here used a Tektronix 4010 CRT display with keyboard for presenting stimuli and receiving the subject's responses. Other paradigms in our laboratory use the teletype to present and record, or go through a general purpose input-output interface to devices like slide-projectors, tachistoscopes, telegraph keys, and so forth. All meshing of RTLAB with this variety of possible devices is done via subroutine ALPHA and its associated routines. Thus, the

user can write his own version of ALPHA, subject to constraints discussed later in this report, to merge RTLAB with his own hardware. This allows rapid and simple conversion of RTLAB, whose general logic is not dependent upon hardware considerations.

As presently written, all subroutines in RTLAB reside in core at the same time. The NOVA 1220 in our lab has 8K of core, and RTLAB occupies approximately 5-6K, depending upon the complexity of ALPHA (which varies with different experiments). A user with only 4K could store subroutines on magnetic tape, paper tape, or disk, and call them in as needed. RTLAB uses a real-time clock for all timing. The following peripherals are available in our laboratory: a KSR 33 teletype, a high speed paper tape punch-reader, a Tektronix 4010 CRT, and a Calma 650 write-only magnetic tape unit. For purposes of exposition the examples of program details and use in this report assume this set of peripherals.

RTLAB was designed as a general purpose on-line system for experiment design, control, and simple data management for virtually any possible reaction time experiment. In E. Smith's¹ terms, all paradigms he discusses except the one-many stimulus-response mapping arrangement are possible. (A one-many paradigm could be run if responses were coded cleverly in ALPHA. See detailed discussion of ALPHA later.) Further, RTLAB is presently designed for paradigms where each stimulus and each foreperiod will appear equally often in a sequence of trials. Although RTLAB is designed primarily for

reaction time paradigms, clearly it can be used without modification for any multiple-stimulus, single-response, discrete-trial learning or problem-solving task where correctness of response or latency of response is a dependent variable. Examples of other tasks particularly well-suited to RTLAB would include continuous paired-associate learning, continuous recognition learning, concept identification, discrimination learning, probability learning, and signal detection. Relatively minor modifications in software would allow the use of RTLAB's basic logic in virtually any discrete-trial experimental paradigm.

The remainder of this report will present details of the overall structure of RTLAB and its use. For simplicity of exposition, we will describe RTLAB in the context of a particular implementation in our own laboratory, namely, the design of Experiment I reported by Olson and Laxar.² But the potential user should keep in mind the wide range of possible uses for RTLAB. Except for several flow charts of main subroutines and a complete assembler listing of all subroutines in RTLAB in the final section of this report, the description of RTLAB will be from the perspective of a user rather than that of a programmer. The single important exception is the discussion of ALPHA, the routine through which each user must mate his own particular experimental setup with RTLAB.

DESIGN CONSTRAINTS

The user has a number of options available in setting up experiments to

be run with RTLAB. The program asks for the experimenter's specifications through a series of simple queries on the teletype (see the section on program use for the format of these queries). The following paragraphs discuss the options available and describe the current software constraints on the values of experimental parameters. All values discussed are decimal, and those whose limits are not explicitly mentioned are constrained by the NOVA 16-bit single precision arithmetic (maximum value = 65565). It is especially important that the user pay attention to the constraints discussed in paragraph E.

A. Number of trials. The present version allows up to 256 trials. A user with adequate core storage could change this limit by increasing the arrays addressed at STIM, FORE, PLACE, and SPACE in subroutine DESIGN and at CODE and TIME in subroutine SCORE.

B. Foreperiods. Up to 16 different foreperiod values can be specified. Entries are in milliseconds.

C. Stimuli. Up to 64 different stimuli can be specified. The user enters the stimuli as digital codes whose meaning is contained in subroutine ALPHA or one of its associated subroutines. See the examples in the section on program use.

D. Stimulus-response mappings. The user specifies one correct response code for each stimulus code he enters. Again, this correct response code is assumed to have an interpretation in ALPHA. Any one-one

or many-one configuration of stimuli and correct response codes is acceptable.

E. Blocking for randomization.

RTLAB contains routines which, given specifications about stimuli, foreperiods, and number of trials, set up a randomized, counterbalanced sequence of trials for presentation to a subject. These routines operate upon a block of n contiguous trials, producing a single random permutation of the n trials. Within a block each combination of stimulus and foreperiod must appear equally often. Thus, the minimum block size for purposes of randomization is the product of the number of stimuli and the number of foreperiods. The general expression for block size is given in (1), where K is an integer greater than zero. The relationship between number of trials

$$(1) \quad B = K (\#Stim)(\#FP)$$

(T) and block size (B) is given in (2), where N is an integer greater than zero.

$$(2) \quad T = N \times B$$

The constraints in (1) and (2) are very important, for if they are violated the sequence of trials presented to the subject will not have proper counterbalancing. Examples of acceptable and unacceptable combinations of number of trials, block size, number of stimuli and number of foreperiods are shown in Table 1. Since the user is responsible for guaranteeing that these constraints are met, a short message is printed on the teletype during the design query

reminding the user of the nature of the constraints (see section on program use).

F. Inter-trial interval. In our research, this has referred to the time between the subject's response and the presentation of the signal that the computer is ready to present the next trial. The subject's acknowledgment of that signal initiates the foreperiod of the next trial. Figure 1 shows this sequence of events.

G. Minimum and maximum reaction times. The user must specify limits on the range of acceptable reaction times. The limits were incorporated in the program to detect lapses by the subject into random responding or inattention and to force the investigator to define the range of acceptable data before rather than after the fact. The range limit can be effectively bypassed by setting them very small and very large.

H. Rerun or discard errors. The user can decide either to rerun all errors or to run the fixed sequence of trials, retaining the error data for analysis or discarding it, as appropriate. When errors are rerun, they are all rerun at the end of the regular sequence in the order in which they were made. The current maximum on rerun trials is 128.

RUNNING THE EXPERIMENT

The user has great flexibility on how each trial will be run. The

Table 1. Examples of Valid and Invalid Combinations of Basic Parameters

No. Stimuli	No. Foreperiods	No. Trials	Block Size	Comment
4	1	112	28	Valid.
48	1	240	48	Valid.
2	2	96	8	Valid.
1	5	100	20	Valid.
16	2	64	64	Valid.
5	3	150	15	Valid.
4	2	112	4	Invalid. Block size does not equal foreperiods times stimuli.
4	1	100	10	Invalid. Block size does not equal an integral function of foreperiods times trials.
16	1	100	16	Invalid. Stimuli $\times$ Foreperiods (or block size) not integral in number of trials.
8	2	80	32	Invalid. Block size not integral in number of trials.

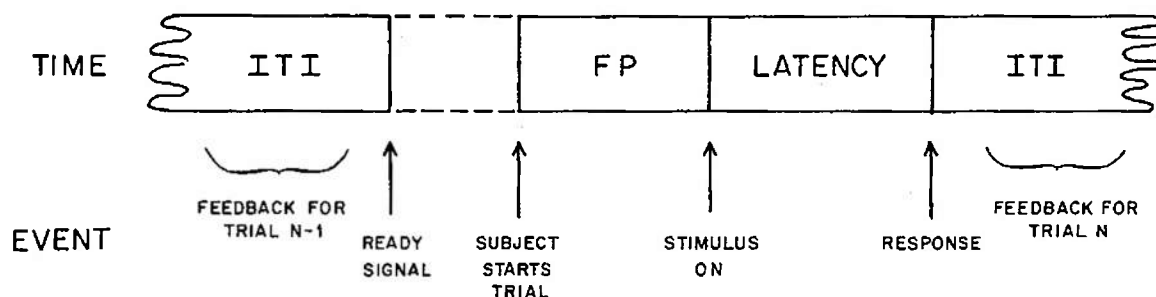


Fig. 1. Sequence of trial events for a subject-paced reaction time experiment.

definition of events in Figure 1 is merely an example from our own research. We use a self-paced task, where the subject initiates each trial after the ready signal. The foreperiod is the time from the subject's initiation of the trial to the presentation of the stimulus. The intertrial interval begins with the subject's response and ends with the presentation of the ready signal. This interval is constant regardless of the magnitude of the subject's reaction time.

RTLAB records two pieces of information on each trial, the subject's reaction time in milliseconds and a response code which indicates whether the subject was correct, incorrect, or violated a constraint on maximum time or minimum time. These are stored by subroutine SCORE, and along with the tables of trial-by-trial stimuli and foreperiods provide a complete record of the experimental events for later analysis and listing. (See details of SCORE and the storage of information in later sections.)

ANALYSIS AND LISTING OPTIONS

RTLAB provides some low-level data analysis for each subject via subroutine RTANL. It is assumed that any complex analysis and analyses across subjects will be done off-line. Users may want to tailor RTANL to their own needs.

The present version of RTANL provides the analysis shown in Table 2. The standard analysis includes mean reaction times for various logical partitions of the data, while the optional analysis allows for a further breakdown into blocks of contiguous trials. The size of these blocks does not have to correspond to the size of the blocks used in randomizing the experimental sequence. However, it is up to the user to create meaningful block sizes for the analysis. Examples of the output of the analysis routine are given in a later section.

RTLAB also allows the user to transfer the raw data to various storage media.

Table 2. Data Analyses Provided by RTANL for each Subject

Standard Analysis Includes --

1. Overall results
2. Analysis by stimuli (if more than one)
3. Analysis by foreperiods (if more than one)
4. Analysis by stimuli $\times$ foreperiods (if both more than one)

-- Mean reaction time and frequency counts are calculated for each of these for:

A. All data

B. Corrects alone

C. Errors alone

Only if errors are possible

Optional analysis

All of the above repeated for blocks of trials, where block size is entered at the time of analysis (N.B. It is up to the user to create meaningful block sizes.).

The version of RTLAB in use at NAVSUBMEDRSCHLAB allows a listing of the raw data on the teletype and a transfer of the raw data to magnetic tape for analysis off-line on a large computer.

THE USE OF RTLAB

This section presents a step-by-step guide to the use of RTLAB. It will be

assumed that the reader is familiar with the general description of RTLAB's structure and constraints presented in previous sections.

Table 3 lists the subroutines of RTLAB and their addresses obtained via the NOVA relocatable loader. This gives the user a rough idea of the size of the various routines. Table 4 is a list of the interprogram symbols and

Table 3. Program Elements and Addresses from Relocatable Loader

Starting Addresses		
Element Name	Normal	Page Zero
RTEXP	440	50
BETA	711	107
RTINT	1706	121
SCORE	2054	122
DESIG	3304	130
MTAPE	6724	220
EOF	7154	224
DATYP	7235	226
RANDI	7376	234
PERMU	7434	235
INOUT	7521	237
BMAN	7774	247
NZMEM	10024	250
CRTXT	10054	251
BOX	10107	252
ALS	10504	253
RTANL	10717	257
CARVE	12076	303
PROQ	12245	304
DPACC	12302	310
	12444	312

Table 4. Interprogram Symbols and Their Addresses from Relocatable Loader

Symbolic Addresses	Octal Addresses	Symbolic Addresses	Octal Addresses	Symbolic Addresses	Octal Addresses
ABEGI	000130	ARTUP	000121	MAX	000052
ABNDC	000237	ASCOR	000122	MCURS	000256
ABNOC	000243	ASTIM	000132	MIN	000051
ACLER	000250	ATIME	000124	MONK	000304
ACODE	000123	ATYPE	000166	NFP	000060
ACS	000140	AWERB	000222	NOVER	000063
ADAT	000232	BTXT	000226	NUN	000305
ADCBN	000240	CHAR	000253	OLD	000236
ADONE	000112	CLEVE	000303	OVINC	000126
AEOF	000224	CLMSK	000050	PHALF	000120
AFORE	000131	CRESP	000062	PRES	000111
AFP	000137	CS	000065	PSYCH	000257
AGETC	000242	CT	000055	QUACC	000310
ALGAE	000306	CTDWN	000071	RCHAR	000254
AMAIN	000102	CTRLS	000076	REPC	000127
AMOEB	000307	EXP	000061	RESP	000075
AOCBN	000244	FP	000064	RLBOX	000252
AOUTP	000247	GROUP	000056	RTIME	000072
AOVER	000125	IDBIN	000246	SEC	000066
APE	000167	IFORE	000133	SHOW	000251
APRMT	000235	IGETI	000245	SKTCH	000255
APUTC	000241	INTER	000070	SUBJ	000057
ARAND	000234	ISTIM	000134	TRLS	000054
ARAP	000101	ITI	000053	TTIME	000073
ARESP	000141	ITYP	000233	UQUER	000116

their absolute addresses for the example in Table 3.

The complete package specified in Table 3 would usually be loaded via an absolute binary tape. The tape would self-start at location MAIN (indirectly addressed by AMAIN), printing the heading shown at the top of Figure 2. As the teletype message reminds the user, all numeric entries are decimal

and all times are in milliseconds. All numbers are entered in the same fashion: the string of decimal digits is followed by a carriage return (the familiar decimal-to-binary routine of Data General). The legality of input parameters is not checked. If a mistake is made the program must be restarted at location MAIN.

PT EXP PROGRAM
ALL ENTRIES ARE DECIMAL, ALL TIMES IN MSEC.

EXPERIMENT CODE = ? 123

NEW CODE. CHANGE DESIGN? (1) NO (2) YES 2

BLOCK SIZE MUST EQUAL $N(\#STIM)(\#FP)$, WHERE
N IS AN INTEGER > 0. BLOCK SIZE MUST ALSO
BE AN INTEGRAL QUOTIENT OF # TRIALS.

NUMBER OF TRIALS = ? 48

BLOCK SIZE = ? 4

ITI = ? 2000

MAX PT = ? 1500

MIN PT = ? 120

NUMBER OF FOREPERIODS = ? 1

FOREPERIODS IN MSEC.

1 1000

NUMBER OF STIMULI = ? 4

TYPE STIMULUS CODE FOLLOWED BY CORRECT RESPONSE CODE

1 STIMULUS CODE = 4
RESPONSE CODE = 1

2 STIMULUS CODE = 3
RESPONSE CODE = 2

3 STIMULUS CODE = 2
RESPONSE CODE = 2

4 STIMULUS CODE = 1
RESPONSE CODE = 1

REIN ERRORS & NULLS (1) OR DISCARD (2)? 2

SUBJECT NUMBER = ?

Fig. 2. Sample teletype print-out for the entry of a new experimental design in RTLAB.

The program first requests an experiment number, as shown in Figure 2. If this identification number is already known to the program, it is assumed that all parameters were entered on a previous run. In the latter case, all queries between the dashed lines in Figure 2 are skipped and the program goes on to the queries, starting with subject number, that have to do with the specific run (see below).

Only the current experiment number is saved, so just one set of experimental parameters is present in memory at a given time. Thus, values can only be saved between contiguous runs of the same experiment by leaving them in the machine, by punching a complete tape of RTLAB, or by punching an overlay of locations containing the parameter values. The overlay would be read in after the complete RTLAB tape. Table 5 shows the memory locations needed for such an overlay.

Table 5. Memory Ranges for Overlay

Program Element	Symbolic Address
RTEXP	MIN to EXP
SCORE	REPC
DESIGN	FPTAB to RTAB+77 ₈

If the experiment number is a new one the teletype queries proceed as in Figure 2. The program acknowledges that the code is a new one, and double checks that the user does in fact want to change all the parameters. If the user responds "yes" with a numeric two, the program proceeds as in Figure 2. If the response is "no", the program asks, as shown in Figure 3, whether the user desires to change the

RT EXP PROGRAM

ALL ENTRIES ARE DECIMAL, ALL TIMES IN MSEC.

EXPERIMENT CODE = ? 301

NEW CODE. CHANGE DESIGN? (1) NO (2) YES 1

CHANGE CODE? (1) NO (2) YES 2

SUBJECT NUMBER = ?

Fig. 3. Sample print-out for changing the experiment code without changing the design.

experiment code. If the user responds "yes" as in Figure 3 the code entered replaces the previous code but all previous parameter values are retained, the program jumping to the subject number query. If the user would have responded "no" to the query about changing the code RTLAB would have returned to the first question and asked for the experiment code again.

In Figure 2 once the program has learned that a set of design parameters is to be entered a short reminder about the randomization constraints described in a previous section is printed and a series of straightforward queries is made about each of the design parameters. Note that once the user specifies the number of foreperiods the program asks for the value of each. Similarly, after the number of stimuli has been specified the program asks for a numeric code for each stimulus and a similar code for the correct response associated with each stimulus. Recall that these numeric codes are given meaning in terms of system interfaces by the user-supplied subroutine ALPHA.

Each of the possible sequences of queries shown in Figures 2 and 3 ended with the query about the subject number. Figure 4 shows the continuation of the user-computer dialog which would hold for any of these cases. The subject number and group number are for identification. On the assumption that the combination of subject number and experiment number will be unique, these two are used to derive the seed for the pseudo-random number generator which controls the randomization of the sequence of trials. Thus, pairing the same subject number with the same

experiment number will result in an identical sequence of trials, given constant experimental parameters.

The queries after group number are specific to a particular application of RTLAB and are part of subroutine ALPHA. All queries up to this point have come from subroutine DESIGN. Usually the user will need to know some additional details about the encoding of responses at the subject's station. In the example shown in Figure 4, the ASCII characters to serve as "true" and "false" responses on the CRT keyboard are required, since this was counterbalanced from subject to subject. Application-specific queries should be part of ALPHA and addressed from RTEXP via the instruction JSR@UQUER. If the user has no special queries, he must at least provide for a return back to RTEXP from ALPHA.

Once all parameters have been entered, RTLAB designs the complete set of trials to be run and halts, printing the teletype message shown at the bottom of Figure 4. As soon as the experimenter and the subject are ready to proceed, the former presses CONTINUE on the NOVA console and the experiment would be run. The details of the running of the experiment are controlled entirely by the user in subroutine ALPHA. A more complete discussion of the user-supplied routines that might compose ALPHA is supplied in the last section of this report on user-modifications. Basically, the main routine RTEXP counts down the number of trials and jumps on each trial to ALPHA via JSR@PRES. Prior to returning to RTEXP

SUBJECT NUMBER = ? 101

GROUP NUMBER = ? 1

"TRUE" CHARACTER = ? /

"FALSE" CHARACTER = ? Z

PRESS CONTINUE TO START SESSION.

Fig. 4. Sample print-out for entering subject-specific information.

from this jump the routines in ALPHA must take care of presenting the trial, monitoring the subject's performance on that trial, and storing the raw data in appropriate tables for later analysis or listing. In a very real sense, ALPHA is where the important work is done on each trial. After the last trial has been run RTEXP jumps to ALPHA via JSR@ADONE for a message or signal to the subject that the session is over. After pressing continue any teletype interrupt prior to the completion of the sequence of trials aborts the run. The experimenter can terminate a session, but it must be remembered that it will only be possible to restart the session at the beginning.

An example of the experiment entered in Figure 2 is now presented. The subject starts each trial by pushing any key on the CRT keyboard. One second later a display like that shown

in Figure 5 would flash on the screen, and the subject would decide as quickly as possible whether the word in the center correctly named the side of the square on which the diamond appeared. As soon as the subject responded the display would disappear and information like that shown in Table 6 would appear as feedback. This feedback would remain on for two seconds (the intertrial interval) and as soon as the feedback disappeared the subject could start the next trial.

When control returns to RTEXP after the JSR@ADONE to ALPHA, the message shown at the top of Figure 6 is printed on the teletype and the program halts. The analysis routines are initiated by pressing CONTINUE on the console. A brief series of queries are made as shown in Figure 6. The output resulting from these choices is also shown in

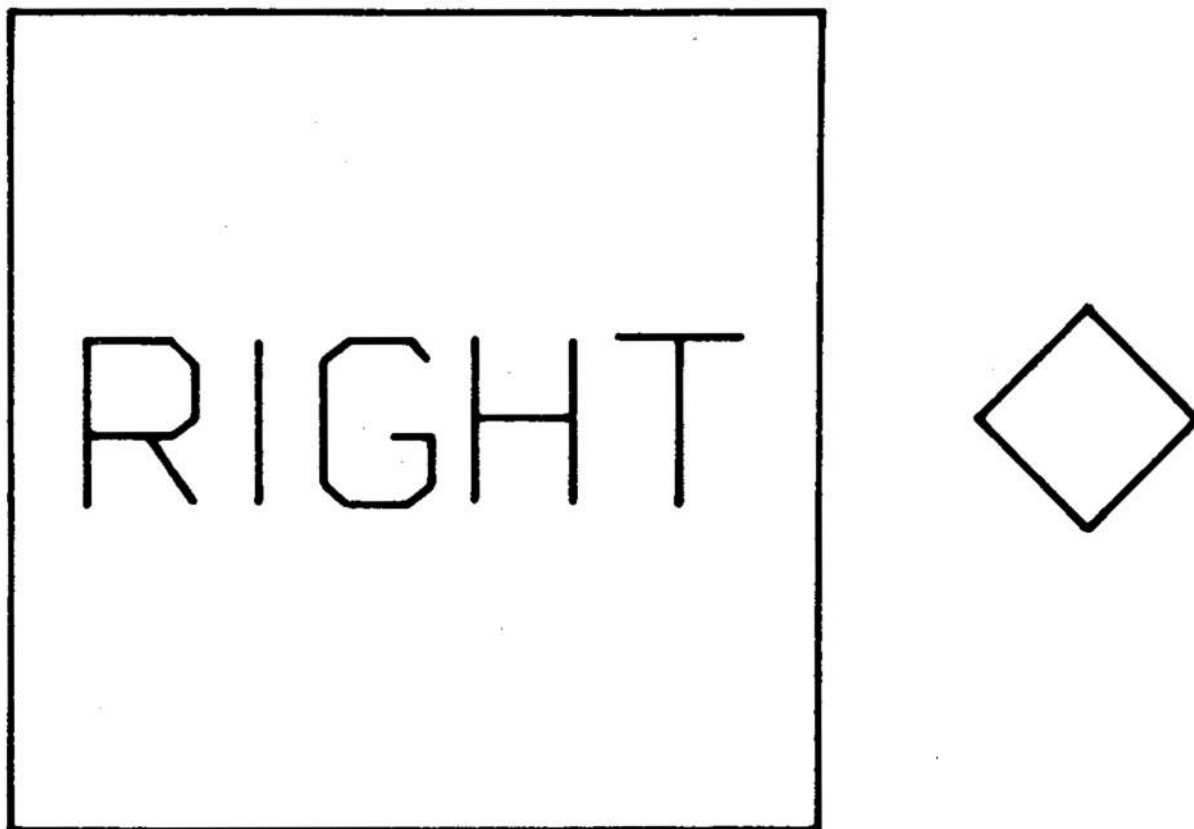


Fig. 5. Sample CRT display (based on Olson and Laxar, 1973).

Table 6. Feedback Messages to Subject from ALPHA via CRT

Event	Message
Correct response	TRUE TIME = 748 MILLISECONDS
Incorrect response	TRUE ERROR TIME = 1039 MILLISECONDS
Response not defined in program	ILLEGAL RESPONSE ERROR TIME = 683 MILLISECONDS
RT less than minimum allowed	FALSE TOO FAST
RT greater than maximum allowed	TOO SLOW
Trial sequence over	END OF SESSION. THANK YOU.

PRESS CONTINUE FOR ANALYSIS

DATA ANALYSIS FOR PT EXPERIMENT NO. 123
SUBJECT NO. 101

ANALYSIS BY BLOCKS? (1) YES (2) NO 2

TYPE OUT DATA? (1) NO OR (2) YES 2

OVERALL ANALYSIS

OVERALL PT	N	CORRECT PT	N	EPPOP PT	N
500.7708	48	501.1627	43	497.4000	5

ANALYSIS BY STIMULI

STIM	OVERALL PT	N	CORRECT PT	N	EPPOP PT	N
1	369.5000	12	369.5000	12	0.0000	0
2	543.7500	12	543.7500	12	0.0000	0
3	576.7500	12	599.4545	11	327.0000	1
4	513.0833	12	499.6250	8	540.0000	4

TRIAL	STIMULUS	F P	P T	CODE
1	2	1000	373	1
2	3	1000	570	1
3	4	1000	501	1
4	1	1000	290	1
5	4	1000	507	1
6	1	1000	329	1
7	2	1000	524	1
8	3	1000	522	1
9	3	1000	889	1
10	1	1000	481	1
11	2	1000	639	1
12	4	1000	482	1
13	3	1000	491	1
14	2	1000	502	1
15	1	1000	382	1
16	4	1000	419	1

Fig. 6. Sample print-out for on-line analysis following the completion of a single subject's session.
(Part 1 of 2 parts)

17	4	1000	532	1
18	1	1000	303	1
19	2	1000	553	1
20	3	1000	808	1
21	1	1000	409	1
22	2	1000	632	1
23	3	1000	594	1
24	4	1000	421	1
25	4	1000	632	2
26	1	1000	485	1
27	2	1000	491	1
28	3	1000	611	1
29	3	1000	327	2
30	4	1000	443	2
31	1	1000	271	1
32	2	1000	519	1
33	4	1000	655	1
34	2	1000	562	1
35	1	1000	289	1
36	3	1000	609	1
37	1	1000	382	1
38	4	1000	480	1
39	2	1000	559	1
40	3	1000	402	1
41	1	1000	469	1
42	3	1000	533	1
43	4	1000	568	2
44	2	1000	654	1
45	2	1000	517	1
46	1	1000	344	1
47	3	1000	565	1
48	4	1000	517	2

MOUNT AND READY TAPE, PRESS CONTINUE.

Fig. 6. Sample print-out for on-line analysis following the completion of a single subject's session.
(Part 2 of 2 parts)

Figure 6, and upon completion of the output the program again halts with a message about the magnetic tape, shown at the bottom of Figure 6. If one had asked for the analysis by blocks the analysis would have been repeated for each block. If one did not ask for a listing the entire listing would be suppressed. If errors had been rerun, no breakdown would be given for corrects and errors, but the number of trials rerun would be shown at the beginning of the analysis. One can suppress all the output by going to location MAIN when the program halts at the end of the session.

The magnetic tape output is straightforward. It can be bypassed by manually returning control to the beginning of RTLAB via the console switches. If a tape is written control automatically returns to the beginning of RTLAB after the end of file is written. Appendix A contains a schematic of our tape format.

FLOW CHARTS AND SUBROUTINE LISTING

Appendix B contains a complete table of subroutines and their primary functions for the sample experiment described in this report. Appendix C has flow-charts for the most important subroutines, and can be used as a guide to the program listings. Appendix D contains these listings for all subroutines used in the sample experiment.

SUMMARY OF USER MODIFICATIONS

Required modifications. For most applications of RTLAB the user must

write a subroutine ALPHA and any subroutines required by ALPHA. RTLAB is designed so that this should be the only detailed programming required by the user. If a user finds he has to extensively modify a significant number of other routines in RTLAB as well, then there may be little advantage to RTLAB over the writing of the experimental package from scratch. But the user must be prepared to write at least this one routine.

The main program RTEXP calls routines in ALPHA for three different purposes. Thus, ALPHA must include indirect entry points for the following calls from RTEXP: (a) UQUER for any user-specific design queries; (b) PRES for the presentation of each trial; and (c) ADONE for the termination of the experiment. (a) and (c) are relatively straightforward, and depend entirely upon the desires or needs of the user. If either or both of these functions are not needed, the user can effectively bypass the calls by inserting a JMP 0,3 at the appropriate address in ALPHA.

However, function (b) is central to the use of RTLAB and thus the requirements for a jump via PRES to ALPHA and the necessary communications with other subprograms in RTLAB must be spelled out in some detail. The jump via PRES must take care of all timing functions, must present stimuli and accept responses, must encode the responses and contrast these with stored tables of appropriate responses, and must store response information and latencies via subroutine SCORE. The stimulus code, correct response code, and foreperiod value for each

trial are passed by RTEXP to ALPHA via locations CS, CRESP, and FP. Those locations are defined in RTEXP and must be declared as displacement externals in ALPHA. Subroutine ALPHA must translate the stimulus code into some actual stimulus for presentation. For instance, the codes 1-4 shown in Figure 2 are used by the ALPHA in Appendix D to present one of four stimulus displays, like the sample shown in Figure 5, on the CRT screen. See the details in Appendix D, especially subroutine BOX, which is called by ALPHA for drawing the displays on the screen. Similarly, the ASCII code received from the CRT keyboard when the subject responds is translated into the appropriate response code for transmission to SCORE. In our example, the encoding of "true" and "false" which had been entered prior to the run (see Figure 4) is stored in locations TRUE and FALSE in ALPHA. The keyboard entry from the subject's response is first compared with these to recover the "true" or "false" response and then compared with the code in CRESP for classification as correct or incorrect.

The data on each trial are stored by means of JSR to ASCOR to subroutine SCORE. The latency is passed via the address TTIME and the scored response code is passed via RESP. Both are defined in RTEXP and declared as displacement externals in ALPHA. The scoring of the response is done in ALPHA, with the following code passed to SCORE: 1 = correct, 2 = error, 3 = too fast, too slow. Subroutine SCORE obtains the trial count, CTRLS, from RTEXP. SCORE keeps track of which trials have to be rerun, and

RTEXP contains the necessary programming to rerun these trials when the rerun option is used.

All timing is coordinated in ALPHA. The real-time clock is set initially by RTEXP to interrupt the program every millisecond when it is enabled, and the interrupt service routine (RTINT) increments various counters on each interrupt. These counters are then cleared and sampled by ALPHA to time out pre-set intervals like the foreperiod or the intertrial interval and to measure the subject's latency. Any of the counters defined in RTINT, RTIME, CTDWN, and INTER may be declared displacement externals in ALPHA. The programming for this is quite straightforward, and the examples in ALPHA and RTINT in Appendix D can be examined for more details.

In order to facilitate the user's comprehension of the interprogram relationships in RTLAB, Table 7 presents the associations among subprograms as indexed by .EXTD statements in the relocatable source programs. This table should help the user trace the communications between ALPHA and the other subroutines in Appendix D as he prepares his own version of ALPHA.

