PROGRAM LIBRARY CATALOG

013-000085-01

Ordering No. 013-000085

Printed in the United States of America
Rev. 01, September 1975

PROGRAM LIBRARY [1]

INTRODUCTION

This catalog contains an abstract of each program currently available to members of the Data General User's Group. A fee to cover reproduction and distribution costs is charged for each program. The current price list is in Appendix A. Orders for programs must specify the name and catalog number of the program(s) requested. For orders under $50.00 a check or money order must be included. For orders of $50.00 or more, a company purchase order will be accepted. To simplify the ordering procedure, Program Library Coupons are available in a $10.00 denomination.

The User's Group Program Library functions as a clearinghouse and distributor of User programs. No technical or programming assistance is available. If a program does not work as described in the abstract, the problem should be fully documented and sent to the User's Group Administrator, Data General Corp., Southboro, MA 01772. Such critiques will be forwarded to the program author for comment. If a program is not usable in the form supplied and described in the documentation, it will be removed from the Library.

The description and availability of the programs contained in this Library are subject to change without notice. Distribution will be in accordance with the current policy of the Data General User's Group.[2]

PROGRAM DELINEATIONS

In order to make the identification of a particular type of program convenient for the User, the following classification system has been adopted.

COMMUNICATIONS - Programs used to transmit or control data flow from the originating point via common carrier or private communications lines to a remote location for processing or distribution. Also includes communications between processors.

COMPUTATIONAL - Programs used in computing for applications such as scientific, engineering, bio-medical, military, aerospace, and all other non-manufacturing areas.

EDUCATIONAL - Programs that are specifically useful to educational institutions. Generally these programs relate to the instructional process, administrative procedures and courseware. Programs that educators or students also will find useful will be listed in other categories, especially Computational, Games, and Utilities.

INSTRUMENTATION AND CONTROL - Programs which will be utilized directly in on-line control of continuous or batch manufacturing processes, or associated with the support of general manufacturing activity, and other non-scientific data acquisition and control systems.

UTILITIES - Programs which facilitate the generation and operation of other programs, e.g., interpreters, editors, assemblers, simulators, I/O handlers. Also the facilitation of the movement of data, for example: tape to disk, disk to tape, disk to disk, etc.

---

1 PROGRAM LIBRARY

Data General Corp. warrants only that the programs supplied are duplicates of the programs submitted by the author listed. Therefore, Data General will not be liable for any special, indirect, or consequential damages arising out of, or in connection with, the use of the materials.

2 LICENSED PROGRAMS

As indicated herein, the availability of certain User's Group Programs is contingent upon the execution of a Data General Corp. Program License Agreement, Program Availability Schedule, and applicable documents. Programs in this category are identified by the letters "LP" following the catalog number.

TABLE OF CONTENTS

SECTION 5 - INSTRUMENTATION AND CONTROL

SECTION 6 - UTILITIES

SECTION 6 - UTILITIES

APPENDIX A - PRICING INFORMATION

APPENDIX B - SAMPLE FORMS

APPENDIX C - MEMBERSHIP INFORMATION

COMMUNICATIONS

Program No. 056

RTOS 2702 EMULATOR (.IBM)

Dr. Hardy J. Pottinger
University of Missouri - Rolla
Rolla, Missouri

.IBM is a special purpose software driver for the DGC 4025 360/370 Interface unit designed to be used with the DGC Real Time Operating System. It is designed to emulate the functional characteristics of an IBM 2702 with a Telegraph Type II adapter. This includes recognizing and responding to the channel command set for the 2702, providing ending status to commands and terminating READ operations on receipt of certain control characters. Data is transferred to and from the host in the byte multiplex mode. Calling sequences for .IBM are similar to other RTOS device handlers and are outlined in the .IBM user's manual.

4K read/write memory and RTOS are required. Necessary peripheral is the 4025 360/370 interface. The program is written in Assembly language.

Program No. 093

AUTO CALLER DRIVER SOFTWARE

This program provides an interface to a Bell auto call unit using a Model 5280 controller from Data General's Custom Products group.

1) The program will run on Rev.3 RDOS or later. Otherwise, modify VIEX for user interrupt return.

2) If device codes are different, change ACU = Definition on page 1 and reassemble.

Program No. 095

UT200 SIMULATOR

Robert J. Collins
Balcones Research Center
Route 4, Box 189
Austin, TX 78757

This program allows a NOVA family computer to communicate with a remote CDC 6600/6400 computer to simulate a 200-user terminal. Jobs may be submitted from the card reader in the normal manner, or from any input device supported by the operating system, including disk files. An assembly time option allows for generation of a communications history file.

All of the routines in the package must be assembled with the Macro Assembler and loaded with the relocatable Math Library (099-000001-02). The programs require 16K, or 20K with the history file. They were developed and tested using RDOS Rev.03. The host operating system was SCOPE EXPORT/IMPORT. A modified CDROV.RB is necessary and included in the package.

The routines operate at 2400 baud; any higher rate is untested.

Program No. 114 (LP)

.IBM APPLICATION SOFTWARE

Dr. Hardy J. Pottinger
University of Missouri
Rolla, Missouri

This package of software is designed for use with program #056, .IBM. It is designed to facilitate the simultaneous servicing of multiple independent applications. Rather than servicing a large number of similar low-speed teletype-like terminals, the software services a small number of rather varied applications such as a direct connected teletype, a card reader, a paper tape reader, and several high speed communication lines. Data need not even be generated by an external device but could be generated internally by software for test purposes.

The routines included are: 1) main monitor; 2) TTY driver; 3) hi-speed PTR driver; 4) full duplex data-link driver; 5) full duplex data-link handler; 6) mark sense card reader; 7) half duplex low speed modem; 8) utility routines.

A revised version of RTOS Rev.03 in use at U. of Missouri with this software is also included.

The package is written in Assembly language and contains instructions for operation.

Program No. 001

FAST FOURIER TRANSFORM AND INVERSE

Charles S. Cox and Ronald Stutheit
Scripps Institute of Oceanography
University of California
San Diego, California

This potentially reentrant program is in subroutine form and may be called from a main program. The JSR statement is followed by the following parameters: forward (F(N*T))/inverse (F(L*T)) transform switch (0 or 1); number of sample points; the real parts of the input series. When control returns from the subroutine, the transformed data will have replaced the input data.

The number of sample points must be an integer power of 2, up to 1024_{10} complex values. Trigonometric computations are preformed by table look-up, and input data is scaled, to reduce the possibility of overflow, with the scaling factor retained to allow for restoration of the output series.

Storage: 610_{10} words (6 on page 0, 348 for routine, 256 for cosine table).

Program No. 005

EIGENVALUE SOLUTION PROGRAM

Professor James A. Leone
Canisius College
Buffalo, New York

Using the Jacobi method of Eigenvalue solution, this BASIC program, written to run on Multi-User Basic (DGC #091-000026) solves for the eigenvalues and eigenvectors of any NxN matrix, The utilization of the matrix manipulation capability in this 71-statement program enables data to be supplied in one DATA statement:
E.G. 540 DATA N,A(1,1), A(2,1),...,A(N,N).

A printout of the A matrix's eigenvalues and associated eigenvectors is produced in a form suitable for such scientific applications as chemistry, physics, and the mathematical sciences.

Program No. 013

BCD TO BINARY CONVERSION BY RADIX REDUCTION

This assembly language routine reduces the 4-digit BCD value in ACØ to an equivalent binary representation, replacing the original BCD value in ACØ. The packed BCD string is treated as a binary value with an "expanded" radix. The routine reduces the radix of each digit until the converted value is obtained. This technique is considerably faster than the isolation of each digit followed by the binary reconstruction. This 12 (decimal) word routine has the following execution times: NOVA = 101.1 us; NOVA 1200 = 31.95 us; SUPERNOVA = 21.0 us; NOVA 800 = 19.2 us; SUPERNOVA SC = 12.6 us.

Program No. 022

FOCAL LENGTH BY MICROSCOPE METHOD

R. C. Bell
Vari-Tech Company
Grand Rapids, Michigan

Using approximate thick lens equations for symmetric lenses, this BASIC program calculates the focal length of symmetric lenses from four microscope measurements from a micrometer indicating microscope. Printout includes the average and standard deviation of the centered focal lengths of a batch of lenses.

Program No. 025

CONVERSION OF LINEAR ENGLISH MEASUREMENT VALUES TO METRIC EQUIVALENTS

M. Cook
Bovay Engineers, Inc.
Houston, Texas

This BASIC language program converts English Linear Measurement values into their Metric System equivalents. Since the computer will not accept any fractional values except those expressed in decimal form, all input data must be expressed as whole feet and/or inches, and decimal fractions thereof.

Four input options are accepted. Data may be entered in feet only (to the nearest whole foot or decimal fraction thereof); feet and inches only (to the nearest whole inch or decimal fraction thereof);

inches only, (to the nearest whole inch or decimal fraction thereof); or in miles and/or fractions, thereof.

Output may be obtained in Millimeters, Centimeters, Meters, or Kilometers, by entering the proper value of B when starting a run.

Another feature of this program gives it capability to convert long strings of values and identify the results by groups.

Program No. 026

CONVERSION OF METRIC LINEAR MEASUREMENT TO ENGLISH EQUIVALENTS

M. Cook
Bovay Engineers, Inc.
Houston, Texas

This BASIC program converts Metric linear measurements, millimeters, centimeters, meters, or kilometers into the English linear measurement equivalent of inches or feet. Provision is made for 17 different conversion options. The user may have his output in inches, to the nearest inch. In this case, the computer will round off all fractional values; those less than ½ are truncated; those greater than ½ are raised to the next whole number. Decimal values may be obtained, or common fractions to the nearest ½, ¼, 1/8, 1/16, 1/32, or 1/64. If the output is desired in feet, feet and inches, or feet, inches, and fractions thereof (decimal or common) similar options may be had.

Program No. 040

POLYNOMIAL CURVE FIT (POLYFIT)

Frank D. Whisler
U. S. Dept. of Agriculture
U. S. Water Conservation Lab.
4331 East Broadway
Phoenix, AZ 85040

POLYFIT will fit least-square polynomials to bivariate data using an orthogonal polynomial method. Limits are 11th-degree fit and a maximum of 100 data points. The program allows user to specify the lowest degree polynomial to be fit, and then fits the polynomials in order of ascending degree. At each stage, the index of determination (r2) is printed, and the user has the choice of going to the next higher degree fit, seeing either of two summaries of fit at that stage, or of stopping the program.

POLYFIT is written in extended BASIC and requires only a teletype for operation.

Program No. 044

CURVEFIT

Frank D. Whisler
U. S. Dept. of Agriculture
U. S. Water Conservation Lab.
4331 East Broadway
Phoenix, AZ 85040

CURVEFIT will determine which of 6 curves best fits the User's data using a least squares linearized fit. The program accepts up to 200 observations on 2 variables.

The program is written in Extended BASIC and requires only a teletype for operation.

