

THE mini- tasker

DECUS

RT-11 SIG NEWSLETTER

August 1984

Volume 10, Number 4

SJ

RTMON

FILEX

CSI

ODT

PIP

LD

RMON

SYSMAC

FB

DIR

KMON

QUEUE

QUEMAN

TECO

PAT

DUP



VM

K52

LIBR

BINCOM

KED

DUMP

BUP

SRCCOM

LINK

SIPP

FORMAT

SLP

RESORC

TTYSET

IND

MACRO

XM

JSW

HELP

The following are trademarks of Digital Equipment Corporation:

DEC	DIBOL	PDT
DECnet	Digital Logo	RSTS
DECsystem-10	EduSystem	RSX
DECSYSTEM-20	IAS	UNIBUS
DECUS	MASSBUS	VAX
DECwriter	PDP	VMS
		VT

UNIX is a trademark of Bell Laboratories.

Copyright © Digital Equipment Corporation 1984
All Rights Reserved

It is assumed that all articles submitted to the editor of this newsletter are with the authors' permission to publish in any DECUS publication. The articles are the responsibility of the authors and, therefore, DECUS, Digital Equipment Corporation, and the editor assume no responsibility or liability for articles or information appearing in the document. The views herein expressed are those of the authors and do not necessarily express the views of DECUS or Digital Equipment Corporation.

Contributions to the newsletter should be sent to:

Ken Demers
Adaptive Automation, Inc.
5 Science Park
New Haven, Conn. 06511
203 786-5050

Other communications can be sent to:

John T. Rasted
JTR Associates
58 Rasted Lane
Meriden, Conn. 06450
203 634-1632

or

RT-11 SIG
C/O DECUS
One Iron Way
MR2-3/E55
Marlboro, Mass. 01752
617 481-9511 Ext. 4141

Table of Contents

SIG CONTACTS	4
USER INPUT	
LSI-11/2 - 11/23 Macros	6
The Windscreen Wiper Story	7
Benchmark Tests - FORTRAN-77 vs FORTRAN-IV	8
Running UCL From a Device Other Than SY:	11
Interfacing a Parrallel Line-Printer to the DRV11	12
Multi-Sorting Method	13
USER RESPONSES	
RT-11 50 Hz Issue	25
DECUS LIBRARY	
A/D Conversions Package For Use With ADV11-C & KVV11-C	26
Adventure	27
BUPRES: Program to Read Individual Files From a Magtape	27
CONEFR.BAS: Cone Frustrum Layout	28
Conpendium Tape From the Australian RT-11 SIG	29
DBSMNG: Small Database Management Package Binary Kit	29
Data Base Management Package for the Pro-300 Series	30
DECOD2: for the Pro-300 Series	32
DECODE4: RT-11 SAV Files Disassembler	33
DIBOL: 83 Screen Handler Package	33
ELLIPTIC: Elliptic Integrals and Functions	34
File Manipulation Commands	34
FODT: FORTRAN-IV On-Line Debugging Tool	35
FORTRAN Callable Subroutines Package for A/D on the MINC	36
Indirect Command File Processing for RT-11 V4.0	36
MJCLOK: Multi-Job Clock Program	37
Monitor Commands for Namelists Package	37
One-Drive Diskette and File Copy	38
PRINIT: Printer Initialization Routine for the LA120	39
RUNOFF: Bonner Lab	39
STONE: A Program for Resolving Mossbauer Spectra	40
Symposium Tape from the European RT-11 SIG, 1982 Warwick	41
TCPL: Terminal Control Function Library	41
TECO Full Screen Editor	42
TRAMP for RT-11	43
TSXLIB: Library of Programmed Requests for TSX-PLUS V5.0	43
TXTWRT: An RT-11 Text Formatting Program	44
VLISP-11	44
Whetstone Benchmark	45
WORD: Document Spelling Checker/Corrector	46
9FETCH.B	46
PAST SYMPOSIUM INFORMATION	
DECUS Symposium RT-11 SIG Tape Directory - Spring 1984	47
RT-11 PRO Memory Issues	54

SIG CONTACTS

RT-11 SIG Chairman
JOHN T. RASTED
JTR Associates
58 Rasted Lane
Meriden, CT 06450
(203) 634-1632

Newsletter Editor
DECNET Contact
KEN DEMERS
Adaptive Automation
5 Science Park
New Haven, CT 06511
(203) 436-1029

Business Working Group Contact
KENNETH L. AYDLOTT
Teledyne Hastings-Raydist
P.O. Box 1275
Hampton, VA 23661
(804) 723-6531

Tape Copy Generation Contact
RALSTON BARNARD
Division 7523
Sandia Laboratories
Albuquerque, NM 87185
(505) 844-5115

Structured Languages Contact
MARK BARTELT
HSC Research Development Corp.
555 University Ave.
Toronto, Ontario
Canada M5G 1X8
(416) 597-1500 Ext 4588
or 598-5982

APL Contact
DOUG BOHRER
Bohrer & Company
903 Ridge Rd
Suite 3
Wilmette, IL 60091
(312) 251-9449

TSX Contact
MACRO Contact
NICK BOURGEOIS / 5238
Sandia Laboratories
P.O. Box 5800
Albuquerque, NM 87185
(505) 844-8088

Symposia Coordinator
NED RHODES
General Scientific Corp.
1684 East Gude Drive
Rockville, MD 20850
(301) 340-2773

RT DECUS Library Contact
Tape Copy Distribution
TOM SHINAL
General Scientific Corp.
1684 East Gude Drive
Rockville, MD 20850
(301) 340-2773

BASIC Contact
ED STEVENS
E M D A Inc.
77 North Oak Knoll
Suite 104
Pasadena, CA 91101
(818) 795-5991

CAMAC Contact
J. W. TIPPIE
Kinetic Systems Inc.
11 Mary Knoll Drive
Lockport, IL 60441
(815) 838-0005

FORTRAN Contact
RT-11 LUG Contact
RON TRELLE
Division 7523
Sandia Laboratories
Albuquerque, NM 87185
(505) 844-0955

Personal Computer Contact
WILLIAM K. WALKER
Monsanto Research Corporation
MRC - Mound
Miamisburg, OH 45342
(513) 865-3557

TECO Contact
Product Planning Contact
JOHN CROWELL
CROW4ELL Ltd.
145 Andanada
Los Alamos, NM 87544
(505) 662-3893

COBOL Contact
Wish List Contact
BILL LEROY
Information Resources, Inc.
470 E. Paces Ferry Rd. N.E.
P.O. Box 52661
Atlanta, GA 30355
(404) 231-3117

RT-11 Hardware Contact
CARL LOWENSTEIN
Univ. of Calif., San Diego
Marine Physical Lab. of the
Scripps Inst. of Oceanography
San Diego, CA 92152
(619) 294-3678

Standards Coordinator
BRADFORD LUBELL
Univ. of Calif., Los Angeles
Los Angeles, CA 90024
(213) 825-6713, 824-6119

C Contact
JACK PETERSON
Horizon Data Systems
1901 Wildflower Terrace
Richmond, VA 23233
(804) 740-9244

FMS Contact
Susan Rasted
Software Dynamics Inc.
1000 Yale Ave.
Wallingford, CT 06492
(203) 265-2226

USER INPUT

At Thermo Electron we were dragged kicking and screaming from the LSI-11/2 to the 11/23 when the 11/2 was cancelled. I wanted the MACRO-11 programs for the 11/23 to look as much as possible like the systems already in the field using the 11/2 and 11/03. The main problem was in the floating point routines. I now add the following macros to a program that uses the 11/23 and the floating routines appear the same in the code.

```
;
    .MACRO  FADD,X
    LDFPS  #7440
    LDF     (SP)+,R0
    ADDF    (SP)+,R0
    STF     R0,-(SP)
    .ENDM   FADD
;
    .MACRO  FMUL,X
    LDFPS  #7440
    LDF     (SP)+,R0
    MULF    (SP)+,R0
    STF     R0,-(SP)
    .ENDM   FMUL
;
    .MACRO  FSUB,X
    LDFPS  #7440
    LDF     (SP)+,R0
    SUBF    (SP)+,R0
    NEGF    R0
    STF     R0,-(SP)
    .ENDM   FSUB
;
    .MACRO  FDIV,X
    LDFPS  #7440
    LDF     (SP)+,R0
    LDF     (SP)+,R1
    DIVE    R0,R1
    STF     R1,-(SP)
    .ENDM   FDIV
```

John T. Davies III
Thermo Electron Corporation
Instruments Division
524 Alpha Drive
Pittsburgh, PA 15238-2912

THE WINDSCREEN WIPER STORY

Users of the Really-good-Time system and the Random Suicide eXperiment have always known they come from different worlds. However, in most cases this is just blind uninformed prejudice. We all know intuitively that there is something different between the two systems, but what is that 'something'?

The story so far:

RT and RSX are both successful automobiles originally designed for the desert.

Around Version 12, the manufacturers decided to export the vehicles to Europe where it always rains and snows. Directives were sent to both design groups insisting that windscreen wipers be added.

The RT-11 group leader read this in his mail when he arrived for lunch. As he was thinking about it, a programmer called Jim Senior walked by his office (most RT-11 programmers are called Jim). "Jim", he called out, "we gotta add wind-screen wipers to V12".

Jim went off to the deli to get his favorite fish and salad sandwich for lunch. On the way back, it suddenly came to him: "Wind Screen Wiper... WSW... WSW... JSW... add a bit to the JSW (is there one left?)." Jim told his group leader that he now owned bit 16 of the JSW. Three hours later the job was done.

Setting WSW\$ in the JSW turned the windscreen wipers on. Clearing turned it off. Implemented as a handler. No SYSGEN required. Three new paragraphs in two different manuals.

Things happened differently in the RSX department. First, the SIG formed a working group that looked at the topic for two years. Then an internal committee was formed to look at the feasibility of the project. They reported:

'As is well known, the purpose of our vehicle is to travel from point A to point B. The generic design of this system requires that all parts of the system be able to interact effectively with all other parts of the system. In other words, all parts of the system must be designed within the framework of the central goal of going from point A to point B.