One other important user modification needs to be mentioned here. Since the Calma 650 write-only tape unit in our laboratory is not common, most users will want to write their own magnetic tape routines (MTAPE and EOF in our programs) or substitute paper tape, disk storage, or other mass storage routines as appropriate. Table 8 provides the information required to make this substitution and

Table 7. Inter Program Communication in RTLAB

Calling Program	Program Communicated With Directly													
	RTEXP	DESIGN	ALPHA	RTINT	SCORE	DATYP	RTANL	MTAPE	BMAN	BOX	CARVE	CRTXT	PERMU	ALS
RTEXP		X	X	X	X		X		X					
DESIG	X				X				X				X	
ALPHA	X	X		X	X					X				
RTINT	X								X					
SCORE	X													
DATYP	X	X			X				X					
RTANL	X	X			X	X					X			
MTAPE	X	X			X									X
BMAN														
BOX														
CARVE														X
CRTXT														
PERMU														X

Table 8. Organization of Information in RTLAB

Information	Symbolic location ^a	Indirect Address ^a (Symbolic -1)	Program Element
Experiment Identification No.	EXP (Z)		RTEXP
Subject No.	SUBJ (Z)		RTEXP
Group No.	GROUP (Z)		RTEXP
Number of Trials	TRLS (Z)		RTEXP
Number of Stimuli	CT (Z)		RTEXP
Number of Foreperiods	NFP (Z)		RTEXP
Rerun Code (1 = errors rerun, 2 = not rerun)	REPC (Z)		SCORE
Table of Stimuli	CSTAB (N)	ACS (Z)	DESIGN
Table of Correct Responses	RTAB (N)	ARESP (Z)	DESIGN
Table of Foreperiods	FPTAB (N)	AFP (Z)	DESIGN
Table of Stimuli Presented in Session ^b	STIM (N)	ISTIM (Z)	DESIGN
Table of Foreperiods Presented in Session ^b	FORE (N)	IFORE (Z)	DESIGN
Table of Trial Outcomes ^b	CODE (N)	ACODE (Z)	SCORE
Table of Reaction times ^b	TIME (N)	ATIME (Z)	SCORE
Table of Rerun Trials ^c	OVER (N)	AOVER (Z)	SCORE
^a (Z)=zero page relocatable, (N) = normal relocatable ^b Stored in a top-to-bottom fashion, such that within the table the first location (e.g., STIM) contains the data for trial <u>n</u> , while the <u>n</u> th location (e.g., STIM+(<u>n</u> -1)) contains it for trial 1. ^c Stored in the order they were rerun.			

additional data found useful in creation of alternate versions of ALPHA. As noted previously, Appendix A shows our tape format as a sample data coding.

Optional modifications. The overall organization of RTLAB as independently assembled relocatable program modules should make it relatively easy for a user to modify any aspects of RTLAB that do not conform to his needs. Each user will have to weigh the utility of modifying our programs versus writing them from scratch. For instance, RTANL is an obvious candidate for modification, but our version is quite long and complex and most user's may find it easiest to use it as is. But should the user want to change or replace any of the routines the information in Tables 7 and 8 and in the Appendices should be quite helpful.

FURTHER INFORMATION

Copies of RTLAB relocatable source program tapes can be obtained from the Data General User's Group by writing to:

Executive Secretary
Data General Users Group
Data General Corporation
Route 9
Southboro, Massachusetts 01772

Further information about RTLAB can be obtained by writing to:

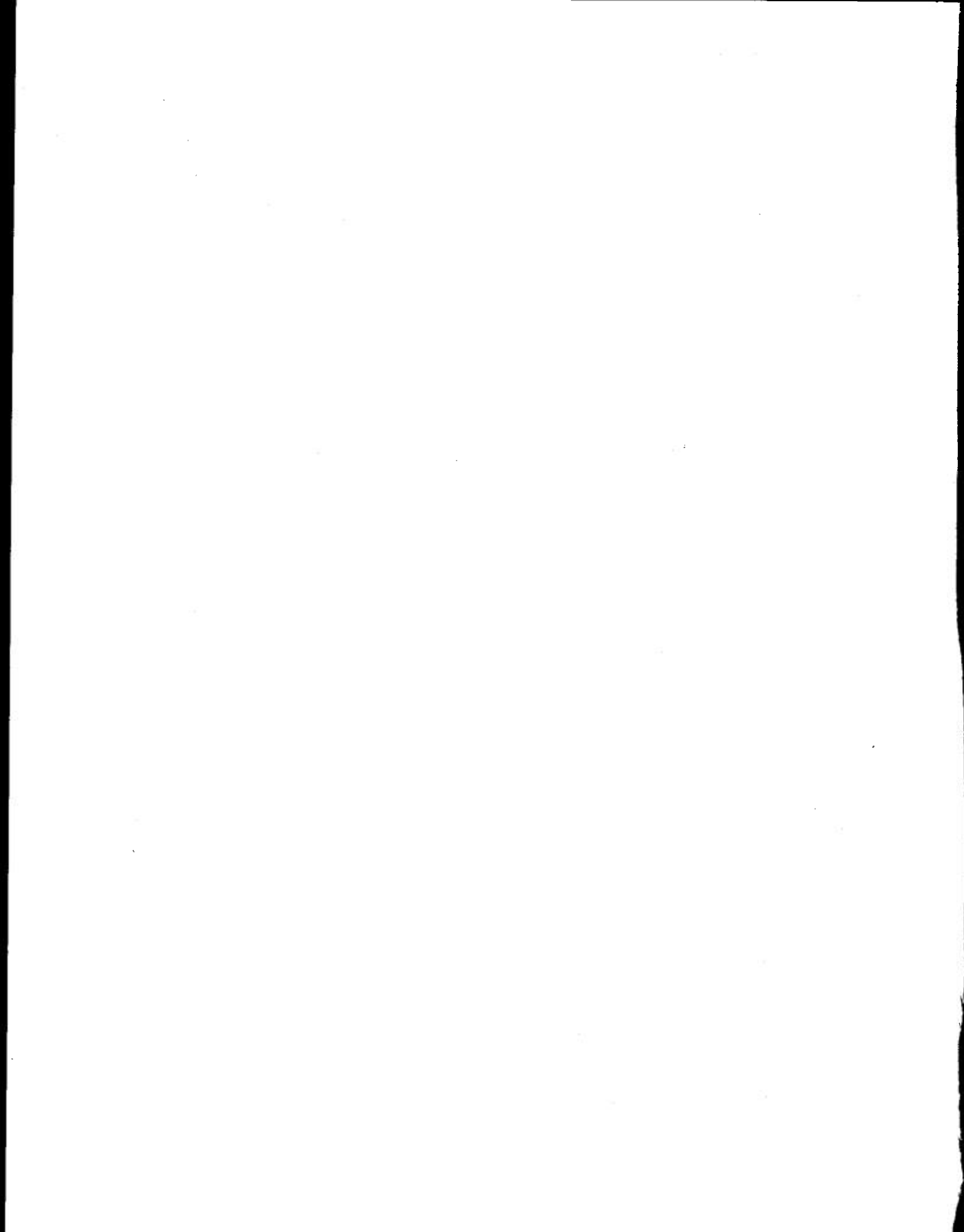
Dr. Gary M. Olson
Department of Psychology
Michigan State University
East Lansing, Michigan 48823

or

Dr. George Moeller
Human Factors Branch
Submarine Medical Research Laboratory
Submarine Base New London
Groton, Connecticut 06340

REFERENCES

1. Smith, E.E. Choice reaction time: An analysis of the major theoretical positions. Psychol Bull. 1968, 69, 77-110.
2. Olson, G.M., & Laxar, K. Asymmetries in processing the terms "right" and "left." J Exp Psychol, 1973, 100, 284-290.



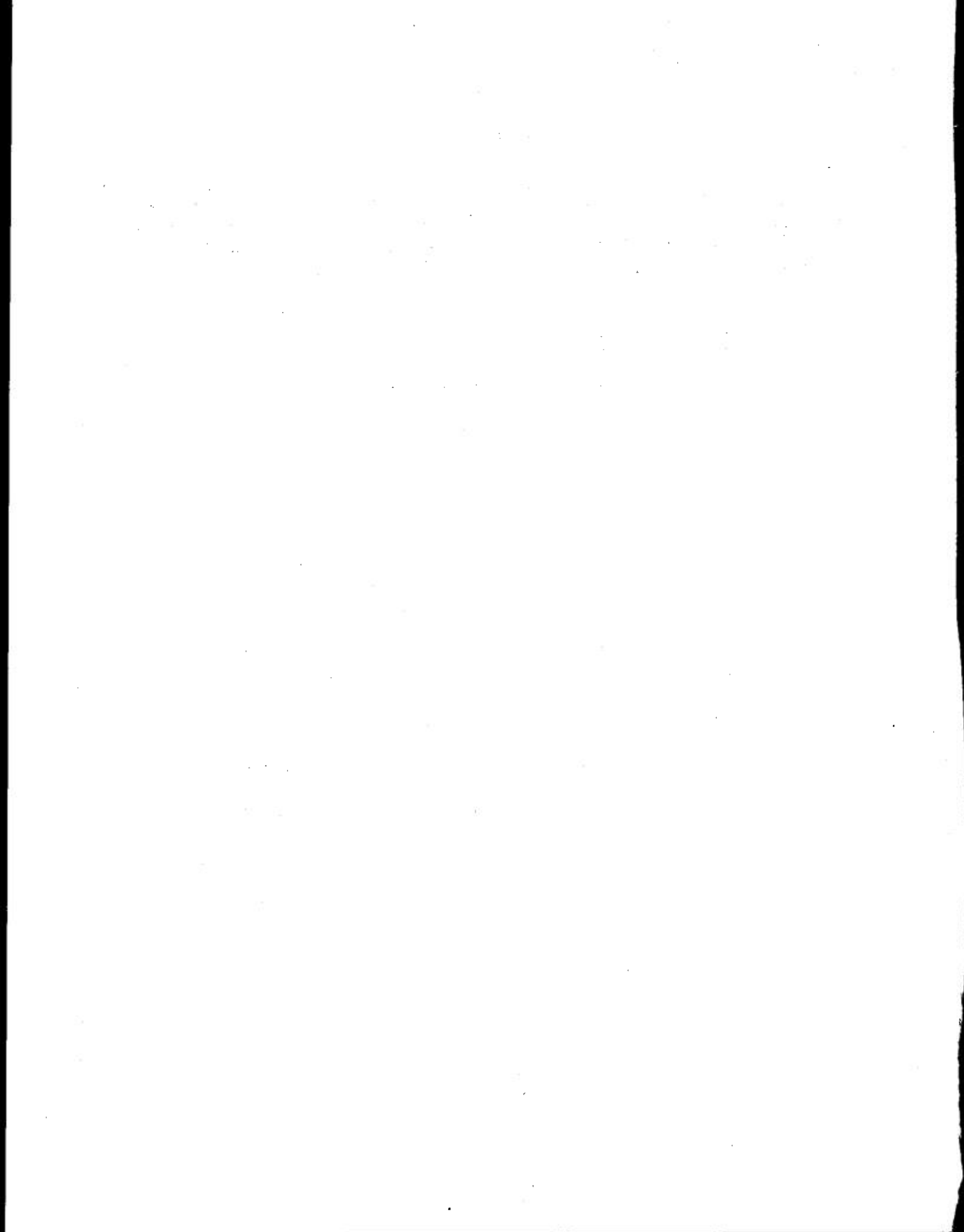
Appendix A

Magnetic Tape Format

In our uses of RTLAB we write the raw data for each subject on magnetic tape for off-line analysis. All data are entered as decimal numbers in

ASCII characters. A separate file is written for each subject-session, with five records in each file organized as follows:

<u>Record</u>	<u>Word(s)</u>	<u>Contents</u>
1	1	Experiment Number
	2	Subject Number
	3	Group Number
	4	Number of Trials
	5	Number of Stimuli
	6	Number of Foreperiods
	7	Error Code (1 = all errors rerun, 2 = errors not rerun)
	8-31	Dummy zeroes
2	1-# trials	Stimuli in order presented
3	1-# trials	Foreperiods in order presented
4	1-# trials	Response codes
5	1-# trials	Reaction times in milliseconds



Appendix B

Subroutines of RTEXP and their functions (Arranged alphabetically)

<u>Subroutine Name</u>	<u>Entry Point (all indirect)</u>	<u>Function</u>	<u>Remarks</u>
ALPHA	UQUER	Query experimenter on teletype for parameters needed in ALPHA or its subroutines.	User supplied (JMP $\emptyset$, 3 as no-op if no queries)
	PRES	Present a trial, handle response and reaction time so that data are stored via SCORE, present feedback (if any).	User supplied
	ADONE	Inform subject that session is over.	User supplied (JMP $\emptyset$, 3 if not used)
	(ARTUP)	Contains address of user's interrupt service routine for RTC, TTY, CRT, or any other devices requesting interrupts during use of RTLAB.	User supplied. In our applications this is in a sub-routine of ALPHA.
ALS		Tektronix Corporation PLOT-10 package of CRT subroutines for Nova mini-computers.	
BMAN	AOUTP	Prints text messages on teletype.	
CARVE	CLEVE	Called by RTANL to divide a double precision value by a single precision one and print the results on the teletype in decimal notation.	Quotient is printed but not saved.

Appendix B cont.

<u>Subroutine Name</u>	<u>Entry Point (all indirect)</u>	<u>Function</u>	<u>Remarks</u>
CRTXT	SHOW	Writes text messages on CRT.	
DATYP	ADAT	Lists raw data by trials in teletype.	
DESIGN	ABEGIN	Queries experimenter for all the basic parameters for designing and running a reaction time experiment. If it recognizes the experiment identification number it assumes it already knows all the basic parameters. Upon return it has constructed tables of appropriately randomized stimulus codes and foreperiods for the sequence of trials.	
	AFORE	Returns foreperiod value for current trial to location FP in RTEXP.	
	ASTIM	Returns stimulus code for current trial to location CS in RTEXP.	
DPACC	QUACC	Called by RTANL to selectively accumulate data in a double precision register.	
EOF	AEOF	Called by MTAPE to write end-of-file marks.	
INOUT		General teletype subroutine package.	All based on Data General routines.
	APUTC	Transmit ASCII character to teletype from CPU.	

Appendix B cont.

<u>Subroutine Name</u>	<u>Entry Point (all indirect)</u>	<u>Function</u>	<u>Remarks</u>
INOUT cont.	AGETC	Transmit ASCII character to CPU from teletype.	
	IGETI	Entry into GETC if via TTY interrupt.	
	ABNOC	Binary to octal conversion (CPU → TTY).	
	AOCBN	Octal to binary. (TTY → CPU).	
	ABNDC	Binary to decimal.	Suppresses leading zeroes but does right justify.
	ADCBN	Decimal to binary.	
MTAPE	AWERB	Write raw data via Calma unit on magnetic tape.	
NZMEM	ACLER	Clears adjacent memory locations.	
PERMUTE	APRMT	Produces one random permutation of <u>n</u> adjacent memory locations. Called by DESIGN to randomize sequence of trials.	
PROQ		Data General package for unsigned multiply-divide. Equivalents in Data General notation are:	
	MONK	.MPYU	
	AMOEB	.DIVU	
	NUN	.MPYA	
	ALGAE	.DIVI	

Appendix B cont.

<u>Subroutine Name</u>	<u>Entry Point (all indirect)</u>	<u>Function</u>	<u>Remarks</u>
RANDI	ARAND	Data General random number generator. Called by PERMUTE.	Seed is controlled originally by DESIGN and then by successive calls to RANDI.
RTANL	PSYCH	Provides simple data analysis.	Analysis is not stored, but is printed directly on teletype.
RTINT	ARTUP	Interrupt service routine for illustrative application.	
SCORE	AScore	Scores and stores responses and reaction times on each trial.	
	ACode	Contains address-1 of table of responses, scored as: 1 = correct 2 = error 3 = min, max time	
	ATime	Contains address-1 of table of reaction times.	
	AOVER	Contains address-1 of table of trial numbers to be run over.	

Appendix C

Flow-Charts of Primary Routines of RTLAB

This appendix contains flow-charts of the following routines:

<u>Figure</u>	<u>Routine</u>	<u>Page</u>
C-1	RTEXP	C-2
C-2	DESIGN	C-6
C-3	ALPHA	C-8
C-4	PERMUTE	C-12

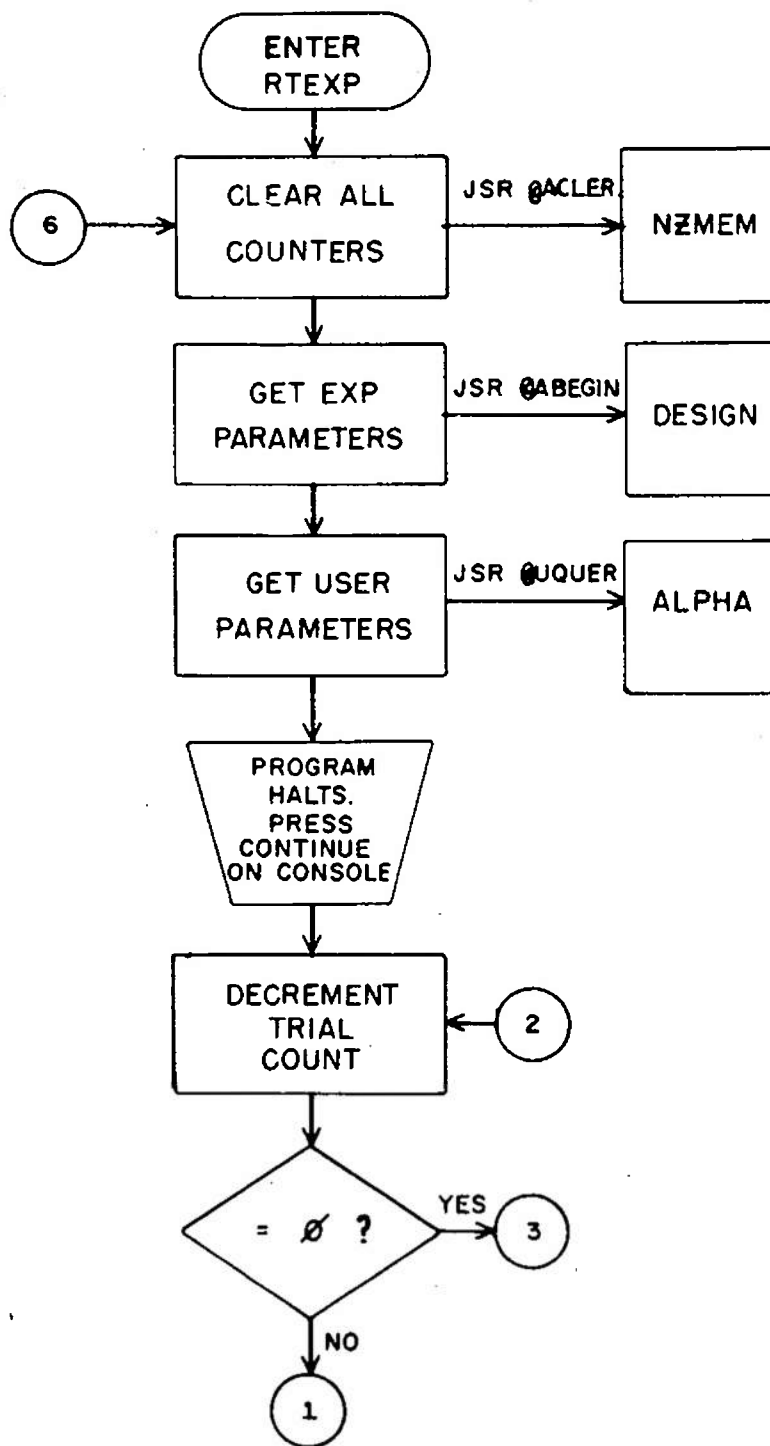


Fig. C-1. Flow-chart for main routine RTEXP. (Part 1 of 4 parts)

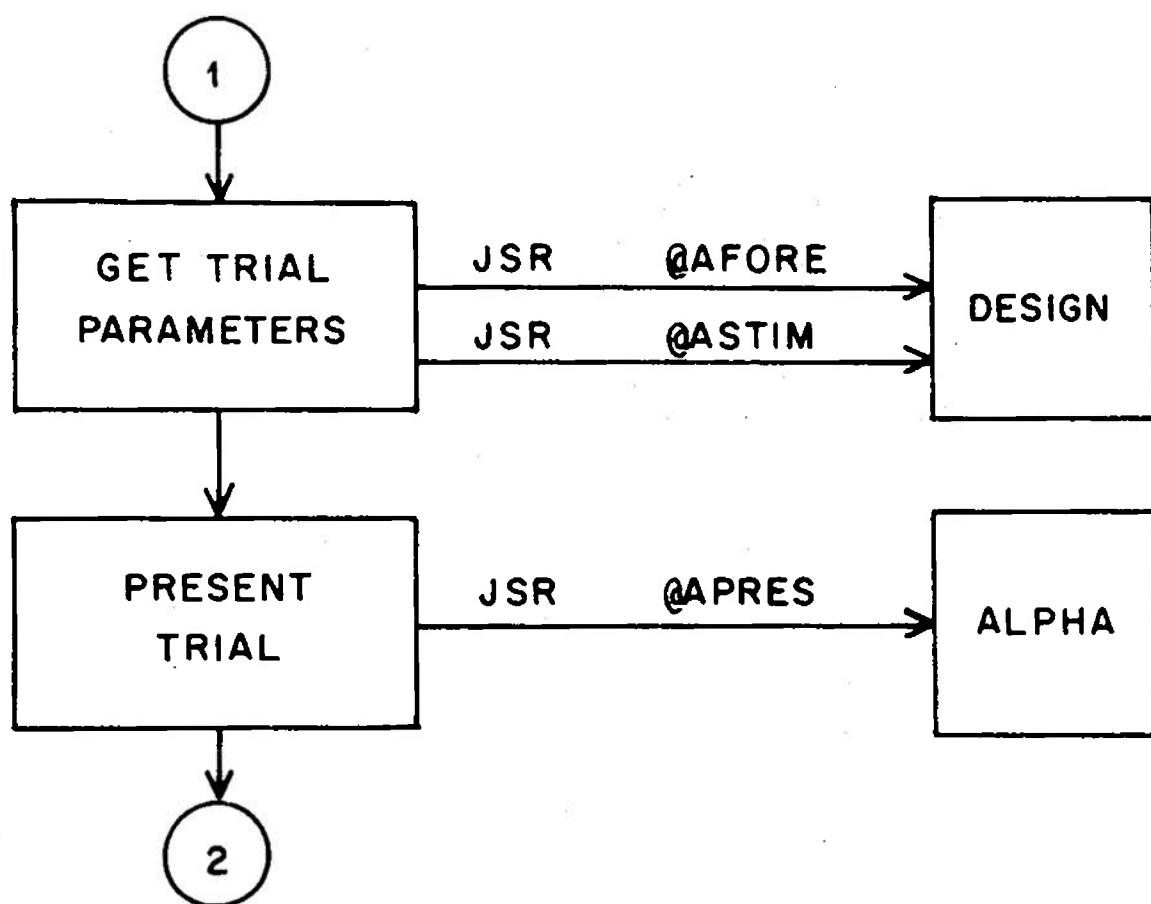


Fig. C-1. Flow-chart for main routine RTEXP. (Part 2 of 4 parts)

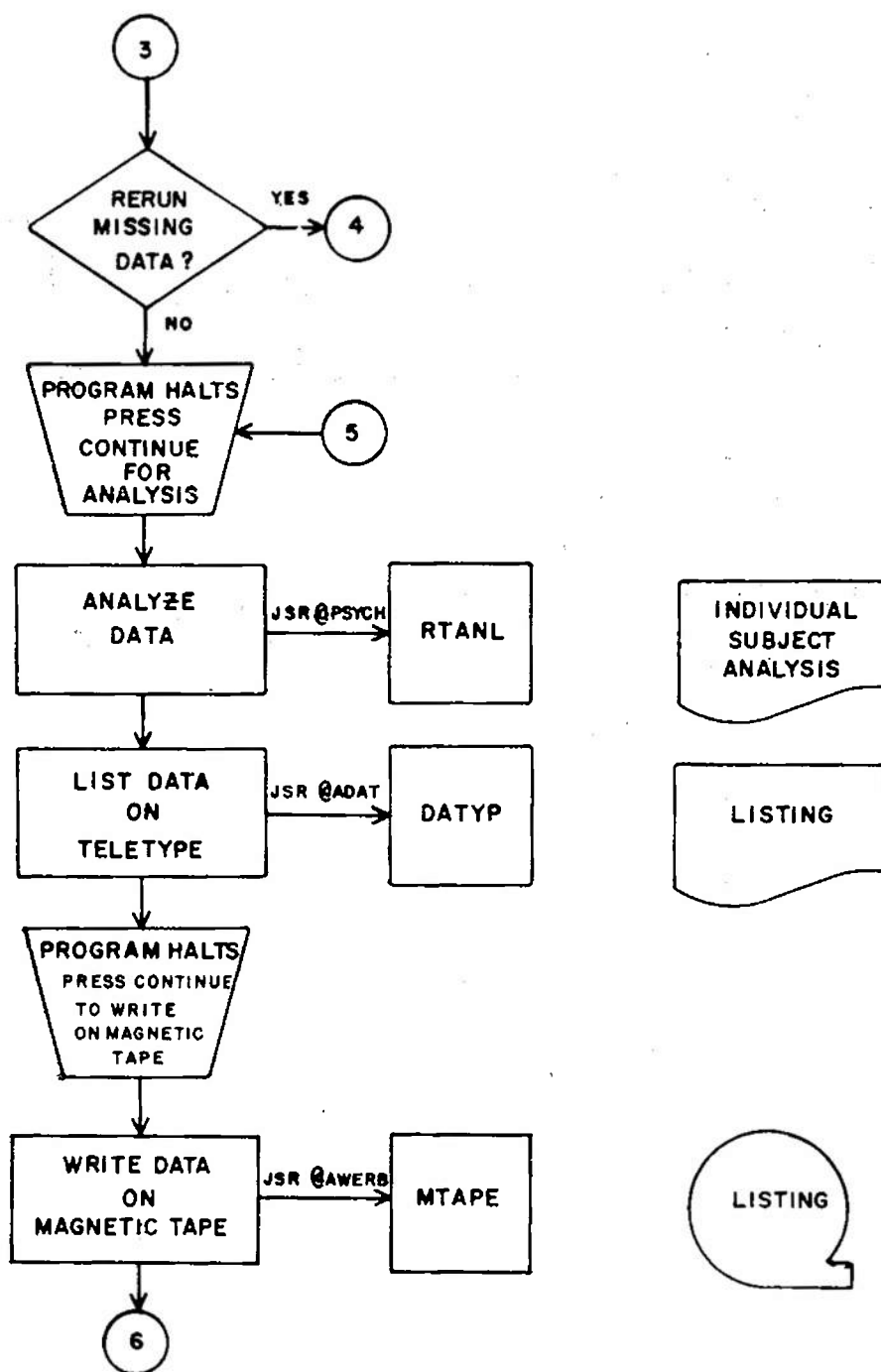


Fig. C-1. Flow-chart for main routine RTEXP. (Part 3 of 4 parts)

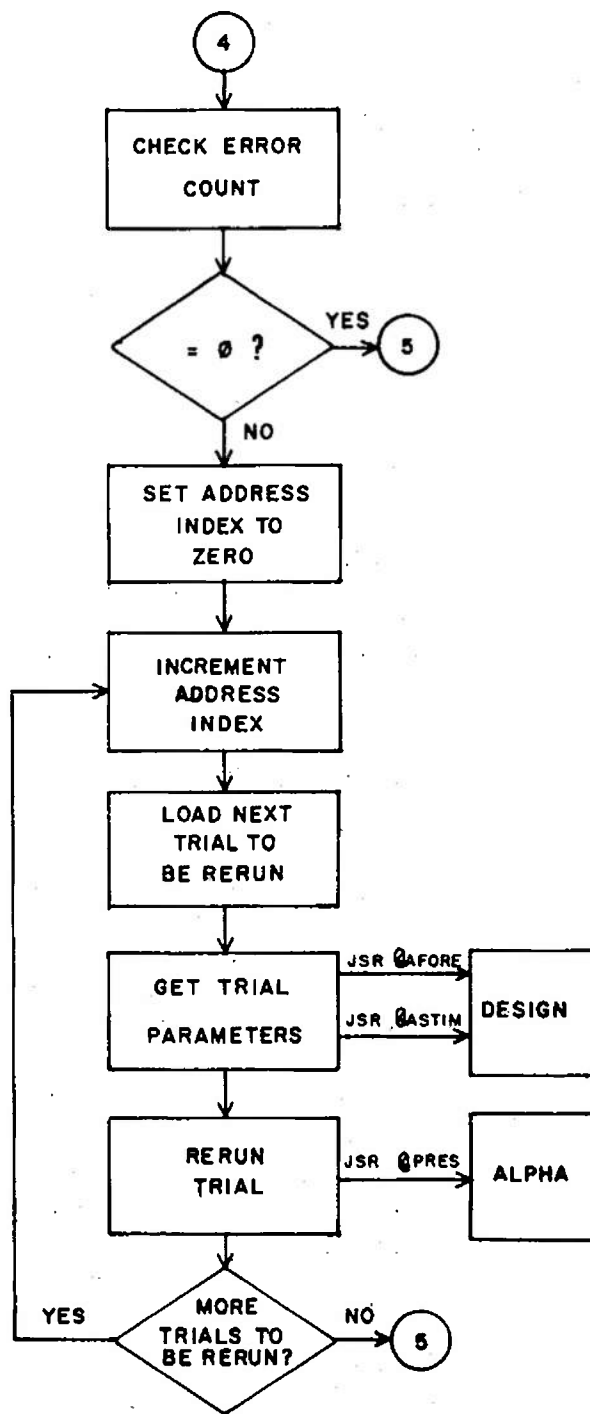


Fig. C-1. Flow-chart for main routine RTEXP. (Part 4 of 4 parts)