Program No. 045

ARCTANGENT

Timothy J. Mulrooney
Naval Underwater Systems Center
Newport, Rhode Island

This program will take the fixed point single precision ARCTANGENT of the sin (t)/cos (t) and return the result in AC3 in the range $-\pi \leq \emptyset \leq \pi$. ARCTANGENT is called as a subroutine with the sin (t) in AC0 and cos(t) in AC1. Return to instruction following call with AC3=ARCTANGENT.

Either Hardware Multiply/Divide or Software Multiply/Divide is necessary. ARCTANGENT is written in assembly language.

Program No. 050

RTLAB

Gary Olson, George Moeller, Kevin Laxar
Naval Submarine Medical Research Lab.
Naval Submarine Base
Groton, CT 06340

RTLAB is a package of 20 relocatable

assembly language routines for on-line experimentation in discrete-trial behavioral experiments where classification of response or reaction time are dependent variables. The routines of RTLAB perform such functions as entering and storing parameters, preparing an appropriately counterbalanced sequence of trials for each subject, controlling the presentation and timing of the events on each trial, scoring and storing subject's responses, providing summary statistics for a subject when the session is over, and transferring data to an appropriate storage medium. The package is constructed to allow maximum User flexibility, especially in mating RTLAB with the User's experimental task and specific instrumentation. RTLAB is not designed for time-sharing between several subject stations.

8K of memory is necessary. Only peripheral required is a teletype, others are up to the User.

Program No. 052

WHOLE

Lawrence Whelan
EDL Co.
Box 2408
Gary, Indiana

This assembler subroutine, callable from Fortran, will truncate the fractional part of a double precision real number and return with only the non fractional part. The user must supply his own mainline program.

Program No. 053

SINGLE PRECISION AND DOUBLE PRECISION BCD TO BINARY

A. Zwahlen
Oy Stromberg Ab
Electronics Group
Helsinki, Finland

This assembly language subroutine converts either an 8-digit BCD-value in AC$\emptyset$, AC1 (double precision, entry: BCDBD) or a 4-digit BCD-value in AC$\emptyset$ (single precision, entry: BCDBS) into its equivalent binary representation replacing the BCD-value. The radix reduction algorithm (User's Group program no. 013) is used to convert a single precision 4-digit value. In double precision conversion, the high order part is converted first, then multiplied by 10000 (decimal) and the converted low order part is added to obtain the final result. The routine is not re-entrant. This 36 (decimal) word routine may be loaded anywhere in core.

8K of memory is needed.

Program No. 075

ACID-BASE TITRATION

Michael S. Spritzer
Dept. of Chemistry
Villanova University
Villanova, PA 19085

This BASIC program handles up to 100 points of acid-base titration data. Provision is made for correction of input data before the start of processing. End point volume and pH is returned. By user choice, the program can be applied to standardization of base, determination of acid concentration, or equivalent weight of the acid. Options include a complete table of normalized data including first derivative -- with end point indicated -- as well as a teletype plot of the titration curve. The program will run with Data General BASIC in 8K of core.

Program No. 081

BASIC CURVE FIT

This program performs a least squares curve fit on the data provided it (by keyboard or test equation). The number of points handled is dependent on core available. Polynomials fitting is possible to degree 10. After fit, the operator has the option to obtain a printer plot, statistics, or a new fit of the data.

Program No. 084

GASOLINE ECONOMY PROGRAM

Walter E. Wahnsiedler
Dept. of Materials Science
Northwestern University
2145 Sheridan Road
Evanston, IL 60201

This program accepts data input from the console regarding fuel usage in an

automobile. Mileages may be input either from a standard cumulative odometer, or from a "trip" odometer, which is reset to zero each time fuel is added to the tank. A list of average mpg at each fill-up is printed, along with a value for the cruising range, and if desired, the user may then delete any suspicious data points. Then an analysis of fuel economy by brand is printed, and the user is permitted to compute economies over portions of his data as needed. Cost can, of course, be substituted for fuel used, and in that case a cost analysis by brand will be obtained.

The program coded in Fortran. Minimum configuration is a NOVA-line CPU with 12K or more memory and teletype.

Program No. 092

FOREIGN EXCHANGE

This BASIC program will compute the Foreign currency equivalence of one U.S. dollar. It includes most of the popular European and Far Eastern currencies. Exchange rate information needed for input can be obtained through any bank or large newspaper.

Program No. 111

CALCULATOR

Ian Kettleborough
Texas A & M University
College Station, Texas 77843

This program simulates a four function (+, -, /, *) calculator having one storage register. The expression is typed on the teletype and when a carriage return is hit, the expression is evaluated from left to right. The expression value is then either typed out or stored in the storage register. The storage register may be used in place of a number in an expression. (Its latest value is substituted when the expression is evaluated.)

Output may be either in octal or decimal (decimal is the default), and is under the control of the user. Input numbers can be specified either in octal or decimal, with decimal being the default.

All calculations are in integer, having a range of -32,768 to +32,767.

The program is written in assembly language and may be run with SOS, RTOS, DOS, or RDOS using the hardware multiply/divide board if available. It is about 550 words in length and requires no page 0 locations.

Program No. 126

ANALYSIS OF VARIANCE PACKAGE

Richard Miller
Dept. of Psychology
Illinois State University
Bloomington-Normal, IL 61761

This package of BASIC programs performs eight analysis of variance routines (ANOVA) including:

1) Independent one-way ANOVA
2) Two way ANOVA
3) Three way ANOVA
4) Randomized blocks ANOVA
5) Two factor randomized blocks ANOVA
6) Split-plot factorial ANOVA (1 between, 1 within)
7) ANOVA for s split-plot factorial (2 between, 1 within)
8) Split-plot factorial ANOVA (1 between, 2 within)

Program No. 127

DARTMOUTH STATISTICAL PACKAGE

This package of 29 statistical programs was developed at Dartmouth College. They are all written in Extended BASIC and include the following categories:

Statistical Measures

- GEOMEN - Geometric mean and standard deviation.
- GROUP - Mean, median, variance, standard deviation, skewness, coefficient of variability.
- SAMPSTAT - Minimum, maximum, range, sum, mean, sum of squares, variance, standard deviation, standard error of mean and coefficient of variation.
- STATMEAS - 34 statistical measures on weighted or unweighted data using formulas from Nat'l Bureau of Standards.

Statistical Probabilities

- T-PROB - One and two tailed probabilities of student-T distribution for given number of degrees of freedom.

One Sample Tests

- STAT06 - Sign test confidence interval using fractional counts.
- STAT07 - Confidence limits for set of data using Wilcoxon signed rank sum procedure.
- STAT08 - Compares two groups of data using median test.
- STAT-09 - Compares two groups of data using Mann-Whitney rank test with confidence intervals.
- TWOSAMP - Analyzes two groups of unpaired data by the two-sample student-T statistic.

Paired Comparison

- SIGNRANK - Computes the Wilcoxon matched-pairs signed-rank statistic for two groups of data.

Contingency Tables

- BACT2L - Bayesian analysis of 2 level contingency table with maximum dimensions of 10x10.
- BACT3L - Bayesian analysis of 3 level contingency table.
- STAT04 - Computes Chi square for 2x2 contingency tables.
- STAT05 - Computes Chi square for any number of M by N contingency tables.

Analysis of Variance

- GRAECO - Computes analysis of variance table for simple Graeco-Latin square design.
- RANBLK - Analyzes randomized complete block design with no replications.
- YOUDEN - Computes analysis of variance table and F-ratio treatments for a Youden square design.

Correlation Analysis

- CORMAT - Computes correlation matrix for N series of data.
- CORREL - Computes Pearson product-moment correlation coefficient.
- PATH - Interactive path analysis.
- SPEARMAN - Computes Spearman rank correlation coefficient of two groups of data occurring together in pairs.

Regression Analysis

- PLR - Uses Piecewise Linear Regression to estimate linear relationships between (a) one variable and time; or (b) two variables.
- REGPREP - Writes data into random access file for use in TUCKREG.
- SIMPREG - Performs simple linear regression analysis on N sets of paired observations.
- STAT-9 - Computes slope, means, Y-intercept, standard deviation, sum-of-squares, and F-ration for a linear regression.
- STEPREG - Performs one multiple linear regression according to Efroymson's algorithm.

- TRNSFORM - Performs seventeen different transformations of data in a file.
- TUCKREG - General program to compute one or more simple or multiple linear regressions and provide all relevant statistical measures.

Program No. 128

STANFORD BUSINESS SCHOOL PACKAGE

This package consists of 36 programs written in Extended BASIC. Although originally written for MBA students at Stanford, the routines should be useful in a variety of situations. RDOS with 32K and Extended BASIC is required. Below are brief descriptions of each program.

NAME	DESCRIPTION
GWBULL	Subjective probability-random values
GCPM1	Critical path analysis
GLPSA1	Linear programming two-phase simplex method
GSPMG	Stanford portfolio management game
GSSS	Small system simulator
VCHART	Investment decisions
GKCOST	Price/earnings ratio calculations
GKASSF	Warrant price calculation
GNMRVB/ GMRGB	Securities portfolio analysis and determination
GVOTE	Committee choice analysis
GTASPD	Selective probability distribution
GDPA	Efficient "corner" portfolios
GDAPI	Abnormal performance index
VRRC	Investment strategy analysis
GRGPLT	Simple regression and plot
GCPATH	Critical path analysis
GVPDQT	Plotting data
GTHOR	Securities EPS growth
GINTLP	Linear programming - variable restricted to one or zero
GSIMEQ	Simultaneous linear equations
GMCR01/ GMCR02	Fiscal policy game
GMCR05/ GMCR06	Economic policy game
GRANK	Ranking statistics
GANOVA	Analysis of variance
GLP	Linear programming
GNETFL	Network flow determination (Max/Min)
GFNRAT	Financial ratios
GRISKA	Risk analysis in capital investment
GSTKVL	Stock valuation
GIRRPV	Investment return (cash flow)
GCHLIN	Rating investment funds
B2AGPDQ	Utility program
B2AGSP5	Utility program

A manual is available as documentation that contains a detailed description of each program, the listing, and a sample run.

Program No. 046

DACBAP (Dana College Basic Algorithm Processor)

Dr. Paul G. Emmerich
Dana College
Blair, Nebraska 68008

This program simulates a single accumulator integer computer. It is especially useful for teaching beginning programmers some of the typical operations of a digital computer in a simplified, pseudo-assembly language. Following training in BASIC, it functions as a transition to a lower-level language. The program should be self-explanatory to a teacher who knows something about assembly language coding, e.g. use of instructions like load, store, etc.

12K of memory is necessary. The program was written to run on Time Share BASIC, but the author thinks it will run on Single User BASIC as well.

Program No. 076

CLASS GRADING PROGRAM

Michael S. Spritzer
Dept. of Chemistry
Villanova University
Villanova, PA 19085

This BASIC program will process class grades for up to 140 students. Input data includes student number, three exam scores and a final exam score. The weight of the final exam is determined by the user. A negative entry for any exam is not counted. Student averages are determined first by averaging the three exam scores and then weighting with the final exam. Provision is made for correction of entries before processing. The corrections list is terminated by a negative entry number. Two output tables are provided: 1) All grades and the average are listed in "gradebook" order, i.e., the same order as input. 2) Grades are listed in order of student number with each student assigned a rank in class. (This table is suitable for posting.) On both tables mean and standard deviation are also indicated for each exam and the average. To facilitate processing intermediate grades, a data tape may be punched. Subsequent scores may then be entered as corrections. The program will run on any Data General BASIC in 8K core.