Until now, we have been able to assume that the vehicle could use wheels. With the addition of windscreen wipers, this will no longer hold. In this case, we will have to cover the road with tissue paper and refashion the shape of the vehicle into a cylinder that rolls along the road. The rolling action will wipe the windscreen.'

The group leader accepted their report and told them to get working on the new turn-indicators and the horn. Six months later, the wiper was installed and a new chapter was added to the documentation.

The results? V12 of RT was not much different than V11; the windscreen wiper operation was largely transparent. V12 of RSX, however, led to reduced usage: a large number of fatal crashes wiped a lot more than the windscreens. In fact, most of the crashes occurred during SYSGEN (you must keep in mind that earlier additions to the task builder had forced RSX to take the shape of a bulldozer).

In conclusion, RT-11 is like a fast sports car that can tow any trailer. RSX is like an army tank. UNIX is a machine with a single wheel and two doors: one to get in, and one to get out, with no windscreen to wipe. VAX/VMS, a larger version of RSX, is a semi-trailer that is only ever partly on the road, a good choice for users who have a collection of the world's telephone books to transport.

Ian Hammond
HAMMOND-software, Am Feldborn 22
D-34 Goettingen, West Germany

Benchmark Tests
FORTRAN-77 vs FORTRAN-IV
 for Various Processors

Tests:

1. CCA1 - Single-precision Whetstone
2. CCA2 - Double-Precision Whetstone
3. CCA5 - FORTRAN Library function test
4. CCA9 - Binomial Expansion
5. CCA10 - IF test

Configurations:

1. EPU - FORTRAN-IV, EIS code with FPU OTS Library
2. FPU - FORTRAN-IV, Threaded code with FPU OTS
3. F77 - FORTRAN-77, FPU/EIS code with FPU OTS

Processors:

1. 11/23 with KEF FPU chip
2. 11/44
3. 11/73 (production model, 15 MHz crystal)

Operating Systems:

RT-11, Version 5.0 and 5.1 were used for all tests, except where noted. The RSX used was Version 4.0

Benchmark CCA1

Processor	Execution times in Minutes				
	EPU	FPU	F77	F77/EPU	F77/FPU
11/23	3.85	4.07	3.07	.80	.75
11/44	.93	1.00	.62	.67	.62
11/73	1.18	1.27	.80	.68	.63
11/44 (RSX)	1.00	1.10	.56	.56	.51

Benchmark CCA2

Processor	Execution times in Minutes				
	EPU	FPU	F77	F77/EPU	F77/FPU
11/23	7.17	7.32	5.57	.78	.76
11/44	1.32	1.43	.80	.61	.56
11/73	2.10	2.28	1.38	.66	.61
11/44 (RSX)	1.48	1.49	.75	.51	.50

Benchmark CCA5

Processor	Execution times in Minutes				
	EPU	FPU	F77	F77/EPU	F77/FPU
11/23	11.33	11.70	11.42	1.01	.98
11/44	1.75	1.90	1.76	1.01	.93
11/73	2.35	2.52	2.57	1.09	1.02
11/44 (RSX)	1.83	1.98	1.83	1.00	.92

Benchmark CCA9

Processor	Execution times in Minutes				
	EPU	FPU	F77	F77/EPU	F77/FPU
11/23	10.45	9.07	6.33	.61	.70
11/44	2.77	2.18	1.24	.45	.57
11/73	3.55	2.85	1.48	.42	.52
11/44 (RSX)	2.94	2.32	1.32	.45	.57

Benchmark CCA10

Processor	Execution times in Minutes				
	EPU	FPU	F77	F77/EPU	F77/FPU
11/23	1.95	6.00	2.34	1.20	.39
11/44	.77	2.23	.90	1.17	.40
11/73	.72	2.32	.87	1.21	.38
11/44 (RSX)	.86	2.51	.87	1.01	.35

Ratios of Processor Performances

These ratios compare the performance of the various processors for the compiler Configurations listed; The ratio RT/RSX is for the 11/44 running under the two operating systems.

Benchmark CCA1

Processors		EPU	FPU	F77
73/23		.31	.31	.26
73/44		1.27	1.27	1.29
RT/RSX		.93	.91	1.11

Benchmark CCA2

Processors		EPU	FPU	F77
73/23		.29	.31	.25
73/44		1.59	1.59	1.73
RT/RSX		.89	.96	1.07

Benchmark CCA5

Processors		EPU	FPU	F77
73/23		.21	.22	.23
73/44		1.34	1.33	1.46
RT/RSX		.96	.96	.96

Benchmark CCA9

Processors		EPU	FPU	F77
73/23		.34	.31	.23
73/44		1.28	1.31	1.19
RT/RSX		.94	.94	.94

Benchmark CCA10

Processors		EPU	FPU	F77
73/23		.37	.39	.37
73/44		.94	1.04	.97
RT/RSX		.90	.89	1.03

R. W. Barnard
 Sandia National Laboratories
 Minicomputer Software Division 7523
 Albuquerque, NM 87185

RUNNING UCL FROM A DEVICE OTHER THAN SY:
(for RT-11 V5 or later)

It is possible to patch the RT-11 monitor to run UCL.SAV from a device other than SY:. This patch is described below. It is NOT an official DEC customization, but if used with a little discretion, it can be very useful. Essentially, this patch involves changing the file name that RT-11 looks for from SY:UCL.SAV to UCL:UCL.SAV. RT-11 will then run UCL from logical device UCL: rather than SY:. I usually assign UCL: to VM: in my start-up command file, thus improving response time considerably.

Those of you who are using the UCL distributed with RT-11 V05.01 will probably want to use the patch in the CUSTOM.TXT file to make the UCL program look for UCL.DAT on UCL: also.

If you are using my UCL+ program, keep in mind that UCL+ updates it's own "disk image" whenever you define/erase/replace a symbol, change the path list, etc. If you, for example, "ASSIGN VM UCL", the copy of UCL+ on VM: will get updated if you make changes, NOT any copy that you may have on SY:. If you run UCL+ off of a transient device like VM:, you should ensure that any updates that you want to make permanent get preserved on some less perishable media. One possible method is to execute an occasional STORE to a file on a selected disk.

If you are using the distributed UCL, the same sort of considerations apply to UCL.DAT. Make sure you keep an up-to-date copy on a non-volatile device!

NOTES: In the patch shown below, "<ret>" indicates lines where only the RETURN key is pressed, "^Z" is a control-Z, "^Y" represents control-Y, "valu" is the address where the Radix-50 string "UCL" is found, "valu-2" represents the address of the word just before "valu", and "RT11xx.SYS" is the name of the monitor file that you wish to patch (for example, "RT11FB.SYS").

I HAVE ONLY CHECKED THIS PATCH FOR RT-11 V05.01!!!

.SIPP RT11xx.SYS


Base? <ret>

Offset? <ret>

Base	Offset	Old	New?
000000	000000	000000	;S
Search for?	;RUCL		
Start? <ret>			
End? <ret>			
Found at valu			

Base	Offset	Old	New?
000000	000000	000000	^Z
Offset? valu-2			

Base	Offset	Old	New?
000000	valu-2	075250	;RUCL
000000	valu	101704	^Y


William K. Walker
Monsanto Research Corp.
P.O. Box 32 OS-123
Miamisburg, OH 45342

Compuserve ID: 71066,24

Perhaps some of the Mini-Tasker readers would profit from our recent experience with an apparently unorthodox interfacing scheme involving the Q-Bus:

Interfacing a Parallel Line-Printer to the DRV11

Since we had an extra line-printer (Integral Data Systems 460 Paper Tiger) that could be "configured for parallel operation," and since we had an extra DRV11 parallel I/O card lying around, I figured it would be no problem to mate the two to provide a working printer for one of our LSI-11 based systems (under RT-11, V.4). Well, the obvious hookup did not work; however, after many hours and headaches, we found that a simple addition to the LP handler provided the needed trick: the marriage was successfully consummated, and the printer now hums merrily along, indeed without problem.

A single cable was led from the output connector on the DRV11 (J1) to the RS232 connector on the printer; the nine active lines included the seven data bits, the NEW DATA READY (STROBE) line, and the REQ A (printer NOT BUSY) line (the line that provides the interrupts). Since the MSB (error bit) of the CSR appears on the input connector (J2) of the DRV11, and shows as asserted with no input, we simply grounded this input (REQ B) on the card. Thus, we have sacrificed the ability to act on any printer error (e.g., paper out). Of course, extra cabling, including some lines to J2, could take advantage of the error checking within the LP handler; however, we have not found this lack of error checking to be a problem.

With the configuration described the printer would successfully print out the entire contents of the first file listed (in a PRINT command), but would then hang and not return to the monitor until two 'C's were typed. Some snooping in core revealed that in the course of printing a file, the driver entry portion of the LP handler is entered twice: the first time prior to output, and a second time after the entire file has been dumped. An interrupt was not occurring this second time when the IE bit was set, though by all rights it should have. It finally became clear to us that, at least with the DRV11 card, in order for an interrupt to be acknowledged, not only must the previous interrupt have been disposed of, but the output buffer register must have been explicitly written to. This was not occurring after the last interrupt of the printing sequence (when it was merely determined that no more characters were to be output); hence, upon the second entry to the driver, the expected interrupt that would allow a return to the monitor was not being acknowledged.

Our solution was to force a "write" to the output buffer register upon the last interrupt in the printing sequence. Toward the end of the LP handler, change

```
LPDONE: CLR      @LPS      ;clear CSR
```

to

```
LPDONE: MOV      LPS,R5    ;
        CLR      (R5)+     ;clear CSR
        CLR      (R5)      ;"write to" OUTPUT BUFFER REGISTER
```

Register R5 is a "safe" register for use at this point, as it is the one the handler has been using to pop characters into the print queue.

I enjoy reading the Mini-Tasker immensely. Keep up the good work!

Sincerely,

Wes Grantham

D. Wesley Grantham, Ph.D.
Director of Auditory Research
Bill Wilkerson Hearing & Speech Center
1114 19th Avenue South
Nashville, TN 37212
(615)320-5353

- Multi-sorting method -

Development of a structured FORTRAN 77 subroutine for multi-sorting without manipulations of the data and extra mass storage during the process.

by Daniel GUINIER (*) and Nikos TOTOS (**),

(*),(**)

Department of Computer Science, NAVAL POSTGRADUATE SCHOOL
CA 93943 MONTEREY, USA.