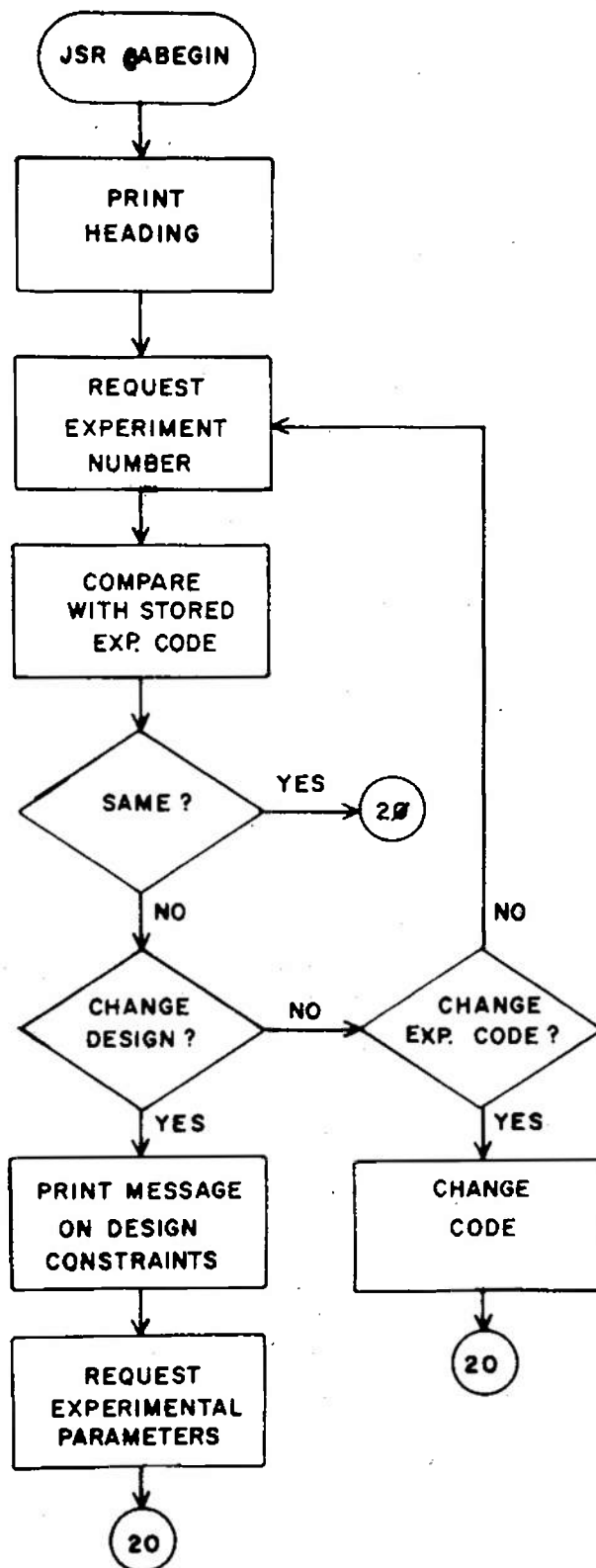


Fig. C-2. Flow-chart for subroutine DESIGN. (Part 1 of 2 parts)

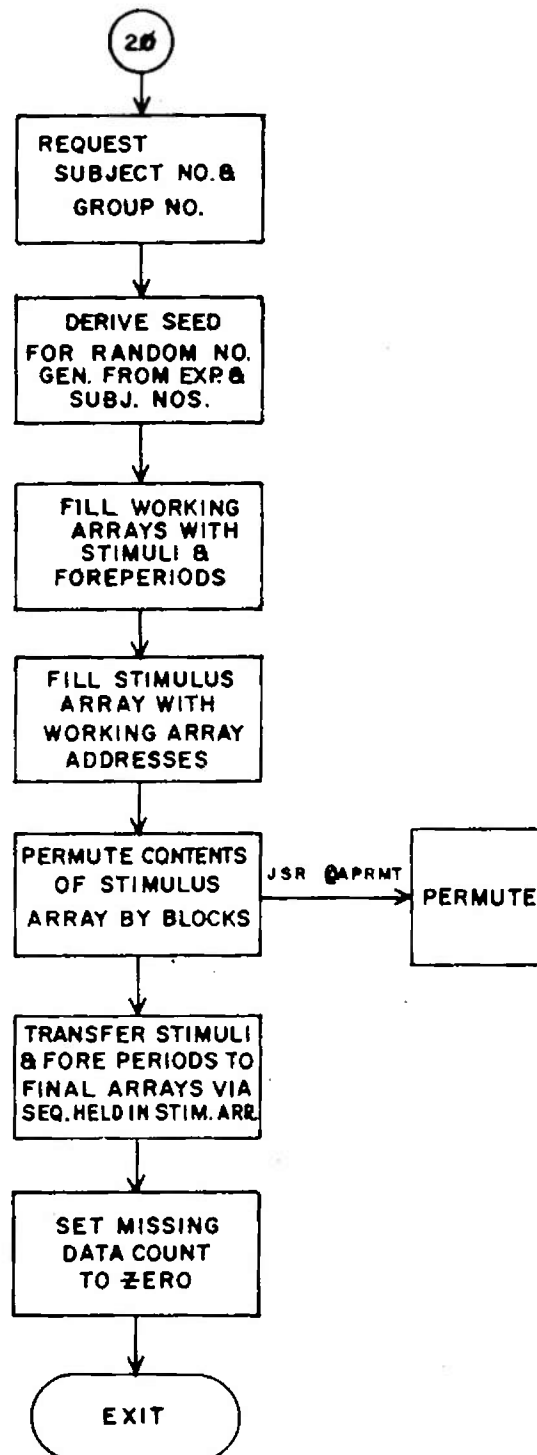


Fig. C-2. Flow-chart for subroutine DESIGN. (Part 2 of 2 parts)

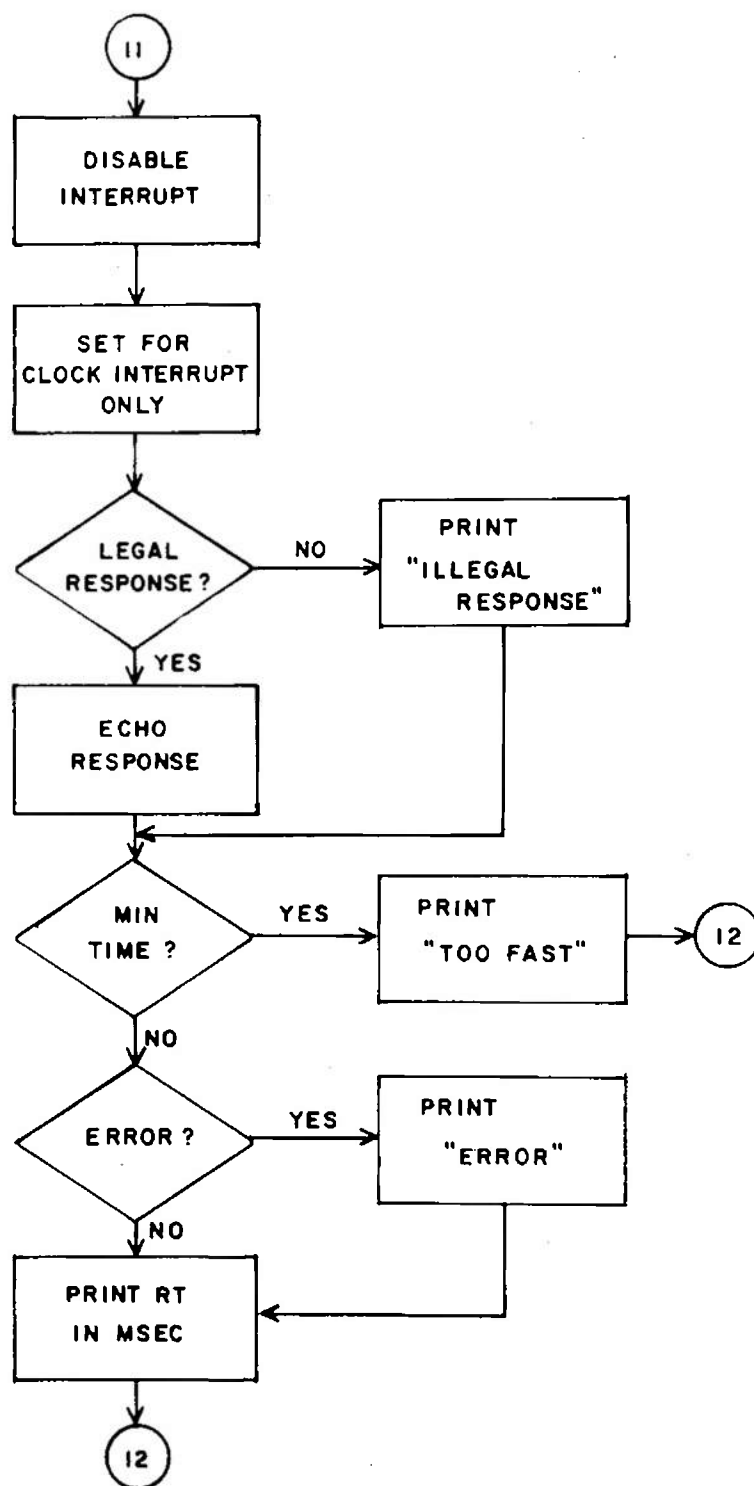


Fig. C-3. Flow-chart for subroutine ALPHA for the experiment discussed in the text. (Part 3 of 4 parts)

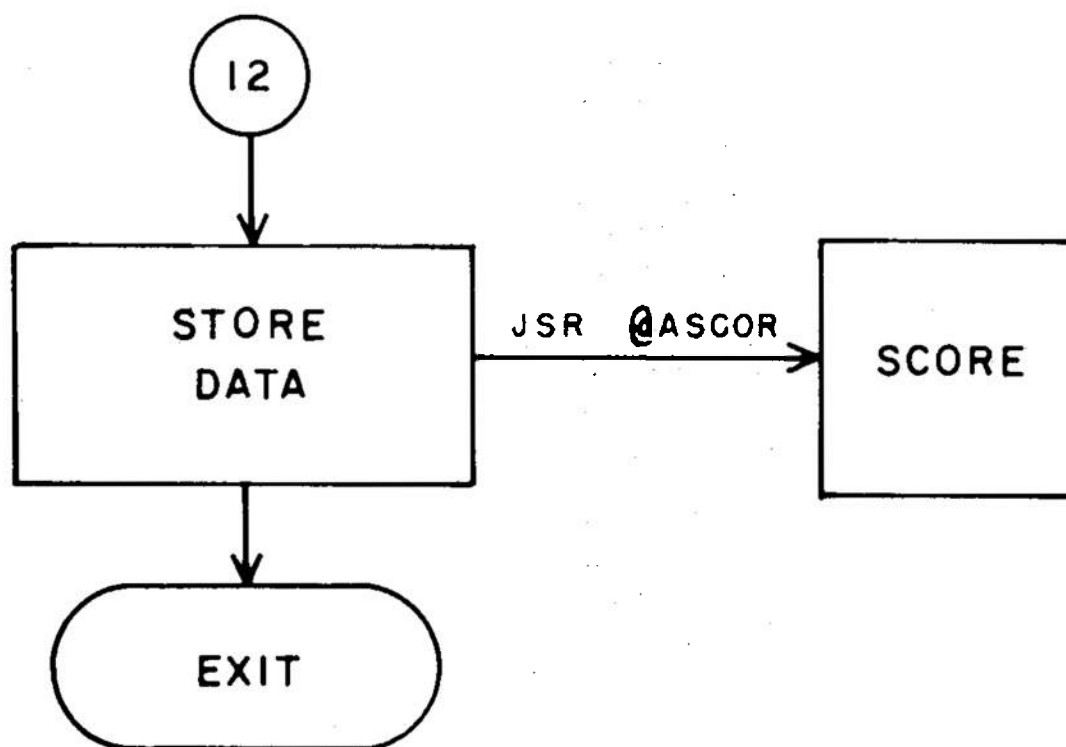
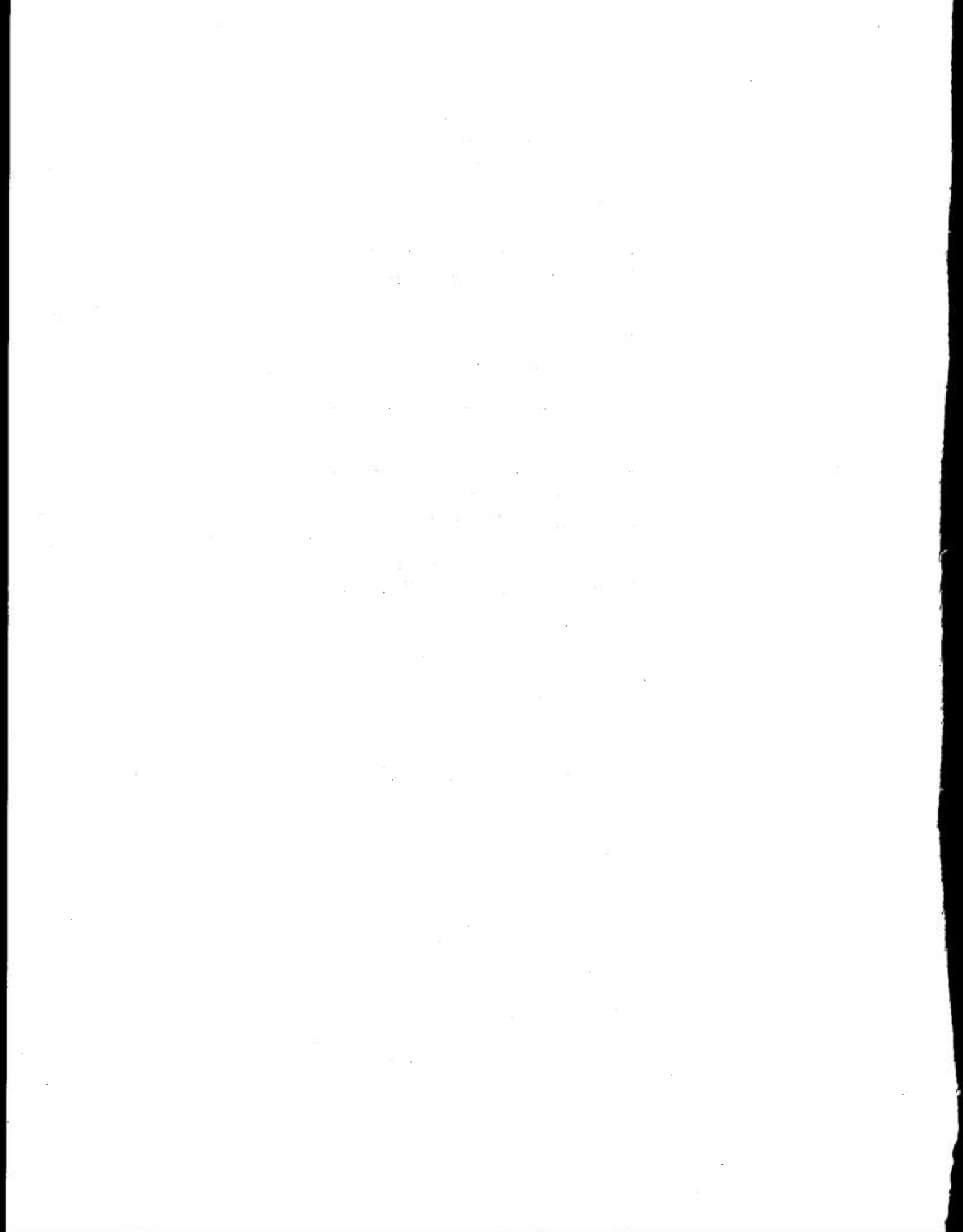


Fig. C-3. Flow-chart for subroutine ALPHA for the experiment discussed in the text. (Part 4 of 4 parts)



Appendix D

Program Listings

All program components of RTLAB are listed in this appendix as output from the NOVA extended assembler. All are in relocatable form. The name or names of the programmer(s) appears in the heading of each listing, along with general information on the use of the routine. Dates and parenthesized letters identify the edition of the subprogram. The listings are in the following order:

<u>Subroutine</u>	<u>Page</u>	<u>Subroutine</u>	<u>Page</u>
RTEXP	D-2	INOUT	D-47
ALPHA	D-7	BMAN	D-52
RTINT	D-16	NZMEM	D-53
SCORE	D-19	CRTXT	D-54
DESIGN	D-21	BOX	D-55
MTAPE	D-34	SALS	D-60
EOF	D-38	RTANL	D-63
DATYP	D-40	CARVE	D-75
RANDI	D-43	PROQ	D-78
PERMUTE	D-45	DPACC	D-80

```

;RTEXP -- MAIN PROGRAM FOR REACTION TIME
;   GENERAL PURPOSE: USER SUPPLIES
;   ROUTINES TO INTERFACE THIS PROGRAM
;   AND ITS PRIMARY SUBROUTINES WITH
;   PARTICULAR DISPLAY AND RESPONSE DEVICES
;
;FOLLOWING SUBROUTINES (WITH ENTRY POINTS) ARE
;   REQUIRED:
;   1. DESIGN (ABEGIN,AFORE,ASTIM) -- ENTER
;       PARAMETER OF EXPERIMENT, SET UP RUN,
;       SUPPLY FOREPERIOD, STIMULUS AND
;       RESPONSE CODES DURING RUN
;   **2. ALPHA (PRES,ARTUP,ADONE,UQUER,PHALF) --
;       ROUTINE TO INTERFACE DISPLAY AND
;       RESPONSE DEVICES WITH RTEXP. MUST
;       INCLUDE INTERRUPT SERVICE ROUTINES.
;       CALL VIA ADONE PRESENTS SIGNAL TO
;       SUBJECT THAT EXPERIMENT IS OVER, VIA
;       PHALF THAT EXPERIMENT IS HALF OVER.
;       CALL VIA UQUER ALLOWS THE ENTRY OF
;       USER-SPECIFIC PARAMETERS DURING THE
;       DESIGN OF THE EXPERIMENT. THE CALLS
;       ADONE,PHALF, AND UQUER CAN, OF COURSE,
;       BE NO-OPS IN THE USER'S PROGRAM.
;   3. NZMEM (ACLER) -- CLEARS MEMORY
;       LOCATIONS
;   4. EMAN (AOUTP) -- PRINTS MESSAGES
;   5. DATYP (ADAT) -- PRINTS DATA ON TTY
;   ++6. MTAPE (AWERB) -- WRITES DATA
;       ON MAGNETIC TAPE
;   ++7. EOF (AEOF) -- CALLED BY MTAPE TO
;       WRITE END OF FILES
;   8. RTANL (PSYCH) -- DOES SIMPLE DATA ANALYSIS
;   9. DPACC (QUACC) -- CALLED BY RTANL TO
;       ACCUMULATE DATA
;   10. CARVE (CLEVE) -- CALLED BY RTANL TO DIVIDE
;       AND PRINT QUOTIENTS
;   11. PROQ (MONK,AMOEB,NUN,ALGAE) -- UNSIGNED
;       DIVIDE, MULTIPLY, CALLED BY CARVE
;   12. INOUT (APUTC,AGETC,ABNOC,AOCEN,ABNDC,
;       ADCEN,IGETI,IDBIN) -- GENERAL
;       TTY SUBROUTINE PACKAGE
;   13. SCORE (ASCOR,ACODE,ATIME,AOVER) --
;       SCORES AND STORES RESPONSES AND
;       REACTION TIMES
;   14. PERMUTE (APRMT) -- PRODUCES RANDOM
;       PERMUTATIONS OF MEMORY LOCATIONS,
;       CALLED BY DESIGN
;   15. RANDI (ARAND) -- RANDOM NUMBER GENERATOR
;       CALLED BY PERMUTE
;   ** USER SUPPLIED
;   ++ USER MAY NEED TO ADAPT FOR OWN TAPE UNIT
;
;RTEXP BEGINS BY QUERYING USER ABOUT THE
;   DESIGN OF THE EXPERIMENT. THEN IT SETS
;   UP A RUN FOR AN INDIVIDUAL SUBJECT
;   AND HALTS, ALLOWING EXPERIMENTER TO START
;   THE EXPERIMENT BY PRESSING "CONTINUE"
;   ON THE CONSOLE. WHEN SESSION IS OVER,
;   A MESSAGE IS GIVEN TO SUBJECT

```

```

;      AND RTEXP HALTS.  THE DATA
;      LISTING AND ANALYSIS ROUTINES CAN
;      THEN BE STARTED WITH "CONTINUE"
;
;      G.O. MOELLER & G.M. OLSON
;      6 APRIL 1973 (A)

```

```

;TITL      PTEXP
;EXTD      ARTUP, ACLER, ABEGIN, AFOPE, ASTIM,
;EXTD      AOUTP, ABNDG, APUTC, ASCOR, PPES, ADONE
;ENT       AMAIN, APAP, INTER, CTDWN, PTIME
;ENT       CLMSK, RESP, TTIME, SEC
;ENT EXP, SUBJ, GROUP, TPLS, ITI, MAX, MIN
;ENT FP, CT, CS, CTRLS, NFP, CRESP, NOVEP
;EXTD AWEPB, REPC, OVINC, AOVEP, PSYCH, AGETC
;EXTD APE, ATYPE, ADAT, UQUEP

102513 .DALC SGT=SUEL# 0,0,SNC ;SGT A,B. A.GT.B
102113 .DALC SGE=ADCL# 0,0,SNC ;SGE A,B. A.GE.B
102512 .DALC SLE=SUEL# 0,0,SZC ;SLE A,B. A.LE.B
102112 .DALC SLT=ADCL# 0,0,SZC ;SLT A,B. A.LT.B
;ZPEL

00000-177773 CLMSK: 177773 ;RTC INTERRUPT ONLY
00001-000000 MIN: 0 ;LB ON PT
00002-000000 MAX: 0 ;UB ON PT
00003-000000 ITI: 0 ;INTER-TPIAL INTERVAL
00004-000000 TRLS: 0 ;# TRIALS
00005-000000 CT: 0 ;# STIMULI
00006-000000 GROUP: 0 ;GROUP #
00007-000000 SUBJ: 0 ;SUBJ #
00010-000000 NFP: 0 ;# FP
00011-000000 EXP: 0 ;EXP ID
00012-000000 CRESP: 0 ;CORRECT RESPONSE CODE
00013-000000 NOVEP: 0 ;CUMULATIVE PERUN TRIALS
00014-000000 FP: 0 ;FOPEPERIOD
00015-000000 CS: 0 ;STIMULUS CODE
00016-000000 SEC: 0
00017-000000 MSEC: 0
00020-000000 INTER: 0 ;ELAPSED ITI
00021-000000 CTDWN: 0 ;ELAPSED FP
00022-000000 RTIME: 0 ;ELAPSED PT
00023-000000 TTIME: 0
00024-000000 TEMP: 0
00025-000000 RESP: 0 ;RESPONSE
00026-000000 CTRLS: 0 ;TRIAL COUNTER
00027-000003 CLKFG: 3
00030-000006
00031-000136 ARAP: RAPUP
00032-000000 AMAIN: MAIN
00033-000002 AHOP: MAIN+2
000003 BLOC1: .BLK 3

;NPEL
000050 CRI=50
000051 CRO=51

;
00000'060277 MAIN: INTDS
00001'060214 NIOC RTC
00002'020015 LDA 0,ARTUP ;ISR ADDRESS TO LOC 1
00003'040001 STA 0,1

```

```

00004'006002$      JSR@   ACLER   ;SET (SEC) TO (SEC + 8) = 0
00005'000011       11
00006'000016-      SEC
00007'006003$      JSP      @ABEGIN ;ENTER PARAMETERS
00010'006025$      JSR      @UQUER ;ENTER USER PARAMTERS
00011'020404       LDA      0,MTXT ;HALT TO ALLOW EXPERIMENTER
00012'006023$      JSP      @ATYPE ;TO READY SUBJECT
00013'063077       HALT
00014'000426       JMP      GOGO
00015'000016' MTXT: TXT4:
00016'005015       .TXT * <15><12>
00017'020012 <12>
00020'051120 PR
00021'051505 ES
00022'020123 S
00023'047503 CO
00024'052116 NT
00025'047111 IN
00026'042525 UE
00027'052040 T
00030'020117 O
00031'052123 ST
00032'051101 AP
00033'020124 T
00034'042523 SE
00035'051523 SS
00036'047511 IO
00037'027116 N.
00040'005015 <15><12>
00041'000000 *
;
00042'020004- GOGO: LDA      0,TRLS
00043'040026- STA      0,CTRLS
00044'020000- LDA      0,CLMSK ;ENABLE RTC
00045'062177 DOBS     0,CPU  ;INTERPUPT
00046'010026- ISZ      CTRLS
00047'020027- LDA      0,CLKF0 ;RTC@1KHZ + START
00050'061114 DOAS     0,RTC

00051'014026- AGIN: DSZ      CTRLS ;DEC TRIALS CTR
00052'006004$ JSR@     AFORE  ;SELECT FP, RETURN (PC+2)
00053'000425 JMP FINIS ;EXIT
00054'006005$ JSR@     ASTIM  ;SELECT STIMULUS
;STIM CODE TO CS
;CORRECT RESP. CODE TO CRESP
00055'004402 JSR TRIAL ;RUN CURRENT TRIAL
00056'000773 JMP AGIN

00057'054420 TRIAL: STA 3,NEXT ;SAVE RETURN
00060'024003- TOVER: LDA      1,ITI ;TRIAL OVER ?
00061'020020- LDA      0,INTEP
00062'106113 SGE      0,1
00063'000776 JMP      --2

00064'060277 INTDS
00065'060214 NIOC     RTC
00066'006002$ JSR@     ACLER   ;RESET TIMES
00067'000010 10
00070'000016- SEC
00071'006012$ JSR      @PRES  ;TIME OUT THE

```

```

                                ;FOREPERIOD, PRESENT
                                ;THE STIMULUS, RECORD
                                ;THE RESPONSE

00072'020440 LDA 0,ZERO
00073'040020- STA 0,INTER
00074'060114 NIOS RTC
00075'060177 INTEN
00076'002401 JMP @NEXT
00077'000000 NEXT: 0
00100'020015$ FINIS: LDA 0,REPC ;GET RERUN INDICATOR
00101'024432 LDA 1,ONE ;GET CONSTANT = 1
00102'106414 SUB# 0,1,SZR ;RERUN?
00103'000433 JMP RAPUP ; NO
00104'020016$ LDA 0,OVINC ; YES -- CHECK IF
00105'024425 LDA 1,ZERO ;ANYTHING TO RERUN
00106'106415 SUB# 0,1,SNR ;
00107'000427 JMP RAPUP
00110'024422 LDA 1,ZERO ;SET RERUN COUNTER
00111'044420 STA 1,MARK ;TO 0
00112'020016$ LDA 0,OVINC
00113'040013- STA 0,NOVER ;STORE RERUN COUNT
00114'020017$ STL: LDA 0,AOVER ;GET ADDRESS OF
00115'010414 ISZ MARK ;RERUN TABLE & CURRENT
00116'030413 LDA 2,MARK ;COUNT
00117'113000 ADD 0,2
00120'025000 LDA 1,0,2
00121'044026- STA 1,CTRLS ;SET TRIAL COUNT
00122'006004$ JSR @AFORE
00123'000000 0
00124'006005$ JSR @ASTIM
00125'004732 JSR TRIAL
00126'014013- DSZ NOVER
00127'000765 JMP STL
00130'000406 JMP RAPUP
00131'000000 MARK: 0
00132'000000 ZERO: 0
00133'000001 ONE: 1
00134'000166' ATAFE: TXTP
00135'000213' ADO: ANAL S
;
00136'024003- RAPUP: LDA 1,ITI
00137'020020- LDA 0,INTER
00140'106113 SGE 0,1
00141'000776 JMP --2
00142'060277 INTDS
00143'060214 NIOC RTC
00144'060250 NIOC CRI
00145'006013$ JSP @ADONE
00146'020767 LDA 0,ADO
00147'101120 MOVZL 0,0
00150'040402 STA 0,++2
00151'006006$ JSP @AOUTP
00152'000000 0
00153'063077 HALT
00154'006020$ JSR @PSYCH
00155'006024$ JSR @ADAT
00156'020756 LDA 0,ATAPE ;WRITE MESSAGE RE TAPE
00157'101120 MOVZL 0,0
00160'040402 STA 0,++2
00161'006006$ JSR @AOUTP

```

```

00162'000000      0
00163'063077      HALT
00164'006014$     JSR @AWERB
00165'002033-     JMP @AHOP
00166'005015      TXTP:  .TXT /<15><12>
00167'005012      <12><12>
00170'047515      MO
00171'047125      UN
00172'020124      T
00173'047101      AN
00174'020104      D
00175'042522      RE
00176'042101      AD
00177'020131      Y
00200'040524      TA
00201'042520      PE
00202'020054      ,
00203'051120      PR
00204'051505      ES
00205'020123      S
00206'047503      CO
00207'052116      NT
00210'047111      IN
00211'042525      UE
00212'000056      ./
00213'005015      ANAL S:  .TXT/<15><12>
00214'051120      PR
00215'051505      ES
00216'020123      S
00217'047503      CO
00220'052116      NT
00221'047111      IN
00222'042525      UE
00223'043040      F
00224'051117      OR
00225'040440      A
00226'040516      NA
00227'054514      LY
00230'044523      SI
00231'000123      S/

```

.END

```

; SUBROUTINE ALPHA -- INTERFACES RTEXT WITH
; CRT FOR SIMPLE RIGHT-LEFT RESEARCH
;(AFTER OLSON & LAXAR, JEP, 1973)
;
; ENTRY POINTS:
; 1. UQUER -- FOR USER-SPECIFIC
;          PARAMETERS
; 2. PRES -- FOR EACH TRIAL
; 3. ADONE -- WHEN RUN IS OVER
;
; USES SUBROUTINES
; 1. BOX
; 2. SCOPE
; 3. RTINT
; 4. CRTXT
; 5. NZMEM
; 6. BMAN
; AND THEIR ASSOCIATED ROUTINES
;
;
; G.M. OLSON
; 5 JUNE 1973 (A)

```

```

.TITL ALPHA
.ENT UQUER,PRES,ADONE
.EXTD ARTUP,FP,CS,RTIME,MAX,MIN
.EXTD CPESP,RESP,CTDWN,TTIME,ASCOR
.EXTD CLMSK,ACLER,SEC,PLBOX
.EXTD SHOW,ATYPE,AGETC

```

```

.ZREL
00000-000064' ATRUE: TRUE
00001-000065' AFALSE: FALSE
00002-000000' PRES: DSPLY
00003-000612' ADONE: FINIS
00004-000544' CLEAR: BLAST
00005-000402' YTIME: TIME
00006-000066' PRINT: WRITE
00007-000660' UQUER: ASK
00010-177771 GMASK: 177771
000050 CRI=50
000051 CRO=51
102113 .DALC SGE=ADCL# 0,0,SNC