Program No. 077

LAB GRADES

Michael S. Spritzer
Dept. of Chemistry
Villanova University
Villanova, PA 19085

This BASIC program will process laboratory grades for up to nine experiments; laboratory examinations are optional. Individual experiments may be weighted; examination weight is determined by the user. Input data includes student number, letter grades for up to nine experiments and exam average, both numerical score and letter grade. Negative entries are not counted. Provision is made for correction of entries before processing. The corrections list is terminated by a negative entry number. Letter grades are input using the following code: F:0, D:1, C-:2, C:3, C+:4, B-:5, B:6, B+:7, A-:8, A:9, A+:10. The output table includes student number, exam average (numerical score and letter grade), experiment average (letter grade), and course letter grade. Multiple copies may be output. Grades awarded: N (incomplete), D, C, C+, B, B+, A, A+. This program will run on any Data General BASIC in 8K of core.

Program No. 120

CO-PILOT

Aram Grayson
Consultant
510 Military Way
Palo Alto, CA 94306

CO-PILOT is a computer language designed for use by teachers writing computer assisted instruction (CAI) lessons. Some of its features include: selective branching to route students through different materials according to their responses, storage of student responses for later insertion in the lesson, and search for a selected word in a student response. No previous programming experience is required to develop and write the programs.

There are two separate routines: the CO-PILOT program itself, which runs the lesson, and the CO-PILOT Editor, in which the lessons are written, edited, and revised. A manual is also available that explains the CO-PILOT commands, the editing commands, and the run-time procedure.

GAMES

Program No. 027

BASIC FOOTBALL AND BASEBALL GAMES

This set of BASIC language programs may be run on Data General's Single User or Time Shared BASIC Interpreters. The set of two tapes includes one Baseball Game and one Football Game (as played by American rules). A Canadian Football Game may be substituted for the American Football Game upon request.

Program No. 058

BATNUM

This BASIC language program may be run on Data General's Single User or Multi User BASIC Interpreters. BATNUM is a number elimination game; see how many tries it takes you to guess the correct number.

Program No. 059

SLOT

This BASIC language program may be run on Data General's Single User or Multi User Interpreters. SLOT is a game that simulates the operation of a slot machine.

Program No. 060

KINGDOM

This game simulates a feudal society. The player is a land owner and must make yearly decisions concerning planting, sale of crops, and use of peasants. See how many years you can stay in power. Be careful, your peasants don't like to be starved, and your neighbors are jealous of your estate.

REV01 of this program is in binary format.

REV02 is a BASIC source tape.

Make sure to specify which REV you want when ordering.

Program No. 062

BLACKJACK

This BASIC language program may be run on Data General's Single User or Multi User BASIC Interpreters. This program simulates a Blackjack game, allowing players to bet.

Program No. 064

TURTLERACE

This BASIC language program may be run on Data General's Single User or Multi User BASIC Interpreters. Pick your favorite turtle, make a bet, then watch them run(?). TURTLERACE requires the use of a CRT for operation.

Program No. 065

LUNAR

This BASIC language program may be run on Data General's Single User or Multi User BASIC Interpreters. LUNAR is a simulation of a spaceship landing on the moon. You are the astronaut and must guide your craft to a safe landing.

Program No. 066

HORSERACE

This BASIC language program may be run on Data General's Single User or Multi User BASIC Interpreters. The program simulates a horserace. Make your bet, and then they're off! All your favorites: Man O'War, Seabiscuit, Citation, are there. This game requires the use of a CRT.

Program No. 067

HEMAN

This BASIC language program may be run on Data General's Single User or Multi User BASIC Interpreters. Can you make the bell ring and win a cigar? For this game it is the RETURN key, not a sledge hammer, that measures your strength. A CRT is necessary.

Program No. 072

2-PLAYER FOOTBALL

The game is played by Canadian rules in that there are only 3 downs, the field is 110 yards, and a touchback yields a single point. Operation of the game is self explanatory.

Hardware Requirements:

1) 2 Teletypes $TTO and $TTO1
2) 24K Memory
3) Real Time Clock
4) Moving Head Disk

Software Requirements:

1) RDOS (3181)
2) Fortran IV compiler and runtime libraries (3212, 3213)

Program No. 078

FORTRAN QUBIC

Fortran Qubic is a programmed three-dimensional game of TIC-TAC-TOE. The computer plays against the person at the console. The routines may be arranged to provide medium to very difficult competition.

Qubic consists of the main program (FQUB), the I/O routine (FQUF), and four playing routines. Selection of one of the four combinations produces a game of a certain difficulty. Below they are listed in order of increasing difficulty:

PLAYA, PLAYB1 (Moves by weighting factor without lookahead)

PLAYA, PLAYB

PLAYA1, PLAYB

PLAYA1, PLAYB (Uses tree search by look-ahead techniques for sequence of moves that can't lose)

The tree search is to a depth of 10 to a maximum of 1500 modes expanded.

The game requires 16K and RDOS or RTOS. It is written in Fortran IV.

Program No. 079

BASIC QUBIC

QUBIC is a three-dimensional game of TIC-TAC-TOE programmed in BASIC. The computer plays against the person at the console. The program does not look-ahead but does select moves based on weighting all currently occupied winning lines. The game presents a medium level of difficulty.

The program requires Extended BASIC for operation.

Program No. 086

BATTLE SHIPS

Charles Eleiott
Avondale Avenue
Hornell, New York

This BASIC language program may be run on Data General's Single User BASIC Interpreter. Battle Ships is a naval battle between 2 players. Each player has five different types of ships to choose from and a handicap system is provided for beginners. The ocean is a 60 x 50 rectangle with an island in the Northwest corner. Both players place their ships anywhere they wish. The plays are made up of one ship moving (throttle and direction) and the other ship taking 2 shots involving: gun direction, elevation and number of charges. A map is printed out after each set showing ship positions and shell landing. (The North-South position is given by a line number in the left column). The amount of fuel and number of charges per ship are limited. One direct hit or three near hits will sink the enemy.

Strategy and skill become determining factors with a little experience. Capsizing, colliding, grounding, and falling off the edge of the earth are all possibilities.

Program No. 087

STAR TREK

Star Trek is an advanced battle-simulation game based on the T.V. series of the same name.

At the computer console, you are captain of a starship of the Federation of Planets. Your ship is of the Enterprise-class; the largest, and most powerful ships in the galaxy. In the simulation, you will battle Klingons, Romulans, and a host of other "enemy" starships.

You have at your command phasers, photon torpedoes, matter-antimatter probes, navigational computers, auto-track radar, and an enormous reserve of power to operate these and other functions. All together, you can give any of 29 different commands.

There are more possibilities in the end of the game than mere win or lose. A total victory is where you have either completely destroyed the enemy, or better yet, forced him to surrender without killing his whole crew. (Romulans have never been known to surrender.) A tactical victory is one where you have out-run, out-maneuvered, or exhausted the fuel supply of the enemy. A moral (phyrric) victory is where you self-destruct rather than surrender; or, in blowing up, manage to take the enemy with you. A loss is when you are either destroyed, or surrender. (Romulans have never taken prisoners.)

This sophisticated Extended BASIC simulation requires in excess of 20K bytes of user space, a CRT, and Rev. 3.6, 3.7, or 3.8 Extended BASIC.

Program No. 096

NIM

Brian Gilbert
Leicester Polytechnic
School of Electronic & Electrical
Engineering
Leicester, England

This BASIC program may be run on any Data General computer with the Extended BASIC Interpreter. NIM is a game in which two players pick up sticks from piles on the ground. See if you can beat the computer by being the one to pick up the last stick in the game. A clever and persistent player should beat the computer fairly regularly.

Program No. 124

BIORHYTHMN

This program is based on the theory that human lives move in predictable undulations involving three separate cycles:

A) Physical 23 Days

B) Emotional 28 Days

C) Intellectual 33 Days

There are three major areas within the chart to be observed:

A) Minus

B) Zero

C) Plus

All you do is enter your name, date of birth, and the number of months you want plotted and out comes your biorhythmn chart.

The program is written in Fortran IV.

Program No. 129

NOVA RADIO MUSIC SYSTEM

This program plays music on a radio held near a NOVA computer by generating radio frequency interference with the memory switching currents. A demo tape is included with the interpreter and five songs:

1) Gentle on my Mind

2) Impossible Dream

3) Windy

4) Walking in the Sunshine

5) Penny Lane

An assembly parameter tape is also provided for the simple transliteration of other musical scores.

Program No. 015

PIN LIST SYSTEM

Bob Williams
W.W. Witt Associates
5827 Star Lane
Houston, Texas

The Pin List System is designed to allow documentation and implementation of logic systems which use wirewrap interconnections. The name and location of each pin is entered on line into the computer. A paper tape is then made of this source list. The source list is then re-entered to produce a sorted listing and a sorted paper tape, both sorted by name and location. The name-sort is used for wire-wrapping, and the location-sort is used for error checking and debugging. A system including 4K Nova-line computer is capable of accepting up to 400 individual points, with expansion being a function of core. The two supplied programs generate the pin list, sort, and punch tapes sorted by name and location respectively.

Program No. 019

WIRE LIST LAYOUT PROGRAM

The wire list generation system is designed to allow the user to quickly layout the wiring list for a group of printed circuit cards that have wirewrap interconnections. The existing system allows the wiring list for eight 86 pin cards to be generated. Each pin is described by six ASCII characters. The program has nine commands:

BLANKS - Blank the storage area.

ENTER - Enter data through the teletype.

READ - Read data from the high-speed reader.

CHANGE - Change data from the teletype.

DELETE - Delete a card entry from the table.

LIST - List all card data on the teletype.

LIST 1 - List one card data on the teletype.

PUNCH - Punch data on the high-speed punch.

WIRE - Layout an optimized wiring diagram.

This program operates in 4096 words of core memory using all locations up to 7630(8).

Program No. 024

CONDUCTOR SELECTION PROGRAM FOR LOW VOLTAGE THREE PHASE SYSTEMS

W.L. Bacica
Bovay Engineers, Inc.
Houston, Texas

This BASIC language program selects either copper conductor or copper feeder duct busway to satisfy the data supplied. Data consists of the following:

L: Three phase volt-amp load

V: Line to line voltage

F: Load power factor in decimal form

B: Circuit length in feet

M: Maximum tolerable voltage drop in percent

Two steps are involved in selecting the proper conductor. The first involves a calculation of the load current and an initial selection of a conductor to satisfy this ampacity. Secondly, the conductor is checked to satisfy the maximum tolerable voltage drop. If the voltage drop is excessive, the program finds the first larger size that satisfies the drop requirement. This conductor (or busway) is the desired result of the program.