(*)

Groupe de Laboratoires du CNRS de STRASBOURG - CRONENBOURG
23 rue du Loess,
B.P. 20 CR,
F 67037 STRASBOURG CEDEX, FRANCE.

(**)

General Staff of Greek Navy
Stratopedo Papagou
Holargos ATHENS, GREECE.

Abstract :

An algorithm for multi-sorting a large amount of data organized in a matrix of columns and rows without intermediate manipulations of the primitive file records and needing no extra mass storage is given on the form of a structured familiar FORTRAN 77 subroutine.

1. Introduction :

Sorting a matrix made up a large amount of data and organized in columns (variables) and rows (observations) is a delicate operation that requires an appropriate method especially if the number of rows is large (several thousands). Inadapted methods can reduce the general efficiency of the program or cause a failure in multi-sorting.

Multi-sorting is a set of sorting operations according several successive columns (descriptive or guide columns) to organize the previous matrix stored in a direct-access file in sets and subsets for future uses such as statistical calculus.

----- Concept : -----

With this operation the system executes a repeated sorting of the data matrix according to successive column numbers, or column names assigned by the user via the terminal. This function permits nested sorting of the data matrix. The concept of the nested sorting is clearly illustrated by the following example. Suppose that we have the data matrix in fig.1(a) and we repeatedly sort it according the columns 4,3,2 and 1. The resultant matrix is shown in fig.1(e). In this matrix the four first rows of column 1 have the same value and the result of the repeated sorting is that the corresponding first four rows of column 2 (which is nested to column 1 according to the assigned sorting sequence 4,3,2,1) are sorted. Likewise, in column 3 the rows 2,3 and 4 have the same value and the result of the repeated sorting is that in column 4 the values in rows 2,3 and 4 are sorted.

The concept is farther clarified in fig.2 where direct sorting of the data matrix is presented according to column 1 (the last column of the sequence of the repeated sorting). The result is a matrix sorted in column 1 but the nested sorting of the previous case does not exist.

This kind of sorting is useful in cases where some variables of the active data matrix are not quantitative (continuous) but descriptive (discrete) and are attached to classes or subclasses of data; successive sorting involving such variables organize the matrix in corresponding order if the collection of data is not presorted in function of the classes or subclasses fig.3.

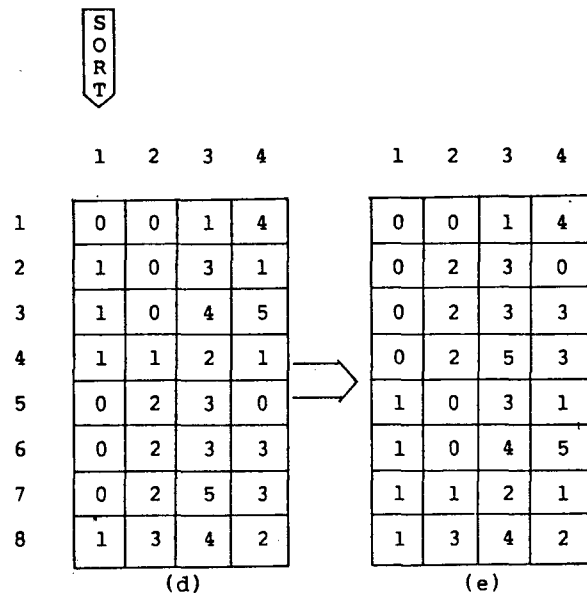
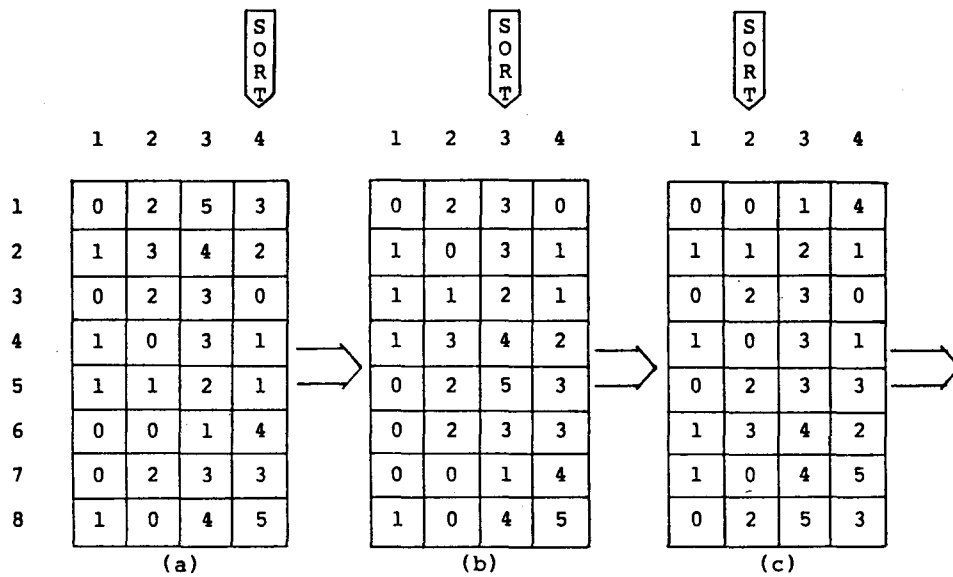


Fig. 1

Nested sorting of the data matrix.

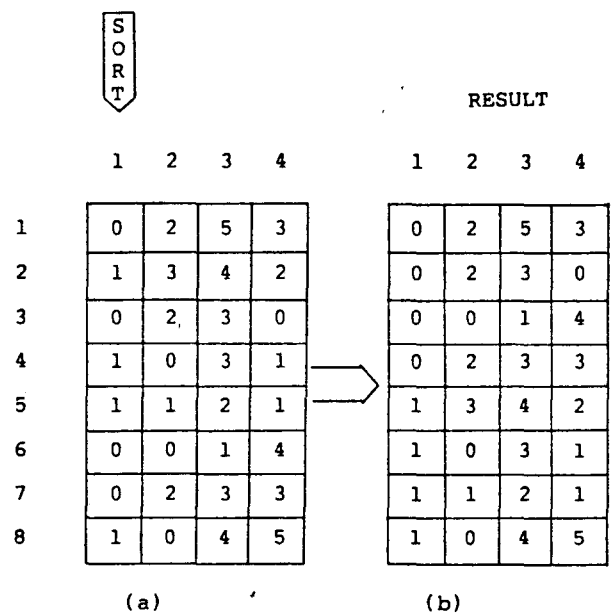


Fig. 2

Single sorting

1	1	3.2	0	0	8.1
0	0	8.1	0	0	12.0
0	0	12.0	0	1	13.4
1	1	3.1	0	2	1.5
0	2	1.5	0	2	2.6
1	0	0.7	1	0	0.7
0	1	13.4	1	1	3.1
0	2	2.6	1	1	3.2
1	2	13.2	1	2	13.2
"Descriptive"	"Quantitative"		"Descriptive"	"Quantitative"	
data	data		data	data	
Before sorting			After sorting		

Fig.3 : Set of data containing two classes and three subclasses.

2. Principle :

The operation of the repeated sorting of the data matrix is executed in two phases. The first phase sorts the data matrix consecutively according to the assigned sequence of column numbers or names, while the second rearranges the rows of the initial matrix according to the result of the repeated sortings.

Phase 1

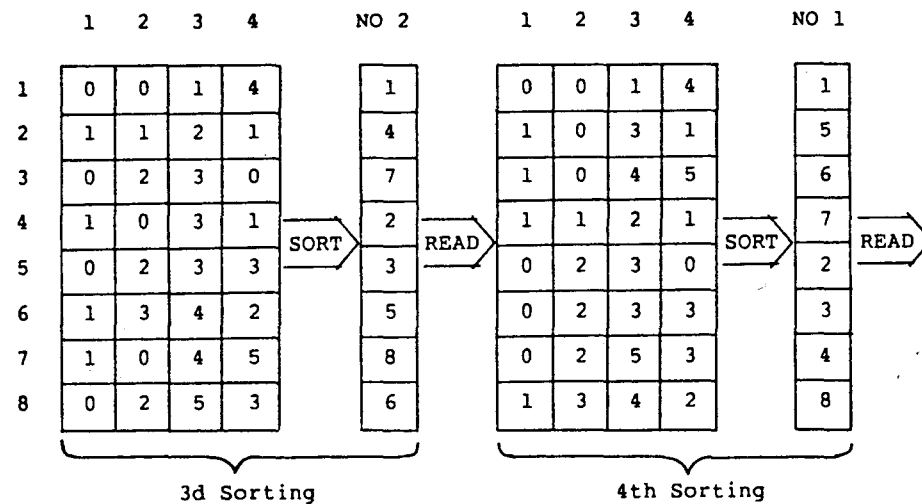
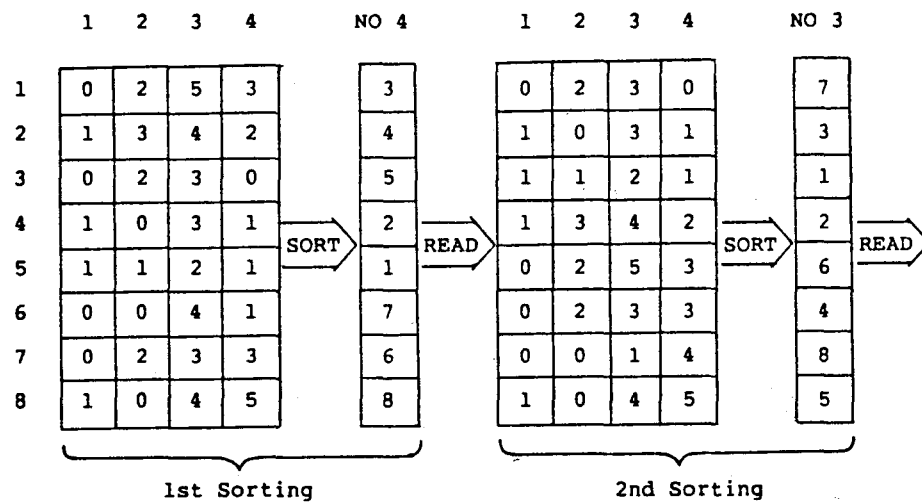
The virtual process according to which the repeated sorting takes place is the following (fig.4) :

- a). Read the first column according to which the data must be sorted. Store the data of this column in an array.
- b). Sort the array and get the invert relative addresses in an array (NO).
- c). Read according the inverted relative addresses. (This means that the rows are rearranged before next sorting.)
- d). Repeat until all the sorting guides have been used.