```

```

.NREL
00000'054533 DSPLY: STA 3,EXIT ;SAVE RETURN
00001'006004- JSR @CLEAR ;ERASE CRT
00002'060250 NIOC CRI ;WAIT FOR SUBJ
00003'063650 SKPDN CRI ;TO START TRIAL
00004'000777 JMP .-1
00005'020544 LDA 0,AREDY ;START
00006'006006- JSR @PRINT
00007'020541 LDA 0,ZEROX
00010'040011$ STA 0,CTDWN
00011'060114 NIOS RTC
00012'060177 INTEN
;COUNT OUT FP
00013'024002$ FPOVR: LDA 1,FP
00014'020011$ LDA 0,CTDWN

```

```

00015'106113      SGE 0,1
00016'000776      JMP .-2

                                ;CLEAR CRT, PPESNT STIM

00017'060277      CSON:      INTDS
00020'060214      NIOC RTC
00021'063551      SKPBZ CRO
00022'000777      JMP .-1
00023'020003$     LDA 0,CS
00024'040402      STA 0,+.2
00025'006017$     JSR @PLBOX
00026'000000      0
00027'020010-     LDA 0,GMASK
00030'060150      NIOS CPI
00031'060114      NIOS RTC
00032'062177      DOBS 0,CPU

                                ;CHECK FOR RESPONSE
                                ;OR FOR MAX TIME

00033'020515      TLIM:      LDA 0,ZEROX
00034'040004$     STA 0,RTIME
00035'024005$     LDA 1,MAX
00036'063450      SKPBZ CRI
00037'000413      JMP SOME
00040'020004$     LDA 0,RTIME
00041'106113      SGE 0,1
00042'000774      JMP .-4
00043'004475      SLOW:      JSR RERUN
00044'006004-     JSR @CLEAR
00045'020505      LDA 0,AOVTM
00046'006006-     JSR @PRINT
00047'020500      LDA 0,THREE
00050'006013$     JSR @ASCOR
00051'002462      JMP @EXIT

                                ;ECHO RESPONSE

00052'004466      SOME:      JSR RERUN
00053'006004-     JSR @CLEAR
00054'020010$     LDA 0,RESP
00055'024407      LDA 1,TRUE
00056'106415      SUB# 0,1,SNR
00057'000415      JMP TRESP
00060'024405      LDA 1,FALSE
00061'106415      SUB# 0,1,SNR
00062'000417      JMP FPESP
00063'000423      JMP IRESP
00064'000000      TRUE:      0
00065'000000      FALSE:    0
00066'054446      WRITE:     STA 3,EXIT+1
00067'101120      MOVZL 0,0
00070'040402      STA 0,SEA
00071'006020$     JSR @SHOW
00072'000000      SEA:      0
00073'002441      JMP @EXIT+1
00074'020460      TRESP:     LDA 0,ATWRD
00075'006006-     JSR @PRINT
00076'020447      LDA 0,ONES
00077'040010$     STA 0,RESP
00100'000412      JMP LAN
00101'020454      FPESP:     LDA 0,AFWRD
00102'006006-     JSR @PRINT
00103'020443      LDA 0,TWO
00104'040010$     STA 0,RESP

```

```

00105'000405      JMP LAN
00106'020450      IRESP: LDA 0,AIWRD
00107'006006-      JSR @PRINT
00110'020437      LDA 0,THREE
00111'040010$      STA 0,RESP
00112'063551      LAN:   SKPBZ CRO          ;ECHO COMPLETE?
00113'000777      JMP .-1
00114'020012$      LDA 0,TTIME          ;CHECK IF RESPONSE TOO FAST
00115'024006$      LDA 1,MIN
00116'106113      SGE 0,1
00117'000416      JMP QUICK          ;CHECK RESPONSE

00120'030007$      LDA 2,CRESP
00121'024010$      LDA 1,RESP
00122'020423      LDA 0,ONES
00123'132405      SUB 1,2,SNR
00124'000404      JMP RECRD          ;RECORD CORRECT RESP
00125'020432      LDA 0,AERR          ;TELL S, "ERROR"
00126'006006-      JSR @PRINT
00127'020417      LDA 0,TWO
00130'006013$      RECRD: JSR @ASCOR
00131'006005-      JSR @YTIME
00132'002401      JMP @EXIT
          000002      EXIT:  .ELK 2

00135'004403      QUICK: JSR RERUN
00136'020415      LDA 0,AFAST
00137'000707      JMP SLOW+3

00140'060277      RERUN: INTDS          ;RESET ISR
00141'060250      NIOC CRI          ;FOR RTC ONLY
00142'020014$      LDA 0,CLMSK
00143'062177      DOBS 0,CPU
00144'001400      JMP 0,3

00145'000001      ONES:  1
00146'000002      TWO:   2
00147'000003      THREE: 3
00150'000000      ZER0X: 0
00151'000160'      AREDY: REDY
00152'000162'      AOVTM: OVTM
00153'000213'      AFAST: FAST
00154'000245'      ATWRD: TWRD
00155'000273'      AFRD:  FWRD
00156'000322'      AIWRD: IWRD
00157'000356'      AERR:  ERROR
00160'003440      REDY:  .TXT * <7>
00161'000040      *
00162'005015      OVTM:  .TXT * <15><12>
00163'005012      <12><12>
00164'005012      <12><12>
00165'005012      <12><12>
00166'020040
00167'020040
00170'020040
00171'020040
00172'020040
00173'020040
00174'020040
00175'020040

```

00176'020040
00177'020040
00200'020040
00201'020040
00202'020040
00203'020040
00204'020040
00205'020040
00206'052040 T
00207'047517 00
00210'051440 S
00211'047514 LO
00212'000127 W*
00213'005015 FAST: .TXT *(<15><12>
00214'005012 <12><12>
00215'005012 <12><12>
00216'005012 <12><12>
00217'020040
00220'020040
00221'020040
00222'020040
00223'020040
00224'020040
00225'020040
00226'020040
00227'020040
00230'020040
00231'020040
00232'020040
00233'020040
00234'020040
00235'020040
00236'020040
00237'052040 T
00240'047517 00
00241'043040 F
00242'051501 AS
00243'020124 T
00244'000000 *
00245'005015 TWPD: .TXT *(<15><12>
00246'005012 <12><12>
00247'005012 <12><12>
00250'005012 <12><12>
00251'020040
00252'020040
00253'020040
00254'020040
00255'020040
00256'020040
00257'020040
00260'020040
00261'020040
00262'020040
00263'020040
00264'020040
00265'020040
00266'020040
00267'020040
00270'051124 TR
00271'042525 UE

```

00272'000040      *
00273'005015  FWRD:  .TXT *(<15><12>
00274'005012  <12><12>
00275'005012  <12><12>
00276'005012  <12><12>
00277'020040
00300'020040
00301'020040
00302'020040
00303'020040
00304'020040
00305'020040
00306'020040
00307'020040
00310'020040
00311'020040
00312'020040
00313'020040
00314'020040
00315'020040
00316'040506  FA
00317'051514  LS
00320'020105  E
00321'000000  *
00322'005015  IWRD:  .TXT *(<15><12>
00323'005012  <12><12>
00324'005012  <12><12>
00325'005012  <12><12>
00326'020040
00327'020040
00330'020040
00331'020040
00332'020040
00333'020040
00334'020040
00335'020040
00336'020040
00337'020040
00340'020040
00341'020040
00342'020040
00343'020040
00344'020040
00345'046111  IL
00346'042514  LE
00347'040507  GA
00350'020114  L
00351'042522  RE
00352'050123  SP
00353'047117  ON
00354'042523  SE
00355'000040  *
00356'005015  ERROR:  .TXT *(<15><12>
00357'020040
00360'020040
00361'020040
00362'020040
00363'020040
00364'020040
00365'020040

```

00366'020040
 00367'020040
 00370'020040
 00371'020040
 00372'020040
 00373'020040
 00374'020040
 00375'020040
 00376'051105 ER
 00377'047522 RO
 00400'020122 R
 00401'000000 *

00402'054464	TIME:	STA 3,IOTA	
00403'020472		LDA 0,ATTXT	
00404'006006-		JSR @PRINT	
00405'024004s		LDA 1,RTIME	;ENDEC FOR CRT
00406'034575		LDA 3,ZERO	
00407'054461		STA 3,XI	
00410'034462		LDA 3,FIVE	
00411'054460		STA 3,LAST	
00412'034450		LDA 3,INST	
00413'054401		STA 3,++1	
00414'000000	LOOP:	0	
00415'020452		LDA 0,C60	
00416'146443		SUB0 2,1,SNC	
00417'101401		INC 0,0,SKP	
00420'147001		ADD 2,1,SKP	
00421'000775		JMP --3	
00422'050442		STA 2,KEEP	
00423'014446		DSZ LAST	
00424'000402		JMP ++2	
00425'000420		JMP TAU	
00426'034442		LDA 3,XI	
00427'030554		LDA 2,ZERO	
00430'156414		SUB# 2,3,SZR	
00431'000414		JMP TAU	
00432'030441		LDA 2,ONE	
00433'034434		LDA 3,C60	
00434'116414		SUB# 0,3,SZR	
00435'000407		JMP TAU-1	
00436'040427		STA 0,KEEP+1	
00437'020424		LDA 0,BLANK	
00440'004440		JSR KAPPA	
00441'020424		LDA 0,KEEP+1	
00442'030422		LDA 2,KEEP	
00443'000404		JMP TAU+2	
00444'050424		STA 2,XI	
00445'004433	TAU:	JSR KAPPA	
00446'030416		LDA 2,KEEP	
00447'010745		ISZ LOOP	
00450'151203		MOVR 2,2,SNC	
00451'000743		JMP LOOP	
00452'020422		LDA 0,AMSEC	
00453'006006-		JSR @PRINT	
00454'002412		JMP @IOTA	
000012		.RDX 10	
00455'023420	TENS:	10000	
00456'001750		1000	
00457'000144		100	

```

00460'000012      10
00461'000001      1
      000010      .RDX 8
00462'030441  INST:  LDA 2, .+TENS-LOOP
00463'000040  BLANK: 40
      000002  KEEP:  .BLK 2
00466'000000  IOTA:  0
00467'000060  C60:   60
00470'000000  XI:    0
00471'000000  LAST:  0
00472'000005  FIVE:  5
00473'000001  ONE:   1
00474'000534' AMSEC: MSEC
00475'000510' ATTXT: TTXT
00476'030410      LDA 2, CONS
00477'050410      STA 2, IDIOT
00500'063551  KAPPA: SKPBZ CRO
00501'000777      JMP .-1
00502'014405      DSZ IDIOT
00503'000777      JMP .-1
00504'061151      DOAS 0, CRO
00505'001400      JMP 0, 3
00506'000250  CONS:  250
00507'000000  IDIOT: 0

00510'005015  TTXT:  .TXT * <15> <12>
00511'020040
00512'020040
00513'020040
00514'020040
00515'020040
00516'020040
00517'020040
00520'020040
00521'020040
00522'020040
00523'020040
00524'020040
00525'020040
00526'020040
00527'020040
00530'044524  TI
00531'042515  ME
00532'036440  =
00533'000040  *
00534'046440  MSEC:  .TXT * M
00535'046111  IL
00536'044514  LI
00537'042523  SE
00540'047503  CO
00541'042116  ND
00542'020123  S
00543'000040  *

00544'054433  BLAST:  STA 3, SAVE
00545'020431      LDA 0, BMSG      ; ERASE CRT
00546'101120      MOVZL 0, 0
00547'040402      STA 0, .+2
00550'006020$     JSR @SHOW
00551'000000      0

```

```

00552'020001      LDA 0,1          ;SET UP ISR
00553'040425      STA 0,SAVE+1      ;FOR COMPLETION
00554'020421      LDA 0,ASR        ;OF ERASE
00555'040001      STA 0,1
00556'020425      LDA 0,ZERO
00557'040425      STA 0,COUNT
00560'020014$     LDA 0,CLMSK
00561'062177      DOBS 0,CPU
00562'060114      NIOS RTC
00563'060177      INTEN
00564'024425      LDA 1,INT
00565'020417      LDA 0,COUNT
00566'106113      SGE 0,1          ;ERASE COMPLETED
00567'000776      JMP .-2          ;NO
00570'060277      INTDS            ;YES
00571'060214      NIOC RTC
00572'020406      LDA 0,SAVE+1
00573'040001      STA 0,1
00574'002403      JMP @SAVE
00575'000605'    ASR:  ISR
00576'000601'    MSG:  MSG
                   000002    SAVE:  .BLK 2
00601'006033    MSG:  .TXT *(<33><14>
00602'000000    *
00603'000000    ZERO:  0
00604'000000    COUNT: 0
00605'010777    ISR:  ISZ COUNT
00606'060114      NIOS RTC
00607'060177      INTEN
00610'002000      JMP @0
                   000012    .RDX 10
00611'001440    INT:  800

                   000010    .RDX 8
00612'054405    FINIS: STA 3,CHETH
00613'006004-    JSR @CLEAR
00614'020404      LDA 0,ETA
00615'006006-    JSR @PRINT
00616'002401      JMP @CHETH
00617'000000    CHETH: 0
00620'000621      ETA:  ZAYIN
00621'005015    ZAYIN: .TXT *(<15><12>
00622'005012    <12><12>
00623'005012    <12><12>
00624'005012    <12><12>
00625'020040
00626'020040
00627'020040
00630'020040
00631'020040
00632'020040
00633'020040
00634'020040
00635'020040
00636'020040
00637'020040
00640'020040
00641'042440      E
00642'042116      ND
00643'047440      0

```

00644'020106 F
 00645'042523 SE
 00646'051523 SS
 00647'047511 IO
 00650'027116 N.
 00651'020040
 00652'044124 TH
 00653'047101 AN
 00654'020113 K
 00655'047531 YO
 00656'027125 U.
 00657'000040 *

00660'054737 ASK: STA 3, CHETH
 00661'020411 LDA 0, TXTAT
 00662'006021\$ JSR @ATYPE
 00663'006022\$ JSR @AGETC
 00664'042000- STA 0, @ATRUE
 00665'020406 LDA 0, TXTAF
 00666'006021\$ JSR @ATYPE
 00667'006022\$ JSR @AGETC
 00670'042001- STA 0, @AFALSE
 00671'002726 JMP @CHETH
 00672'000674' TXTAT: BOZO
 00673'000711' TXTAF: SNUB
 00674'005015 BOZO: .TXT * < 15 > < 12 >
 00675'020012 < 12 >
 00676'052042 "T
 00677'052522 PU
 00700'021105 E"
 00701'041440 C
 00702'040510 HA
 00703'040522 RA
 00704'052103 CT
 00705'051105 ER
 00706'036440 =
 00707'037440 ?
 00710'000040 *
 00711'005015 SNUB: .TXT * < 15 > < 12 >
 00712'020012 < 12 >
 00713'043042 "F
 00714'046101 AL
 00715'042523 SE
 00716'020042 "
 00717'044103 CH
 00720'051101 AP
 00721'041501 AC
 00722'042524 TE
 00723'020122 R
 00724'020075 =
 00725'020077 ?
 00726'000000 *

.END

```

; INTERRUPT SUBRT FOR RT EXPERIMENTS
;   USING CRT AND RTC
;
;   CUMULATE TIME AT CLOCK FREQ IN
;   ;LOCS INTER, CTDWN, + RTIME

;   G.O. MOELLER
;   MODIFIED -- G.M. OLSON
;   28 FEBRUARY 1973(A)

;
;   .TITL RTINT
;   .ENT ARTUP
;   .EXTD ARAP,AOUTP,CLMSK,TTIME
;   .EXTD RESP,INTER,CTDWN,RTIME
;   .NREL

;
00000'040435 INRUP: STA 0,CSPAR
00001'061477 DIB 0,CPU
00002'060214 NIOC RTC
00003'101005 MOV 0,0,SNR
00004'000432 JMP SRCRT
00005'101200 MOVR 0,0
00006'101200 MOVR 0,0
00007'101202 MOVR 0,0,SZC
00010'000406 JMP SRCL
00011'101200 MOVR 0,0
00012'101200 MOVR 0,0
00013'101202 MOVR 0,0,SZC
00014'000455 JMP IOER
00015'000475 JMP TELE

;CLOCK SERVICE ROUTINE
00016'020417 SRCL: LDA 0,CSPAR ;SAVE AC0
00017'040414 STA 0,CLSV
00020'020000 LDA 0,0 ; " LOC 0
00021'040413 STA 0,CLSV+1

;
00022'010006$ ISZ INTER
00023'010007$ ISZ CTDWN
00024'010010$ ISZ RTIME

;RETURN
00025'020407 LDA 0,CLSV+1
00026'040000 STA 0,0
00027'020404 LDA 0,CLSV
00030'060114 NIOS PTC
00031'060177 INTEN
00032'002000 JMP0 0

;
00033'000000 CLSV: 0
00034'000000 0
00035'000000 CSPAP: 0
000050 CRI=50

;CRT ROUTINE
;
00036'020777 SPCRT: LDA 0,CSPAP
00037'040426 STA 0,TINT ;SAVE AC0
00040'020000 LDA 0,0 ; " LOC 0
00041'040425 STA 0,TINT+1

```

```

00042'044425      STA      1,TINT+2
00043'063650      SKPDN    CRI
00044'000777      JMP      .-1
00045'060650      DIAC     0,CRI
00046'024416      LDA      1,PMSK
00047'123400      AND      1,0      ;STPIP PARITY BIT
00050'040005$     STA      0,RESP
00051'020010$     LDA      0,RTIME
00052'040004$     STA      0,TTIME
;
00053'020003$ DLIV:  LDA      0,CLMSK ;PTC INT ONLY
00054'062077      MSKO     0
00055'020411      LDA      0,TINT+1
00056'024411      LDA      1,TINT+2
00057'040000      STA      0,0
00060'020405      LDA      0,TINT
00061'060114      NIOS     RTC
00062'060177      INTEN
00063'002000      JMP0     0
;
00064'000177      PMSK:    177
000004          TINT:     .BLK 4
;
; ERROR ROUTINE
00071'020406      IOEP:   LDA      0,ACAUS
00072'101120      MOVZL    0,0
00073'040402      STA      0,.-+2
00074'006002$     JSR0     AOUTP
00075'000000      0
00076'002001$     JMP0     ARAP
00077'000100'     ACAUS:   CAUS
;
00100'005015      CAUS:    .TXT*<15><12>
00101'044412      <12>1
00102'052116      NT
00103'051105      ER
00104'052522      RU
00105'052120      PT
00106'042440      E
00107'051122      PR
00110'051117      OR
00111'000040      *
;
; TELETYPE ROUTINE
00112'020406      TELE:   LDA      0,ATELE
00113'101120      MOVZL    0,0
00114'040402      STA      0,.-+2
00115'006002$     JSR      @AOUTP
00116'000000      0
00117'002001$     JMP      @ARAP
00120'000121'     ATELE:   TELEX
;
00121'006440      TELEX:   .TXT * <15>
00122'005012      <12><12>
00123'042524      TE
00124'042514      LE
00125'054524      TY
00126'042520      PE
00127'044440      I
00130'052116      NT

```

00131'051105 ER
00132'052522 RU
00133'052120 PT
00134'026440 -
00135'020055 -
00136'052522 RU
00137'020116 N
00140'051511 IS
00141'040440 A
00142'047502 BO
00143'052122 RT
00144'042105 ED
00145'000040 *

00000-000000' ARTUP: .ZREL
; INRUP
 .END

```

;SUBROUTINE SCORE -- STORES INFORMATION DURING
;REACTION TIME EXPERIMENT FOR LATER ANALYSIS.
;KEEPS TRACK OF WHICH TRIALS NEED TO BE RERUN
;AT END OF REGULAR SEQUENCE.
;ENTRY:      JSR @ASCOR
;(AC0) ON ENTRY WILL BE FOLLOWING CODE:
;      1 = CORRECT RESPONSE
;      2 = ERROR RESPONSE
;      3 = NULL (MIN,MAX TIME LIMITS)
;OVINC IS SET TO ZERO BY SUBROUTINE DESIGN
;FOR EACH NEW RUN.  REPC, WHICH IS AN
;INDICATOR FOR HOW TO HANDLE ERRORS AND
;NULLS, IS SET IN DESIGN.  1 = RERUN ALL
;ERRORS AND NULLS, 2 = THROW THEM OUT.
;NOTE:  BUILDS TABLES IN A TOP-TO-BOTTOM
;ORDER, CONSISTENT WITH SUBROUTINES
;DESIGN AND RTANL.
;
;      G.M. OLSON
;      9 NOVEMBER 1972 (A)

```

```

.TITL SCORE
.ENT ASCOR,ACODE,ATIME,AOVER,OVINC,REPC
.EXTD CTRLS,RTIME,NOVER

```

```

.ZREL
00000-000000' ASCOR:  START
00001-000027' ACODE:  CODE-1
00002-000427' ATIME:  TIME-1
00003-001027' AOVER:  OVER-1
00004-000000' OVINC:  0
00005-000000' REPC:   0

```

```

.NREL
00000'054426  START:  STA 3,ONYX      ;SAVE RETURN
00001'024001$ LDA 1,CTRLS      ;GET TRIAL CT
00002'030001-  LDA 2,ACODE      ;ADDR. OF CODE ARRAY
00003'133000   ADD 1,2
00004'041000   STA 0,0,2      ;STORE CODE
00005'030002-  LDA 2,ATIME      ;ADDR. OF TIME ARRAY
00006'133000   ADD 1,2
00007'034002$  LDA 3,RTIME
00010'055000   STA 3,0,2      ;STORE TIME
00011'030416   LDA 2,ONES
00012'112415   SUB# 0,2,SNR
00013'002413   JMP @ONYX
00014'020005-  NULL:  LDA 0,REPC      ;CHECK FOR RERUN OPTION
00015'101233   MOVZR# 0,0,SNC
00016'002410   JMP @ONYX      ;NO RERUN
00017'010004-  ISZ OVINC      ;ADD TO RERUN CT
00020'010003$  ISZ NOVER      ;INCREMENT TEMP. RERUN CT
00021'030003-  LDA 2,AOVER      ;ADDRESS OF RERUN ARRAY
00022'020004-  LDA 0,OVINC
00023'113000   ADD 0,2
00024'045000   STA 1,0,2      ;STORE CURRENT TRIAL CT
00025'002401   JMP @ONYX
00026'000000  ONYX:   0

```

00027'000001 ONES: 1
000400 CODE: .BLK 400
000400 TIME: .BLK 400
000200 OVER: .BLK 200
.END

```

; SUBROUTINE FOR DESIGN OF RT EXPERIMENTS
;   QUERIES "E" FOR DESIGN PARAMETERS
;   SETS UP SET OF TRIALS FOR RUN
;   FETCHES VALUES OF STIMULI AND
;   FOREPERIODS DURING RUN
; CALLS:
;   JSR @ABEGIN      FOR INITIAL DESIGN
;                   OR RUN SETUP
;   JSR @AFORE       TO FETCH FOREPERIOD
;                   --RETURNED IN LOC. FP
;   JSR @ASTIM       TO FETCH STIMULUS CODE
;                   --RETURNED IN LOC. CS
;                   --CORRECT RESPONSE CODE
;                   --RETURNED IN LOC. CRESP
; REQUIRES SUBROUTINES PERMUTE, BMAN, & INOUT
; USES ALL ACCUMULATORS DESTRUCTIVELY
; SAVES CONTENTS OF LOCS 20-25
; ASSUMES:
;   1. STIMULI AND RESPONSES ARE REPRESENTED
;   BY NUMERIC CODES WHICH WILL BE PASSED TO
;   A STIMULUS GENERATION SUBROUTINE AT RUN
;   TIME
;   2. ALL NUMERIC INPUTS DURING
;   QUERIES ARE DECIMAL
;   3. MAX VALUES ARE:
;       # STIMULI = 64
;       # RESPONSES = 64
;       # FOREPERIODS = 16
;       # TRIALS = 256
;   4. BLOCKING FACTOR WILL BE USED
;   TO DETERMINE THE MANNER IN WHICH
;   THE INDIVIDUAL RUNS WILL BE RANDOMIZED
;
;                               G.M. OLSON
;                               19 MARCH 1973 (A)

```

```

.TITL DESIGN
.ENT ABEGIN,AFORE,ASTIM,AFP,ACS,ARESP
.ENT ISTIM,IFORE,ATYPE,APE
.EXTD AOUTP,ADCEN,AGETC,ABNDC,APRMT
.EXTD EXP,SUBJ,GROUP,TRLS,ITI,MAX,MIN
.EXTD FP,CT,CS,CTRLS,NFP,OLD,ARAND
.EXTD CRESP,OVINC,REPC

```

```

.ZPEL
00000-000000' ABEGIN: BEGIN
00001-000050' AFORE:  EWR
00002-000060' ASTIM:  LGA
00003-002577' IFORE:  FORE-1
00004-002177' ISTIM:  STIM-1
00005-001177' ISPACE: SPACE-1
00006-001577' IPLACE: PLACE-1
00007-003177' AFP:    FPTAB-1
00010-003217' ACS:    CSTAB-1
00011-003317' APESP:  RTAB-1
          000010  SAVE:  .BLK 10
          000010  WORK:  .BLK 10

```

```

00032-001024' ARRANGE: PUNT
00033-000001 ONE: 1
00034-000000 ZERO: 0
00035-000000 BLK: 0
00036-000256' ATYPE: PRINT
00037-000264' APE: PGET
00040-000270' ATXT: TXT1
00041-000302' TXT2
00042-000317' TXT3
00043-000346' TXT4
00044-000367' TXT5
00045-000403' TXT6
00046-000416' TXT7
00047-000433' TXT8
00050-000441' TXT9
00051-000451' TXT10
00052-000461' TXT11
00053-000500' TXT12
00054-000516' TXT13
00055-000534' TXT14
00056-000572' TXT15
00057-000621' TXT16
00060-000651' TXT17
00061-000677' TXT18
00062-000711' TXT19
00063-000713' TXT20
00064-000714' TXT21
00065-000744' TXT22
00066-000773' TXT23
00067-001006' TXT24

```

```

                                .NREL
00000'054012- BEGIN: STA 3,SAVE ;SAVE (AC3)
00001'020040- LDA 0,ATXT ;PRINT HEADING
00002'006036- JSR @ATYPE
00003'020064- LDA 0,ATXT+24
00004'006036- JSR @ATYPE
00005'020041- WHY: LDA 0,ATXT+1 ;EXP #
00006'006037- JSR @APE
00007'044440 STA 1,TEXP ;TEMP. EXP. NO.
00010'020006$ LDA 0,EXP
00011'106415 SUB# 0,1,SNR ;CHECK VS. OLD EXP #
00012'000402 JMP .+2
00013'004475 JSR CHANGE ;EXP #S DO NOT MATCH
00014'020044- LDA 0,ATXT+4 ;SUBJ #
00015'006037- JSR @APE
00016'044007$ STA 1,SUBJ
00017'020045- LDA 0,ATXT+5 ;GROUP #
00020'006037- JSR @APE
00021'044010$ STA 1,GROUP
00022'024007$ LDA 1,SUBJ ;ROUTINE TO SET
00023'030006$ LDA 2,EXP ;RANDOM NUMBER
00024'133000 ADD 1,2 ;GENERATOR SEED
00025'150000 COM 2,2
00026'024416 LDA 1,RMSK
00027'133400 AND 1,2
00030'020416 LDA 0,RCON
00031'113000 ADD 0,2
00032'021005 LDA 0,5,2
00033'040022$ STA 0,OLD

```

00034'107400		AND 0,1	
00035'044410		STA 1,MCT	
00036'006023\$		JSR @ARAND	
00037'000022\$		OLD	
00040'014405		DSZ MCT	
00041'000775		JMP .-3	
00042'006032-		JSR @APRANGE	; SET UP RIN
00043'002012-		JMP @SAVE	; RETURN
00044'000077	PMASK:	000077	
00045'000000	MCT:	0	
00046'000400	RCON:	400	
00047'000000	TEXP:	0	
00050'054012-	EWR:	STA 3,SAVE	; SAVE (AC3)
00051'010012-		ISZ SAVE	; INC. PC FOR RETURN
00052'030003-		LDA 2,IFORE	; ADDRESS OF FOREPERIOD ARRAY
00053'024020\$		LDA 1,CTRLS	; CURRENT TRIAL #
00054'133000		ADD 1,2	; GENERATE NEW ADDRESS
00055'025000		LDA 1,0,2	; (AC1) = FOREPERIOD
00056'044015\$		STA 1,FP	; STORE IT
00057'002012-		JMP @SAVE	; RETURN
00060'054012-	LGA:	STA 3,SAVE	; SAVE (AC3)
00061'030004-		LDA 2,ISTIM	; ADDRESS OF STIMULUS ARRAY
00062'024020\$		LDA 1,CTRLS	; CURRENT TRIAL #
00063'133000		ADD 1,2	; GENERATE NEW ADDRESS
00064'025000		LDA 1,0,2	; (AC1) = STIMULUS CODE
00065'044017\$		STA 1,CS	; STORE IT
00066'020020		LDA 0,20	; SAVE (20)
00067'040013-		STA 0,SAVE+1	
00070'020034-		LDA 0,ZERO	; ZERO COUNTER
00071'040022-		STA 0,WORK	
00072'020010-		LDA 0,ACS	; ADDR. OF STIM. TABLE
00073'040020		STA 0,20	
00074'010022-		ISZ WORK	; INCR. COUNT
00075'022020		LDA 0,020	
00076'106414		SUB# 0,1,SNR	; CHECK IF STIM. MATCH
00077'000775		JMP .-3	
00100'030011-		LDA 2,ARESP	; ADDR. OF RESP. TABLE
00101'020022-		LDA 0,WORK	; COUNT OF STIM. POSITION
00102'113000		ADD 0,2	; CREATE ADDR. OF COR. RESP. CODE
00103'021000		LDA 0,0,2	; FETCH CORRECT RESP. CODE
00104'040024\$		STA 0,CRESP	; STORE IT
00105'020013-		LDA 0,SAVE+1	; RESTORE (20)
00106'040020		STA 0,20	
00107'002012-		JMP @SAVE	
00110'054013-	CHANGE:	STA 3,SAVE+1	
00111'020042-		LDA 0,ATXT+2	; QUERY RE DESIGN CHANGE
00112'006037-		JSR @APE	
00113'020033-		LDA 0,ONE	; CHECK FOR "NO"
00114'106415		SUB# 0,1,SNR	
00115'000402		JMP .+2	; IF "NO"
00116'000411		JMP BOS	; IF "YES"
00117'020043-		LDA 0,ATXT+3	; CHECK FOR CODE CHANGE
00120'006037-		JSR @APE	
00121'020033-		LDA 0,ONE	; CHECK FOR "NO"
00122'106415		SUB# 0,1,SNR	
00123'000662		JMP WHY	; IF "NO"
00124'020723		LDA 0,TEXP	; IF "YES"