Program No. 037

TAD (Tape to/from analog conversion program)

Merle G. Hooten
U.S. Army Electronics Command
AMSEL-NL-V-4
Ft. Monmouth, N.J. 07703

TAD is an analog (A/D converter) to-tape and tape-to-analog (D/A converter) program. The program accepts a 12-bit word from an A/D converter and reformats the data to be stored on a 7-track magnetic tape deck. The program also reads the stored data on tape and outputs it in the correct format to a 12-bit D/A converter. The interrupt system is used to allow the transfers. An external

clock is supplied to the A/D converter to provide the start-to-convert pulse.

TAD is an assembly language program requiring 4K of memory. Necessary peripherals include: A/D, D/A, 7 track mag tape, teletype, and an external clock for A/D converter.

Program No. 113

D-A CONVERTER PLOTTING SUBROUTINES

Walter Wahnsiedler
Northwestern University
Evanston, IL 60626

These subroutines are designed to provide the user with a flexible plotting capability using a two-channel digital to analog converter system and a two-dimensional voltage recording device, such as an oscilloscope or X-Y recorder. Routines are provided for the drawing of a labelled axis, output of character strings and real numbers, scaling sets of numbers, output of single points, and output of lines connecting points. Communication between the computer and the external interface is left to the user to program, thus making the subroutines quite general in nature. As supplied, these routines will operate any recording device which has a recording mode which does not need repeated renewal, i.e., these routines will operate a storage oscilloscope or a conventional pen control circuit, but will not operate an X-Y recorder equipped with an event marker, or an oscilloscope without the storage option.

Routines are provided to be used with programs written in DGC FORTRAN IV or DGC Extended BASIC. The FORTRAN routines are contained in a library tape set, and the BASIC routines are supplied as source code to be edited into the user's program or ENTERed along with the user's program. The axis-drawing subroutine is not implemented in BASIC.

Program No. 125

SOS EXTENDED BASIC WITH RTC AND DIGITAL I/O

David Hale and Tony Payne
Queen's Univ. of Belfast
Psychology Dept.
No. 1 Lennoxvale
Belfast, N. Ireland

This is an assembler subroutine to give CALL statements in SOS (Rev.03) Single User Extended BASIC for real time functions of clocking and digital I/O. Four interrupt driven software clocks provide timing to an accuracy of 10 milliseconds. Each clock can be independently zeroed or have its contents read into the user program. Digital I/O is handled on a pair of Type 4066 boards with access at the whole word or individual bit level. In addition, the console data switches and the output registers can be used for sensing input. The program is provided as a source tape that is assembled and loaded to produce a new working version of SOS BASIC. It is a convenient vehicle for teaching process control or running a variety of experiments.

Program No. 002 (LP)

BASIC INTERPRETER SYSTEM WITH CALL

This modified version of the DGC Single User BASIC Program (091-000018) features a CALL statement which may be used for calling assembly-language statements from a BASIC program.

The user's subroutine is preassembled and loaded utilizing the specified subroutine format. The modified BASIC program is then loaded, with data switch 1 set up while the tape is loading---this tells the program that a user's subroutine is in core and location 10 contains a pointer to the S/R starting address. The CALL statement is of the form

```
      100 CALL S/R#, P1, P2, P3
E.G.  210 CALL 2,A (2), B
      Storage:  6K or more
```

Program No. 003

TELETYPE TAPE EDIT

Dr. R.A. Koster
Narisco
Anaheim, California

This is a paper tape editing program which can co-exist in a 4K system with the Paper Tape Assembler (DGC 091-000002) by residing in the symbol table space. Without affecting the assembler, a paper tape may be edited on the teletype by loading the original tape in the teletype reader, starting the tape edit program, and turning on the teletype punch. With data switch 15 up, a single character is copied only when any one of the other 15 data switches is inverted. Thus, toggling a data switch copies at a controlled rate to permit omission or insertion of characters.

Storage: 14_{10} words

Program No. 004

BEAP EDITOR/ASSEMBLER PROGRAM

Robert C. Baskin

BEAP is a combination editor and assembler program with primary applications on systems having only a teletype for I/O. BEAP's 10 keyboard commands enable the programmer to enter his source program into a storage buffer from either the teletype keyboard or reader, examine, modify, output, and assemble the source program in the buffer any number of times without reloading BEAP or the source program.

Programs may be written and debugged in locations 0 through 77 and 7100 thru 7600 without affecting BEAP, the storage buffer, or the binary and bootstrap loaders. BEAP requires approximately 1500_{10} words of storage and has a buffer capacity of 3700 characters.

Program No. 008 (LP)

RELOCATABLE EXTENDED FLOATING POINT INTERPRETER

J.D. Wright
McMaster University
Hamilton, Ontario
Canada

This version of the Extended Floating Point Interpreter (DGC 091-000013) is in standard relocatable binary format for use as a subroutine to be read in by the Relocatable Loader (DGC 091-000016). This version of the Interpreter will operate with the DGC Disk Operating System if inserted into the SYS.LB File. Absolute locations 4, 5, 6 and 7 have been replaced by page zero relocatable entries called LOC4, LOC5, LOC6, LOC7. Thus FETR and FINI must now become for example, JSR @ LOC4 and JSR @ LOC5 respectively. The user simply provides pointers to GET and PUT character routines in locations 40 and 41 respectively.

Program No. 009 (LP)

A CARD READER VERSION OF THE NOVA RELOCATABLE ASSEMBLER

Joe Sukonik
Calma Corporation
Sunnyvale, California

This version of the NOVA Relocatable Assembler (DGC 091-000017) contains an additional input assignment allowing the source program to be input from the card reader. On input, card data is translated from IBM 029 format to ASCII. Absolute location 6571 can be set to the number of columns to be ignored at the end of the card---this is preset at 20. This allows the end of the card to carry sequence numbers, or other identifiers, without causing the assembler to flag them as errors. Minimum configuration of 8K memory is required.

Program No. 010

PUNCH CARD TO TAPE CONVERTER

Joe Sukonik
Calma Corporation
Sunnyvale, California

This program converts cards punched in IBM 029 format to ASCII formatted punched paper tape. If data switch 0 is up, output is directed to the high speed punch. Absolute locations 15134, 15135, and 15136 may be initialized to contain the following information, respectively: number of columns of the card to be converted; number of columns at the beginning of the card to be ignored; number of lines per page---between Form Feed punches. Version supplied resides in the upper 4K module of an 8K system configuration (S.A. -- 15000).

Program No. 011

NOVA MAGNETIC TAPE INPUT/OUTPUT SYSTEM

George U. Ramos
GTE Sylvania Inc.
Mountain View, California

This system program enables core images to be stored on, and retrieved from, magnetic tape by simple teletype commands. Communication with the system is by filename (operator generated). Tape searches are done for the filename, and files retrieved are loaded into core with termination options of HALT or automatic program start (analogous to start - block features of the DGC Binary Loader). The system is also stored on magnetic tape and is "booted" into core by a special bootstrap program which replaces the DGC Binary Loader in core and has a S.A. of X7777. Minimum system configuration is 8K of memory, teletype, and one 9-track magnetic tape unit.

Program No. 012

HOLLERITH TO ASCII CONVERSION SUBROUTINE

David L. Wasley
University of California
Berkeley, California

This assembly language subroutine, with a calling sequence of JSR HOLL, converts a 12-bit Hollerith code in ACØ to its equivalent 8-bit, even parity ASCII code. The ASCII code is returned in ACØ and all other accumulators are modified by the subroutine. The Hollerith code is assumed to be a card image, right justified, with the "12" punch in bit position4. The 64_{10} work subroutine is self-contained and may be assembled anywhere in core.

Program No. 016 (LP)

QUICK LOAD LOADER

W.A. Foley
R.F. Halley
Astrophysics Research Corporation
Norco, California

This routine is a modification of Data General's "Self-Loading Bootstrap Loader" (Tape 081-000001-00), so that the bootstrap does not HALT after the loader is loaded, but rather branches to the loader and begins execution. Thus, if this tape is copied onto the beginning of an absolute binary tape, any binary tape may be loaded using the hardware bootstrap feature of the NOVA computers.

Program No. 017

INPUT/OUTPUT PACKAGE

Dr. Charles S. Cox
University of California
San Diego, California

This is a package of ten subroutines to facilitate I/O operations with the

teletype. Subroutines are provided for:

1) Character input from TTY
2) Character output to TTY
3) Generation of N spaces
4) Generation of CR and LF
5) Output of character string into storage
6) Reads a character string from TTY into storage
7) Reading of a signed, single or double precision, octal or decimal numeral string from TTY and conversion into binary
8) Branching on reception of a break character
9) Conversion of a binary number into signed or unsigned, decimal or octal, numeral string and printing on TTY
10) Formatting of numbers generated by routine #9. These subroutines reside in 742_8 memory locations.

Program No. 020 (LP)

OVERLAY EDITOR-ASSEMBLER SYSTEM

J.E. Pierce
Southwestern Steel Rolling Door Company
Dallas, Texas

An executive routine, a modified Data General editor, and a modified Data General absolute assembler make up this overlay system. The editor stores a text buffer in core and the assembler will assemble the contents of this text buffer. Routines have been added to the editor which facilitate source compression (by deleting comments) if core storage is not adequate. In between overlay operations, other programs may be run in low core (at least up to address 4463) and the text buffer may still be recovered. Core above address 4463 is divided into the assembler user symbol table and the text buffer. The system may be run with a teletype and 4K of memory, but a high speed paper tape reader and at least 8K of memory allow convenient operation.

Program No. 021 (LP)

RESUB

S.M. Heider
State University of New York at Buffalo
Buffalo, New York

RESUB, Relocatable Extended Single User Basic is a BASIC Interpreter System with features to allow concurrent operation with background programs. RESUB is a modified version of Data General's Time Share Basic. The following features should be useful in several single user applications:

1) Tape, Card, Dump, and Punch commands. These allow loading and dumping programs with user supplied routines.
2) Subroutine Call - RESUB'S SUBROUTINE call may be used as a console command. It is also possible to pass a string variable to a subroutine.
3) Stack processor for handling nested interrupts.
4) Provisions for handling background programs when BASIC is not running or at bottom of interrupt stack.
5) Subroutines may be initialized at start up.
6) Location 20 contains address of first unused location in upper core.
7) Entry points are supplied for setting Run Time accumulator, substituting line printer for teletype, etc.
8) No location below 20 is used until RESUB is started. Consequently, it should be possible to load under DOS.

RESUB is about $14{,}000_8$ words long. About 20 words in page zero are available.

The minimum configuration required for using RESUB is an 8K NOVA computer built by Data General with a standard teletype interface. It is advisable to have at least a 12K core.

RESUB will function as a stand alone program. Externals are checked before they are used.

Program No. 023 (LP)

SINGLE USER BASIC WITH MATRICES AND STRINGS

This version of BASIC is a modification of Data General's Time Sharing BASIC as described in the pamphlet called NOVA Line Time Sharing BASIC. The communications multiplexor handler has been removed and replaced with a handler that operates the console teletype only.

The Time Sharing Basic pamphlet should be consulted for the operation of this version of BASIC. It gives the

operator all the program control commands described on page 3 of the pamphlet. An additional command (control P) can be used instead of a carriage return to have a listing made to the line printer, i.e., LIST P, or the results could be put to the line printer, i.e., RUN P.