In order for rearranging to be avoided and thereby save in process time and utilized space, the following method is used (Fig.5).

A one dimension array (NOR) is used to store the relative addresses according which the data file will be read and which remains unchanged during the whole process.

- a). Initialize the NOR.
- b). Read the first column according which storing will take place.

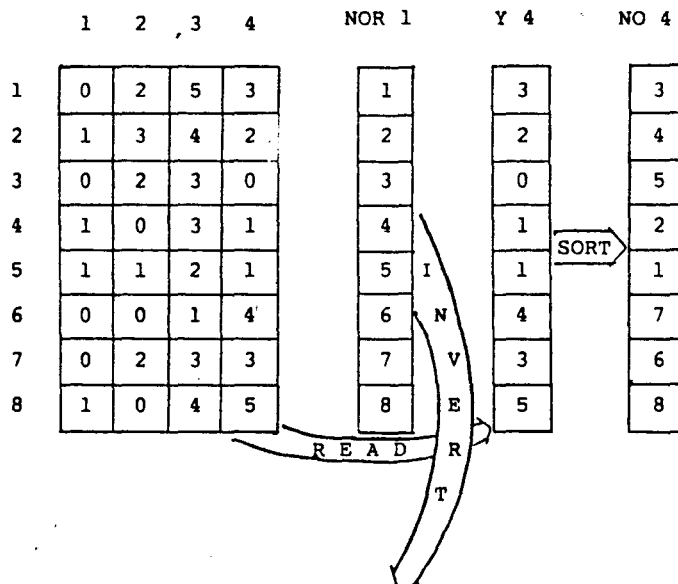


1 2 3 4

1	0	0	1	4
2	0	2	3	0
3	0	2	3	3
4	0	2	5	3
5	1	0	3	1
6	1	0	4	5
7	1	1	2	1
8	1	3	4	2

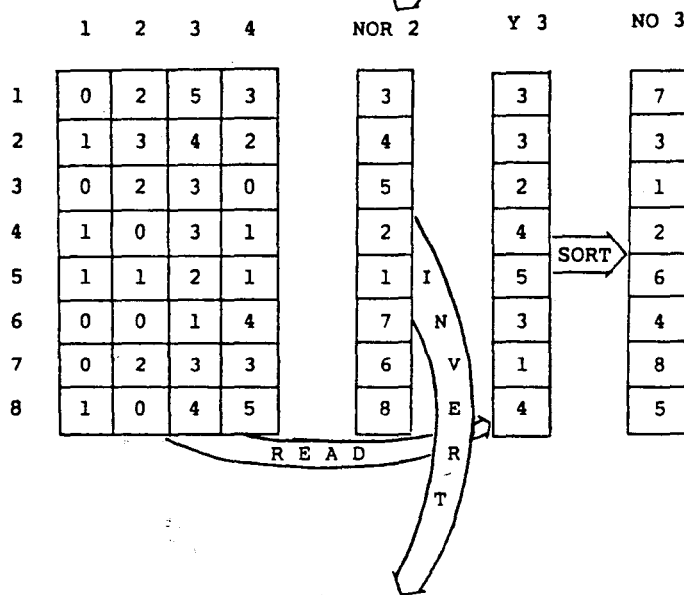
Result

Fig. 4
virtual process of repeating sorting



STEP 1

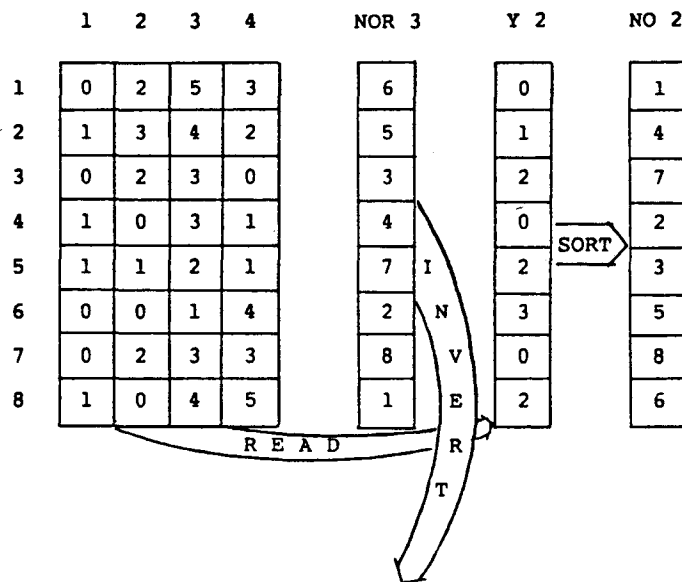
- Initiate NOR 1.
- Read column 4 according to NOR 1.
- Sort Y 4.



STEP 2

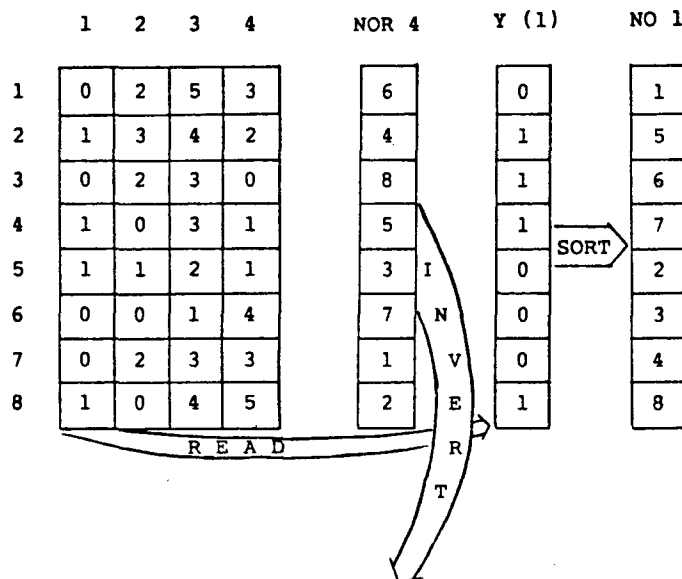
- Invert NOR 1 according to NO 4.
- Read Y 3 according to NOR 2.
- Sort Y 3.

Fig. 5 (Cont)



- STEP 3**

- Invert NOR 2 according NO 3.
 - Read Y 2 according NOR3.
 - Sort Y 2.



- STEP 4**

- Invert NOR 3 according NO 2.
 - Read Y 1 according NOR 4.
 - Sort Y 1.

(NO). -c). Sort the column and get the inverted relative addresses

-d). Invert the NOR according the NO.

-e). Repeat until all sorting guides have been used.

Phase 2

The second phase is the rearranging of the data matrix according the last inverted relative addresses, using the same external file. A pictorial presentation of the algorithm used is shown in fig.6.

The reasoning of this algorithm is the following : In fig.6 the array containing the inverted relative addresses is noted as NOR. The first pointer of NOR1 is pointing to row number 6 which means that the row 6 must move to the first relative address of the matrix. The contents of this address however must be saved. For this reason a mutual exchange between the first and sixth row of the matrix is executed. By this way the first row already contains the appropriate values. A consequence of this exchange however is that the first row which should move to the fourth row of the matrix is now moved to the sixth position. The fourth relative address therefore of NOR must change from 1 to 6. The algorithm is searching the NOR from 1 to 8 until it locates the relative address 1 which exchange with the relative address 6.

Likewise, in the second step the column 3 of the matrix must move to the second relative address. Therefore a mutual exchange between rows 2 and 3 is taking place. In the NOR now a search is taking place until relative address 2 is located. This is at the eighth position and therefore the number 3 is placed at the eighth position. The same process is repeated until all the rows of the data matrix are rearranged according to the inverted relative addresses (NOR).

The choice of a sorting method must be such that the expected result of the repeated sorting to be achieved. In this case each set of data to be sorted (the data of a column) is not independent but rather it is bound with the data of the other columns of the matrix. Whenever duplication of values appears in the column to be sorted, these values must not be treated equally since their bound with the values of the neighboring columns differentiates them. In order to keep the order of the neighboring columns unaltered, the equivalued entries of the matrix must maintain their order after sorting. The following example clarifies the discussed concept. Suppose that we have the matrix of fig.7 and we