00125'040006\$	STA 0,EXP	
00126'002013-	JMP @SAVE+1	
00127'020056- BOS:	LDA 0,ATXT+16	;MESSAGE
00130'006036-	JSP @ATYPE	
00131'020057-	LDA 0,ATXT+17	
00132'006036-	JSR @ATYPE	
00133'020060-	LDA 0,ATXT+20	
00134'006036-	JSR @ATYPE	
00135'020712	LDA 0,TEXP	
00136'040006\$	STA 0,EXP	
00137'020046-	LDA 0,ATXT+6	;# TRIALS
00140'006037-	JSR @APE	
00141'044011\$	STA 1,TRLS	
00142'020061-	LDA 0,ATXT+21	
00143'006037-	JSR @APE	
00144'044035-	STA 1,BLK	
00145'020047-	LDA 0,ATXT+7	
00146'006037-	JSR @APE	;ITI
00147'044012\$	STA 1,ITI	
00150'020050-	LDA 0,ATXT+10	;MAX RT
00151'006037-	JSR @APE	
00152'044013\$	STA 1,MAX	
00153'020051-	LDA 0,ATXT+11	;MIN RT
00154'006037-	JSR @APE	
00155'044014\$	STA 1,MIN	
00156'024020	LDA 1,20	;SAVE (20)
00157'044023-	STA 1,WORK+1	
00160'024021	LDA 1,21	
00161'044024-	STA 1,WORK+2	
00162'020052-	LDA 0,ATXT+12	;# FOREPERIODS
00163'006037-	JSR @APE	
00164'044021\$	STA 1,NFP	
00165'044025-	STA 1,WORK+3	
00166'010025-	ISZ WORK+3	
00167'020053-	LDA 0,ATXT+13	
00170'006036-	JSR @ATYPE	
00171'024007-	LDA 1,AFP	;ADDRESS OF FP TABLE
00172'044020	STA 1,20	;LOC 20 = AFP
00173'024034-	LDA 1,ZERO	
00174'044022-	STA 1,WORK	;SET PRINT AT 0
00175'010022- ORD:	ISZ WORK	;INCREMENT PRINT NO.
00176'014025-	DSZ WORK+3	;DECREMENT STEP CT
00177'000402	JMP .+2	;GO TO FP REQ PRINT
00200'000410	JMP MSP	;SKIP IF DONE
00201'024022-	LDA 1,WORK	
00202'006004\$	JSR @ABNDC	;PRINT FP NO.
00203'020063-	LDA 0,ATXT+23	;SPACE
00204'006036-	JSR @ATYPE	
00205'006002\$	JSR @ADCBN	;GET FP VALUE
00206'046020	STA 01,20	;STORE FP VALUE
00207'000766	JMP ORD	;LOOP
00210'020054- MSP:	LDA 0,ATXT+14	;# STIMULI
00211'006037-	JSR @APE	
00212'044016\$	STA 1,CT	
00213'044025-	STA 1,WORK+3	
00214'010025-	ISZ WORK+3	
00215'020055-	LDA 0,ATXT+15	
00216'006036-	JSR @ATYPE	
00217'024034-	LDA 1,ZERO	
00220'044022-	STA 1,WORK	

00221'024010-	LDA 1,ACS	;ADDRESS OF STIM TABLE
00222'044020	STA 1,20	
00223'024011-	LDA 1,ARESP	;ADDRESS OF RESP TABLE
00224'044021	STA 1,21	
00225'010022- OAK:	ISZ WORK	
00226'014025-	DSZ WORK+3	
00227'000402	JMP .+2	
00230'000416	JMP SLC	
00231'024022-	LDA 1,WORK	
00232'006004\$	JSR @AENDC	;PRINT STIM NO.
00233'020066-	LDA 0,ATXT+26	
00234'006036-	JSR @ATYPE	
00235'006002\$	JSR @ADCEN	;GET, STORE STIM CODE
00236'046020	STA 01,20	
00237'020067-	LDA 0,ATXT+27	
00240'006036-	JSR @ATYPE	
00241'006002\$	JSR @ADCEN	;GET,STORE CORRECT RESP CODE
00242'046021	STA 01,21	
00243'020062-	LDA 0,ATXT+22	;CR AND LF
00244'006036-	JSR @ATYPE	
00245'000760	JMP OAK	
00246'020065- SLC:	LDA 0,ATXT+25	;QUERY RE NULLS & ERRORS
00247'006037-	JSR @APE	
00250'044026\$	STA 1,PEPC	
00251'024023-	LDA 1,WORK+1	;PESTORE (20)
00252'044020	STA 1,20	
00253'024024-	LDA 1,WORK+2	;PESTORE (21)
00254'044021	STA 1,21	
00255'002013-	JMP @SAVE+1	
00256'054014- PRINT:	STA 3,SAVE+2	
00257'101120	MOVZL 0,0	
00260'040402	STA 0,JFK	
00261'006001\$	JSR @AOUTP	
00262'000000 JFK:	0	
00263'002014-	JMP @SAVE+2	
00264'054015- PGET:	STA 3,SAVE+3	
00265'006036-	JSR @ATYPE	
00266'006002\$	JSR @ADCEN	
00267'002015-	JMP @SAVE+3	
00270'005015 TXT1:	.TXT *(<15><12>	
00271'020012	<12>	
00272'052122	RT	
00273'042440	E	
00274'050130	XP	
00275'050040	P	
00276'047522	RO	
00277'051107	GR	
00300'046501	AM	
00301'000040	*	
00302'005015 TXT2:	.TXT *(<15><12>	
00303'020012	<12>	
00304'054105	EX	
00305'042520	PE	
00306'044522	RI	
00307'042515	ME	
00310'052116	NT	
00311'041440	C	
00312'042117	OD	

00313'020105	E	
00314'020075	=	
00315'020077	?	
00316'000000	*	
00317'005015	TXT3:	.TXT *(<15><12>
00320'020012	<12>	
00321'042516	NE	
00322'020127	W	
00323'047503	CO	
00324'042504	DE	
00325'020056	.	
00326'041440	C	
00327'040510	HA	
00330'043516	NG	
00331'020105	E	
00332'042504	DE	
00333'044523	SI	
00334'047107	GN	
00335'020077	?	
00336'030450	(1	
00337'020051	)	
00340'047516	NO	
00341'024040	(	
00342'024462	2)	
00343'054440	Y	
00344'051505	ES	
00345'000040	*	
00346'005015	TXT4:	.TXT *(<15><12>
00347'020012	<12>	
00350'044103	CH	
00351'047101	AN	
00352'042507	GE	
00353'041440	C	
00354'042117	OD	
00355'037505	E?	
00356'024040	(	
00357'024461	1)	
00360'047040	N	
00361'020117	O	
00362'031050	(2	
00363'020051	)	
00364'042531	YE	
00365'020123	S	
00366'000000	*	
00367'005015	TXT5:	.TXT *(<15><12>
00370'020012	<12>	
00371'052523	SU	
00372'045102	BJ	
00373'041505	EC	
00374'020124	T	
00375'052516	NU	
00376'041115	MB	
00377'051105	ER	
00400'036440	=	
00401'037440	?	
00402'000040	*	
00403'005015	TXT6:	.TXT *(<15><12>
00404'043440	G	
00405'047522	RO	
00406'050125	UP	

00407'047040 N
 00410'046525 UM
 00411'042502 BE
 00412'020122 R
 00413'020075 =
 00414'020077 ?
 00415'000000 *
 00416'005015 TXT7: .TXT *(<15><12>
 00417'047040 N
 00420'046525 UM
 00421'042502 BE
 00422'020122 R
 00423'043117 OF
 00424'052040 T
 00425'044522 RI
 00426'046101 AL
 00427'020123 S
 00430'020075 =
 00431'020077 ?
 00432'000000 *
 00433'005015 TXT8: .TXT *(<15><12>
 00434'044440 I
 00435'044524 TI
 00436'036440 =
 00437'037440 ?
 00440'000040 *
 00441'005015 TXT9: .TXT *(<15><12>
 00442'046440 M
 00443'054101 AX
 00444'051040 R
 00445'020124 T
 00446'020075 =
 00447'020077 ?
 00450'000000 *
 00451'005015 TXT10: .TXT *(<15><12>
 00452'046440 M
 00453'047111 IN
 00454'051040 R
 00455'020124 T
 00456'020075 =
 00457'020077 ?
 00460'000000 *
 00461'005015 TXT11: .TXT *(<15><12>
 00462'047040 N
 00463'046525 UM
 00464'042502 BE
 00465'020122 R
 00466'043117 OF
 00467'043040 F
 00470'051117 OR
 00471'050105 EP
 00472'051105 ER
 00473'047511 IO
 00474'051504 DS
 00475'036440 =
 00476'037440 ?
 00477'000040 *
 00500'005015 TXT12: .TXT *(<15><12>
 00501'043040 F
 00502'051117 OR

00503'050105 EP
 00504'051105 ER
 00505'047511 IO
 00506'051504 DS
 00507'044440 I
 00510'020116 N
 00511'051515 MS
 00512'041505 EC
 00513'020056 .
 00514'005015 <15><12>
 00515'000000 *
 00516'005015 TXT13: .TXT *<15><12>
 00517'020012 <12>
 00520'052516 NU
 00521'041115 MB
 00522'051105 ER
 00523'047440 O
 00524'020106 F
 00525'052123 ST
 00526'046511 IM
 00527'046125 UL
 00530'020111 I
 00531'020075 =
 00532'020077 ?
 00533'000000 *
 00534'005015 TXT14: .TXT*<15><12>
 00535'052040 T
 00536'050131 YP
 00537'020105 E
 00540'052123 ST
 00541'046511 IM
 00542'046125 UL
 00543'051525 US
 00544'041440 C
 00545'042117 OD
 00546'020105 E
 00547'047506 FO
 00550'046114 LL
 00551'053517 OW
 00552'042105 ED
 00553'041040 B
 00554'020131 Y
 00555'047503 CO
 00556'051122 RR
 00557'041505 EC
 00560'020124 T
 00561'042522 RE
 00562'050123 SP
 00563'047117 ON
 00564'042523 SE
 00565'041440 C
 00566'042117 OD
 00567'020105 E
 00570'005015 <15><12>
 00571'000000 *
 00572'005015 TXT15: .TXT *<15><12>
 00573'041040 B
 00574'047514 LO
 00575'045503 CK
 00576'051440 S

00577'055111	IZ
00600'020105	E
00601'052515	MU
00602'052123	ST
00603'042440	E
00604'052521	QU
00605'046101	AL
00606'047040	N
00607'021450	(#
00610'052123	ST
00611'046511	IM
00612'024051	)(<
00613'043043	#F
00614'024520	P)
00615'020054	,
00616'044127	VH
00617'051105	ER
00620'000105	E*
00621'005015	TXT16: .TXT *<15><12>
00622'047040	N
00623'044440	I
00624'020123	S
00625'047101	AN
00626'044440	I
00627'052116	NT
00630'043505	EG
00631'051105	ER
00632'037040	>
00633'030040	0
00634'020056	.
00635'041040	B
00636'047514	LO
00637'045503	CK
00640'051440	S
00641'055111	IZ
00642'020105	E
00643'052515	MU
00644'052123	ST
00645'040440	A
00646'051514	LS
00647'020117	O
00650'000000	*
00651'005015	TXT17: .TXT *<15><12>
00652'041040	B
00653'020105	E
00654'047101	AN
00655'044440	I
00656'052116	NT
00657'043505	EG
00660'040522	RA
00661'020114	L
00662'052521	QU
00663'052117	OT
00664'042511	IE
00665'052116	NT
00666'047440	O
00667'020106	F
00670'020043	#
00671'051124	TR
00672'040511	IA

00673'051514 LS
 00674'020056 .
 00675'005015 <15><12>
 00676'000000 *
 00677'005015 TXT18: .TXT *(<15><12>
 00700'041040 B
 00701'047514 LO
 00702'045503 CK
 00703'051440 S
 00704'055111 IZ
 00705'020105 E
 00706'020075 =
 00707'020077 ?
 00710'000000 *
 00711'005015 TXT19: .TXT *(<15><12>
 00712'000000 *
 00713'000040 TXT20: .TXT *(<40>*

 00714'005015 TXT21: .TXT *(<15><12>
 00715'040440 A
 00716'046114 LL
 00717'042440 E
 00720'052116 NT
 00721'044522 FI
 00722'051505 ES
 00723'040440 A
 00724'042522 RE
 00725'042040 D
 00726'041505 EC
 00727'046511 IM
 00730'046101 AL
 00731'020054 ,
 00732'046101 AL
 00733'020114 L
 00734'044524 TI
 00735'042515 ME
 00736'020123 S
 00737'047111 IN
 00740'046440 M
 00741'042523 SE
 00742'027103 C.
 00743'000000 *
 00744'005015 TXT22: .TXT *(<15><12>
 00745'051040 R
 00746'051105 ER
 00747'047125 UN
 00750'042440 E
 00751'051122 PR
 00752'051117 OR
 00753'020123 S
 00754'020046 &
 00755'052516 NU
 00756'046114 LL
 00757'020123 S
 00760'030450 (1
 00761'020051)
 00762'051117 OR
 00763'042040 D
 00764'051511 IS
 00765'040503 CA
 00766'042122 RD

```

00767'024040 (
00770'024462 2)
00771'020077 ?
00772'000000 *
00773'020040 TXT23: .TXT *
00774'020040
00775'052123 ST
00776'046511 IM
00777'046125 UL
01000'051525 US
01001'041440 C
01002'042117 OD
01003'020105 E
01004'020075 =
01005'000040 *
01006'020040 TXT24: .TXT *
01007'020040
01010'020040
01011'020040
01012'051040 R
01013'051505 ES
01014'047520 PO
01015'051516 NS
01016'020105 E
01017'047503 CO
01020'042504 DE
01021'036440 =
01022'020040
01023'000000 *

```

```

01024'054014- PUNT: STA 3,SAVE+2 ;SAVE (AC3)
01025'030011S LDA 2,TPLS ;# TRIALS
01026'050022- STA 2,WORK
01027'030020 LDA 2,20 ;SAVE (20)
01030'050015- STA 2,SAVE+3
01031'030021 LDA 2,21 ;SAVE (21)
01032'050016- STA 2,SAVE+4
01033'030022 LDA 2,22 ;SAVE (22)
01034'050017- STA 2,SAVE+5
01035'030023 LDA 2,23 ;SAVE (23)
01036'050020- STA 2,SAVE+6
01037'030024 LDA 2,24 ;SAVE (24)
01040'050021- STA 2,SAVE+7
01041'030004- LDA 2,1STIM ;ADDRESS OF STIM ARRAY
01042'050020 STA 2,20
01043'030005- LDA 2,1SPACE ;ADDRESS OF SPACE
01044'050021 STA 2,21
01045'030006- LDA 2,1PLACE ;ADDRESS OF PLACE
01046'050022 STA 2,22
01047'152400 SUB 2,2 ;(AC2) = 0
01050'050031- STA 2,WORK+7 ;FP INDICATOR = 0
01051'020525 AUDI: LDA 0,CHK ;CHECK STIM
01052'113404 AND 0,2,SZR ;INDICATOR
01053'000423 JMP FORD ;NOT FINISHED
01054'020016S LDA 0,CT ;RESET FOR NEW BLOCK
01055'040023- STA 0,WORK+1
01056'020010- LDA 0,ACS
01057'040023 STA 0,23
01060'020516 LDA 0,CHK ;CHECK FP IND
01061'024031- LDA 1,WORK+7

```

01062'107404		AND 0,1,SZP	
01063'000405		JMP .+5	
01064'020021s		LDA 0,NFP	
01065'040024-		STA 0,WORK+2	
01066'020007-		LDA 0,AFP	
01067'040024		STA 0,24	
01070'026024		LDA 01,24	
01071'044030-		STA 1,WORK+6	
01072'176400		SUB 3,3	
01073'014024-		DSZ WORK+2	;CHECK IF THROUGH
01074'034033-		LDA 3,ONE	;FOREPERIODS
01075'054031-		STA 3,WORK+7	
01076'026023	FORD:	LDA 01,23	;GET STIM FROM TABLE
01077'046021		STA 01,21	;STORE STIM IN SPACE
01100'024030-		LDA 1,WORK+6	;GET CURRENT FP
01101'046022		STA 01,22	;STORE FP IN PLACE
01102'030021		LDA 2,21	;STORE CURRENT ADDRESS
01103'052020		STA 02,20	;OF SPACE IN STIM ARRAY
01104'014022-		DSZ WORK	;CHECK TRIAL COUNT
01105'000402		JMP .+2	
01106'000405		JMP MIX	;GO TO MIX IF DONE
01107'152400		SUB 2,2	
01110'014023-		DSZ WORK+1	;CHECK STIM COUNT
01111'030033-		LDA 2,ONE	;SET INDICATOR TO ONE
01112'000737		JMP AUDI	
01113'102400	MIX:	SUB 0,0	
01114'040022-		STA 0,WORK	;ZERO TRIAL COUNT
01115'020035-		LDA 0,BLK	;BLOCK SIZE
01116'040406		STA 0,SIZE	;SET PERMUTATION PARAMETER
01117'024004-		LDA 1,1STIM	;SET PERMUTATION START
01120'125400		INC 1,1	
01121'044402		STA 1,PARIS	;ADDRESS
01122'006005s	RMIX:	JSR @APRMT	;CALL PERMUTE
01123'000000	PARIS:	0	
01124'000000	SIZE:	0	
01125'020035-		LDA 0,BLK	
01126'024022-		LDA 1,WORK	
01127'107000		ADD 0,1	;INCREMENT TRIAL COUNT
01130'044022-		STA 1,WORK	
01131'030011s		LDA 2,TRLS	
01132'132405		SUB 1,2,SNR	;CHECK IF DONE
01133'000406		JMP TRANS	;TO COPY
01134'020767		LDA 0,PARIS	;SET UP FOR NEXT
01135'024035-		LDA 1,BLK	;PERMUTATION
01136'123000		ADD 1,0	
01137'040764		STA 0,PARIS	
01140'000762		JMP RMIX	
01141'020011s	TRANS:	LDA 0,TRLS	;SET UP COUNT
01142'040022-		STA 0,WORK	
01143'020004-		LDA 0,1STIM	;ADDRESS OF STIM APPAY
01144'040020		STA 0,20	
01145'020003-		LDA 0,1FOPE	;ADDRESS OF FP APPAY
01146'040021		STA 0,21	
01147'032020	SFO:	LDA 02,20	;GET ADDRESS STORED
01150'014020		DSZ 20	;CURRENTLY IN STIM ARRAY
01151'025000		LDA 1,0,2	;LOAD AC1 FROM SPACE
01152'046020		STA 01,20	;STORE IN STIM ARRAY
01153'020424		LDA 0,ACON	;GET CONSTANT
01154'113000		ADD 0,2	;CREATE NEW ADDRESS
01155'025000		LDA 1,0,2	;LOAD AC1 FROM PLACE

01156'046021	STA @1,21	;STORE IN FORE ARRAY
01157'014022-	DSZ WORK	;CHECK IF DONE
01160'000767	JMP SFO	
01161'030034-	LDA 2,ZERO	;ZERO REGISTER OVINC
01162'050025\$	STA 2,OVINC	
01163'030015-	LDA 2,SAVE+3	
01164'050020	STA 2,20	;RESTORE (20)
01165'030016-	LDA 2,SAVE+4	
01166'050021	STA 2,21	;PESTORE (21)
01167'030017-	LDA 2,SAVE+5	
01170'050022	STA 2,22	;PESTORE (22)
01171'030020-	LDA 2,SAVE+6	
01172'050023	STA 2,23	;RESTORE (23)
01173'030021-	LDA 2,SAVE+7	
01174'050024	STA 2,24	;RESTORE (24)
01175'002014-	JMP @SAVE+2	
01176'177777	CHK: 177777	
01177'000400	ACON: 400	

000400	SPACE:	.BLK 400
000400	PLACE:	.BLK 400
000400	STIM:	.BLK 400
000400	FORE:	.BLK 400
000020	FPTAB:	.BLK 20
000100	CSTAB:	.BLK 100
000100	RTAB:	.BLK 100
		.END

```

.TITL MTAPE
.ENT AWERB
.EXTD EXP, SUBJ, GROUP, TRLS, CT, NFP
.EXTD REPC, ISTIM, IFORE, ACODE, ATIME
.EXTD AEOF

```

```

;
;
;
;
;
SUBROUTINE TO DUMP RT DATA ONTO TAPE
;
G.M. OLSON
11 DECEMBER 1972 (A)

```

102113

```
.DALC SGE=ADCL# 0,0,SNC
```

```
.ZREL
```

```

00000-000200' AWD: WD
00001-000204' ATB: TENDC
00002-000000' AWERB: START
00003-000120' ASR: ISR

```

```
.NREL
```

```
00000'054556 START: STA 3,SWAV ;SAVE RETURN
```

```

00001'060474 DIA 0,74 ;GET READY BIT
00002'101100 MOVL 0,0
00003'101113 MOVL# 0,0,SNC ;READY?
00004'000775 JMP -3 ;WAIT

```

```

00005'004467 JSR LZERO ;WRITE LEADING 0
00006'024001S LDA 1,EXP ;WRITE EXP. NO.
00007'006001- JSP @ATB
00010'004464 JSR LZERO
00011'024002S LDA 1,SUBJ ;WRITE SUBJ. NO.
00012'006001- JSP @ATE
00013'004461 JSR LZERO
00014'024003S LDA 1,GROUP ;WRITE GROUP NO.
00015'006001- JSP @ATE
00016'004456 JSR LZERO
00017'024004S LDA 1,TRLS ;WRITE NO. TRIALS
00020'006001- JSP @ATE
00021'004453 JSR LZERO
00022'024005S LDA 1,CT ;WRITE NO. STIMULI
00023'006001- JSP @ATE
00024'004450 JSR LZERO
00025'024006S LDA 1,NFP ;WRITE NO. FOREPERIODS
00026'006001- JSP @ATE
00027'004445 JSR LZERO
00030'024007S LDA 1,REPC ;WRITE ERROR CODE
00031'006001- JSP @ATE
00032'020477 LDA 0,BIG
00033'040537 STA 0,WORK+5
00034'004440 MOPE: JSR LZERO
00035'024471 LDA 1,ZERO
00036'006001- JSP @ATE
00037'014533 DSZ WORK+5
00040'000774 JMP MORE

```

```

00041'004437      JSR STALL
00042'020533      LDA 0,IRG          ;WRITE IFG
00043'062074      DOB 0,74
00044'004434      JSR STALL
00045'020010$ TSTM: LDA 0,ISTIM      ;STORE STIM
00046'040402      STA 0,.+2
00047'004463      JSR PPEP
00050'000000      0
00051'004472      JSR TAPE
00052'020011$ TFP:  LDA 0,IFORE      ;STORE FOREPERIODS
00053'040402      STA 0,.+2
00054'004456      JSR PREP
00055'000000      0
00056'004465      JSR TAPE
00057'020012$ TCD:  LDA 0,ACODE      ;STORE CODES
00060'040402      STA 0,.+2
00061'004451      JSR PREP
00062'000000      0
00063'004460      JSR TAPE
00064'020013$ TTM:  LDA 0,ATIME      ;STORE REACTION TIMES
00065'040402      STA 0,.+2
00066'004444      JSR PREP
00067'000000      0
00070'004453      JSR TAPE

00071'006014$      JSR @AE0F          ;WRITE EOF
00072'000001      1
00073'002463      JMP @SWAV
00074'054464 LZERO: STA 3,SWAV+2      ;ROUTINE TO WRITE LEADING 0S
00075'020501      LDA 0,C60
00076'006000-      JSR @AVD
00077'002461      JMP @SWAV+2
00100'054461 STALL: STA 3,SWAV+3      ;STORE RETURN
00101'020003-      LDA 0,ASR          ;STORE ISP IN LOC 1
00102'040001      STA 0,1
00103'020423      LDA 0,ZERO          ;SET COUNT TO 0
00104'040423      STA 0,COUNT
00105'020420      LDA 0,CLMSK        ;SET CLOCK INTERRUPT
00106'062177      DOBS 0,CPU
00107'020415      LDA 0,CLKF0        ;SET CLOCK TICKING
00110'061114      DOAS 0,RTC
00111'024417      LDA 1,CONS        ;SET CHECK VALUE
00112'020415      LDA 0,COUNT
00113'106113      SGE 0,1            ;INTERVAL OVER?
00114'000776      JMP .-2
00115'060277      INTDS
00116'060214      NIOC RTC
00117'002442      JMP @SWAV+3
00120'010407 ISR:  ISZ COUNT
00121'060114      NIOS RTC
00122'060177      INTEN
00123'002000      JMP 00
00124'000001 CLKF0: 1
00125'177773 CLMSK: 177773
00126'000000 ZERO: 0
00127'000000 COUNT: 0
00130'000050 CONS: 50
00131'000030 BIG: 30
;
;ROUTINE TO SET UP TRANSFER TO TAPE

```

```

00132'054425  PREP:   STA 3,SWAV+1      ;SAVE (AC3) + 1
00133'010424      ISZ  SWAV+1
00134'021400      LDA 0,0,3          ;GET START ADDRESS
00135'024004$     LDA 1,TRLS        ;GET NUMBER OF TRIALS
00136'044427      STA 1,WORK        ;STORE NUMBER OF TRIALS
00137'107000      ADD 0,1          ;GENERATE FIRST STORE ADDRESS
00140'125400      INC 1,1          ;(DATA STORED IN REVERSE
                                ;ORDER BY RT PROGRAM)
00141'044030      STA 1,30         ;STORE ADDR. TO AUTO-DEC. LOC
00142'002415      JMP @SWAV+1      ;RETURN

```

; ROUTINE TO WRITE ON TAPE

```

00143'05'414  TAPE:   STA 3,SWAV+1
00144'004730      JSR LZEP0        ;WRITE LEADING ZERO
00145'026030      LDA 1,030        ;GET NEXT DATUM
00146'006001-   JSR @ATB          ;WRITE ON TAPE
00147'014416      DSZ WORK
00150'000774      JMP TAPE+1
00151'004727      JSR STALL
00152'020423      LDA 0,IRG        ;WRITE IRG
00153'062074      DOB 0,74
00154'004724      JSR STALL
00155'002402      JMP @SWAV+1

```

```

000007  SWAV:   .BLK 7
000007  WORK:   .BLK 7
00174'000005  FIVE:   5
00175'000004  IRG:    4
00176'000060  C60:    60
00177'000000  SAVE:   0

```

; G.O. MOELLER BINARY TO ASCII DECIMAL
 ; TAPE WRITING ROUTINES
 ; BASED UPON ENDEC IN "INTRODUCTION
 ; TO PROGRAMMING THE NOVA COMPUTER"
 ; COPYRIGHT 1972, DATA GENERAL CORPORATION
 ; REPRODUCED BY PERMISSION

```

00200'063574  WD:     SKPEZ 74
00201'000777      JMP .-1
00202'061174      DOAS 0,74
00203'001400      JMP 0,3
00204'054773  TENDC:  STA 3,SAVE
00205'034422      LDA 3,INST
00206'054401      STA 3,.-+1
00207'000000  LOOP:   0
00210'020766      LDA 0,C60
00211'146443      SUBO 2,1,SNC
00212'101401      INC 0,0,SKP
00213'147001      ADD 2,1,SKP
00214'000775      JMP .-3
00215'004763      JSR WD
00216'010771      ISZ LOOP
00217'151203      MOVR 2,2,SNC
00220'000767      JMP LOOP
00221'002756      JMP @SAVE

```

```

000012      .PDX 10

```

00222'023420	TENS:	10000
00223'001750		1000
00224'000144		100
00225'000012		10
00226'000001		1
000010		.RDX 8
00227'030413	INST:	LDA 2, .+TENS-LOOP
		.END

```

.TITL EOF
.ENT AEOF

;
; SUBROUTINE TO WRITE EOF MARKS ON
; CALMA TAPES, WITH PROPER TIMING
; CALL IS VIA JSR @AEOF
; NUMBER OF EOF MARKS
; RETURN
;
; G.M. OLSON
; 9 NOVEMBER 1972 (A)
102113 .DALC SGE=ADCL# 0,0,SNC

.ZREL
00000-000000' AEOF: EOF
00001-000037' ASP: ISR

.NREL

00000'054450 EOF: STA 3,SAVE ;SAVE (AC3) + 1
00001'010447 ISZ SAVE
00002'025400 LDA 1,0,3 ;STORE NO. OF FILE
00003'044446 STA 1,SAVE+1 ;MARKS
00004'020001- LDA 0,ASP ;STORE ADDRESS OF
00005'040001 STA 0,1 ;INT. ROUTINE
00006'020437 LDA 0,ZERO ;ZERO TIME
00007'040440 STA 0,COUNT ;COUNT
00010'020434 LDA 0,CLMSK ;SET MASK
00011'062177 DOBS 0,CPU
00012'020431 LDA 0,CLKF0 ;SET CLOCK
00013'061114 DOAS 0,RTC ;TICKING
00014'024443 LDA 1,INT1
00015'004414 JSR CHK
00016'020430 WRT: LDA 0,FMARK ;WRITE EOF
00017'062074 DOB 0,74
00020'020425 LDA 0,ZERO ;ZERO TIME
00021'040426 STA 0,COUNT ;COUNT
00022'024436 LDA 1,INT2 ;SET SECOND
00023'060114 NIOS RTC ;INTERVAL
00024'060177 INTEN
00025'004404 JSR CHK
00026'014423 DSZ SAVE+1
00027'000767 JMP WRT
00030'002420 JMP @SAVE
00031'020416 CHK: LDA 0,COUNT
00032'106113 SGE 0,1
00033'000776 JMP --2
00034'060277 INTDS
00035'060214 NIOC PTC
00036'001400 JMP 0,3
00037'010410 ISP: ISZ COUNT
00040'060114 NIOS PTC
00041'060177 INTEN
00042'002000 JMP @0
00043'000001 CLKF0: 1
00044'177773 CLMSK: 177773
00045'000000 ZERO: 0
00046'000010 FMARK: 10

```

00047'000000 COUNT: 0
000007 SAVE: .BLK 7
000012 .RDX 10
00057'000062 INT1: 50
00060'000113 INT2: 75
 .END

```
; SUBROUTINE TO TYPE OUT THE DATA
;   FOR EACH SUBJECT
```

```
.TITL   DATYP
.ENT ADAT, ITYP, BTXT
.EXTD AENDC, APUTC, ADCEN, AGETC, AOUTP
.EXTD APE, TRLS, ATYPE, ISTIM, IFORE
.EXTD ATIME, ACODE
```

```
; ENTRY IS VIA JSR@ ADAT
```

```
; 27 DEC 1972 -- KEVIN LAXAR
; MODIFIED BY G.M.O. -- 19 MARCH 1973(A)
```

```

00000-000064' BTXT:  TXT30
00001-000106'      TXT31
00002-000137'      SPACE
00003-000135'      LINE
00004-000005' ADAT:  GODAT
00005-000000  ITYP:  0

.ZREL
00000'000000  ICT:   0
00001'000000  DCT:   0
00002'000000  ZERO:  0
00003'000000  EVAS:  0
00004'000001  ONE:   1

00005'054776  GODAT:  STA      3, EVAS
00006'024005-  LDA      1, ITYP      ; QUERY RE DATA TYPE OUT.
00007'020775  LDA      0, ONE      ; CHECK FOR "NO"
00010'106415  SUB#     0, 1, SNR
00011'002772  JMP      @EVAS      ; IF NO, PROCEED
00012'020001-  LDA      0, BTXT+1    ; IF YES, TYPE HEADING
00013'006010$  JSR      @ATYPE
00014'020007$  LDA      0, TRLS
00015'040764  STA      0, DCT      ; STORE # TRIALS IN DCT
00016'024764  LDA      1, ZERO
00017'044761  STA      1, ICT      ; INITIALIZE CTR.
00020'010760  POOL:   ISZ      ICT
00021'024757  LDA      1, ICT
00022'006001$  JSR@     AENDC      ; PRINTS TRIAL NUMBER.
00023'020002-  LDA      0, BTXT+2
00024'006010$  JSR@     ATYPE      ; SPACES FOR NEXT CHAR.
00025'034754  LDA      3, DCT
00026'030011$  LDA      2, ISTIM    ; (ISTIM) INTO AC2
00027'173000  ADD      3, 2        ; AC2=ISTIM + TRIALS COUNT
00030'025000  LDA      1, 0, 2    ; PUTS (ISTIM + TRIALS) INTO AC0.
00031'006001$  JSR@     AENDC      ; TYPES THE STIMULUS CODE.
00032'020002-  LDA      0, BTXT+2
00033'006010$  JSR@     ATYPE      ; SPACES
00034'034745  LDA      3, DCT
00035'030012$  LDA      2, IFORE
00036'173000  ADD      3, 2
00037'025000  LDA      1, 0, 2    ; PUTS (IFORE + TRIALS) INTO AC1.
00040'006001$  JSR@     AENDC      ; TYPES FOREPERIOD.