Program No. 028 (LP)

MACHINE LANGUAGE I/O SUBROUTINES FOR T.S. BASIC

R.M. Thomas
GTE Automatic Electric (Canada) ltd.
Brockville, Ontario
Canada

These subroutines allow TIME-SHARED BASIC users to manipulate data in binary or other arbitrary formats such as EIA, and to read and punch character string data using the high-speed reader and punch or a teletype reader/punch.

Manipulation of binary or other data is achieved by unpacking 8-bit characters into equivalent strings of ASCII 1's and 0's which can be manipulated in BASIC. Results can be repacked for compact storage or output.

The machine-language subroutines are accessed using CALL statements in BASIC. A version of TIME-SHARED BASIC incorporating these subroutines is available; it occupies about 150 extra words of core, but is otherwise identical to normal TIME-SHARED BASIC.

The supplied program includes the modified DGC Time-Shared BASIC (091-000026) with the machine language subroutines.

Program No. 031

NOVA OPERATING SYSTEM PROGRAM

Timothy Mulrooney, Sr.
Naval Underwater Systems Center
Newport, Rhode Island

This program, written for the Data General NOVA series of minicomputers, is used along with other user programs to list, enter, or fill core, and to read, punch, copy, or verify paper tapes. It can also jump completely out of itself to a user program somewhere else in core. The eight available commands provide for specified core dump, core modification (individual and block), core to binary format punch, tape copy and verify, core compare, and program start-up. Program resides in the 1K below Binary Loader. Minimum configuration is CPU with 4K of memory, Teletype ASR, and/or high speed paper tape reader. An 8K system is provided; changing 3 instructions in the source tape configures the program to any core size.

Program No. 033

EMUL-8

Eb Wulff
68 Lucinda Avenue
Wahroonga 2076, Australia

This emulator of a PDP-8/S/L/I computer runs on any NOVA with 8K of memory (min. 5K). Most instructions are emulated in 20-25 US. on a SUPERNOVA computer. The Emulator is organized so that the operator can carry out all control functions required for a PDP-8 from the NOVA console. In particular, it is possible to stop a PDP-8 program.

Allowance has been made for the special ASR 33 Teletypes used on PDP-8 computers. The address space is 4K 12 bit words.

This version was modified extensively to adopt fast routines used in SIMUL-8, while retaining the machine-like operation and the fast interrupt scheme of earlier EMUL-8's.

Program No. 035

CORE COMPARE AND LOAD

Ed Wulff
68 Lucinda Avenue
Wahroonga 2076, Australia

This is a program which compares absolute binary paper tapes with the contents in core locations. Any differences are printed out:

address core value tape value

This version also corrects any differences; in other words, the core values are restored to the values on tape.

Program No. 036

BINARY TAPE MAPPER

Eb Wulff
68 Lucinda Avenue
Wahroonga 2076, Australia

The Binary Tape Mapper will print the starting and finishing addresses of all non-contiguous blocks on the tape. The end block address is also printed.

Note: Only tapes without "fill" blocks are read by this program. (Compatible with Vers. 1 of Loader)

Program No. 038

BINARY TAPE MERGER

Eb Wulff
68 Lucinda Avenue
Wahroonga 2076, Australia

The Binary Tape Merger Program allows the merging of a number of absolute binary tapes into one composite absolute binary tape. The output tape is punched in increasing address order and contains data for all those addresses defined in the input tapes.

Program No. 039

SYSTEM DIRECTORY SORTER (DSORT)

Bill McAndrew
Automatic Electronic Systems Inc.
5455 Pare Street
Montreal, Quebec
Canada

This program is used to sort the listing of the system directory (SYS.DR) into alphabetical order. This allows for easy reference to all files stored on disk.

The program calculates the file attributes; an octal word count for each file, and formats the output onto the lineprinter.

This NOVA language program may be run on all DOS driven NOVA computers where the system includes a lineprinter. Available core must be large enough to hold at least ½ of SYS.DR file.

Program No. 041 (LP)

COMBO

Stephen Heider
Department of Physics
State University of New York at Buffalo
Buffalo, NY 14214

COMBO facilitates editing and assembly of programs on NOVA systems with at least 8K of core. This program contains modified versions of the Extended Assembler and Editor programs. Additional Editor commands have been incorporated to control the Assembler as well as I/O device selection. Assembler input may be either from the Edit Buffer or from paper

tape. The user determines the symbol table size at startup. Additional features make it possible to punch library tapes and to obtain partial listings.

Minimum peripheral required is Teletype, but COMBO will support high-speed paper tape reader and punch as well as lineprinter.

Program No. 042

FAST SPECIAL FORMAT FLOATING POINT SUBROUTINES

Thomas Schulz
Institut fur Angewandte Physik
Universitat
23 Kiel
Ohlshausenstrabe

These nine subroutines were developed for fast real time applications. They include:

·FFLT	Fix to floating convert
·FFIX	Floating to fix convert
·FMOV	Move FLPT number from one location to another
·FSR	Store FLPT number (out of register) relative to AC2
·FLRS	Load FLPT register relative to AC2 and skip if FLPT number is zero
·FADD	Addition of two floating point numbers
·FSUB	FLPT subtraction
·FMUL	FLPT multiplication
·FDIV	FLPT division

The program uses 713_8 locations and requires hardware multiply/divide. A short Basic program is included for putting constant floating point data into an assembler source program to calculate the mantissa and exponent out of a given value.

Program No. 043

INCREMENTAL PLOTTER SUBROUTINES FOR BASIC

Christer Berg
Oy Stromberg Ab
P.O. Box 118
00101 Helsinki, Finland

This program extends BASIC with plotting capabilities by use of the CALL statement. Six calls are available: program initialization, XY-movements both absolute relative, pen up/down, ASCII character drawing, and coordinate inquiry. Program loading is described in the Extended Basic Manual.

Two sets of tapes are available with this program:

REV 05 - for use with Single User Basic with Call (User's Group Program #002)

REV 06 - for Extended Basic and Time Share Basic

When ordering, please specify which REV you want.

Program No. 048

MEMORY TEST PROGRAM

Bruce K. Ray
Polymorphic Computer Systems
P.O. Box 3581
Boulder, Colorado 80303

The Memory Test Program provides a comprehensive test of every location, excluding that required by the program, in a Data General read/write memory system. Every location is tested by using every possible bit pattern in that location. Under switch register control one has the option of using only a rotating bit pattern for each location, as well as the option of continuous cycling through the program. Errors are indicated by a halt, with the accumulators containing pertinent error information.

Any Data General processor with read/write memory is sufficient for operation.

Program No. 049

ASSEMBLER SOURCE PROGRAM ANALYZER

Bruce K. Ray
Polymorphic Computer Systems
P.O. Box 3581
Boulder, Colorado 80303

This ALGOL program scans an assembler source program and tabulates the number of instructions of each NOVA instruction type used. Group percentage usage as well as actual instruction type count is given as output.

Operating Instructions - Load the ASCII tape and compile the program according to the supporting operating systems procedures. After the object is loaded and started, the filename of the assembler program to be analyzed, including any ex-

tentions, is requested. After the program analyzes the source file, the statistics are printed.

Any Algol supportable memory is sufficient for program operation, and it will run under DOS, RDOS, or SOS.

Program No. 051

BASIC RENUMBERER

Walter E. Wahnsiedler
Northwestern University
Materials Science Department
Evanston, Illinois

This assembly language program will renumber BASIC source code with evenly spaced line numbers; the starting number and spacing is at the User's option. A new input tape is created for BASIC, permitting easier editing of programs under development and producing better appearing finished programs. References in the code (GOSUB, GOTO, THEN) are updated as the tape is punched.

SOS REV 08 is necessary to recompile. Teletype is sufficient, but program supports high speed reader and punch.

Program No. 054

MINI EDITOR II

Eb Wulff
68 Lucinda Avenue
Wahroonga 2076, Australia

The Mini Editor can be used to edit paper tapes on a teletype with reader and punch. It requires only 100 octal words of memory and can thus be used while the paper tape assembler is in memory in a 4K NOVA.

Program No. 055

FLOATING POINT SORT

This is a callable, relocatable overlay-able subroutine written in Fortran. Its function is to take a tag file loaded onto any disk by the User and arrange that tag file in either ascending or decending order. The ordered information is left on the input file.

User must provide the mainline program.

Program No. 068

ERROR

This module contains the main system error reporting routines. The errors are considered either fatal or non-fatal, and may involve a filename or not. If a fatal entry is used, a '.RTN' system call is issued after the error message is printed. A non-fatal error entry returns to the location following the error call, after the error message is issued. If a filename is assumed, the byte pointer must be in AC0 on entry to the fatal or non-fatal file error routine. The error module entry points are:

1) ERR1 - Fatal file entry point.
2) ERR2 - Non-fatal file entry point.
   ERR3 - Fatal non-file entry point.
4) ERR4 - Non-fatal non-file entry point.

The error routines are intended to be called in the event of a system call error return. Example:

```
LDA 0, TEXTP     Try to open channel zero.
SUBO 1,1

.SYSTEM          Big system call...
. OPEN0

JSR@.ERR1        Assumed to be a fatal
  ....           error involving a file
                 name.

.EXTN ERR1
.ERR1:ERR1       Fatal file error is
                 declared external normal.
```

Supporting routines in this module are:

OUT - Output a single byte to the current console.

PRINT - Output text to the console (JSR+1=byte pointer).

PRNT1 - Print text to the console (AC1=byte pointer).

OCTAL - Output a five digit value (AC1=value) to console.

Each of the programs has been declared in an entry statement so that other programs may utilize them. All segments of the module are completely self-contained and do not call on any external application routines.

Program No. 071

NOVA ACCOUNTING SYSTEM

John Karch & Walter Wahnsiedler
Northwestern University
Materials Research Center
Evanston, Illinois

This accounting system consists of a pair of programs capable of updating and editing a log file kept on a disk attached to a NOVA computer with at least 16K of core and RDOS. The system is primarily designed for machines operating in single-user mode, but is compatible with Data General Batch and Time Share Basic. Each user is known by name and four digit number, and must specify these correctly to gain access to the CLI. At the end of each session, the user calls the log program, which computes his total time used and then places the machine in an idle state. The source code for the programs is in Fortran IV, and locations for commonly made alternations have been marked with comments. The contents of the log file can be displayed at the console at any time by logging on with a special user name.

Program No. 080

PRINTX

PRINTX is a utility program which accomplishes the function of the RDOS "PRINT" command and, in addition, labels each printout with a header block containing the following:

1) Name of the file being printed
2) Current date
3) Current time
4) Current disk directory
5) Disk Pack identification

This is a convenient way of keeping track of the disk pack and directory in which a certain file resides.

PRINTX is used in the same way as the RDOS "PRINT" command, and runs under RDOS Rev. 3 in either foreground or background.

Program No. 082

GENERAL PURPOSE FORTRAN MACROGENERATOR

The general purpose macrogenerator is a string processor in that it takes as its input a character stream and produces a character stream as output. The output stream contains strings directly copied from the input, and strings produced as the result of macro calls.