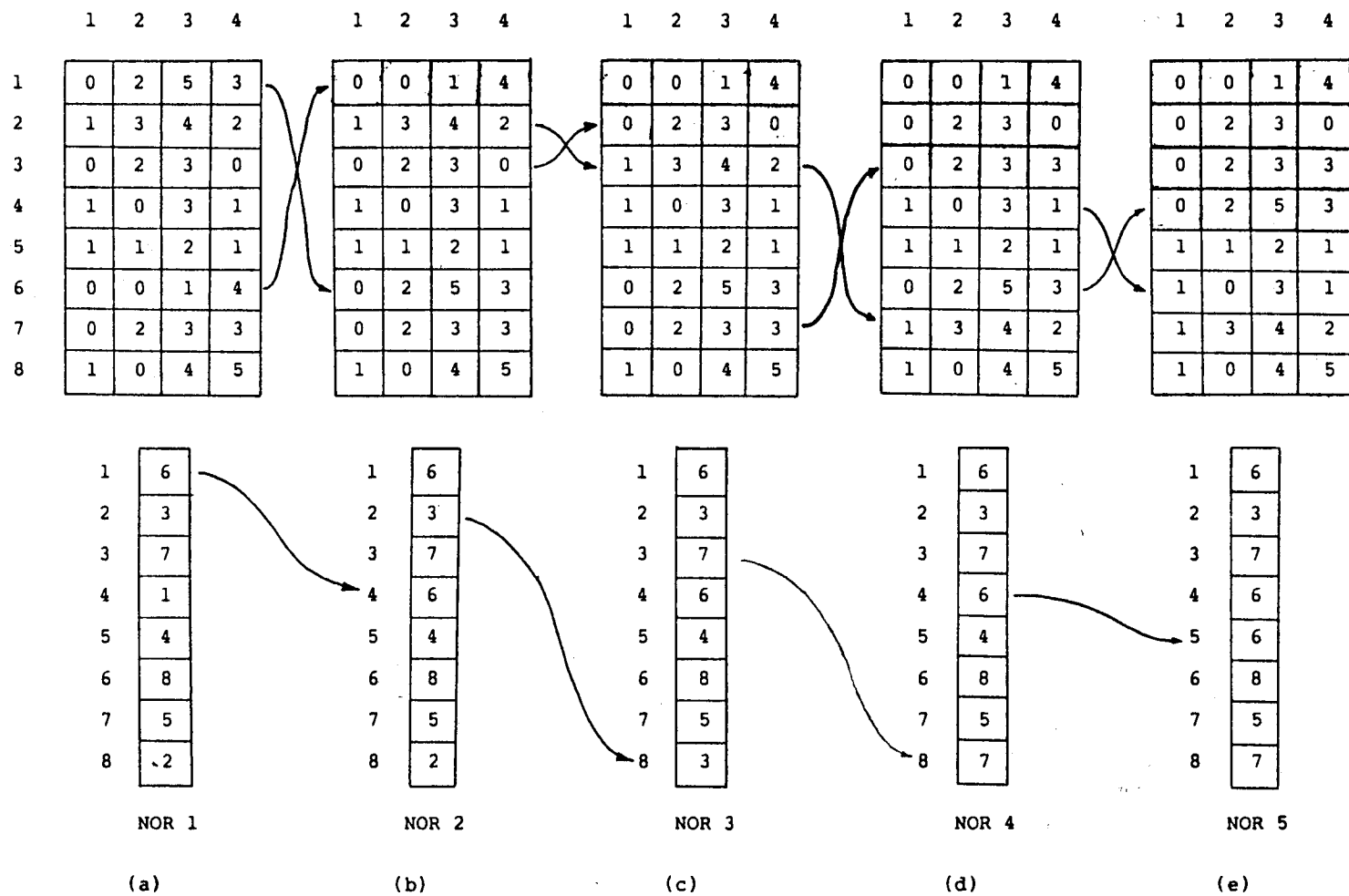


Fig. 6

Rearranging of the rows of data matrix
according the inverse relative addresses.

0	1	3	1	1	1	1	0	2	0	1	1
0	2	2	0	2	2	0	1	1	0	1	2
0	1	2	0	1	2	0	1	2	0	2	2
1	1	1	1	0	2	1	1	3	1	0	2
1	0	2	0	1	3	0	2	2	1	1	3
(a)			(b)			(c)			(d)		

Fig.7

repeatedly sort it according to columns 3,2,1 but considering the sequence of the duplicate values in each column. The resultant matrix in fig.7(d) has the desired property. Repeating now the same sorting but not considering the sequence of the duplicate values the result is different as illustrated in fig.8.

0	1	3	1	1	1	1	0	2	0	1	3
0	2	2	1	0	2	0	1	2	0	2	2
0	1	2	0	2	2	1	1	1	0	1	2
1	1	1	0	1	2	0	1	3	1	1	1
1	0	2	0	1	3	0	2	2	1	0	2
(a)			(b)			(c)			(d)		

Fig.8

The above consideration imposes limitations in the selection of the sorting method to be used. Considering only the speed and the consumed memory space in the method selecting process is not enough. The selected method must also reverse the existing order of the equivalued elements, for this reason, the Shell's and the Singleton's method must be left and the "Tree monkey puzzle"'s can be used.

3. FORTRAN 77 source code for multi-sorting :

```

C*****
C      SUBROUTINE REAR(LEC,IMP,NCOL,NROW,AR,NSORT)
C*****
C      Subroutine used to repeatedly call the subroutine EXTREE
C      for sorting according the column guides and then to
C      rearrange the data file according to the last NO()
C      (relative addresses for inverted data).
C
C      ARGUMENTS:

```



```

C   LEC   : Logical number for terminal input.
C   IMP   : Logical number for terminal output.
C   NCOL  : Number of columns ( 128 maximum).
C   NROW  : Number of rows    (2048 maximum).
C   AR    : Array containing the sorting column guides.
C   NSORT : The number of repeated sortings.
C
C*****
      IMPLICIT INTEGER*2 (I-N)

      INTEGER*2 AR(128),NO(2048),NOR(2048),DUM(2048)

      REAL*4 X(128),Y(128)

C   -----
C   Assign 'OLD' direct-access data file by any user sequence,
C   or statement OPEN() which records fixed length is NBYTES
C   and number is NROW and logical number LOGOLD.

      DATA LOGOLD/2/

      OPEN (....)

C   -----
C   -----
C   Phase 1 : Successive actualizations of NO() according to the
C   address calculations after each sort over.
C   For each sorting colum guide initialise the NOR to a natural
C   sequence of numbers (1,2,3,...,NROW).
C   -----

      DO I=1,NROW
         NOR(I)=I
      END DO

C   For each sorting column guide.

      DO J=1,NSORT

C   Read the data matrix according to NOR

         DO I=1,NROW
            READ(LOGOLD'NOR(I)) (X(K),K=1,NCOL)
            Y(I)=X(AR(J))
         END DO

C   Tree sort is needed for multi sorts, if the data are in
C   previous order, the user will "burst" them by a dummy
C   sort according to any unsorted column to restore the
C   performances of this method.

C   Call the "Tree monkey puzzle"'s method for sorting.

         CALL EXTREE(Y,NO,NROW,1)

C   Create the next NOR according to the produced by EXTREE NO().

```

```

DO L=1,NROW
  K=NOR(NO(L))
  DUM(L)=K
END DO
DO N=1,NROW
  NOR(N)=DUM(N)
END DO
END DO

```

```

C -----
C Phase 2 : Rearranging lines according to last NO(), without
C extra storage.
C -----

```

```

DO I=1,NROW
  READ (LOGOLD'I      ) (X(J),J=1,NCOL)
  READ (LOGOLD'NOR(I)) (Y(J),J=1,NCOL)
  WRITE(LOGOLD'I      ) (Y(J),J=1,NCOL)
  WRITE(LOGOLD'NOR(I)) (X(J),J=1,NCOL)
  J=I
  DO WHILE(NOR(J).NE.I)
    J=J+1
  END DO
  NOR(J)=NOR(I)
END DO
RETURN
END

```

4. Conclusion :

With such a set of tools, it is possible to proceed to numerical data multi-sorting organized in columns and rows, according to the guiding columns given by the user, without any intermediate manipulation of data and extra mass storage, with an unlimited number of keys.

With the algorithms given directly in a familiar structured language, the user can immediatly introduce in his/her own project.

5. References and bibliography :

GUINIER D. (1980) : Comparison of several sorting methods for numbers of characters running on a PDP11/05 under RT11 in FORTRAN IV language, DECUS RT11 SIG, Mini-Tasker, vol.6, no.2, 4-1980.

GUINIER D., TOTOS N. (1984) : - Sorting methods and evaluation -, Development of structured FOTRAN 77 subroutines for sorting by address calculation using deviations of the "Tree monkey puzzle"'s, Shell's and Singleton's methods and performances evaluation, DECUS VAX11 SIG and ACM SIGIR (submitted).

HOARE C.A.R. (1961) : Algorithms 63, Partition, and 64, Quicksort, Comm. ACM, 4 July, P.321.

SINGLETON R.C. (1969) : Algorithm 347, SORT, an efficient algorithm for sorting with minimal storage, Comm. ACM, vol.12, pp.185-186.

USER RESPONSES

NPE 6/023

Ken Demers
RT-11 SIG Newsletter Editor
MS-48
United Tech Research Ctr
Silver Lane
East Hartford, CT 06108
USA

18 May 1984

Concerning Ian Hammond's wishlist on page 3 of the January 1984 issue of the Mini-Tasker -

I think that DEC should have RT-11 master copies in Europe already set for 50 Hz, changing the GEN manual to reflect this and to show explicitly how to change the monitor frequency from 50 to 60 and from 60 to 50. A global symbol could be used in the source files, corresponding to "U.K." which affects modem support.

I too would like a KED start-up file, to be able to preset SET WRAP 70 and SET SECTION 10. I find that KED often rewrites the screen where a continuous or split scroll would be more congenial.

Yours sincerely

JR Stockton

J R Stockton (Dr)
Division of Electrical Science

DECUS LIBRARY

ADCON: A/D Conversions Package for Use with ADV11-C and K WV11-C

Version: V1.1, March 1984

Author: G.C. Scott, Trenton, NJ

Operating System: RT-11 V4.0

Source Language: FORTRAN IV, MACRO-11

Memory Required: 21KW

Other Software Required: DUMP Utility Program

Special Hardware Required: ADV11-C analog-to-digital conversion pc board, K WV11-C programmable realtime clock pc board

ADCON is a software package which can be used with Digital Equipment Corporation's ADV11-C analog-to-digital conversion and K WV11-C programmable realtime clock pc boards. The characteristics of the software include: (1) digitizations from 8 differential analog input channels; (2) up to 1KHz sampling rate, (3) software programmable gain, (4) 12 bit data resolution, and (5) DC offset corrections and data calibrations. The package includes the following: (1) DY1:READ.ME, user instructions and programming example, (2) DY1:ADCON.FOR, a FORTRAN IV program which sets up for the A/D Conversions, (3) DY1:CON.MAC, a MACRO-11 program which controls the A/D conversions, (4) DY1:CHANGE.FOR, a FORTRAN IV program which makes available the digitized data in decimal integer format and performs corrections and calibrations on the data, and (5) listing files for the above FORTRAN IV and MACRO-11 programs. Error checking is performed throughout DY1:ADCON.FOR and DY1:CON.MAC. Suggestions for software and hardware setup are included for users who require more customized A/D conversion schemes.

Changes and Improvements: Mistake in passage of digitizing rate from DY1:ADCON.FOR to DY1:CON.MAC corrected; sample run added to DY1:READ.ME and DY1:ABSTR.TXT so noted.

Documentation on magnetic media.