```

00041'020002-	LDA	0,BTXT+2	
00042'006010\$	JSR@	ATYPE	; SPACES.
00043'034736	LDA	3,DCT	
00044'030013\$	LDA	2,ATIME	
00045'173000	ADD	3,2	
00046'025000	LDA	1,0,2	
00047'006001\$	JSR@	ABNDC	
00050'020002-	LDA	0,BTXT+2	
00051'006010\$	JSR@	ATYPE	
00052'034727	LDA	3,DCT	
00053'030014\$	LDA	2,ACODE	
00054'173000	ADD	3,2	
00055'025000	LDA	1,0,2	
00056'006001\$	JSR@	ABNDC	
00057'020003-	LDA	0,BTXT+3	
00060'006010\$	JSR@	ATYPE	; CR AND LF.
00061'014720	DSZ	DCT	; DECR CTR
00062'000736	JMP	POOL	; IF CTR NOT 0, RETURN & DO NEXT ; TRIAL.
00063'002720	JMP	@EVAS	; IF CTR=0, RETURN TO MAIN PROG.

00064'005015	TXT30:	.TXT	*<15><12>
00065'052012	<12>T		
00066'050131	YP		
00067'020105	E		
00070'052517	OU		
00071'020124	T		
00072'040504	DA		
00073'040524	TA		
00074'020077	?		
00075'030450	(1		
00076'047051	)N		
00077'020117	0		
00100'051117	OR		
00101'024040	(		
00102'024462	2)		
00103'042531	YE		
00104'020123	S		
00105'000040	*		
00106'005015	TXT31:	.TXT	*<15><12>
00107'052012	<12>T		
00110'044522	RI		
00111'046101	AL		
00112'020040			
00113'052123	ST		
00114'046511	IM		
00115'046125	UL		
00116'051525	US		
00117'020040			
00120'043040	F		
00121'050040	P		
00122'020040			
00123'020040			
00124'051040	R		
00125'052040	T		
00126'020040			
00127'020040			
00130'041440	C		
00131'042117	OD		

00132'006505 E<15>
00133'005012 <12><12>
00134'000000 *
00135'005015 LINE: .TXT *<15><12>
00136'000000 *
00137'020040 SPACE: .TXT *
00140'000040 *
 .END

```

; RANDOM NUMBER GENERATOR (RELOCATABLE SUBROUTINE)
; GENERATES A (PSEUDO) RANDOM SEQUENCE OF INTEGERS
; IN THE RANGE 0 <= N <= 2**16-1

; INPUT:      ADDRESS OF OLD VALUE POINTED TO BY WORD
;             AFTER JSP

; OUTPUT:     16-BIT NEW RANDOM NUMBER IN AC0
;             AND IN STORAGE REPLACING OLD VALUE

; CALLING SEQUENCE:
;     JSR     @APAND
;     ADDRESS OF OLD VALUE
;     RETURN

; METHOD:      GENERATES A LINEAR CONGRUENTIAL
;             SEQUENCE OF THE FORM:
;              $X(N+1) = (X(N)*A+C) \text{MOD } 2^{**}16$ 

; CAUTION:    IF A K-BIT RANDOM NUMBER (AS OPPOSED
;             TO A 16-BIT NUMBER) IS NEEDED,
;             USE THE MOST SIGNIFICANT
;             K-BITS (THE LEAST SIGNIFICANT K-BITS
;             ARE NOT AS RANDOM).
;             FOR EXAMPLE, TO OBTAIN RANDOM N MOD 2,
;             USE THE SIGN BIT OF THE RESULT

; UNCHANGED:  AC1, AC2
; DESTROYED:  AC0, AC3, CARRY
;
; ROUTINE ADAPTED FROM NOVA MATH
; SUBROUTINE LIBRARY
; COPYRIGHT 1969, DATA GENERAL CORPORATION
; REPRODUCED BY PERMISSION
;
;     ADAPTED BY G.M. OLSON
;     22 AUGUST 1972 (A)
;     .TITL RANDI
;     .ENT APAND

```

```

; .NREL
000000'054432 .PAND: STA 3,.UD03      ; SAVE RETURN
000001'010431      ISZ .UD03      ; BUMP PAST ADDRESS CONSTANT
000002'044426      STA 1,.UD01     ; *SAVE AC1
000003'050426      STA 2,.UD02     ; *SAVE AC2
000004'031400      LDA 2,0,3       ; GET ADDRESS OF OLD VALUE
000005'021000      LDA 0,0,2       ; OLD VALUE TO AC0
000006'004407      JSR .UD50       ; N*A
000007'034425      LDA 3,.UD20     ; GET INCREMENT, C
000010'163000      ADD 3,0         ; (N*A+C) MOD 2**16
000011'041000      STA 0,0,2       ; STORE IT
000012'024416      LDA 1,.UD01     ; *RESTORE AC1
000013'030416      LDA 2,.UD02     ; *RESTORE AC2
000014'002416      JMP 0,.UD03     ; RETURN

```

```

; COMPUTE N*(2**11+2**2+1)

```

```

00015'024420 .UD50: LDA 1,.UD21 ; GET COUNT, X
00016'044415 STA 1,.UD10 ; FOR ITERATION
00017'105120 MOVZL 0,1 ; N*2**(X+1)
00020'125120 MOVZL 1,1
00021'014412 DSZ .UD10
00022'000776 JMP .-2
00023'107000 ADD 0,1
00024'125120 MOVZL 1,1
00025'125120 MOVZL 1,1 ; 2**2*(N*2**(X+1)+N)
00026'123000 ADD 1,0
00027'001400 JMP 0,3 ; N+N*2**2+N*2**(X+3)

```

```

00030'000000 .UD01: 0 ; *SAVE AC1
00031'000000 .UD02: 0 ; *SAVE AC2
00032'000000 .UD03: 0 ; SAVE RETURN

00033'000000 .UD10: 0 ; ITERATION COUNT

00034'033031 .UD20: 33031 ; INCREMENT
00035'000010 .UD21: 10 ; ITERATION COUNT, X
                .ZREL
00000-000000' APAND: .RAND
                .END

```

.TITL PERMUTE
 .ENT APPMT,OLD
 .EXTD APAND

; PELOCATABLE SUBROUTINE TO RANDOMLY
 ; PERMUTE THE CONTENTS OF N ADJACENT
 ; MEMORY LOCATIONS. PERQUIRES PELOCATABLE
 ; RANDOM NUMBER GENERATOR ADDRESSED BY
 ; @APAND.

; CALL TO PERMUTE IS:
 ; JSR @APPMT
 ; ADDRESS OF ARRAY
 ; SIZE OF ARRAY
 ; RETURN
 ; SEED FOR RANDOM NUMBER GENERATOR IS
 ; PASSED VIA LOCATION "OLD" ON ZERO PAGE.

; G.M. OLSON
 ; 22 AUGUST 1972 (A)

	.ZPEL	
00000-000000	APPMT:	PEP
00001-000000	OLD:	0
	.NPEL	
00000'054463	PEP:	STA 3,PMSAV
00001'010462		ISZ PMSAV ; INCREMENT RETURN
00002'010461		ISZ PMSAV ; ADDRESS BY 2
00003'030020		LDA 2,20 ; SAVE (20)
00004'050460		STA 2,PMSAV+1
00005'031401		LDA 2,1,3 ; SIZE OF ARRAY
00006'050447		STA 2,PCTR
00007'014446		DSZ PCTR ; DECREMENT FOR RANGE TESTS
00010'050446		STA 2,ZCTR
00011'102460		SUBC 0,0
00012'040445		STA 0,BITC ; ZERO BIT COUNTER
00013'151123		MOVZL 2,2,SNC ; CHECK NEXT BIT
00014'000402		JMP .+2 ; IF = 0
00015'000403		JMP .+3 ; IF = 1
00016'010441		ISZ BITC ; INCREMENT BIT COUNT
00017'000774		JMP .-4 ; ITERATE
00020'031400		LDA 2,0,3 ; ADDRESS OF ARRAY
00021'050020		STA 2,20 ; SAVED IN LOC 20
00022'050437		STA 2,ASAV
00023'014020		DSZ 20 ; BACK UP ONE
00024'020433	AGAIN:	LDA 0,BITC ; GET BIT COUNT
00025'040435		STA 0,LAX ; STORE IN LAX
00026'006001\$		JSR @APAND ; GET A RANDOM NUMBER
00027'000001-		OLD
00030'101220		MOVZP 0,0 ; SHIFT RANDOM NUMBER
00031'014431		DSZ LAX ; DECREMENT BIT COUNT
00032'000776		JMP .-2 ; LOOP
00033'024422		LDA 1,PCTR ; WHEN DONE SHIFTING,
00034'122033		ADCZ# 1,0,SNC ; CHECK IF GREATER
00035'000402		JMP .+2 ; THAN SIZE OF ARRAY
00036'000766		JMP AGAIN ; GET NEW RANDOM NUMBER
00037'026020		LDA 1,020 ; GET ADDRESS OF NEXT
		; ARRAY ELEMENT

00040'044420	STA 1,TEMP	;SET ASIDE
00041'014020	DSZ 20	;BACK UP ADDRESS COUNT
00042'030417	LDA 2,ASAV	
00043'113000	ADD 0,2	;CREATE ADDRESS FOR
		;SWAP
00044'025000	LDA 1,0,2	;SWAP OLD (20)
00045'046020	STA 1,020	;AND (START + X)
00046'024412	LDA 1,TEMP	
00047'045000	STA 1,0,2	
00050'014406	DSZ ZCTR	;DECREMENT RANGE COUNT
00051'000753	JMP AGAIN	;LOOP
00052'030412	LDA 2,PMSAV+1	;RESTORE (20)
00053'050020	STA 2,20	
00054'002407	JMP @PMSAV	
00055'000000	PCTP: 0	
00056'000000	ZCTP: 0	
00057'000000	BITC: 0	
00060'000000	TEMP: 0	
00061'000000	ASAV: 0	
00062'000000	LAX: 0	
000002	PMSAV: .BLK 2	
	.END	

```

; INOUT -- SUBROUTINE PACKAGE CONTAINING
; PUTC, GETC, ENOCT, OCTEN, ENDEC, AND
; DECB IN PFLOCATABLE FORM,
; WITH ENTRY POINTS APUTC, AGETC, ABNOC,
; AOCEN, ABNDC, AND ADCEN. FOLLOWING
; ASSUMPTIONS ARE MADE ON ENTRY:
; PUTC: CHARACTER TO TIO
; IN AC0
; GETC: CHARACTER FROM TTI
; IN AC0
; ENOCT: BINARY NUMBER INPUT
; TO ENOCT IS IN AC1
; OCTEN: BINARY NUMBER OUTPUT
; FROM OCTEN WILL BE
; IN AC1
; ENDEC: BINARY NUMBER INPUT
; TO ENDEC IS IN AC1
; DECB: BINARY NUMBER OUTPUT
; FROM DECB WILL BE
; IN AC1
; ALL SUBROUTINES REQUIRE AC0, AC1, AND AC3.
; IN ADDITION, ENOCT AND ENDEC ALSO
; REQUIRE AC2.
; SPECIAL NOTES: IF GETC IS TO BE
; ENTERED BECAUSE OF A TTI
; INTERRUPT, THE ENTRY POINT MUST BE
; IGETI, AND IT WILL BE ASSUMED
; THAT NIOS TTI OR THE EQUIVALENT
; WILL BE PART OF THE INTERRUPT SERVICE
; ROUTINE.
; *****ALL THESE ROUTINES EXCEPT DECB
; ARE COPIES OR ADAPTATIONS OF THOSE
; DESCRIBED IN THE DATA GENERAL MANUAL,
; "INTRODUCTION TO PROGRAMMING THE NOVA
; COMPUTERS." PROGRAM LISTINGS AND
; COMMENTS ARE IN APPENDIX A OF THAT
; MANUAL. DECB IS COPIED FROM THE
; MATH ROUTINE LIBRARY, AND IS
; DESCRIBED IN THE APPROPRIATE
; DOCUMENTATION. COPYRIGHT 1969,
; 1970, 1971 DATA GENERAL CORPORATION
; REPRODUCED BY PERMISSION.
; THE SPECIAL FEATURES OF DECB
; ARE PRESERVED IN THIS VERSION.
; CONSULT THE MATH LIBRARY DOCUMENTS
; FOR DETAILS.

```

```

; G.M. OLSON
; 14 DECEMBER 1972 (A)

```

```

.TITL INOUT
.ENT APUTC, AGETC, ABNOC, AOCEN
.ENT ABNDC, ADCEN, IGETI, IDBIN
.NREL

```

```

00000'054416 ENOCT: STA 3, SAVE
00001'152620 SUBZR 2, 2
00002'020413 LOOP: LDA 0, C60

```

00003*146443	SUBO	2, 1, SNC
00004*101401	INC	0, 0, SKP
00005*147001	ADD	2, 1, SKP
00006*000775	JMP	.- 3
00007*004437	JSR	PUTC
00010*151220	MOVZR	2, 2
00011*151220	MOVZR	2, 2
00012*151224	MOVZR	2, 2, SZR
00013*000767	JMP	LOOP
00014*002402	JMP	@SAVE
00015*000060	C60:	60
000007	SAVE:	.BLK 7
00025*054517	GETI:	STA 3, SGET
00026*000403		JMP .+3
00027*054515	GETC:	STA 3, SGET
00030*060110		NIOS TTI
00031*063610		SKPDN TTI
00032*000777		JMP .- 1
00033*060610		DIAC 0, TTI
00034*034513		LDA 3, MSK
00035*163400		AND 3, 0
00036*004410		JSR PUTC
00037*034507		LDA 3, CR
00040*116404		SUB 0, 3, SZR
00041*002503		JMP @SGET
00042*020506		LDA 0, LF
00043*004403		JSR PUTC
00044*020502		LDA 0, CR
00045*002477		JMP @SGET
00046*063511	PUTC:	SKPBZ TTO
00047*000777		JMP .- 1
00050*061111		DOAS 0, TTO
00051*101004		MOV 0, 0, SZR
00052*001400		JMP 0, 3
00053*054472		STA 3, SPUT
00054*020472		LDA 0, CR
00055*004771		JSR PUTC
00056*020472		LDA 0, LF
00057*004767		JSR PUTC
00060*102400		SUB 0, 0
00061*002464		JMP @SPUT
00062*054734	ENDEC:	STA 3, SAVE
00063*034460		LDA 3, ZERO
00064*054454		STA 3, SUPP
00065*034454		LDA 3, FIVE
00066*054451		STA 3, LAST
00067*034446		LDA 3, INST
00070*054401		STA 3, .+1
00071*000000	KOOP:	0
00072*000723		LDA 0, C60
00073*146443		SUBO 2, 1, SNC
00074*101401		INC 0, 0, SKP
00075*147001		ADD 2, 1, SKP
00076*000775		JMP .- 3
00077*050720		STA 2, SAVE+1
00100*014437		DSZ LAST

00101'000402		JMP	+.2
00102'000420		JMP	SOAR
00103'034435		LDA	3, SUPP
00104'030437		LDA	2, ZEP0
00105'156414		SUB#	2, 3, SZR
00106'000414		JMP	SOAR
00107'030433		LDA	2, ONE
00110'034705		LDA	3, C60
00111'116414		SUB#	0, 3, SZR
00112'000407		JMP	SOAR-1
00113'040705		STA	0, SAVE+2
00114'020422		LDA	0, BLANK
00115'004731		JSR	PUTC
00116'020702		LDA	0, SAVE+2
00117'030700		LDA	2, SAVE+1
00120'000404		JMP	SOAR+2
00121'050417		STA	2, SUPP
00122'004724	SOAR:	JSR	PUTC
00123'030674		LDA	2, SAVE+1
00124'010745		ISZ	KOOP
00125'151203		MOVR	2, 2, SNC
00126'000743		JMP	KOOP
00127'002667		JMP	@SAVE
000012		.RDX	10
00130'023420	TENS:		10000
00131'001750			1000
00132'000144			100
00133'000012			10
00134'000001			1
000010		.RDX	8
00135'030437	INST:	LDA	2, .+TENS-KOOP
00136'000040	BLANK:		40
00137'000000	LAST:		0
00140'000000	SUPP:		0
00141'000005	FIVE:		5
00142'000001	ONE:		1
00143'000000	ZERO:		0
00144'000000	SGET:		0
00145'000000	SPUT:		0
00146'000015	CR:		15
00147'000177	MSK:		177
00150'000012	LF:		12
00151'054645	OCTEN:	STA	3, SAVE
00152'126400		SUB	1, 1
00153'004654	OCT1:	JSR	GETC
00154'034772		LDA	3, CR
00155'116415		SUB#	0, 3, SNR
00156'002640		JMP	@SAVE
00157'034406		LDA	3, C7
00160'163400		AND	3, 0
00161'127120		ADDZL	1, 1
00162'125120		MOVZL	1, 1
00163'107000		ADD	0, 1
00164'000767		JMP	OCT1
00165'000007	C7:		7

```

00166'054455 .DENI: STA 3,.EC03 ; SAVE RETURN
00167'050453 STA 2,.EC02 ; SAVE AC2
00170'020462 LDA 0,.EC24 ; GET "S"
00171'006002- JSP @APUTC ; SEND "S"
00172'102400 SUB 0,0
00173'006002- JSP @APUTC ; SEND NULL
00174'000403 JMP .+3
00175'054446 DECEN: STA 3,.EC03 ; SAVE AC3
00176'050444 STA 2,.EC02 ; SAVE AC2
00177'102400 SUB 0,0
00200'040444 STA 0,.EC10 ; CLEAR SIGN WORD
00201'040444 STA 0,.EC11 ; CLEAR SUM WORD
00202'006003- JSP @AGETC ; GET A CHARACTER
00203'024443 LDA 1,.EC20 ; TEST FOR "+"
00204'106405 SUB 0,1,SNR
00205'000405 JMP .EC97 ; YES
00206'024441 LDA 1,.EC21 ; NO, TEST FOR "-"
00207'106404 SUB 0,1,SZP
00210'000403 JMP .EC96 ; NO EXPLICIT SIGN
00211'010433 ISZ .EC10 ; SET FLAG WORD FOR NEGATIVE
; NUMBER
00212'006003- .EC97: JSP @AGETC ; GET ANOTHER CHARACTER
00213'024435 .EC96: LDA 1,.EC22 ; ASCII "0"
00214'030435 LDA 2,.EC23 ; ASCII "9"
00215'142033 ADCZ# 2,0,SNR ; SKIP IF > 9
00216'106032 ADCZ# 0,1,SZC ; SKIP IF >= 0
00217'000406 JMP .EC95 ; NOT A DIGIT, THEREFORE A BREAK
; CHARACTER
00220'122400 SUB 1,0 ; REDUCE DIGIT TO 0-9 BINARY
; RANGE
00221'024424 LDA 1,.EC11 ; SUM WORD
00222'004412 JSR .EC50 ; MULTIPLY BY 10 AND ADD
00223'044422 STA 1,.EC11 ; SAVE SUM
00224'000766 JMP .EC97 ; GET NEXT CHARACTER

00225'024420 .EC95: LDA 1,.EC11 ; RESULT TO AC1
00226'125120 MOVZL 1,1
00227'014415 DSZ .EC10 ; TEST SIGN
00230'125221 MOVZR 1,1,SKP ; POSITIVE
00231'124640 NEGOR 1,1 ; NEGATIVE
00232'030410 LDA 2,.EC02 ; RESTORE AC2
00233'002410 JMP @.EC03

; ROUTINE TO MULTIPLY AC1 BY 10 AND ADD AC0

00234'131120 .EC50: MOVZL 1,2 ; N*2
00235'151120 MOVZL 2,2 ; N*4
00236'147000 ADD 2,1 ; N*5
00237'125120 MOVZL 1,1 ; N*5*2 = N*10
00240'107000 ADD 0,1 ; ADD AC0
00241'001400 JMP 0,3 ; SUCCESS RETURN

00242'000000 .EC02: 0 ; SAVE AC2
00243'000000 .EC03: 0 ; SAVE AC3

```

00244'000000	.EC10:	0	; FLAG WORD FOR SIGN OF RESULT
00245'000000	.EC11:	0	; RUNNING SUM WORD
00246'000053	.EC20:	"+"	; ASCII "+"
00247'000055	.EC21:	"-"	; ASCII "-"
00250'000060	.EC22:	"0"	; ASCII "0"
00251'000071	.EC23:	"9"	; ASCII "9"
00252'000123	.EC24:	"S"	; ASCII "S" FOR INDICATION
			; ENTRY
		.ZPEL	
00000-000062'	ABNDC:	ENDEC	
00001-000175'	ADCEN:	DECEN	
00002-000046'	APUTC:	PUTC	
00003-000027'	AGETC:	GETC	
00004-000000'	ABNOC:	ENOCT	
00005-000151'	AOCEN:	OCTEN	
00006-000025'	IGETI:	GETI	
00007-000166'	IDBIN:	.DENI	
		.END	

```

; BYTE POINTER SUBPT, STEPS THRU MESSAGE
; TYPING UNTIL ENCOUNTERS NULL CHARACTER.
;
; WRITE TO WRITE+7 TAKEN FROM
; "HOW TO USE THE NOVA COMPUTERS"
; COPYRIGHT 1971, DATA GENERAL CORPORATION
; REPRODUCED BY PERMISSION
;
; CALL AS FOLLOWS: (MESS=MESSAGE ADDRESS)
;     LDA     0, MESS
;     MOVZL   0, 0
;     STA     0, .+2
;     JSR     @AOUTP
;     0
;     RETURN HERE
;
; EMAN + PUTC USE ALL AC. SAVE AC1+AC2
; ON ENTRY, RESTORE ON RETURN
;
;     G.O. MOELLEN
;     AUGUST 19 1972 (A)

. TITL EMAN
. ENT AOUTP
. EXTD APUTC
. NREL

;
00000'054424  OUTP:  STA     3, SVOUT
00001'054424          STA     3, SVOUT+1
00002'050424          STA     2, SVOUT+2
00003'044424          STA     1, SVOUT+3
00004'010420          ISZ     SVOUT
; STEP THRU MESSAGE
00005'032420  WRITE:  LDA@    2, SVOUT+1          ; CF "HOW TO...", 2-21
00006'012417          ISZ@    SVOUT+1
00007'151220          MOVZP   2, 2
00010'021000          LDA     0, 0, 2
00011'030412          LDA     2, C377
00012'101002          MOV     0, 0, SZC
00013'101300          MOVS    0, 0
00014'143405          AND     2, 0, SNR
00015'000403          JMP     .+3
00016'006001$        JSR@     APUTC
00017'000766          JMP     WRITE
00020'030406          LDA     2, SVOUT+2
00021'024406          LDA     1, SVOUT+3
00022'002402          JMP@    SVOUT
;
00023'000377  C377:   377
000004  SVOUT:  .ELK 4
.2REL
00000-000000' AOUTP:  OUTP
;
. END

```

```

; SUBROUTINE TO CLEAR N LOCATIONS,
; ANY AREA BEYOND END OF ZMEM.
; ZMEM REQUIRES 30 WORDS.
;
; CALL WITH JSR@ ACLER.
; IF JSR LOCATED AT PC
; PC+1 = NO. LOCS TO CLEAR
; PC+2 = ADDRESS OF AREA.
;
; ZMEM USES LOC 20, AC2, AC3
; SAVES PREVIOUS (20) + (AC2)
; AS ANY JSR, SAVE (AC3) BEFORE
; CALL - IF NECESSARY.
;
;
; G.O. MOELLER
; 11 AUGUST 1972 (A)
;

```

```

.TITL NZMEM
.ENT ACLER
.NREL

```

```

;
00000'054425 CLEAR: STA 3, CTS ; SET RETURN ADDRESS,
00001'010424 ISZ CTS ; SAVE (AC2), (AC3) AND
00002'010423 ISZ CTS ; (20).
00003'050423 STA 2, CTS+1
00004'030020 LDA 2, 20
00005'050422 STA 2, CTS+2
00006'031400 LDA 2, 0, 3 ; SET CTR, #LOCS TO ZERO
00007'050414 STA 2, ZCTR
00010'031401 LDA 2, 1, 3 ; START ADDRESS, AREA TO BE ZEROED
00011'050020 STA 2, 20 ; TO LOC 20.
00012'014020 DSZ 20
00013'030411 LDA 2, ZERCL
00014'052020 STA@ 2, 20
00015'014406 DSZ ZCTR
00016'000776 JMP .-2
00017'034410 LDA 3, CTS+2
00020'054020 STA 3, 20
00021'030405 LDA 2, CTS+1
00022'002403 JMP@ CTS
00023'000000 ZCTR : 0
00024'000000 ZERCL : 0
000003 CTS : .BLK 3
.ZREL
00000-000000' ACLER : CLEAR
;
.END

```

```

; POUTINE TO PRINT MESSAGES
; ON CRT -- STOPS WHEN
; ENCOUNTERS NULL CHARACTER
;
; DELTA TO DELTA+7 BASED ON EXAMPLE
; IN "HOW TO USE THE NOVA COMPUTERS"
; COPYRIGHT 1971, DATA GENERAL CORPORATION
; REPRODUCED BY PERMISSION
;
; G.M. OLSON
; 2 MARCH 1973 (A)

```

```

.TITL CRTXT
.ENT SHOW
.EXTD CHAP

```

```

.ZPEL
00000-000000' SHOW: GAMMA

```

```

.NPEL
00000'054426 GAMMA: STA 3,OUT ;SAVE (AC3) FOR RETURN
00001'054426 STA 3,OUT+1 ;SAVE (AC3) FOR CHAP. APPAY
00002'050426 STA 2,OUT+2 ;SAVE (AC2)
00003'044426 STA 1,OUT+3 ;SAVE (AC1)
00004'010422 ISZ OUT ;BUMP RETURN
00005'032422 DELTA: LDA 2,OUT+1 ;LOAD CHAP. WORD
00006'012421 ISZ OUT+1 ;INCREMENT
00007'151220 MOVZP 2,2 ;ROTATE TO RIGHT
00010'021000 LDA 0,0,2 ;LOAD CHAR
00011'030421 LDA 2,C377 ;LOAD MASK
00012'101002 MOV 0,0,SZC ;CHECK CARRY
00013'101300 MOVS 0,0 ;SWAP BYTES
00014'143405 AND 2,0,SNR ;CHECK IF NULL
00015'000405 JMP MU ;EXIT
00016'040407 STA 0,TEMP
00017'006001$ JSP @CHAP ;CHARACTER TO CRT
00020'000025' TEMP
00021'000764 JMP DELTA ;LOOP
00022'030406 MU: LDA 2,OUT+2 ;PESTORE (AC2) +
00023'024406 LDA 1,OUT+3 ; (AC1)
00024'002402 JMP OUT
00025'000000 TEMP: 0
000004 OUT: .BLK 4
00032'000377 C377: 377
.END

```

```

;SUBROUTINE BOX -- DRAWS RIGHT-LEFT DIAGRAMS
;ON CRT (BASED ON OLSON AND LAXAP, JOURNAL
;OF EXPERIMENTAL PSYCHOLOGY, 1973)
;CALLING SEQUENCE: JSP @PLBOX
; STIMULUS CODE
; RETURN
;STIMULUS CODE: 1 = "RIGHT"-RIGHT
; 2 = "RIGHT" - LEFT
; 3 = "LEFT" - RIGHT
; 4 = "LEFT" - LEFT
;
; G.M. OLSON
; 26 MARCH 1973 (A)