This macro processor will process arbitrary text strings. They need not be legal input to the Data General Macro Assembler. Features include recursive calls and definition and redefinition of macros within macros. One example of the program's usefulness would be for generating tables and test data during program debug.

Program No. 083

S-DEBUG

Diplomphysiker Jurgen Rathlev
Institut fur Angewandte Physik der
Universitat Kiel
23 Kiel
Ohlshausenstr. 40-60 West Germany

The Simulating Debugger (S-Debug) is a program to debug NOVA stand alone assembler programs. Because every instruction of the user's program is interpreted by the S-Debug program rather than by the NOVA hardware, functions like "instruction step" and "stop" can be performed from the terminal. It is also possible to print a run-time listing of the user's program and to change the interrupt status from the terminal. Furthermore, most of the features of the standard NOVA debugger II are available (e.g. four breakpoints, four program counters, binary punch option). Because S-Debug is self protecting, it cannot be destroyed by faulty user programs.

Moreover, there are two versions (S-Debug 2 and 3) which allow symbolic debugging of stand-alone programs. S-Debug 3 even accepts assembler mnemonics instead of arithmetic expressions with symbols and octal numbers. The required ASCII-paper-tape containing the user symbols is generated by the standard assembler using mode 5. S-Debug is located at the upper end of memory and needs no "page-zero" locations. There is no special hardware equipment necessary, but it is advantageous to have a high-speed paper tape reader and punch.

Two versions are available:

REV01 -- 8K
REV02 -- 16K

Program No. 085

CASSETTE/MAG TAPE REVERSIBLE READ SUB-ROUTINES

Walter E. Wahnsiedler
Dept. of Materials Science
Northwestern University
2145 Sheridan Road
Evanston, IL 60201

These are a set of Fortran-callable subroutines which will enable a calling program to read a cassette, mag tape, or disc file in either direction in binary. The calling program must open the file, and provide a small portion of code which will rewind the file when returned to. Paper tape synchronization facilities are also included (sync flag is two completely punched frames) and the subroutines can even be used to "reversibly" read paper tape if the operator can re-load the tape each time a "rewind" is required. End of data can be sensed optionally by the detection of a zero datum word, or, of course, by detection of end-of-file.

Minimum configuration is a NOVA-line CPU supporting RDOS or SOS FORTRAN with appropriate peripherals to process the reversible file being read. Memory requirements will vary with size of calling program.

Program No. 088

RTOS SNAPSHOT DUMP

Snap is a snapshot dump that can be run with RTOS Rev. 5 or the current RTOS Rev. 3. Snap outputs the contents of the accumulators, the carry bit, important page zero information (current TCB address, system mode indicator, current system interrupt mask, user stack pointer), all TCB queues with each task's priority, status, system call word, and identifier, and then returns to the caller.

Interrupts are disabled upon entry to snap and are not enabled until control is passed back to the calling program. Snap does not restore the contents of the accumulators to their values upon entry to snap.

To use snap it must be included as an external normal (.EXTN SNAP), redefined with some variable (.SNAP: SNAP), and then perform a subroutine jump to it (JSR @.SNAP). The relocatable binary SNAP.RB is then included in the command line of the relocatable load.

NOTE: Snap must be assembled with PARR.SR (ASM PARR/S SNAP)

Snap is written in Assembly language.

Program No. 089

TASK - S SIMPLE MULTI-PROGRAMMING SUPERVISOR

Steve Geller
Geophysical Institute
University of Alaska
College, Alaska 99701

Any number of parallel-running tasks may be defined and run under the supervision of the TASK supervisory module. The design objective was to allow programs to be written using the basic I/O facilities of the NOVA (DOAS, NIOC instructions etc.) in a shared environment with other I/O driving tasks. Control is given to the task of highest priority that is not waiting for an event. No time-slicing is used; the interval timer is not required. No device drivers are directly supported, since it is expected that the tasks which run with the TASK module will in fact be doing direct I/O. The TASK module traps interrupts, and enables any tasks which are waiting on the interrupt. A POST facility allows the use of pseudo-interrupts. One may use this to simulate an interrupt on any "event", such as a task completion.

TASK by itself requires 243_8 NREL locations and 5 ZREL locations. It also uses auto-incrementing location 20. It is written in Assembly language.

Program No. 090

BASIC PRINTING UTILITIES

This package of routines allows the Extended BASIC user considerable flexibility in listing ASCII files. PPT.SR and its two associated programs, PRTXREF.SR and PRTMAP.SR, are written in Extended BASIC to allow a user to:

1) List ASCII source and data files to any device or to the user's console with optional page headings and/or line numbers.

2) Copy sequentially organized ASCII files.

3) Obtain statement cross-reference and/ or variable map listings for a BASIC source program file.

These programs require both RDOS and Extended BASIC Rev. 03 for operation.

Program No. 091

STRIP (STRING PROCESSOR)

'STRIP' (String Processor) is a text-processing system for the NOVA family computers. In the 'STRIP' language, one can write procedures for accepting, naming and storing any character string from the input device; for modifying any string in any way; for treating any string at any time as an executable procedure, or as a name, or as text; and for outputting any string. The 'STRIP' language is based upon an extension and generalization to character strings of the programming concept of the "Macro". Through the ability of 'STRIP' to accept and store definitions of procedures, the capabilities of the language can be indefinitely extended. 'STRIP' can handle iterative and recursive procedures and can deal with character strings, integers, and boolean vector variables.

'STRIP' is written in Assembly language and requires a minimum RDOS configuration for operation.

Program No. 097

MUMPS

James Cruce

MUMPS is an interactive language and operating system that is very useful in dealing with large textual data bases. String handling is easy and efficient with the various string manipulation language elements. The hierarchial array and file structure provides easy organization of data. The language also has basic arithmetic capability.

This version provides the user with a basic MUMPS system. It includes almost all MUMPS language features and basic integer arithmetic capability. The system will run stand-alone and can provide a multi-user environment. 12K of core is necessary for a single user system; an increase in the number of users is a function of core available.

For simple systems with CPU, fixed head disk and teletype or CRT, very little tailoring is necessary. However, for larger and more complex systems, some software/hardware tailoring may be required.

Program No. 098

FILEDUMP

Dr. Hardy J. Pottinger
University of Missouri
Rolla, Missouri

FILEDUMP is an RDOS Rev. 02 program which is callable from the CLI and is used to produce an octal dump of an RDOS disk file. The general format of the CLI command used to call FILEDUMP is FDUMP filename. Input is from a file called "filename", output is to a file called SYSOUT. In most cases, it will be desirable to link SYSOUT to $LPT since FDUMP produces quite a lot of output.

The octal dump produces an output in the following format:

$$loc_i \ (loc_i) \ \ldots \ (loc_{i+7}) \ /C_1 \ \ldots \ C_{16}/$$

where loc_i is the starting word location of the 8 word line and (loc_i) is the contents of the location. If C_i is a printable byte it will be printed, otherwise a "." will be output. The parity bit is ignored for printing purposes.

An optional description dump may also be printed. The descriptive dump decodes and prints the User Status Table, active TCB chain, and Overlay Node Table in an easy to read format. The descriptive dump may be printed alone or with the octal dump.

The dumped file may be any file capable of being read sequentially by RDOS, and may start at either 0 or 16. Save files normally start at location 16, data files start at "location" 0. The descriptive dump would of course be meaningless for data files and would not be used.

Program No. 100

RDOS/FORTRAN IV I/O SUBROUTINES

This package contains six subroutines that allow Fortran IV programs to use RDOS character, sequential binary and ASCII line modes of I/O without the requirement of using the Fortran formatted I/O package. The subroutines complete the Fortran IV interface to RDOS that previously only included record and block modes of disk I/O and free formatted mag tape I/O. The routines included are: character I/O, sequential I/O and line I/O subroutines, and they are written in Assembly language.

Program No. 101

CHARACTER PROCESSING ROUTINES

This package of Fortran routines consists of the following:

INDEX - will search the specified elements of an integer array for a particular word. If the word is not found, the function returns a value of zero (0). If the word is found, the function returns the location of the word as its value.

IFIND - will search the specified elements of an integer array for a particular string. If the string is not found, the function returns a value of zero (0). If the string is found, the function returns the location of the first position of the string in the integer array as its value.

RFIELD - will right-justify (with leading blanks) characters in a field. Imbedded blanks will be retained in the field.

IBYTE - will pick-up a byte from a string array.

The routines are an extension to the Data General Commercial Subroutine Package, and require the Fortran Runtime Library for operation.

Program No. 102

RTOS/RDOS SYSTEM UTILIZATION TASK

The System Utilization task "UTIL" allows the user to monitor the amount of CPU time not used by the application program and the operating system. It operates with either RTOS or RDOS and utilizes the multi-tasking capability of the systems. Once a minute the task "UTIL" will output, to a user specified file or device, a message of the following format:

MM:SS idle time = HH.TT%

where MM:SS is the current time in minutes and seconds
HH.TT is the percentage of CPU time that is not being used by the operating systems or user application tasks.

The program is written in Assembly language.

Program No. 103

DTECT

This routine can be used to monitor the line activity on all unopened QTY lines. If QTY;64 is opened on the supplied channel and this routine is called, it will return after sensing activity on any unopened lines. Upon return, "Line" will contain the line number where the activity took place, and "Noise" will contain the character that was entered.

DTECT is written in Assembly language and requires the Fortran IV Runtime Library.

Program No. 104

CALENDAR

This Fortran routine will get the time of day and the date and format them into ASCII strings for output. Example:

```
10:10 AM     Wednesday     June 26, 1974
```

The Fortran Runtime Library is necessary for execution.

Program No. 105

F1130 - IBM 1130 FORTRAN PROGRAM CONVERTER

This program converts IBM 1130 Fortran programs (source level) to DGC Fortran source code.

The program performs the following checks and conversions:

a) Converts (Label) Read (CHL, RECNO) List
   to (Label) Call FSEEK (CHL, RECNO)
   Read Binary (CHL) List

b) Converts (Label) Write (CHL, RECNO) List
   to (Label) Call FSEEK (CHL, RECNO)
   Write Binary (CHL) List

c) Converts (Label) Find (CHL, RECNO)
   to (Label) Call FSEEK (CHL, RECNO)

d) Does the following subroutine calling sequence changes:

```
Call ADD  (...)  To  Call QADD  (...)
Call SUB  (...)  To  Call QSUB  (...)
Call MPY  (...)  To  Call QMPY  (...)
Call DIV  (...)  To  Call QDIV  (...)
Call MOVE (...)  To  Call QMOVE (...)
Call READ (...)  To  Call QREAD (...)
```

e) Converts (Label) Call LINK (Name)
   to (Label) Call CHAIN ("NAME.SV", IER)
   X Type 'CHAIN ERROR =', IER
   X Stop

f) Puts all variables initialized by a data statement in labelled common via the following statement "COMMON /LBCOM/ ... /

g) Places comments at appropriate places in the program text to make the program more readable.

h) Optionally converts all Ø26 keypunch codes to Ø29 codes.

```
& =  +
% =  (
15392 =  )
# =  =
@ =  '
```

i) Converts the following numeric code representations:

```
5440   =   3360
16448  =   8224
24896  =   12064
11552  =   24640
```

j) Converts (Label) Call P1403 (...)
   (Label) Call PRINT (...)

k) Converts (Label) Call S1403 (...)
   Call S1403 (...)
   (Label) Continue

l) Does not insure that variables equivalenced together exist in common.