Media (Service Charge Code): Floppy Diskette (KA),
600' Magtape (MA)

Format: RT-11

Adventure

Version: January 1984

Author: W. Crowther, D. Sutton, K. Blackett, R. Supnik

Submitted By: Ian Calhaem, Ph.D., Dept. of Scientific and
Industrial Research, Wellington, New Zealand

Operating System: RT-11 V5.1 (PRO support requires V5.1 or later)

Source Language: FORTRAN IV, MACRO-11

Adventure is a magical, unpredictable and often addicting computer game that has caught on in the United States in now epidemic proportions.

It is a treasure hunt with all the trimmings - mysteries and challenges that grow more and more complex as the game unravels. Adventure is more of a puzzle than a game. Once solved, it's mastered! The mastering, however, often takes months of drawing maps and planning strategy.

Adventure's sweeping popularity lies in the power to enchant. Players are projected in to a world of fantasy, one that blends the heart-pounding suspense of Treasure Island with the magic of Alice in Wonderland.

Restrictions: Requires XM monitor

Documentation on magnetic media.

Media (Service Charge Code): 5 1/4" Floppy Diskette (JA)

Format: RT-11

BUPRES: A Program Which Reads Individual Files from a Magtape

Version: February 1984

Author: Michael L. Johnson, University of Virginia,
Charlottesville, VA

Submitted By: Susan G. Frasier, University of Virginia,
Charlottesville, VA

Operating System: RT-11 V5 required, TSX-PLUS

Source Language: FORTRAN-IV, MACRO-11

Memory Required: 22KB

Other Software Required: EXTMT (DECUS No. 11-337), a copy of which is included in this distribution.

Special Hardware Required: MT Magtape Drive

BUPRES is a FORTRAN IV program which reads individual files from a magnetic tape (MT:) created by the RT-11 version 5 BACKUP/DEVICE command.

The program prompts the user for all answers. Typically, the answer will be yes or no (Y or N is all that is required and a return is an N). The only other type of answer is the output file name. A return will use the original file name. No wildcards are allowed.

Warning, the program does NOT check for already existing files by the same name! Furthermore, the program does NOT protect restored files which were originally protected. Supports only single volume tapes!

The .SAV file is from TSX-PLUS Version 4.1 and FORTRAN IV Version 2.5. Use the .COM file to build your own, if you want.

Documentation on magnetic media.

Media (Service Charge Code): Floppy Diskette (KA),
600' Magtape (MA)

Format: RT-11

CONEFR.BAS: Cone Frustrum Layout

Version: February 1984

Author: Fred Fortman, National Metal Fabricators, Elk Grove
Village, IL

Operating System: RT-11 V2, V3

Source Language: BASIC-11

Memory Required: 16K minimum

The program "CONEFR" (Cone Frustrum) is for obtaining all the data needed to layout any size cone frustum which has a common centerline through the large and small diameters. The data the program requires is self explanatory. It will ask for the material thickness, diameters, vertical height, slant height, or the angle off the vertical centerline of the side. Any three of these are all the program requires.

There are then options as to the number of segments you wish to make or the size material you wish to use. Using the size option the program will give you the number of segments needed.

The data will be shown on the CRT and, if required, will produce a hard copy of the data on the line printer.

Documentation on magnetic media.

Media (Service Charge Code): Floppy Diskette (KB),
600' Magtape (MA)

Format: RT-11

Compendium Tape from the Australian RT-11 SIG

Version: Spring 1980 - Fall 1983

Author: Various

Submitted By: R.N. Caffin, CSIRO Textile Physics, Ryde NSW,
Australia

Operating System: RT-11 V4, V5

Source Language: BASIC-11, C, FOCAL, FORTRAN IV, MACRO-11, PASCAL

This is a collection of software acquired by the submitter over the years from various sources. Some of it comes from other SIG tapes from past DECUS symposia, some of it comes direct from various authors around the world. It is arranged as a series of .DSK files which may be treated as logical disks by LD or XD, or they may be copied as device images to RX01 discs. Included on the tape are a couple of non-DSK files which summarize and index the contents of most of the DSK files.

No guarantees are made as to the completeness, usability or quality of the programs on the tape. The material has not been checked or reviewed and documentation may or may not be included.

Sources may or may not be included. Documentation may or may not be included on the magnetic media.

Media (Service Charge Code): 2400' Magtape (PC)

DBSMNG: Small Database Management Package Binary Kit

Version: V2, November 1983

Author: R. R. DiMarco, South East QLD Electricity Board,
Brisbane, Australia

Operating System: RSTS/E V7 under RT-11 emulator, RT-11 V4,
TSX-PLUS V3

Source Language: MACRO-11

Memory Required: 20KB

Special Hardware Required: EIS

The DBSMNG package was developed from a suite of modules used to facilitate file I/O and complex data manipulation; as the number of modules in the package expanded, management routines were added to combine the individual modules into programs to perform commonly used functions; especially those needed to create, maintain and exploit small databases.

DBSMNG is designed to support small, homogeneous databases (i.e., of approximately 4000 records), and allows databases to be quickly established and easily maintained. Included in the package are:

DBSEDT ... a screen oriented editor
REPORT ... a report writer
INSPEC ... an on-line query program
BACKUP ... an incremental backup utility
MENU ... a menu program...and

other utility programs, and FORTRAN/ASSEMBLER callable routines that allow user application programs to interface with a database.

Restrictions: Binary kit only. For sources refer to Small Data Base Management Package Source Kit, DECUS No. 11-529.

Documentation on magnetic media.

Media (Service Charge Code): RX02 Floppy Diskettes (KB),
600' Magtape (MA)

Format: RT-11

Data Base Management Package, Source Only, for the Professional-300 Series

Version: V2, February 1984

Author: R. DiMarco, South East Queensland Electricity Board,
Brisbane, Australia

Submitted By: Ian Calhaem, Ph.D., Dept. of Scientific and
Industrial Research, Wellington, New Zealand

Operating System: RT-11 V5.1 (PRO support requires V5.1 or later)

Source Language: MACRO-11

Memory Required: 20KW

Special Hardware Required: VT52, VT100, ISC or 7VI912 terminal

This package contains all the sources for the Data Base Management Package version (see abstract for DECUS No. 11-529). The manual is NOT included here and must be obtained by ordering No. 11-529 (EB), which also includes demonstration packages.

Media (Service Charge Code): Floppy Diskettes (JC)

Format: RT-11

Data Base Management Package, Binary Kit Only, for the Professional-300 Series

Version: V2, February 1984

Author: R. DiMarco, South East Queensland Electricity Board,
Brisbane, Australia

Submitted By: Ian Calhaem, Ph.D., Dept. of Scientific and
Industrial Research, Wellington, New Zealand

Operating System: RT-11 V5.1 (PRO support requires V5.1 or later)

Source Language: MACRO-11

Memory Required: 20KW

Special Hardware Required: VT52, VT100, ISC or 7VI912 terminal

The database package was designed to allow small, homogeneous databases to be quickly set up and manipulated. The package provides the user with the following facilities:

1. An extremely simple method of defining the structure of the records which make up the database.
2. A screen oriented database editor, which allows records to be created, edited and deleted. The editor supports protected and database managed fields. The editor was designed so that persons with little computer experience (i.e., typists, clerks, etc.) can use it.
3. A report generation package, which allows specific records to be selected, and reports generated. The records to be included in the report can be sorted by any field within the record, and the report format can be easily modified.
4. A FORTRAN interface which allows records in the database to be readily accessed from a FORTRAN mainline program. Fields within a record can be accessed via their logical names. A FORTRAN interface is also provided to the sorting module, in the event that the user needs to implement a special application program. The FORTRAN interface also allows the user to access two or more databases simultaneously.
5. A menu program is included to allow the user to change between the various application programs without the knowledge of their computer interfacing.

This version (V2) of the Data Base Management Package includes an easier method of defining and establishing a database, and a much larger manual. The procedure discussed in the manual are illustrated in a pre-built database for managing telephone information which is included.

Restrictions: The sources are not included in this package. To obtain the sources order DECUS No. PRO-119.

Documentation on magnetic media.

Media (Service Charge Code): Floppy Diskettes (JB)

Format: RT-11

new
11-695

DBSMNG: Small Database Management Package Binary Kit

Version: V2, November 1983

Author: R. R. DiMarco, South East QLD Electricity Board,
Brisbane, Australia

Operating System: RSTS/E V7 under RT-11 emulator, RT-11 V4,
TSX-PLUS V3

Source Language: MACRO-11

Memory Required: 20KB

Special Hardware Required: EIS

The DBSMNG package was developed from a suite of modules used to facilitate file I/O and complex data manipulation; as the number of modules in the package expanded, management routines were added to combine the individual modules into programs to perform commonly used functions; especially those needed to create, maintain and exploit small databases.

DBSMNG is designed to support small, homogeneous databases (i.e., of approximately 4000 records), and allows databases to be quickly established and easily maintained. Included in the package are:

DBSEDT ... a screen oriented editor
REPORT ... a report writer
INSPEC ... an on-line query program
BACKUP ... an incremental backup utility
MENU ... a menu program...and

other utility programs, and FORTRAN/ASSEMBLER callable routines that allow user application programs to interface with a database.

Restrictions: Binary kit only. For sources refer to Small Data Base Management Package Source Kit, DECUS No. 11-529.

Documentation on magnetic media.

Media (Service Charge Code): RX02 Floppy Diskettes (KB),
600' Magtape (MA)

new
PRO-109

DECOD2 for the Professional-300 Series

Version: V02, December 1983

Author: Yukoh Kobayashi

Submitted By: Ian Calhaem, Ph.D., Dept. of Scientific and
Industrial Research, Wellington, New Zealand

Operating System: RT-11 V5.1 (PRO support requires V5.1 or later)

Source Language: MACRO-11

DECOD2 is a program written to run under the RT-11 operating system to decode (DISASSEMBLY.SAV and .LDA files) into their equivalent MACRO-11 assembly code complete with self generated labels, absolute location, symbol table, cross reference table, octal, ASCII and RADIX 50 equivalents.

DECOD2 is a modified and improved version of the DECODE V01J program, which was written by Michael Levine at the Naval Weapons Center.