```

```

.TITL BOX
.ENT PLBOX
.EXTD SKTCH

```

```

.ZPEL
00000-000026' PLEBOX: START

```

```

.NPEL
00000'000000 TEMP: 0
00001'000000 MODE: 0
00002'000000 ZEP0: 0
00003'000001 ONE: 1
00004'000005 FIVE: 5
00005'000006 SIX: 6
00006'000000 COUNT: 0
00007'000000 TEMPX: 0
00010'000000 TEMPY: 0
00011'000000 TEMPT: 0
00012'000000 MODOC: 0
00013'000137' FXLOC: XR
00014'000145' FYLOC: YR
00015'000153' RTLOC: TOTP
00016'000161' LXLOC: XL
00017'000166' LYLOC: YL
00020'000173' LTLOC: TOTL
00021'000202' LDIAX: XLD
00022'000200' PDIAX: XPD
00023'000201' DIAY: YD
00024'000203' DIAT: TOTD
00025'000000 OBAN: 0

```

```

00026'054764 START: STA 3,MODOC ;SAVE (AC3)+1 FOR
00027'010763 ISZ MODOC ;RETURN
00030'025400 LDA 1,0,3 ;LOAD CODE NO.
00031'020752 LDA 0,ONE ;SUBTRACT 1 FROM CODE
00032'106400 SUB 0,1
00033'125220 MOVZR 1,1 ;SAVE CODE BIT FOR
00034'152460 SUBC 2,2 ;DIAMOND
00035'151100 MOVL 2,2 ;
00036'050767 STA 2,OBAN
00037'125222 MOVZR 1,1,SZC ;CHECK IF WORD IS:
00040'000413 JMP LEFT ;"LEFT"
00041'030752 LDA 2,RXLOC ;"RIGHT" -- LOAD PARAMETERS
00042'050745 STA 2,TEMPX
00043'030751 LDA 2,RYLOC

```

00044'050744		STA 2,TEMPY	
00045'030750		LDA 2,RTLOC	
00046'050743		STA 2,TEMPT	
00047'030736		LDA 2,SIX	
00050'050736		STA 2,COUNT	
00051'004427		JSR DRAW	;DRAW "RIGHT"
00052'000412		JMP YUOK	;GO TO DRAW DIAMOND
00053'030743	LEFT:	LDA 2,LXLOC	;LOAD "LEFT" PARAMETERS
00054'050733		STA 2,TEMPX	
00055'030742		LDA 2,LYLOC	
00056'050732		STA 2,TEMPY	
00057'030741		LDA 2,LTLOC	
00060'050731		STA 2,TEMPT	
00061'030723		LDA 2,FIVE	
00062'050724		STA 2,COUNT	
00063'004415		JSR DRAW	;DRAW "LEFT"
00064'030735	YUOK:	LDA 2,LDIAX	;DRAW APPROPRIATE
00065'014740		DSZ OBAN	;DIAMOND
00066'030734		LDA 2,RDIAX	
00067'050720		STA 2,TEMPX	
00070'030733		LDA 2,DIAY	
00071'050717		STA 2,TEMPY	
00072'030732		LDA 2,DIAT	
00073'050716		STA 2,TEMPT	
00074'030707		LDA 2,ONE	
00075'050711		STA 2,COUNT	
00076'004402		JSR DRAW	
00077'002713		JMP @MODOC	;EXIT FROM SUBROUTINE
00100'054436	DRAW:	STA 3,TEJON	;DRAW ON CRT
00101'020701		LDA 0,ZERO	;SET FOR DARK VECTOR
00102'040677		STA 0,MODE	
00103'030704		LDA 2,TEMPX	;SET X
00104'021000		LDA 0,0,2	
00105'040414		STA 0,XVAR	
00106'030702		LDA 2,TEMPY	;SET Y
00107'021000		LDA 0,0,2	
00110'040412		STA 0,YVAR	
00111'030700		LDA 2,TEMPT	;SET COUNT OF X,Y PAIRS
00112'021000		LDA 0,0,2	
00113'040665		STA 0,TEMP	
00114'000403		JMP .+3	
00115'020666	PLOT:	LDA 0,ONE	;SET FOR BRIGHT VECTOR
00116'040663		STA 0,MODE	
00117'006001\$		JSR @SKTCH	;JUMP TO ALS
00120'000001'		MODE	
00121'000000	XVAR:	0	
00122'000000	YVAR:	0	
00123'010776		ISZ XVAR	;INCREMENT POINTERS
00124'010776		ISZ YVAR	
00125'014653		DSZ TEMP	;CHECK IF FINISHED
			;WITH BRIGHT VECTOR
00126'000767		JMP PLOT	;CONTINUE WITH BRIGHT VECTOR
00127'014657		DSZ COUNT	;CHECK IF DRAWING DONE
00130'000402		JMP .+2	
00131'002405		JMP @TEJON	;EXIT
00132'010655		ISZ TEMPX	;SET FOR NEXT DARK
00133'010655		ISZ TEMPY	;VECTOR
00134'010655		ISZ TEMPT	
00135'000744		JMP DRAW+1	
00136'000000	TEJON:	0	

00137'000204'	XR:	BX
00140'000216'		RX
00141'000240'		IX
00142'000254'		GX
00143'000300'		HX
00144'000314'		TPX
00145'000211'	YR:	BY
00146'000227'		RY
00147'000246'		ITY
00150'000266'		GY
00151'000306'		HY
00152'000246'		ITY
00153'000005	TOTR:	5
00154'000011		11
00155'000006		6
00156'000012		12
00157'000006		6
00160'000004		4
00161'000204'	XL:	EX
00162'000324'		LX
00163'000332'		EX
00164'000350'		FX
00165'000320'		TLX
00166'000211'	YL:	BY
00167'000327'		LY
00170'000341'		EFY
00171'000341'		EFY
00172'000246'		ITY
00173'000005	TOTL:	5
00174'000003		3
00175'000007		7
00176'000006		6
00177'000004		4
00200'000356'	XPD:	DRX
00201'000370'	YD:	DY
00202'000363'	XLD:	DLX
00203'000005	TOTD:	5

000012		.RDX 10
00204'000620	BX:	400
00205'001274		700
00206'001274		700
00207'000620		400
00210'000620		400
00211'001130	BY:	600
00212'001130		600
00213'000454		300
00214'000454		300
00215'001130		600
00216'000654	RX:	428
00217'000654		428
00220'000714		460
00221'000724		468
00222'000724		468
00223'000714		460
00224'000654		428
00225'000710		456
00226'000724		468
00227'000644	RY:	420
00230'000740		480

00231'000740		480
00232'000730		472
00233'000704		452
00234'000674		444
00235'000674		444
00236'000674		444
00237'000644		420
00240'000750	IX:	488
00241'000760		496
00242'000754		492
00243'000754		492
00244'000750		488
00245'000760		496
00246'000740	ITY:	480
00247'000740		480
00250'000740		480
00251'000644		420
00252'000644		420
00253'000644		420
00254'001034	GX:	540
00255'001054		556
00256'001054		556
00257'001044		548
00260'001014		524
00261'001004		516
00262'001004		516
00263'001014		524
00264'001044		548
00265'001054		556
00266'000674	GY:	444
00267'000674		444
00270'000654		428
00271'000644		420
00272'000644		420
00273'000654		428
00274'000730		472
00275'000740		480
00276'000740		480
00277'000730		472
00300'001140	HX:	608
00301'001140		608
00302'001140		608
00303'001074		572
00304'001074		572
00305'001074		572
00306'000644	HY:	420
00307'000740		480
00310'000702		450
00311'000702		450
00312'000644		420
00313'000740		480
00314'001160	TRX:	624
00315'001240		672
00316'001210		648
00317'001210		648
00320'001150	TLX:	616
00321'001230		664
00322'001200		640
00323'001200		640
00324'000670	LX:	440

00325'000670		440
00326'000740		480
00327'000740	LY:	480
00330'000644		420
00331'000644		420
00332'001034	EX:	540
00333'000764		500
00334'000764		500
00335'001010		520
00336'000764		500
00337'000764		500
00340'001034		540
00341'000740	EFY:	480
00342'000740		480
00343'000702		450
00344'000702		450
00345'000702		450
00346'000644		420
00347'000644		420
00350'001130	FX:	600
00351'001060		560
00352'001060		560
00353'001104		580
00354'001060		560
00355'001060		560
00356'001510	DRX:	840
00357'001440		800
00360'001370		760
00361'001440		800
00362'001510		840
00363'000404	DLX:	260
00364'000454		300
00365'000524		340
00366'000454		300
00367'000404		260
00370'000702	DY:	450
00371'000752		490
00372'000702		450
00373'000632		410
00374'000702		450

.END

```

;               SUBROUTINE SALS
;   ABBREVIATED VERSION OF TEKTRONIX PLOT-10
;   FOR NOVA MINICOMPUTERS
;   "*" = NAVSUBMEDRSCHLAB MODIFICATION
;
;   G.M. OLSON
;   6 APRIL 1973 (A)
;
;               .TITL    SALS      ;*
;               .ENT CHAR,SKTCH ;*
000050          CRI=50           ;*
000051          CRO=51           ;*
;
;               .ZREL      ;*
000000-000000' CHAR: CHOUT      ;*
000001-000010' SKTCH: TPLOT     ;*
;
;               .NREL
;
;   RELEASE 2   72-MAY-11
;   COPYRIGHTED BY TEKTRONIX, INC.  1972
;   REPRODUCED BY PERMISSION
;
;
;   CHOUT
;   THIS ROUTINE IS CALLED TO OUTPUT
;   AN ASCII CHARACTER TO THE 4010
;   GRAPHIC COMPUTER TERMINAL
;
;   TO CALL:
;*   JSR    @CHAR
;   ADDRESS OF WORD WITH CHARACTER
;   RETURN HERE
;
000000'040407 CHOUT: STA    0,CCAC0 ;SAVE AC0
000001'063551      SKPBZ   CRO      ;SKIP IF NOT BUSY
000002'000777      JMP     .-1
000003'023400      LDA     0,00,3  ;GET CHARACTER
000004'061151      DOAS    0,CRO   ;SHIP CHARACTER
000005'020402      LDA     0,CCAC0 ;RESTORE AC0
000006'001401      JMP     1,3
;
000007'000000 CCAC0: 0           ;TEMP FOR AC0
;
;
;   TPLOT
;   THIS ROUTINE IS CALLED TO PLOT
;   IN LINEAR INTERPOLATE (VECTOR),
;   POINT PLOT MODE
;   DEPENDING ON THE VALUE OF THE MODE
;   PARAMETER
;
;   IF MODE
;   = 0    INITIALZE AND DARK VECTOR TO X,Y
;   > 0    BRIGHT VECTOR TO X,Y
;   < 0    POINT PLOT TO X,Y

```

```

;
;
;
; TO CALL:
;* JSR @SKTCH
; ADDRESS OF MODE PARAMETER
; ADDRESS OF X
; ADDRESS OF Y
; RETURN HERE
;

00010'040516 TPL0T: STA 0,TPTAC0;SAVE AC0
00011'023401 LDA 0,01,3 ;GET X
00012'040516 STA 0,TPTX
00013'023402 LDA 0,02,3 ;GET Y
00014'040515 STA 0,TPTY
00015'023400 LDA 0,00,3 ;GET MODE
00016'040514 STA 0,TPMOD
00017'054510 STA 3,TPTADD;SAVE CALL ADDRESS
00020'101015 MOV# 0,0,SNR ;SKP IF NEO 0
00021'000405 JMP TPTDV ;= 0 INITIALIZE AND DARK VECTOR
00022'101113 MOVL# 0,0,SNC ;SKIP IF < 0
00023'000405 JMP TPTNFM ;NORMAL BRIGHT VECTOR
00024'006000- JSR @CHAR ;*SET TO ALPHA
00025'000120' US
00026'006000- TPTDV: JSR @CHAR ;*DARK VECTOR
00027'000117' GS
;

00030'020501 TPTNFM: LDA 0,TPTY ;GET Y
00031'101112 MOVL# 0,0,SZC ;SKP IF +
00032'102400 SUB 0,0 ;MAKE 0
00033'034471 LDA 3,D780 ;UPPER Y BOUND
00034'162513 SUBL# 3,0,SNC ;SKP IF ON SCREEN
00035'161000 MOV 3,0 ;SET TO EDGE
00036'040473 STA 0,TPTY ;SAVE GOOD Y
00037'103120 ADDZL 0,0 ;*USE UPPER 5 BITS
00040'101120 MOVZL 0,0
00041'101300 MOVS 0,0 ;AND SWAP HALVES
00042'034457 LDA 3,B040 ;HI Y TAG
00043'163000 ADD 3,0 ;PUT IN CHAR
00044'040467 STA 0,TPTTMP;USE A TEMP
00045'006000- JSP @CHAR ;*SHIP HI Y 5
00046'000133' TPTTMP
00047'020462 LDA 0,TPTY ;GET Y
00050'034450 LDA 3,B037 ;MASK
00051'163400 AND 3,0 ;LEAVE LOW Y 5
00052'034451 LDA 3,B140 ;LOW Y TAG
00053'163000 ADD 3,0 ;SET IN CHAR
00054'040457 STA 0,TPTTMP
00055'006000- JSP @CHAR ;*SHIP LOW Y
00056'000133' TPTTMP
;

00057'020451 LDA 0,TPTX ;GET X VALUE
00060'101112 MOVL# 0,0,SZC ;*
00061'102400 SUB 0,0 ;*
00062'034443 LDA 3,D1023 ;*
00063'162513 SUBL# 3,0,SNC ;*
00064'161000 MOV 3,0 ;*
00065'040443 STA 0,TPTX ;*
00066'103120 ADDZL 0,0 ;*AND SO ON
00067'101120 MOVZL 0,0
00070'101300 MOVS 0,0 ;HI X 5

```

```

00071'034430      LDA      3,B040 ;HI X TAG
00072'163000      ADD      3,0    ;ADD IN TAG
00073'040440      STA      0,TPTTMP
00074'006000-     JSR      @CHAR  ;*SHIP HI X 5
00075'000133'     TPTTMP
00076'020432      LDA      0,TPTX  ;GET X
00077'034421      LDA      3,B037 ;GOODIE MASK
00100'163400      AND      3,0    ;LEAVE LOW X 5
00101'034421      LDA      3,B100 ;LOW X TAG
00102'163000      ADD      3,0    ;PUT IN TAG
00103'040430      STA      0,TPTTMP
00104'006000-     JSR      @CHAR  ;*
00105'000133'     TPTTMP
00106'020424      LDA      0,TFMD
00107'101113      MOVL#    0,0,SNC
00110'000404      JMP      TPTXT
00111'102400      SUB      0,0
00112'040420      STA      0,TFMD
00113'000715      JMP      TPTNPM
00114'020412      TPTXT: LDA      0,TPTAC0;RESTORE AC0
00115'034412      LDA      3,TPTADD;CALL ADDRESS
00116'001403      JMP      3,3    ;EXIT

```

```

;
;
;
;      CONSTANTS AND VARIABLES
;

```

```

00117'000035      GS:      035
00120'000037      US:      037

```

```

000120' B037=US
00121'000040      B040:    040
00122'000100      B100:    100
00123'000140      B140:    140

```

```

;
00124'001414      D780:    1414      ;MAX VISIBLE Y
00125'001777      D1023:   1777      ;MAX VISIBLE X
;

```

```

00126'000000      TPTAC0:  0
00127'000000      TPTADD:  0
00130'000000      TPTX:    0
00131'000000      TPTY:    0
00132'000000      TFMD:    0
00133'000000      TPTTMP:  0

```

```

.END

```

```

; SUBROUTINE PTANL -- DOES SIMPLE ANALYSIS
; OF PT DATA:
;   STANDARD ANALYSIS INCLUDES--
;   1. OVERALL RESULTS
;   2. ANALYSIS BY STIMULI (IF MORE THAN ONE)
;   3. ANALYSIS BY FOREPERIODS (IF MORE THAN ONE)
;   4. ANALYSIS BY STIMULI X FOREPERIODS (IF BOTH
;      MORE THAN ONE)
;   ALL OF THESE ARE DONE FOR
;   A. ALL DATA
;   B. CORRECTS ALONE   ) ONLY IF ERRORS
;   C. ERRORS ALONE     ) ARE POSSIBLE

;   OPTIONAL ANALYSIS --
;   ALL OF THE ABOVE DONE BY BLOCKS,
;   WHERE BLOCK SIZE IS ENTERED AT
;   TIME OF ANALYSIS
;   ***NOTE: IT IS UP TO THE USER
;   TO CREATE MEANINGFUL BLOCK SIZES

; PTANL REQUIRES SUBROUTINES DESIGN, DPACC, CARVE,
; INOUT, AND THEIR ASSOCIATED SUBROUTINES.
; ALSO REQUIRES ACCESS TO MANY GLOBAL
; NAMES IN PTEXP

```

```

; G.M. OLSON
; 19 MARCH 1973 (A)

```

```

.TITL PTANL
.ENT PSYCH
.EXTD EXP,SUBJ,ATYPE,AFE,TRLS
.EXTD REFC,APP,NFP,CT,ACS,ETXT
.EXTD ABNDG,QUACC,CLEVE,ITYP,OVINC
.EXTD ATIME,ACODE,ISTIM,IFOPE

```

```

.ZPEL
00000-000000' PSYCH: WHY      ; ENTRY POINT
00001-000514' PTXT:  TXT1     ; ZERO PAGE TEXT ADDRESSES
00002-000542'      TXT2
00003-000553'      TXT3
00004-000600'      TXT4
00005-000612' ITXT:  TXT5
00006-000635'      TXT6
00007-000707'      TXT7
00010-000765'      TXT8
00011-001045'      TXT9
00012-001135'      TXT10
00013-001137'      TXT11
00014-001141'      TXT12
00015-001142'      TXT13
00016-000475' ANIND: NIND
00017-000474' ASIZE: SIZE
00020-000000' ARET:  0
00021-000407' AITAB: ITAB
00022-000440' ASOPT: SOPT

```

00023-000473' ASTAPT: START

```

.NPEL
00000'054520 WHY: STA 3,WORK ;STORE RETURN
00001'020001- LDA 0,PTXT ;PRINT HEADINGS
00002'006003$ JSP @ATYPE
00003'024001$ LDA 1,EXP
00004'006014$ JSP @ABNDC
00005'020002- LDA 0,PTXT+1
00006'006003$ JSP @ATYPE
00007'024002$ LDA 1,SUBJ
00010'006014$ JSR @ABNDC
00011'020006$ LDA 0,REPC
00012'040511 STA 0,WORK+3
00013'014510 DSZ WORK+3
00014'000405 JMP .+5
00015'020015- LDA 0,ITXT+10
00016'006003$ JSR @ATYPE
00017'024020$ LDA 1,OVINC
00020'006014$ JSP @ABNDC
00021'020003- LDA 0,PTXT+2 ;QUERY FOR ANALYSIS
00022'006004$ JSR @APE ;BY BLOCKS
00023'044516 STA 1,ELKF
00024'020531 LDA 0,ONE
00025'106414 SUB #0,1,SZF
00026'000404 JMP EASY ;NO BLOCK ANALYSIS
00027'020004- LDA 0,PTXT+3 ;
00030'006004$ JSR @APE ;GET BLOCKSIZE FOR
00031'044511 STA 1,BLKSZ ;ANALYSIS
00032'020013$ EASY: LDA 0,BTXT
00033'006004$ JSR @APE
00034'044017$ STA 1,ITYP
00035'022506 LDA 0,@APPAS ;SET UP APPAY
00036'040511 STA 0,ENTER ;ADDRESSES FOR
00037'022505 LDA 0,@APPAS+1 ;OVERALL ANALYSIS
00040'040510 STA 0,ENTER+1
00041'022504 LDA 0,@APPAS+2
00042'040507 STA 0,ENTER+2
00043'022503 LDA 0,@APPAS+3
00044'040506 STA 0,ENTER+3
00045'020005$ LDA 0,TFLS
00046'042017- STA 0,@ASIZE
00047'004516 JSR GAP ;DO OVERALL ANALYSIS
00050'020471 LDA 0,ELKF ;CHECK IF ANALYSIS
00051'024504 LDA 1,ONE ;IS TO BE DONE BY
00052'106414 SUB #0,1,SZF ;BLOCKS
00053'002445 JMP @WORK ;NO-- RETURN
00054'020464 LDA 0,ZERO
00055'040457 STA 0,WORK+14
00056'020005$ LDA 0,TFLS ;SET UP TRIAL COUNT
00057'040453 STA 0,WORK+12
00060'020462 LDA 0,BLKSZ ;LOAD BLOCK SIZE
00061'042017- STA 0,@ASIZE ;INTO CALL TO DPACC
00062'040451 STA 0,WORK+13
00063'020475 NUB: LDA 0,FOUR ;SET UP TEMPORARY
00064'040445 STA 0,WORK+11 ;ADDRESS LOADING COUNTER
00065'034466 LDA 3,IARPS ;SET UP POINTERS FOR
00066'030466 LDA 2,IENT ;PICKING OUT DATA BY BLOCKS
00067'024005$ NINC: LDA 1,TFLS ;
00070'023400 LDA 00,0,3 ;GET ADDRESS

```

```

00071'107000      ADD 0,1          ;INCREMENT BY TRIAL COUNT
00072'020441      LDA 0,WORK+13
00073'106400      SUB 0,1          ;DECREMENT BY BLOCK COUNT
00074'045000      STA 1,0,2        ;STORE IN ENTER
00075'151400      INC 2,2          ;SET FOR NEXT
00076'175400      INC 3,3          ;ADDRESSES
00077'014432      DSZ WORK+11      ;CHECK IF DONE
00100'000767      JMP NINC         ;LOOP
00101'020005-     LDA 0,ITXT       ;PRINT BLOCK HEADING
00102'006003$     JSP @ATYPE
00103'010431      ISZ WORK+14      ;CHANGE BLOCK NUMBER
00104'024430      LDA 1,WORK+14
00105'006014$     JSR @AENDC       ;PRINT BLOCK NUMBER
00106'004457      JSR GAP          ;DO ANALYSIS
00107'020423      LDA 0,WORK+12    ;
00110'024423      LDA 1,WORK+13    ;CHECK IF DONE
00111'122405      SUB 1,0,SNR
00112'002406      JMP @WORK        ;EXIT
00113'020420      LDA 0,WORK+13    ;INCR BLOCK COUNTER
00114'024426      LDA 1,BLKSZ      ;BY BLOCK SIZE
00115'107000      ADD 0,1
00116'044415      STA 1,WORK+13
00117'000744      JMP NUB          ;LOOP

      000020  WORK:  .BLK 20
00140'000000      ZERO:  0
00141'000000      BLKF:  0
00142'000000      BLKSZ:  0
00143'000021$     APRAS:  ATIME
00144'000022$     ACODE
00145'000023$     ISTIM
00146'000024$     IFORE
00147'000000      ENTER:  0
00150'000000      0
00151'000000      0
00152'000000      0
00153'000143'     IAPPS:  APRAS
00154'000147'     IENT:   ENTER
00155'000001      ONE:    1
00156'000002      TWO:    2
00157'000003      THREE:  3
00160'000004      FOUR:   4
00161'000401'     AT1:    WTABC
00162'000374'     AT2:    MTABI
00163'000404'     AT3:    WTABI
00164'000410'     AT4:    ITAB+1

```

ROUTINE FOR ORGANIZING ANALYSIS

```

00165'054020-     GAR:    STA 3,APET      ;SAVE RETURN
00166'020752      LDA 0,ZERO          ;SET UP FOR
00167'042016-     STA 0,@AVIND        ;OVERALL ANALYSIS
00170'020757      LDA 0,ENTER
00171'042023-     STA 0,@ASTART
00172'020763      LDA 0,ONE
00173'042766      STA 0,@AT1
00174'020762      LDA 0,TWO
00175'042765      STA 0,@AT2
00176'042765      STA 0,@AT3
00177'020006-     LDA 0,ITXT+1
00200'006003$     JSR @ATYPE

```

00201'020747	LDA 0, ENTER+1	
00202'042021-	STA 0, @AITAB	
00203'006022-	JSR @ASORT	; DO ANALYSIS
00204'020011\$	LDA 0, CT	; CHECK NUMBER OF
00205'040717	STA 0, WORK+4	; STIMULI
00206'014716	DSZ WORK+4	; = 1?
00207'000402	JMP .+2	; NO
00210'000434	JMP FPTST	; YES - SKIP TO FP
00211'020007-	LDA 0, ITXT+2	
00212'006003\$	JSR @ATYPE	
00213'020736	LDA 0, ENTER+2	
00214'042021-	STA 0, @AITAB	
00215'020733	LDA 0, ENTER+1	
00216'042746	STA 0, @AT4	
00217'020547	LDA 0, MSK7	; SET VARIABLE MASKS AND WORDS
00220'040554	STA 0, MTABI	
00221'020735	LDA 0, TWO	
00222'040553	STA 0, MTABI+1	
00223'040562	STA 0, WTABI+1	
00224'020731	LDA 0, ONE	
00225'040555	STA 0, WTABC+1	
00226'010676	ISZ WORK+4	
00227'020726 RSTIM:	LDA 0, ONE	; RESET INDICATOR COUNT
00230'042016-	STA 0, @ANIND	
00231'030012\$	LDA 2, ACS	; LOAD ADDR OF STIM TABLE
00232'020672	LDA 0, WORK+4	; LOAD CURRENT COUNT
00233'113000	ADD 0, 2	; ADD TO CREATE ADDR OF STIM
00234'025000	LDA 1, 0, 2	; (AC1)=CURRENT STIM
00235'044542	STA 1, WTAB0	; STORE IN WORD TABLES
00236'044543	STA 1, WTABC	
00237'044545	STA 1, WTABI	
00240'006014\$	JSR @AENDC	; PRINT STIM NO.
00241'006022-	JSR @ASOPT	; DO ANALYSIS
00242'014662	DSZ WORK+4	; CHECK IF DONE
00243'000764	JMP RSTIM	; LOOP
00244'020010\$ FPTST:	LDA 0, NFP	; CHECK NO OF FPS
00245'040660	STA 0, WORK+5	
00246'014657	DSZ WORK+5	
00247'000402	JMP .+2	
00250'002020-	JMP @APET	
00251'020010-	LDA 0, ITXT+3	
00252'006003\$	JSR @ATYPE	
00253'020677	LDA 0, ENTER+3	
00254'040533	STA 0, ITAB	
00255'020673	LDA 0, ENTER+1	
00256'040532	STA 0, ITAB+1	
00257'020507	LDA 0, MSK7	
00260'040514	STA 0, MTABI	
00261'020675	LDA 0, TWO	
00262'040513	STA 0, MTABI+1	
00263'040522	STA 0, WTABI+1	
00264'020671	LDA 0, ONE	
00265'040515	STA 0, WTABC+1	
00266'010637	ISZ WORK+5	
00267'020666 PFOFE:	LDA 0, ONE	
00270'042016-	STA 0, @ANIND	
00271'030007\$	LDA 2, AFP	; LOAD ADDR OF FP TABLE
00272'020633	LDA 0, WORK+5	; LOAD CURRENT BLOCK COUNT
00273'113000	ADD 0, 2	; CREATE ADDR OF NEXT FP

00274'025000	LDA 1,0,2	;LOAD NEXT FP
00275'044502	STA 1,WTAB0	;STORE IN WORD TABLES
00276'044503	STA 1,WTABC	
00277'044505	STA 1,WTABI	
00300'006014\$	JSR @ABNDC	;PRINT CURRENT FP
00301'006022-	JSR @ASORT	;DO ANALYSIS
00302'014623	DSZ WORK+5	;CHECK IF DONE
00303'000764	JMP RFORE	
00304'020011\$ BOTH:	LDA 0,CT	;CHECK FOR NUMBER
00305'040513	STA 0,TEMP+6	;OF STIM
00306'014512	DSZ TEMP+6	
00307'000402	JMP .+2	
00310'002020-	JMP @ARET	
00311'020011-	LDA 0,ITXT+4	
00312'006003\$	JSR @ATYPE	
00313'020636	LDA 0,ENTER+2	
00314'040473	STA 0,ITAB	
00315'020635	LDA 0,ENTER+3	
00316'040472	STA 0,ITAB+1	
00317'020631	LDA 0,ENTER+1	
00320'040471	STA 0,ITAB+2	
00321'020445	LDA 0,MSK7	
00322'040452	STA 0,MTABI	
00323'040452	STA 0,MTABI+1	
00324'020631	LDA 0,ONE	;SET UP FOR STIM & FP
00325'040456	STA 0,WTABC+2	;ANALYSIS
00326'020630	LDA 0,TWO	
00327'040447	STA 0,MTABI+2	
00330'040456	STA 0,WTABI+2	
00331'010467	ISZ TEMP+6	
00332'030012\$ SAIL:	LDA 2,ACS	;OUTER LOOP - STIM
00333'020465	LDA 0,TEMP+6	
00334'113000	ADD 0,2	
00335'025000	LDA 1,0,2	
00336'044441	STA 1,WTAB0	
00337'044442	STA 1,WTABC	
00340'044444	STA 1,WTABI	
00341'020010\$	LDA 0,NFP	;RESET FP COUNT
00342'040450	STA 0,TEMP	
00343'020613 FLY:	LDA 0,TWO	
00344'042016-	STA 0,@ANIND	
00345'030007\$	LDA 2,AFP	;INNER LOOP - FPS
00346'020444	LDA 0,TEMP	
00347'113000	ADD 0,2	
00350'025000	LDA 1,0,2	
00351'044427	STA 1,WTAB0+1	
00352'044430	STA 1,WTABC+1	
00353'044432	STA 1,WTABI+1	
00354'024423	LDA 1,WTAB0	
00355'006014\$	JSR @ABNDC	
00356'024422	LDA 1,WTAB0+1	
00357'006014\$	JSR @ABNDC	
00360'006022-	JSR @ASORT	;DO ANALYSIS
00361'014431	DSZ TEMP	;CHECK FP COUNT
00362'000761	JMP FLY	
00363'014435	DSZ TEMP+6	;CHECK STIM COUNT
00364'000746	JMP SAIL	
00365'002020-	JMP @ARET	;EXIT
00366'177777 MSK7:	177777	
00367'177777 MTAB0:	177777	

```

00370'177777      177777
00371'177777 MTABC: 177777
00372'177777      177777
00373'177777      177777
      000003 MTABI:  .BLK 3
      000002 WTABO:  .BLK 2
      000003 WTABC:  .BLK 3
      000003 WTABI:  .BLK 3
      000003 ITAB:   .BLK 3
      000020 TEMP:   .BLK 20
00432'000366' AMTAB: MTABO-1
00433'000370'      MTABC-1
00434'000373'      MTABI-1
00435'000376' AWTAB: WTABO-1
00436'000400'      WTABC-1
00437'000403'      WTABI-1