Program No. 106

DISASSEMBLER

Ian Kettleborough
Texas A&M University
College Station, TX 77843

The Disassembler is a 2 pass program that reads in an absolute binary file and produces the appropriate assembly language. The program will produce symbolic references (in the form AXXX, where XXX is a 4 digit octal number) for all memory reference instructions. This requires that the binary file be read twice.

At assembly time, one must specify if the program is to be run standalone, under DOS, or under RDOS. With standalone mode, output is a listing on the line printer and an optional source tape in the teletype punch. Input is from the high speed paper tape reader. (A user written print routine is necessary to accept a character and print it in this mode).

For DOS or RDOS, input may be from any file, and output is either a listing or a source file (but not both), depending on whether the /A global switch is present. Either the source file from

the standalone or operating system version may then be input to the assembler.

Output consists of the instruction in Assembly language, the instruction decoded to 2 ASCII characters (if printable ones) and for DOS or RDOS the appropriate .SYSTM call word if it could be interpreted as such.

Several unfamiliar instructions such as NIO 2, LPT are produced but the appropriate .DIOA instructions are appended to the source file to handle these instructions when they are input to the assembler. The Disassembler requires about 3700 (octal) core locations and sufficient core storage for the number of memory reference instructions that reference different addresses. (A minimum of 4K is suggested for the standalone version.)

The program is written in Assembly language.

Program No. 107

MCA BINARY LOADER

Ian Kettleborough
Texas A&M University
College Station, TX 77843

The MCA loader allows an absolute binary program to be loaded via the MCA (Multiprocessor Communications Adapter #4038). One computer must be running with unmapped RDOS. The program will load any RDOS file (disk or paper tape.) The receive routine is a modified binary loader which when loaded replaces the standard loader at the top end of memory. The modifield binary loader will either accept input from the high speed reader or from the MCA. This program is about 150 words larger than the standard binary loader, and utility programs such as the assembler and BASIC may overwrite a section of this loader necessitating the reloading of it before it can be used again. Once the receive routine is ready (in execution) the binary file can then be loaded via the MCA.

Note: The program as now written, uses MCA receiver 1 and 2. These two receiver codes are noted in the program and must be changed if the MCA that is being used does utilize these two receiver codes.

The program is written in Assembly language.

Program No. 108

FILECOM

Ian Kettleborough
Texas A&M University
College Station, TX 77843

This program is similar to the RDOS CLI FILCOM command except that it does a byte by byte comparison (instead of a word) of two files. Also leading nulls on one or both of the input files are skipped. This allows a file on disk to be compared to a file on paper tape that has leader on it. Dissimilar bytes are printed either to the system console ($TTO) or to a listing file, if specified. The displacement into the field where the difference occurred is also printed.

FILECOM is written in Assembly language.

Program No. 109

NOVA INTERPRETER

Ian Kettleborough
Texas A&M University
College Station, TX 77843

This program interprets another Assembly language program insuring that users do not attempt to reference or execute outside of their area of core. (The valid region of core is established at load time.) The program will catch the following user errors:

1) Not starting address.
2) Attempt to execute outside area.
3) Attempt to reference outside area.
4) Too many levels of indirect addressing.
5) Attempt to branch to itself.
6) Execute time exceeded.
7) System call not allowed.
8) Direct I/O not allowed.
9) Job cancelled by operator.

If any of these error conditions occur, program execution is terminated

and a dump is produced. Execution of a 'HALT' instruction causes program execution to terminate normally and to produce a post execution dump.

Either at assembly time of this routine or during program execution, I/O instructions and/or .SYSTM calls can be made valid.

Tracing of up to 10 statements or complete sections of a program is provided. At any time during program execution, a dump of the user area or just of the accumulators may be requested.

The requirement for execution is that the user program be relocatable and have the symbol 'START' declared as an entry point (using the .ENT statement) and that 'START' be defined on the statement where program execution is to begin.

The program is written in Assembly language and requires about 2000 octal words of core. It has been used in a student environment to prevent inexperienced users from storing over part of the operating system or branching off wildly.

Program No. 110

FORTRAN RANDOM NUMBER GENERATOR

Ian Kettleborough
Texas A&M University
College Station, TX 77843

This program is a function subprogram that is callable from a Fortran program, and will generate a sequence of pseudo numbers in the range from 0.0 to 1.0 (0.0 = X 1.0). The program has two entry points, RAND and RND, which produce the same results. In the first call to this routine the argument given in the call is the initial seed for the sequence of numbers to be generated. A sequence is repeatable if the same seed is given in the first call.

Note: The argument given must not be a constant because this routine changes its value every time that it is called.

The subroutine is written in Assembly language and requires 36 octal words of core.

Program No. 112

CHARGE

John P. Walsh
Kaman Avidyne
Burlington, MA

CHARGE is an accounting program designed to process the log file generated by the RDOS batch monitor. The user sets two variables in the main program for the minimum charge per job and the dollar charge per hour of machine time.

As each job on the log file (as indicated by the batch control card !JOB) is processed, the following information is output by CHARGE:

1) Date and time at start of Batch job.
2) Entire contents of user job card.
3) Date and time at end of user job.
4) Elapsed time in seconds used by job.
5) Total cost for job.

At the end of the log file, the total charges for that file are also output. In addition to this standard output, all "JOB TERMINATED BY OPERATOR" messages are also preserved in the edited file. Output from CHARGE is usually to a disk file in order that multiple copies can be made.

The program is written in Fortran IV.

Program No. 115

BASIC EDITOR

This program is written in Extended BASIC and it allows a user to edit programs in BASIC using either single or timesharing mode. It is primarily an implementation of a subset of the NOVA Text Editor features. MACRO capability is not present. The following commands are available:

a) Insert following lines
b) Type the lines
c) Delete the lines
d) Change the string to a new string
e) Search for line containing string
f) Specify input file
g) Specify output file

h) Yank a page
i) Punch a page
j) Current line position
k) Total # of lines
l) Set line position to first line
m) Set line position to after last line
n) Append next page to buffer
o) Close output file
p) Type condition of the text buffer
q) Return back to the monitor

Program No. 117 (LP)

RDOS BINARY LOADER

Ian Kettleborough
Texas A&M University
College Station, TX 77843

This routine is a modification of Data General's "BINARY LOADER" (Document 093-000003-06) so that it can be loaded via RDOS HIPBOOT. This allows shifting from RDOS operation to a standalone one and having the binary loader already in core. It is loaded either with the CLI 'BOOT Filename' command or by responding to HIPBOOT 'FILENAME?' with the save file name. The routine sizes available core and relocates the binary loader to the high end of core.

Program No. 118

DISK COPY

This is a standalone program for D.G.C. 203-cylinder moving-head disks of the type 4047A, B, 4048A, and 4057A. The program does a complete disk copy, all sectors. The time required is about 90 seconds for the 4047A, B; and 7 minutes for the 4057A.

Error messages are provided. The program will also run in a shared disk environment. Upon completion, the drives are recalibrated so that it is unnecessary to manually recalibrate for a boot.

DISK COPY is written in Assembly language.

Program No. 119

PREP

E.R. Fritz
Intersystems B.V.
Herengracht 244
Amsterdam, The Netherlands

This program provides a facility which is often called the 'include feature'. In general, PREP scans a file containing source statements for statements of the form:

	(SP)	Include (SP) (FILENAMES) (CR)
WHERE:	(SP)	Means any number of spaces and/or tab characters
	(FILENAMES)	Means any number of filenames, separated by blanks, tab characters and/or commas
	(CR)	Means carriage return character

If such a statement is found, it is replaced by the contents of the successive files named in it. These contents, in their turn, are also inspected for the occurrence of include statements. The depth of nesting is four.

The main program is written in Algol. An Assembly language routine called ('FGRND'), which is used to investigate whether PREP is running in the background or foreground, is also included.

Program No. 121

TYPER

TYPER allows the computer system to be used to emulate a typewriter, with several additional features, which include the ability to insert, delete or change any portion of the input text without the necessity of reformatting the data structure.

TYPER implements all the features of a typewriter (including margin and tab control, spacing, margin release, etc.); functions for input/output (read, write, etc.); special functions (insert, delete, rubout, clear, duplicate, and erase); and screen control functions (roll up/down, cursor movement). There are also some

functions which are controlled automatically by the program after being initially set up by the user, including double line spacing, form feeds after page writes, and automatic rolling of the screen after vertical cursor positioning.

TYPER runs under RDOS Rev. 3.03 or later. The program is written in Algol.

Program No. 122

SCRIPT

This program reads an unformatted text file, and, acting upon the formatting commands in the text, formats the text and writes the formatted text to some output device. The text produced is justified with page numbers.

The power of script is not so much in its ability to justify text, but in its other formatting capabilities. By using the formatting commands, a writer can change the line length, the left margin, the length of a page, the top margin on a page, and many other variables involved in producing text.

SCRIPT runs under RDOS Rev. 3.03 or later. The program is written in Algol.

Program No. 123

FORTRAN POST PROCESSOR

This Algol program will perform the following modifications given an assembly source output from the Fortran IV compiler:

a) All occurrences of:

```
JSR @  .LDX  (or .STX where X = 0, 1,
or 2).
(ABS. ADDRESS)
```

Are changed to:

```
LDA @  .+2
JMP    .+2
(ABS. ADDRESS)
```

b) All single dimension array subscript calculations are converted to 'in-line' code.

Program No. 155

BRTIOS (A-D I/O and Logical Functions for the BASIC User)

BRTIOS uses the CALL feature of Extended BASIC to provide analog and digital I/O and logical functions to the BASIC user. The package supports the 4120 Series A/D, 4180 Series D/A and 4066 Digital I/O. It is supplied as a source file utilizing CALL numbers 10 - 19 and must be assembled and SYSGENed into Extended BASIC following the instructions in the manuals (93-65 and 93-119). The 10 subroutines are: Analog Input, Analog Output, Digital Input, Digital Output, Set Bit, Clear Bit, Get Bit Status, Exclusive Or, Inclusive Or and Logical And.

PRICING INFORMATION

Fees are charged for User's Group programs to cover the costs of reproduction and mailing, and to support other User's Group services.

Prices for programs listed in this catalog are provided in the Program Library Price List Schedule, available to all members of the Data General User's Group. For details, please contact:

User's Group Administrator
Data General Corporation
Southboro, Massachusetts 01772

European members may contact:

User's Group Office
United Kingdom Training Center
Westway House
320 Ruislip Road East
Greenford, Middlesex, England

SAMPLE: Copy this form or use facsimile.

PROGRAM REQUEST FORM

Date _____

CATALOG #	QUANTITY	PROGRAM NAME	TAPE	DOCUMENTATION ONLY	UNIT PRICE	TOTAL PRICE

TOTAL _____

Payment is made: ☐ Check enclosed
☐ Coupons
☐ P.O.