The output can be assembled by the MACRO-11 assembler and linked to recreate the original file by the use of switches the output code can be made to reflect the instruction set of one machine configuration only instead of all possible instructions.

Documentation on magnetic media.

Media (Service Charge Code): 5 1/4" Floppy Diskette (JA)

DECODE4: RT-11 SAV Files Disassembler

Version: February 1984

Author: Henry O. Peterson, Bend, OR

Operating System: RT-11 V3, HEATH HT-11 H101A-5

Source Language: MACRO-11

Memory Required: 5.4KW

Other Software Required: RUNOFF (DECUS 11-530), documentation is in RUNOFF format

Now, even if your computer is the result of a pact between the fortress at Maynard and an entity 1500 km to its west (and slightly left); even if as might be expected its software matured well before the LSI-11 revolution you can more likely, using that software, get away with fitting the decoder to and using it more efficiently with that computer.

No more are your programs at the mercy of unpredictable or underdeveloped foreign systems! Now you can render such a system user friendly (as well as capitalize on it). Or one might become ensnared by the pages of Chapter 11. Perhaps it might be more feasible to negotiate. Though KGB one, there is a lot of exacting and tedious work to espionage.

DECODE 4.0 is a modified version of DECODE 3.0 (DECUS No. 11-342). The file is stored in DECOD4.MAC and DECOD4.SAV. The program provides some additional features over version 3.0 at the expense of being initially somewhat awkward to use. DECOD4 is not intended to totally replace DECOD3. In some cases, DECOD3 might be preferable.

Changes and Improvements: Assemblies with HEATH HT-11 System and Utilities, uses separate file to supply repetitive command line information, listing controls, etc.

Restrictions: Has not been checked with LDA files.

Documentation on magnetic media.

Media (Service Charge Code): Floppy Diskette (KA),
600' Magtape (MA)

Format: RT-11

DIBOL '83 Screen Handler Package

Version: V4.0, February 1984

Author: David L. Wyse, Projects Unlimited, Inc., Dayton, OH

Operating System: RSTS/E V8, RSX-11M-PLUS V2.1, Micro/RSX V1.0,
RT-11 V5.0, CTS-300 V8.0, CTS-500 V5.0, VAX/VMS V3.2

Source Language: DIBOL

Other Software Required: DIBOL '83 Compiler

This is a Screen Handling package written in DIBOL '83 and is transportable across all DIBOL supported operating systems. The package consists of three DIBOL '83 subroutines: "DISPL", a screen display subroutine which allows full use of VT100 type advanced video, line and special character drawing features and will format numeric fields with decimal points; the "ACCP" which accepts input from a VT100 type terminal including decimal point numeric fields, using the optimized IO features of DIBOL '83; and "CENT" which uses the DISPL subroutine to display a literal in the center of a line with the capability of using the advanced and special video features of VT100 type terminals.

Changes and Improvements: Subroutines have been updated to use DIBOL '83 features including improved screen IO and faster execution times.

Documentation on magnetic media.

Media (Service Charge Code): Floppy Diskette (KA) Format: RT-11,
600' Magtape (MA) Format: DOS-11

ELLIPTIC: Elliptic Integrals and Functions

Version: V2.7, March, 1984

Author: Donald Polzin, Universitaet Kiel, Kiel, West Germany

Operating System: RT-11, MUMPS-11

Source Language: FORTRAN IV

Elliptic consists of a set of FORTRAN callable subroutines which calculate complete elliptic integrals of first, second and third kind and jacobian elliptic functions am, sn, cn, dn for real modulus and argument.

Documentation on magnetic media.

Media (Service Charge Code): Floppy Diskette (KA),
600' Magtape (MA)

Format: RT-11

File Manipulation Commands

Version: V1.0, November 1983

Author: John Alexander, Shiva Associates, Sepulveda, CA

Operating System: RT-11

Source Language: TECO-11

Other Software Required: TECO-11

This is a group of file manipulation "Executives" that is intended to allow the user to make "global" changes to all lines of a file. The "executives" call up TECO files to perform the work. The TECO executive gets the name of modules and creates further com lines that operate on the specified files. This can be very useful to

the user that desires to utilize the code that has already been written and write supporting documentation, or to manipulate data files, etc. Some examples of "operations" that can be performed are: 1) Number all lines of a file, 2) Move a set of columns to a new column location for all lines, 3) Pad out the end of line to a given column, 4) Cut excessively long lines to a given column, 5) Fix all lines (long or short) to a given column, 6) Eliminate all "tabs" from a file, 7) Replace spaces with tabs, where possible, 8) Shuffle two files together, 9) Combine columns of one file with columns of another file, 10) Strip comments out of code, and 11) Strip code out of comments, etc., etc. To execute any of these "executives", type in "@" in front of the exec name and a carriage return. The executive's will prompt the user for input.

Restrictions: Dependent upon RT-11 Executives that utilize TECO.

Documentation on magnetic media.

Media (Service Charge Code): Write-Up (AA), Floppy Diskettes (KB), 600' Magtape (MA)

Format: RT 11

new
PRO-114

FODT: FORTRAN-IV On-Line Debugging Tool for RT-11, for the Professional-300 Series

Version: December 1983

Author: David Beckwith and Michael Kawasaki

Submitted By: Ian Caihaem, Ph.D., Dept. of Scientific and Industrial Research, Wellington, New Zealand

Operating System: RT-11 V5.1 (PRO support requires V5.1 or later)

Source Language: MACRO-11

This debugging aid allows the user of RT-11 FORTRAN-IV to monitor the operation of his program using commands similar to ODT. FODT provides a subset of the commands available with ODT such as single stepping, breakpoints and inspection of several types of variables. FODT can be built to a task without editing or recompiling the source and does not interfere with the normal operation of FORTRAN IV.

Documentation on magnetic media.

Media (Service Charge Code): 5 1/4" Floppy Diskette (JA)

FORTTRAN Callable Subroutines Package for Fast Continuous A/D on the MINC

Version: V1.0, November 1983

Submitted By: Digital Equipment Corporation

Operating System: RT-11 V5.0

Source Language: FORTRAN IV, MACRO-11

Memory Required: 280 (decimal) Words

Other Software Required: MACRO-11 Assembler, RT-11 O/S, FORTRAN IV compiler

Special Hardware Required: MNCAD-MINC A/D Module, MNCKW-MINC Clock Module

Assembly routines, ADCONT, WAITFD, and STOPIT, constitute a FORTRAN callable package capable of providing dedicated, continuous (buffer management with transfer to peripheral storage) analog-to-digital acquisition at rates two to eight times faster than the maximum rates provided by REAL-11 routines in a fraction of the memory space. The interface is also less complex and was modeled after MINC BASIC. Experienced users should be able to modify the sources for use with K and V series logic running under RT-11. A programmable clock and A/D modules are required.

Documentation on magnetic media.

Media (Service Charge Code): Floppy Diskette (KA),
600' Magtape (MA)

Format: RT-11

Indirect Command File Processing for RT-11 V4.0

Version: February 1983

Author: Russell L. Morrison II, Plessey Peripheral Systems,
Irvine, CA

Operating System: RT-11 V4.0

Other Software Required: RT-11 V4.0 Autopatch Revision D

Special Hardware Required: Line Time Clock

This manual consists of a description of a patch to the RT-11 V4.0 Indirect Command File Processor from Autopatch Revision D. Users may patch IND.SAV which is found in the RT-11 Autopatch Revision D in the manner described in the manual. IND.SAV provides RT-11 with RSX-like command files, having features such as parameter substitution, terminal input ("ASK"), limited math, and flow control (".GOTO" and ".GOSUB").

Note: The media contains a manual only.

Restrictions: The patch described in this manual is operating system dependent. It will only work with IND.SAV from RT-11 V4.0 Autopatch Revision D. In order for the patch to be effective, RT-11 must be SYSGENed with timer support in the Single Job (SJ) monitor.

Media (Service Charge Code): Floppy Diskette (KA),
600' Magtape (MA)

Format: RT-11

MJCLOCK: Multi-Job Clock Program

Version: V01, February 1984

Author: Stephen Cribbs, Atomic Energy of Canada, Pinawa, Canada

Operating System: RT-11 V4

Source Language: MACRO-11

Memory Required: 1K Words

Other Software Required: This program requires the use of either the RT-11 Foreground/Background Monitor (FB) or the Extended Memory Monitor (XM).

Special Hardware Required: VT100 terminal

MJCLOCK is a foreground or system job utility that displays current calendar and time values, and user defined messages, on a VT100 RT-11 console terminal in a manner transparent to the background job. This program demonstrates the feasibility of routing output to the console from foreground or system jobs through background job completion routines thus avoiding terminal context switching. MJCLOCK has the facility to avoid conflicts with background jobs that require full use of the video display.

Note: MJCLOCK requires the programmed requests available under RT-11 V3 and V4.

Documentation on magnetic media.

Media (Service Charge Code): Manual (EA), Floppy Diskette (KA),
600' Magtape (MA)

Format: RT-11

Monitor Commands for Namelists Package

Version: V1.0, November 1983

Author: John Alexander, Shiva Associates, Sepulveda, CA

Operating System: RT-11

Source Language: TECO-11

Other Software Required: TECO-11

This is a group of keyboard monitor "Executives" that is intended to allow the user to utilize a "Namelist" file to perform keyboard monitor commands, and others on a group of modules. The "executives" call up TECO files to perform the work. The TECO executive gets the namelist of modules, or single files, and creates further com lines that operate on the specified files.

This can be very useful to the user that has 5 to 105 modules that make up a major program. A typical executive is one that will merge files into a single file to allow global edits. When finished the user may then utilize split .N to split them out again. To execute any of these "executives", type in "@" in front of the executive name and a carriage return. The executive's name will prompt the user for inputs.

For example: To create an alphabetized name list of files type in @NAM.N To see a directory of the available executives type in @D.N. To copy a group of modules in a namelist to a device type in @COPY.N, etc.

In general any executives that end in "E" are more general executives, e.g., COPYE.N allows the user to specify a different file type to copy than the module names have in the name list file. RENAME.N allows a source namelist and a destination namelist, etc.

Restrictions: Dependent upon RT-11 Executives that utilize TECO.