```

```

;ROUTINE FOR ORGANIZING EACH ANALYSIS
;INTO OVERALL, CORRECT, INCORRECT

```

```

00440'054754 SORT:  STA 3,TEMP+2      ;SAVE RETURN
00441'024771      LDA 1,AMTAB          ;MASK TABLE ADDR
00442'030773      LDA 2,AWTAB          ;WORD TABLE ADDR
00443'004422      JSR MEAN              ;DO ANALYSIS
00444'020006$     LDA 0,PEPC            ;CHECK IF ANY ERRORS
00445'040746      STA 0,TEMP+1          ;POSSIBLE IN DESIGN
00446'014745      DSZ TEMP+1
00447'000404      JMP .+4
00450'020013-     LDA 0,ITXT+6
00451'006003$     JSP @ATYPE            ;CR + LF
00452'002742      JMP @TEMP+2          ;EXIT - NO ERRORS
00453'010422      ISZ NIND              ;BUMP INDICATOR COUNT
                                          ;SET FOR CORRECT
                                          ;RESP ANALYSIS
00454'024757      LDA 1,AMTAB+1
00455'030761      LDA 2,AWTAB+1
00456'004407      JSP MEAN              ;DO ANALYSIS
                                          ;SET FOR ERROR
                                          ;ANALYSIS
00457'024755      LDA 1,AMTAB+2
00460'030757      LDA 2,AWTAB+2
00461'004404      JSR MEAN              ;DO ERROR ANALYSIS
00462'020013-     LDA 0,ITXT+6
00463'006003$     JSP @ATYPE            ;CR + LF
00464'002730      JMP @TEMP+2          ;EXIT

```

```

;ROUTINE TO CALL DPACC AND CARVE

```

```

00465'044412 MEAN:  STA 1,AMASK          ;MASK TABLE ADDR
00466'050412      STA 2,AWORD          ;WORD TABLE ADDR
00467'054726      STA 3,TEMP+3          ;RETURN
00470'006015$     JSR @QUACC           ;GO TO ACCUM ROUTINE
00471'000504'     TOTAL                  ;ADDR OF D.P. SUM
00472'000506'     COUNT                  ;ADDR OF DATA COUNT
00473'000000 START: 0                    ;START OF DATA ARRAY
00474'000000 SIZE:  0                    ;SIZE OF ARRAYS
00475'000000 NIND:  0                    ;NUMBER OF INDICATORS
00476'000406'     ITAB-1                 ;ADDR OF IND. ADDRESSES
00477'000000 AMASK: 0                    ;ADDR OF MASKS
00500'000000 AVORD: 0                    ;ADDR OF WORDS
00501'020012-     LDA 0,ITXT+5
00502'006003$     JSP @ATYPE

```

```

00503'006016$      JSR @CLEVE      ;DO DIVISION
00504'000000      TOTAL:      0
00505'000000      0
00506'000000      COUNT:      0
00507'020014-      LDA 0,I TXT+7
00510'006003$      JSR @ATYPE      ;SINGLE SPACE
00511'024775      LDA 1,COUNT
00512'006014$      JSR @ABNDC      ;PRINT DATA COUNT
00513'002702      JMP @TEMP+3      ;EXIT

00514'006440      TXT1: .TXT * <15>
00515'005012      <12><12>
00516'005012      <12><12>
00517'042040      D
00520'052101      AT
00521'020101      A
00522'047101      AN
00523'046101      AL
00524'051531      YS
00525'051511      IS
00526'043040      F
00527'051117      OR
00530'051040      P
00531'020124      T
00532'054105      EX
00533'042520      PE
00534'044522      RI
00535'042515      ME
00536'052116      NT
00537'047040      N
00540'027117      O.
00541'000040      *
00542'006440      TXT2: .TXT * <15>
00543'020012      <12>
00544'052523      SU
00545'045102      BJ
00546'041505      EC
00547'020124      T
00550'047516      NO
00551'020056      .
00552'000000      *
00553'006440      TXT3: .TXT * <15>
00554'005012      <12><12>
00555'040440      A
00556'040516      NA
00557'054514      LY
00560'044523      SI
00561'020123      S
00562'054502      BY
00563'041040      B
00564'047514      LO
00565'045503      CK
00566'037523      S?
00567'024040      (
00570'024461      1)
00571'054440      Y
00572'051505      ES
00573'024040      (
00574'024462      2)
00575'047040      N

```

00576'020117 0
 00577'000000 *
 00600'006440 TXT4: .TXT * <15>
 00601'020012 <12>
 00602'046102 BL
 00603'041517 OC
 00604'020113 K
 00605'044523 SI
 00606'042532 ZE
 00607'036440 =
 00610'037440 ?
 00611'000040 *
 00612'006440 TXT5: .TXT * <15>
 00613'005012 <12><12>
 00614'040440 A
 00615'040516 NA
 00616'054514 LY
 00617'044523 SI
 00620'020123 S
 00621'054502 BY
 00622'041040 B
 00623'047514 LO
 00624'045503 CK
 00625'020123 S
 00626'005015 <15><12>
 00627'041040 B
 00630'047514 LO
 00631'045503 CK
 00632'047040 N
 00633'027117 O.
 00634'000040 *
 00635'005015 TXT6: .TXT * <15><12>
 00636'005012 <12><12>
 00637'053117 OV
 00640'051105 ER
 00641'046101 AL
 00642'020114 L
 00643'047101 AN
 00644'046101 AL
 00645'051531 YS
 00646'051511 IS
 00647'005015 <15><12>
 00650'020012 <12>
 00651'020040
 00652'053117 OV
 00653'051105 ER
 00654'046101 AL
 00655'020114 L
 00656'052122 RT
 00657'020040
 00660'020040
 00661'047040 N
 00662'020040
 00663'041440 C
 00664'051117 OR
 00665'042522 RE
 00666'052103 CT
 00667'051040 R
 00670'020124 T
 00671'020040

00672'020040
 00673'020116 N
 00674'020040
 00675'020040
 00676'051105 ER
 00677'047522 RO
 00700'020122 R
 00701'052122 RT
 00702'020040
 00703'020040
 00704'047040 N
 00705'005015 <15><12>
 00706'000012 <12>*
 00707'005015 TXT7: .TXT *(<15><12>
 00710'005012 <12><12>
 00711'047101 AN
 00712'046101 AL
 00713'051531 YS
 00714'051511 IS
 00715'041040 B
 00716'020131 Y
 00717'052123 ST
 00720'046511 IM
 00721'046125 UL
 00722'006511 I<15>
 00723'005012 <12><12>
 00724'051440 S
 00725'044524 TI
 00726'020115 M
 00727'020040
 00730'053117 OV
 00731'051105 ER
 00732'046101 AL
 00733'020114 L
 00734'052122 RT
 00735'020040
 00736'020040
 00737'047040 N
 00740'020040
 00741'041440 C
 00742'051117 OR
 00743'042522 PE
 00744'052103 CT
 00745'051040 R
 00746'020124 T
 00747'020040
 00750'020040
 00751'020116 N
 00752'020040
 00753'020040
 00754'051105 ER
 00755'047522 RO
 00756'020122 R
 00757'052122 RT
 00760'020040
 00761'020040
 00762'047040 N
 00763'005015 <15><12>
 00764'000012 <12>*
 00765'005015 TXT8: .TXT *(<15><12>

00766'005012 <12><12>
 00767'047101 AN
 00770'046101 AL
 00771'051531 YS
 00772'051511 IS
 00773'041040 B
 00774'020131 Y
 00775'047506 FO
 00776'042522 PE
 00777'042520 PE
 01000'044522 RI
 01001'042117 OD
 01002'006523 S<15>
 01003'005012 <12><12>
 01004'020040
 01005'043040 F
 01006'020120 P
 01007'020040
 01010'053117 OV
 01011'051105 ER
 01012'046101 AL
 01013'020114 L
 01014'052122 RT
 01015'020040
 01016'020040
 01017'047040 N
 01020'020040
 01021'041440 C
 01022'051117 OR
 01023'042522 PE
 01024'052103 CT
 01025'051040 R
 01026'020124 T
 01027'020040
 01030'020040
 01031'020116 N
 01032'020040
 01033'020040
 01034'051105 ER
 01035'047522 RO
 01036'020122 R
 01037'052122 RT
 01040'020040
 01041'020040
 01042'047040 N
 01043'005015 <15><12>
 01044'000012 <12>*<12>
 01045'005015 TXT9: .TXT *<15><12>
 01046'005012 <12><12>
 01047'047101 AN
 01050'046101 AL
 01051'051531 YS
 01052'051511 IS
 01053'041040 B
 01054'020131 Y
 01055'052123 ST
 01056'046511 IM
 01057'046125 UL
 01060'020111 I
 01061'020130 X

01062'047506	FO
01063'042522	RE
01064'042520	PE
01065'044522	RI
01066'042117	OD
01067'006523	S<15>
01070'005012	<12><12>
01071'051440	S
01072'044524	TI
01073'020115	M
01074'020040	
01075'050106	FP
01076'020040	
01077'047440	O
01100'042526	VE
01101'040522	PA
01102'046114	LL
01103'051040	R
01104'020124	T
01105'020040	
01106'020040	
01107'020116	N
01110'020040	
01111'047503	CO
01112'051122	RR
01113'041505	EC
01114'020124	T
01115'052122	PT
01116'020040	
01117'020040	
01120'047040	N
01121'020040	
01122'020040	
01123'042440	E
01124'051122	RR
01125'051117	OR
01126'051040	R
01127'020124	T
01130'020040	
01131'020040	
01132'006516	N<15>
01133'005012	<12><12>
01134'000000	*
01135'020040	TXT10: .TXT *
01136'000040	*
01137'005015	TXT11: .TXT *<15><12>
01140'000000	*
01141'000040	TXT12: .TXT * *
01142'005015	TXT13: .TXT *<15><12>
01143'052040	T
01144'052117	OT
01145'046101	AL
01146'042440	E
01147'051122	RR
01150'051117	OR
01151'020123	S
01152'042522	PE
01153'052522	RU
01154'020116	N
01155'020075	=
01156'000040	*

```

; SUBROUTINE CARVE -- ROUTINE TO DIVIDE A
; DOUBLE PRECISION FIXED POINT NUMBER BY
; A SINGLE PRECISION FIXED POINT NUMBER,
; YIELDING AND PRINTING OUT A FLOATING
; POINT NUMBER IN ORDINARY DECIMAL
; NOTATION
; ***NOTE: CARVE DOES NOT SAVE THE
; QUOTIENT, ONLY PRINTS IT OUT
;
; CALL TO CARVE IS:
;     JSR @CLEVE
;     HIGH ORDER DIVIDEND
;     LOW ORDER DIVIDEND
;     DIVISOR
;     RETURN
;
; >>>>USES SUBROUTINES PROO & INOUT
;
; PORTIONS OF THIS ROUTINE ARE
; BASED ON THE ROUTINE ENDEC
; FOUND IN THE PUBLICATION,
; "HOW TO USE THE NOVA COMPUTERS"
; COPYRIGHT 1971, DATA GENERAL CORPORATION
; REPRODUCED BY PERMISSION
;
;     G.M.OLSON
;     19 MARCH 1973 (A)

```

```

.TITL CARVE
.ENT CLEVE
.EXTD APUTC,ALGAE,MONK

```

```

.ZREL
00000-000000' CLEVE: CUT

```

```

.NREL
00000'020444 CUT: LDA 0,THREE ;SAVE (AC3) + 3
00001'163000 ADD 3,0 ;FOR RETURN
00002'040450 STA 0,SPLIT
00003'021400 LDA 0,0,3 ;GET HIGH ORDER DVD.
00004'025401 LDA 1,1,3 ;GET LOW ORDER DVD.
00005'031402 LDA 2,2,3 ;GET DIVISOR
00006'050431 STA 2,DIVSR ;STORE "
00007'006002$ JSR @ALGAE ;DIVIDE
00010'040424 STA 0,REND ;STORE REMAINDER
00011'020431 LDA 0,ZERO ;SET INDICATOR TO PRINT
00012'040423 STA 0,BLUR ;LEADING BLANKS
00013'121000 MOV 1,0
00014'004445 JSR SLICE ;PRINT INTEGRAL PART
00015'020433 LDA 0,PUNCT
00016'006001$ JSR @APUTC ;PRINT DECIMAL POINT
00017'020426 LDA 0,FOUR ;SET COUNTER TO 4
00020'040421 STA 0,SHARP
00021'024413 POINT: LDA 1,REND ;GET REMAINDER
00022'030522 LDA 2,TENS+3 ;GET CONSTANT 10
00023'006003$ JSR @MONK ;MULTIPLY
00024'030413 LDA 2,DIVSR ;GET DIVISOR
00025'006002$ JSR @ALGAE ;DIVIDE AGAIN
00026'040406 STA 0,REND ;SAVE REMAINDER

```

```

00027'121000      MOV 1,0
00030'004431      JSR SLICE      ;PRINT NEXT DECIMAL PART
00031'014410      DSZ SHARP      ;DECREMENT COUNTER
00032'000767      JMP POINT      ;REPEAT
00033'002417      JMP @SPLIT     ;SPLIT TO CALLING ROUTINE

```

```

00034'000000      PEND: 0
00035'000000      BLUR: 0
00036'000000      DICE: 0
00037'000000      DIVSP: 0
00040'000000      LAST: 0
00041'000000      SHARP: 0
00042'000000      ZERO: 0
00043'000001      ONE: 1
00044'000003      THREE: 3
00045'000004      FOUR: 4
00046'000005      FIVE: 5
00047'000060      C60: 60
00050'000056      PUNCT: 56
00051'000040      BLANK: 40
000007      SPLIT: .BLK 7

```

```

;ROUTINE TO PRINT DECIMAL NUMBERS
;WITHOUT LEADING ZEROES (BASED ON
;BNDEC), RIGHT JUSTIFIED

```

```

00061'054772      SLICE: STA 3,SPLIT+1
00062'034760      LDA 3,ZERO      ;SET INDICATOR TO
00063'054753      STA 3,DICE      ;SUPPRESS LEADING ZEROES
00064'034762      LDA 3,FIVE      ;SET PLACE COUNT
00065'054753      STA 3, LAST
00066'034460      LDA 3, INST
00067'054401      STA 3, .+1
00070'000000      MINCE: 0
00071'020756      LDA 0, C60
00072'146443      SUBO 2, 1, SNC
00073'101401      INC 0, 0, SKP
00074'147001      ADD 2, 1, SKP
00075'000775      JMP .-3
00076'050757      STA 2, SPLIT+3  ;SAVE (AC2)
00077'014741      DSZ LAST        ;DECREMENT PLACE COUNT
00100'000402      JMP .+2
00101'000424      JMP SPOIL
00102'034734      LDA 3, DICE      ;GET INDICATOR
00103'030737      LDA 2, ZERO      ;AND ZERO, CHECK
00104'156414      SUB# 2, 3, SZR   ;IF EQUAL
00105'000420      JMP SPOIL        ;NOT EQUAL -- PRINT CHAR.
00106'030735      LDA 2, ONE
00107'034740      LDA 3, C60
00110'116414      SUB# 0, 3, SZR   ;CURRENT DIGIT 0?
00111'000413      JMP SPOIL-1     ;NO, CHANGE INDICATOR
00112'034723      LDA 3, BLUR
00113'030727      LDA 2, ZERO
00114'156414      SUB# 2, 3, SZR
00115'000405      JMP POKE        ;DO NOT PRINT LEADING ZEROES
00116'040737      STA 0, SPLIT+3
00117'020732      LDA 0, BLANK
00120'006001$     JSR @APUTC      ;PRINT LEADING BLANK
00121'020734      LDA 0, SPLIT+3  ;RESTORE (AC0)
00122'030732      POKE:  LDA 2, SPLIT+2 ;RESTORE (AC2)

```

00123'000410		JMP PARE	
00124'050712		STA 2,DICE	
00125'006001\$	SPOIL:	JSR @APUTC	;PRINT CHARACTER
00126'030712		LDA 2, LAST	;CHECK IF LAST
00127'034713		LDA 3,ZERO	;CHARACTER
00130'156415		SUB# 2,3,SNR	
00131'000405		JMP PARE+3	
00132'030722		LDA 2,SPLIT+2	
00133'010735	PARE:	ISZ MINCE	;BACK INTO ENDEC
00134'151203		MOVR 2,2,SNC	
00135'000733		JMP MINCE	
00136'020705		LDA 0,ONE	
00137'040676		STA 0,BLUR	
00140'002713		JMP @SPLIT+1	

000012		.RDX 10
00141'023420	TENS:	10000
00142'001750		1000
00143'000144		100
00144'000012		10
00145'000001		1
000010		.RDX 8
00146'030451	INST:	LDA 2,..+TENS-MINCE
		.END

```

;UNSigned DIVIDE, MULTIPLY
;
;COPYRIGHT 1969, DATA GENERAL CORPORATION
;REPRODUCED BY PERMISSION.
;      ADAPTED FROM DATA GENERAL
;      MATH ROUTINE PACKAGE BY
;      G.M. OLSON
;      20 NOVEMBER 1972 (B)

```

```

.TITL PPOG
.ENT MONK,AMOEB,NIN,ALGAE

```

```

.ZPEL
00000-000021' MONK: .MPYU'
00001-000022' NIN: .MPYA
00002-000001' ALGAE: .DIVU'
00003-000000' AMOEB: .DIVI

```

```

.NPEL

```

```

00000'102400 .DIVI: SUB 0,0 ; INTEGER DIVIDE, CLEAR AC0
00001'054416 .DIVU: STA 3,.CC03 ; SAVE AC3
00002'142432 ; SUB# 2,0,SZC ; TEST FOR OVERFLOW
00003'000412 JMP .CC99 ; SET CARRY AND RETURN
00004'034414 LDA 3,.CC20 ; 16 ITERATIONS
00005'125120 MOVZL 1,1 ; SHIFT LOW DIVIDEND
00006'101100 .CC98: MOVL 0,0 ; SHIFT HIGH DIVIDEND
00007'142412 SUB# 2,0,SZC ; DOES DIVISOR GO IN?
00010'142400 SUB 2,0 ; YES
00011'125100 MOVL 1,1 ; SHIFT LOW DIVIDEND
00012'175404 INC 3,3,SZP ; CHECK COUNT
00013'000773 JMP .CC98 ; NOT DONE
00014'176441 SUBO 3,3,SKP ; DONE, CLEAR CARRY
00015'176420 .CC99: SUBZ 3,3 ; SET CARRY
00016'002401 JMP 0,CC03 ; RETURN

```

```

00017'000000 .CC03: 0 ; SAVE AC3

```

```

00020'177760 .CC20: -20 ; - 16 DECIMAL

```

```

00021'102460 .MPYU: SUBC 0,0 ; CLEAR AC0, DON'T DISTURB
; CARRY
00022'054411 .MPYA: STA 3,.CB03 ; SAVE AC3
00023'034411 LDA 3,.CB20 ; 16 TIMES THRU LOOP
00024'125203 .CB99: MOVR 1,1,SNC ; CHECK NEXT MULTIPLIER BIT
00025'101201 MOVR 0,0,SKP ; 0, JUST SHIFT
00026'143220 ADDZP 2,0 ; 1, ADD MULTIPLICAND AND SHIFT
00027'175404 INC 3,3,SZP ; CHECK FOR 16TH TIME THRU
00030'000774 JMP .CB99 ; NO, CONTINUE
00031'125260 MOVR 1,1 ; YES, SHIFT LAST LOW BIT
; (NOTE IT WAS COMPLEMENTED BY
; FINAL INC)
00032'002401 JMP 0,CB03 ; RETURN

```

```

00033'000000 .CB03: 0 ; RETURN ADDRESS

```

```

00034'177760 .CB20: -20 ; -16 DECIMAL

```

```

; SUBROUTINE DPACC -- ACCUMULATES THE SUM
; OF A SERIES OF ARBITRARY SINGLE PRECISION
; VALUES IN A DOUBLE PRECISION ACCUMULATOR.
; ALLOWS OPTIONAL INCORPORATION OF SELECTIVE
; ACCUMULATION OUT OF THE USER'S DATA ARRAY
; BY MEANS OF A SET OF INDICATOR ARRAYS THAT
; PARALLEL THE STRUCTURE OF THE DATA ARRAY.
; BOTH A MASK AND A TEST WORD ARE USED
; TO SELECT DATA ON THE BASIS OF A GIVEN
; INDICATOR ARRAY, AND WHEN MORE THAN ONE
; INDICATOR IS USED THE SELECTION IS BASED
; ON ALL INDICATORS SIMULTANEOUSLY, I.E.,
; THE INDIVIDUAL YES-NO DECISIONS ARE AND'ED,
; SO THAT A DATUM MUST PASS ALL TESTS BEFORE
; IT WILL BE ACCUMULATED.
; THE CALLING SEQUENCE IS:
;   JSP @QUACC
;   ADDRESS OF HIGH ORDER WORD FOR D.P.
;   SUM (FOR OUTPUT)
;   ADDRESS OF DATA COUNT (FOR OUTPUT)
;   ADDRESS-1 OF DATA ARRAY
;   SIZE OF DATA ARRAY
;   NUMBER OF INDICATOR ARRAYS
;   ADDRESS-1 OF ADDRESS-1S OF INDICATOR
;   ARRAYS
;   ADDRESS-1 OF TABLE OF INDICATOR MASKS
;   ADDRESS-1 OF TABLE OF INDICATOR WORDS
;   RETURN
;
; NOTE: IF NUMBER OF INDICATOR ARRAYS IS
; ZERO, ENTIRE DATA ARRAY IS ACCUMULATED.
; RELEVANT ARGUMENTS IN THE SUBROUTINE CALL MUST
; BE FILLED WITH DUMMY VALUES. THIS
; ROUTINE ALWAYS SETS THE D.P. SUM TO
; ZERO ON ENTRY.
;
;   G.M. OLSON
;   5 DECEMBER 1972 (A)
;

```

```

.TITL DPACC
.ENT QUACC

```

```

.ZPEL

```

```

00000-000000' QUACC: FORTH
00001-000125' FLASH: .DADD

```

```

.NPEL

```

```

00000'030507 FORTH: LDA 2,EIGHT ;SET RETURN AT
00001'173000 ADD 3,2 ;(AC3) + 8
00002'050500 STA 2,BACK
00003'031400 LDA 2,0,3 ;STORE ADDR. OF
00004'050464 STA 2,ASUM ;D.P.SUM
00005'024500 LDA 1,ZERO ;ZERO D.P. SUM
00006'045000 STA 1,0,2
00007'045001 STA 1,1,2
00010'044506 STA 1,WORK ;ZERO COUNTER
00011'044477 STA 1,TEMCT ;ZERO DATA COUNT
00012'021401 LDA 0,1,3 ;STORE ADDR. OF
00013'040476 STA 0,ACT ;DATA COUNT

```

00014'021402		LDA 0,2,3	;STORE ADDR.-1
00015'040020		STA 0,20	;OF DATA ARRAY
00016'021403		LDA 0,3,3	;STORE SIZE OF
00017'040464		STA 0,SIZE	;DATA ARRAY
00020'021404		LDA 0,4,3	;STORE NUMBER OF
00021'040471		STA 0,GNAPP	;INDICATOR ARRAYS
00022'021405		LDA 0,5,3	;STORE LOC. OF
00023'040470		STA 0,ARTAB	;TABLE OF IND. ADDR.
00024'021406		LDA 0,6,3	;STORE LOC. OF
00025'040467		STA 0,AMTAB	;MASK TABLE
00026'021407		LDA 0,7,3	;STORE LOC. OF
00027'040466		STA 0,AWTAB	;WORD TABLE
00030'010466	NACC:	ISZ WORK	;INCR. COUNTER
00031'020454		LDA 0,ZERO	;SET SKIP IND. TO 0
00032'040452		STA 0,IND	
00033'024457		LDA 1,GNAPP	;GET NO. OF IND. ARRAYS
00034'106415		SUB# 0,1,SNR	;= 0?
00035'000424		JMP ACCUM	;JUMP TO ACCUM
00036'044461		STA 1,WORK+1	;STORE NO. IND.
00037'020454		LDA 0,ARTAB	;TRANSFER ADDR. OF
00040'040021		STA 0,21	;PARAMETERS TO
00041'020453		LDA 0,AMTAB	;AUTO INC. LOCS
00042'040022		STA 0,22	
00043'020452		LDA 0,AWTAB	
00044'040023		STA 0,23	
00045'022021	OUFIE:	LDA 0,021	;GET NEXT IND. ADDR.
00046'030450		LDA 2,WORK	;GET CURRENT COUNT
00047'113000		ADD 0,2	;ADD TO GET IND. VALUE
00050'021000		LDA 0,0,2	
00051'026022		LDA 1,022	;LOAD IND. MASK
00052'123400		AND 1,0	;MASK
00053'026023		LDA 1,023	;LOAD IND. WORD
00054'030432		LDA 2,ONE	
00055'122404		SUB 1,0,SZR	;IND. = WORD?
00056'050426		STA 2,IND	;NO - SET ACCUM. SKIP
00057'014440		DSZ WORK+1	;CHECK IF THROUGH
00060'000765		JMP OUFIE	
00061'026020	ACCUM:	LDA 1,020	;GET NEXT DATUM
00062'020423		LDA 0,ZERO	
00063'014421		DSZ IND	;CHECK IND. SKIP
00064'000402		JMP .+2	
00065'000407		JMP ASUM+4	;SKIP ACCUM
00066'010422		ISZ TEMCT	;INC. DATA COUNT
00067'006001-		JSR @FLASH	;ACCUMULATE
00070'000000	ASUM:	0	
00071'030777		LDA 2,ASUM	;TRANSFER NEW SUM
00072'041000		STA 0,0,2	;TO STORAGE
00073'045001		STA 1,1,2	
00074'014407		DSZ SIZE	;CHECK IF DONE
00075'000733		JMP NACC	
00076'024412		LDA 1,TEMCT	;TRANSFER DATA COUNT
00077'030412		LDA 2,ACT	
00100'045000		STA 1,0,2	
00101'002401		JMP @BACK	
00102'000000	BACK:	0	
00103'000000	SIZE:	0	
00104'000000	IND:	0	
00105'000000	ZERO:	0	

```

00106'000001 ONE: 1
00107'000010 EIGHT: 10
00110'000000 TEMCT: 0
00111'000000 ACT: 0
00112'000000 GNARR: 0
00113'000000 ARTAB: 0
00114'000000 AMTAB: 0
00115'000000 AWTAB: 0
000007 WOPK: .BLK 7

; DATA GENERAL ROUTINE FOR D.P. ADDITION
; COPYRIGHT 1969, DATA GENERAL CORPORATION
; REPRODUCED BY PERMISSION

00125'054414 .DADD: STA 3,.BD03 ; SAVE RETURN
00126'010413 ISZ .BD03 ; BUMP PAST ADDRESS CONSTANT
00127'050411 STA 2,.BD02 ; *SAVE AC2
00130'035400 LDA 3,0,3 ; ADDRESS OF D2
00131'031400 LDA 2,0,3 ; HIGH ORDER OF D2
00132'035401 LDA 3,1,3 ; LOW ORDER OF D2
00133'167022 ADDZ 3,1,SZC ; LOW ORDER ADD
00134'101400 INC 0,0 ; CARRY TO HIGH ORDER
00135'143000 ADD 2,0 ; HIGH ORDER ADD
00136'030402 LDA 2,.BD02 ; *RESTORE AC2
00137'002402 JMP 0.BD03 ; AND RETURN

00140'000000 .BD02: 0 ; *SAVE AC2
00141'000000 .BD03: 0 ; SAVE RETURN
.END

```

UNCLASSIFIED

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) NAVAL SUBMARINE MEDICAL RESEARCH LABORATORY Naval Submarine Medical Center		2a. REPORT SECURITY CLASSIFICATION Unclassified	
		2b. GROUP	
3. REPORT TITLE RTLAB: A REACTION-TIME LABORATORY FOR THE DATA GENERAL NOVA MINICOMPUTER			
4. DESCRIPTIVE NOTES (Type of report and inclusive dates) Interim report			
5. AUTHOR(S) (First name, middle initial, last name) Gary M. OLSON, George MOELLER, and Kevin LAXAR			
6. REPORT DATE 7 August 1973		7a. TOTAL NO. OF PAGES (21 plus 4 App.(98 pp))	7b. NO. OF REFS 2
8a. CONTRACT OR GRANT NO.		9a. ORIGINATOR'S REPORT NUMBER(S) NSMRL Report Number 748	
b. PROJECT NO. MF51.524.004-2002DX5G		9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)	
c.			
d.			
10. DISTRIBUTION STATEMENT Approved for public release; distribution unlimited.			
11. SUPPLEMENTARY NOTES		12. SPONSORING MILITARY ACTIVITY Naval Submarine Medical Center Box 600 Naval Submarine Base Groton, Connecticut 06340	
13. ABSTRACT RTLAB is a package of relocatable assembly-language subroutines for the design, control, and analysis of behavioral experiments on the Data General Corporation NOVA family of computers. It was originally designed for use in reaction time research but is readily adaptable to any discrete-trial task. RTLAB accepts design parameters from the experimenter for a given experiment and then produces an appropriately randomized and counterbalanced set of trials for presentation to a subject. During an experimental session RTLAB controls the presentation of stimuli, the acceptance and scoring of responses, and all of the timing and input-output constraints associated with control and data acquisition. Its current design does not allow for time-sharing among more than one subject station. A history of the experiment is saved for later analysis and transfer to storage media. User modifications of any of the details of these operations is facilitated by the organization of RTLAB into subroutine modules. The organization of RTLAB and its use are presented along with sufficient documentation of the routines to allow user modification where appropriate.			

DD FORM 1 NOV 65 1473 (PAGE 1)

S/N 0102-014-6600

UNCLASSIFIED

Security Classification

Security Classification

DD FORM 1473 (BACK)
(PAGE 2)

Security Classification