Name _____

Ship to _____

Telephone: Area Code _____ No. _____ Ext. _____

Return completed form to: Data General Users Group Administrator, Data General Corp., Southboro, Mass. 01772

SAMPLE: Copy this form or use facsimile.

PROGRAM SUBMISSION FORM

Date _____

Name of Author _____

Company Name _____

Address _____

Telephone: Area Code _____ No. _____ Ext. _____

Program Title _____

Language _____

Make sure that the following items are included:

- ☐ an abstract (200 words maximum)
- ☐ full description of software's function
- ☐ Operating Instructions
- ☐ Source code tape and listing
- ☐ Source language

Minimum hardware required (specify Data General model numbers if known) _____

_____

Memory Size _____ Peripherals _____

Other programs or subroutines required _____

Operating system required _____ Revision Level _____

Known restrictions, deficiencies or problems _____

_____

_____

Date of planned or possible future revisions: _____

I give full permission to Data General to publish information regarding this program and to reproduce and distribute it in full or in part for use by all interested parties, in accordance with the current policies of the Data General Users Group. I further warrant and represent that I have good and sufficient title and all rights and interest in and to the program to grant such permission.

_____

Date Authorized Company Agent

Return this form to: Users Group Administrator, Data General Corp., Southboro, Mass. 01772

SAMPLE: Copy this form or use facsimile.

PROGRAM REVIEW SHEET

To be completed in the event that a problem is encountered with a User's Group program.

Please complete all relevant items.

Date _____

Title of Program _____ Catalog # _____

1) Does the Library Abstract adequately describe the program? Yes ☐ No ☐

Suggested changes

_____

_____

2) Are the operating instructions:
   a) Clear Yes ☐ No ☐
   b) Understandable Yes ☐ No ☐
   c) Complete Yes ☐ No ☐

3) Does this program duplicate other program(s) in the Users Group Library? Yes ☐ No ☐
   Catalog #(s) _____ If yes, is it an improvement? Yes ☐ No ☐
   Explanation _____

_____

4) Operation--Please check any section below which caused trouble:
   Paper Tape— Compile ☐ Load ☐ Start Up ☐
   Program Operation— Running ☐ Peripheral Handling ☐ Input/Output ☐

Comments _____

_____

_____

5) Did you discover any restrictions not mentioned by the author? Yes ☐ No ☐ Specify _____

_____

_____

6) Can you suggest corrections or improvements to this program? Yes ☐ No ☐ (Please complete Program Revision Form)

7) Should this program:
   a) Be retained in its present form? Yes ☐ No ☐
   b) Remain after appropriate changes? Yes ☐ No ☐
   c) Be removed? Yes ☐ No ☐ Why? _____

_____

8) Any other comments _____

_____

_____

_____

_____

_____

Name _____

Address _____

Telephone: Area Code _____ No. _____ Ext. _____

Would you have any objections if we forward a copy of this review to the program author? Yes ☐ No ☐

Return to: User Group Administrator, Data General Corp., Southboro, Mass. 01772

013000082

SAMPLE: Copy this form or use facsimile.

PROGRAM REVISION FORM

Date _____

Original Title of Program _____ Catalog # _____

Original Author _____

Revising Author _____

Company or Affiliation _____

Address _____

Telephone: Area Code _____ No. _____ Ext. _____

Please list all changes in the following:

1) Monitor/operating system _____

2) Core storage required _____

3) Peripherals required _____

4) Other software required _____

5) Program logic _____

Reasons for Revision:

1) Debug, correct known problem _____

2) Extend to handle new or different configurations _____

3) Operate under new or different monitor system _____

4) Increased operational efficiency _____

5) Other _____

_____

Outstanding deficiencies or problems after revision _____

_____

_____

Documentation
All revisions should include a full statement of all changes.
Revised abstract ☐ Revised write-up ☐ New listing ☐

I give full permission to Data General to publish information regarding this program and to reproduce and distribute it in full or in part for use by all interested parties, in accordance with the current policies of the Data General Users Group. I further warrant and represent that I have good and sufficient title and all rights and interest in and to the program to grant such permission.

_____

Date Authorized Company Agent

013000083

OBJECTIVES

The purpose of the Data General User's Group is to promote the more effective use of computer systems manufactured by Data General Corporation through a program of information and software exchange. Specific objectives include:

1) Maintenance of program library
2) Periodic publication of a newsletter
3) Sponsorship of annual national meeting
4) Active sponsorship of regional and local chapters
5) Promotion of special projects, publications, and special interest groups (SIG) when deemed necessary by the membership

ACTIVITIES

Program Library

The Library consists of programs submitted by Group members. It is growing rapidly and includes programs in the following categories: communications, computational, data processing, education, games, instrumentation and control, and utilities. Abstracts of these programs are compiled and printed as a Program Library Listing which is periodically updated. Programs may be ordered by sending a check or Library coupons; or a company purchase order if the total is $50.00 or more. Program submission forms (which explain the requirements for submitting a program to the Library) and order forms are available upon request from the User's Group Administrator in Southboro.

INPUT/OUTPUT

Input/Output is the official bimonthly publication of the User's Group. It serves as a forum for discussion among members and includes articles on hardware, software, applications, and seminars. Special columns such as "Help Wanted" provide a vehicle for members seeking answers to programming and other problems.

National Meeting

Once a year the national group sponsors its annual business meeting and technical seminar. At this time new officers are elected and plans submitted for the coming year. For the technical sessions members are invited to present papers on their research and applications.

Regional and Local Chapters

The national group actively supports the formation of regional and local chapters. These groups serve the needs of members within a given geographic area with meetings, seminars and other activities. Each group sends a representative to Executive Board meetings.

Special Interest Groups

These groups are formed by members sharing a common interest area such as education, medicine, or communications. They hold meetings during the year as well as at the annual meeting.

MEMBERSHIP CATEGORIES

Installation Member

An employee of a company, organization, or school, which now has, or which has on definite order, a computer manufactured by Data General Corporation. Installation members have the right to vote in elections for User's Group officers and to participate fully in all activities.

Student Member

Any student from a high school, college, or university which now has, or which has on definite order, a computer manufactured by Data General Corporation. Student members participate fully in all User's Group activities.

Associate Member

Any individual with a bona fide interest in the Data General User's Group. Associate members will receive relevant correspondence, the Newsletter, and invitations to National Meetings. The Membership fee is $5.00.

GOVERNANCE

The User's Group activities are coordinated by the Executive Board which is elected from the membership, and the Executive Secretary who is Data General's representative.

For further information and membership forms contact:

User's Group Administrator
Data General Corporation
Southboro, MA 01772

SAMPLE: Copy this form or use facsimile.

INSTALLATION MEMBERSHIP FORM

Date _____

Name _____ Position _____

Company, organization or school _____

Address _____

City _____ State _____ Zip _____

Telephone: Area Code _____ No. _____ Ext. _____

# of machines on order _____ # of machines installed _____

1) Account category: OEM ☐ End User ☐ System House ☐

2) Mode of operation (check all applicable):

☐ 01 Batch (Central)
☐ 02 Conversational
☐ 03 Time Sharing
☐ 04 Real Time
☐ 05 On Line
☐ 06 In Line
☐ 07 Dedicated
☐ 08 Inquiry
☐ 09 Multi-Program.
☐ 10 Multi-Proc.
☐ 11 Batch (Remote)

3) SOFTWARE:

Operating System— SOS ☐ RTOS ☐ RDOS ☐ Other ☐ Specify _____

Languages— Fortran IV ☐ Fortran V ☐ Basic ☐ Algol ☐ Assembler ☐ Other ☐

Specify _____

Application Packages (Specify): _____

4) From whom was your machine(s) purchased?

Data General Corp. ☐ Other ☐ Specify _____

5) Configuration

CENTRAL PROCESSOR	QTY	MEM (K)
NOVA 2/4		
NOVA 2/10		
1200		
1210		
1220		
800		
820		
840		
NOVA		
SUPERNOVA		
OPTIONS		
FP		
M/D		
MMPU		
COMMUNICATIONS		
ASYNCH		
4023/4029		
MULTILINE		
4026		
4060		
4100		
SYNCH		
SINGLE LINE		
4015		
4074		
MULTI-LINE 4073		
360 INTERFACE		
MCA		

PERIPHERALS	QTY	
TERMINALS		
TTY		
NOVA DISPLAY		
INFOTON		
DISC		
NOVA DISC		
ALPHA DISC		
DIABLO		
CENTURY		
MAGNETIC TAPE		
WANG		
NOVA CASSETTE		
P. T. READER		
DGC		
DIGITRONICS		
P. T. PUNCH		
TELETYPE		
CARD READER		
MOHAWK		
DOCUMATION		
LINE PRINTER		
DATA PRODUCTS		
CENTRONICS		
PLOTTER		
SPECIALS (% OF SYS)		

6) Application Category

Major Class of System (Check one):

- ☐ 1 Computational
- ☐ 2 Data Processing
- ☐ 3 Communications
- ☐ 4 Instrumentation/Control
- ☐ Other (Specify)_____

Check those applicable:

- ☐ 03 Access Security
- ☐ 06 Air Traffic Control
- ☐ 09 Business D.P.
- ☐ 12 Closed Loop Cntrl.
- ☐ 15 Command & Control
- ☐ 18 Computer Modeling
- ☐ 21 Data Acquisition
- ☐ 24 Data Communications
- ☐ 27 Data Formating
- ☐ 30 Design Automation
- ☐ 33 Distribution
- ☐ 36 Eng. Calculations
- ☐ 39 Environ. Control
- ☐ 42 Graphics
- ☐ 45 Information Mgmt.
- ☐ 48 Instruction
- ☐ 51 Instrumentation Cntrl.
- ☐ 54 Instrum. Monitoring
- ☐ 57 Machine Control
- ☐ 60 Machine Monitoring
- ☐ 63 Point-of-Sale
- ☐ 66 Pre-Processing
- ☐ 69 Process Control
- ☐ 72 Production Control
- ☐ 75 Process Monitoring
- ☐ 78 Research
- ☐ 81 Word Processing
- ☐ 83 Other (Specify)_____

Application description:_____

Return to: Users Group Administrator, Data General Corp., Southboro, Mass. 01772

013000073

SAMPLE: Copy this form or use facsimile.

MEMBERSHIP APPLICATION FORM

Please check which category of membership you are applying for:

☐ Student Membership ☐ Associate Membership

Date _____

Name _____

Company, organization, or school _____

Address _____

City _____ State _____ Zip _____

Telephone: Area Code _____ No. _____ Ext. _____

FOR STUDENT MEMBERSHIP ONLY
Please have a member of your school's staff sign below to confirm that you are presently enrolled and have access to a Data General computer.

_____
Signature

_____
Position

FOR ASSOCIATE MEMBERSHIP ONLY
Please explain your interest in the Data General Users Group: _____

_____

_____

_____

_____

_____

_____

_____

_____

Return completed form to: Users Group Administrator, Data General Corp., Southboro, Mass. 01772