Documentation on magnetic media.

Media (Service Charge Code): Write-Up (AA), Floppy Diskette (KA),
600' Magtape (MA)

One-Drive Diskette and File Copy

Version: March 1984

Author: Henry O. Peterson, Bend, OR

Operating System: RT-11 V3, Heath HT-11 H101A-5 USCD PASCAL
Version II.0

Source Language: MACRO-11, USCD PASCAL

Memory Required: 8KW

Other Software Required: For PASCAL programs: USCD PASCAL, Version II.0 including the means to: Copy from RT-11, compile and assemble. For documentation: RUNOFF with hyphenation (DECUS No. 11-530).

Special Hardware Required: At least one RX01 floppy drive with: CSR at 177170, DBR at 177172, octal. One drive must be DX0.

These programs are intended to allow, on a Heath HT-11, or DEC PDP-11 V03 system which uses RX01 floppy drives; copying of files or diskettes with only one drive (DX0:) working. Additionally, if you also have USCD PASCAL, version II.0, on the same computer those text files can be copied between the same RT-11 or HT-11 system with the same one drive working. In some cases, read-after-write is used. Also, in case of some errors, recopying of previously copied data may not be necessary.

It seems that these programs may be especially useful if you have purchased a system for which you undertake most repairs yourself. In the case of a hardware problem with a floppy drive, these programs may allow the use of your system, if it can function with the remaining drive, while the faulty drive is being repaired.

Restrictions: There is a risk of diskette damage while using these programs if any of the following occurs: an incorrect diskette is written on, certain characters are typed, or a hardware error occurs. One objective in writing these programs was to minimize such risk.

Documentation on magnetic media.

Media (Service Charge Code): Manual (EA), Floppy Diskette (KA),
600' Magtape (MA)

Format: RT-11

PRINIT: Printer Initialization Routine for the LA120 Printer

Version: V01.0, February 1984

Author: Kirk R. Stauffer, Copperweld Robotics, Madison Heights,
MI

Operating System: RT-11 V5.0

Source Language: MACRO-11

Memory Required: 1.5 KW

Special Hardware Required: LA120 Printer

This program remotely configures a DEC LA120 printer from a system terminal. Upon entry, the printer is initialized with default values for the horizontal and vertical pitch, form length and margins. The operator is then able to modify any of these four parameters from the terminal.

Documentation on magnetic media.

Media (Service Charge Code): Floppy Diskette (KA),
600' Magtape (MA)

Format: RT-11

RUNOFF, Bonner Lab

Version: V4.0, January 1984

Author: John Clement, Ph.D., Rice University, Houston, TX

Operating System: IAS, RSX-11M V3.2, RT-11 V4.0, VAX/VMS V3.1

Source Language: MACRO-11

Memory Required: 32KB

Bonner Lab RUNOFF is a text processor which when used with your favorite editor makes a complete word processor. Its syntax is almost a complete emulation of DSR (Digital Standard Runoff) and it is very compatible with previous versions of RUNOFF. The intent of this program is to support complete scientific word processing to produce publication quality output. It has been used to produce thesis, progress reports, and scientific papers here at Rice University.

This version allows complete control of any special printer available via user definable escape sequences. In addition, a macro facility allows text or sequences of commands to be abbreviated to a single label. If the printer has the correct features then variable spacing, subscripting, superscripting, and equation formatting are possible. By properly defining escape sequences, the user may support different printers in a transparent fashion. In other words the same input text will print in identical fashion on different printers with different control codes and escape sequences.

Table of contents, indexing and sub-indexing are all supported. A variety of LAYOUT and STYLE commands may completely change the look of the pages. Table layout is simplified by right justified and 'decimal' justified tabs. Permanent margins which apply to the page headers are available. These simplify the setting of header and text margins.

RNO occupies a minimum of 31 to 37 kbytes depending on the operating system.

Documentation on magnetic media.

Media (Service Charge Code): 500' Magtape (MA)

Format: DOS-11

STONE: A Program for Resolving Mossbauer Spectra

Version: July 1983

Author: A.J. Stone, K.M. Parkin and M.D. Dyar

Submitted By: M. Darby Dyar, Massachusetts Institute of
Technology, Cambridge, MA

Operating System: RT-11 V4.0, MINC V1.2

Source Language: FORTRAN IV

Memory Required: 40KB

Special Hardware Required: (Two) RX02 floppy Disk ports or
storage greater than 1600 blocks

This program is an overlaid, condensed version of the larger program MOSSPEC, which is in use worldwide. It fits a sum of Lorentzian or Lorentzian/Gaussian combined lines to a given Mossbauer spectrum by means of the Gauss non-linear regression procedure with a facility for constraining any set of parameters or linear combination of parameters. Results are output as a table of the fitted parameters, including the statistical values for standard deviation, x^2 , and Misfit.

This program was developed to enable sophisticated Mossbauer curve-fitting to be done on a very small computer, such as would be present in a laboratory environment. Use of the detailed User's Manual is strongly recommended.

Note: Use of the accompanying manual is strongly recommended.

Restrictions: Program requires approximately 900 blocks of free space for swapping on and off the disk; therefore it can only be run off RX02 (double density) floppies: one floppy for the operating system +STONE.SAV and one floppy for the few .TMP files and the empty blocks (contiguous). This program can also be run off any storage medium with a total of 1600 blocks.

Media (Service Charge Code): Manual (EA), RX02 Floppy Diskette (KA), 600' Magtape (MA)

Format: RT-11

Symposium Tape from the European RT-11 SIG, 1982, Warwick

Version: Fall 1982

Author: Various

Submitted By: Ray Carpenter

Operating System: RT-11

Source Language: Various

This symposium tape from the European RT-11 SIG is a collection of programs collated at the DECUS Europe Symposium held in September 1982, in Warwick.

No guarantees are made as to the completeness, usability, or quality of the programs on the tape. The material has not been checked or reviewed and documentation may or may not be included.

Restrictions: Sources may or may not be included.

Media (Service Charge Code): 2400' Magtape (PS)

Format: RT-11

TCFL: Terminal Control Function Library

Version: V1.0, July 1983

Author: Stephen Cribbs, Atomic Energy of Canada Limited, Pinawa, Canada

Operating System: RT-11 V4 or later

Source Language: MACRO-11

Memory Required: Varies with features required

TCFL is a subroutine library that provides RT-11 programmers with a convenient method of controlling "smart" terminals. FORTRAN callable routines have been provided for the issuing of most ANSI terminal control sequences and also for the receipt and interpretation of response from the terminal. RT-11 multi-terminal or single console I/O options can be selected during program assembly.

Documentation on magnetic media

Media (Service Charge Code): Floppy Diskette (KA),
600' Magtape (MA)

Format: RT-11

TECO Full Screen Editor

Version: March 1984

Author: Paul Mennen, Palo Alto, CA

Operating System: RSX-11M V4.0, RT-11 V4, TSX-PLUS

Source Language: TECO

Other Software Required: TECO-11 version 35 or later

Special Hardware Required: VT100 compatible terminal, Heath H19 terminal

This package consists of two TECO macros which provide the TECO user with the convenience of a full screen editor with a clean user interface, while retaining the full power and generality of TECO. One of the macros requires a VT100 compatible terminal and the second macro requires a Heath H19 or a Zenith Z19 terminal. The macros are well commented, which makes them more easily adapted to other terminals or other special requirements. These macros have been fully tested under the following configurations:

- 1). RT-11SJ version 4 with TECO-11 version 36
- 2). TSX-PLUS version 3.1 with TECO-11 version 36
- 3). RSX-11M version 4.0 with TECO-11 version 35

However, it should also run with most RT-11 and RSX releases and with other operating systems which support TECO-11. Requires TECO-11 version 35 or later. Supplied in RT-11 format, but instructions for converting it to RSX format via FILEX are included.

Changes and Improvements: Changed slightly for TSX-PLUS compatibility, fixed minor bugs, improved manual.

Restrictions: Requires TECO-11 version 35 or later. See DECUS No. 11-333.

Documentation on magnetic media.

Media (Service Charge Code): Manual (EA), Floppy Diskette (KA),
600' Magtape (MA)

Format: RT-11

TRAMP for RT-11

Version: November 1983

Author: F.A. Minkema and R. Beetz

Submitted By: R. Beetz, Akzo Pharma, The Netherlands

Operating System: RT-11

Source Language: FORTRAN IV

Memory Required: 58KB

Other Software Required: RT-11 SYSLIB

Special Hardware Required: VT100 series

TRAMP software contains the tools for developing menu controlled applications, form applications, and report applications. TRAMP now brings the speed, convenience, accuracy, and low cost of computerized processing to users who, in the past have been building the same kind of controlling mechanism more than once. Menus, forms, and reports are designed by the use of a description language. TRAMP associates constant data with the menu, form and report layout, not with the application program, resulting in simplified application program maintenance and increased application program flexibility. Some menu, form, and report layout parameters can later be modified without the need to recompile the application program. A complete set of program documentation is delivered automatically.

TRAMP software has three main components for developing and executing application programs:

- menu, form, and report ASCII source file
- MPR, TFR, and RGR compilers for checking and converting source files and generating documentation
- driver subroutines

Additionally, there is a general purpose routine library RTLIB (or RSXLIB) that is used with or without TRAMP.

Associated Documentation: Proceedings from DECUS Europe 1983 p.305, DECUS No. PRC-EUR-83, Media Service Charge Code (YA).

Documentation on magnetic media.

Media (Service Charge Code): RX02 Floppy Diskettes (KB),
500' Magtape (MA)

Format: RT-11

TSXLIB: A Library Implementation of Programmed Requests for
TSX-PLUS V5.0

Version: 84a29a, January 1984

Author: N.A. Bourgeois, Jr., NAB Software Services,
Albuquerque, NM

Operating System: RT-11 V4.0, V5.0, TSX-PLUS V5.0 required

Source Language: MACRO-11

Like RT-11, TSX-PLUS offers the MACRO programmer a number of system services via programmed requests or EMTs. RT-11 makes its system services available to the FORTRAN programmer through the system subroutine library, SYSLIB. TSXLIB makes the TSX-PLUS EMTs available to the FORTRAN programmer as a library of callable routines. The package includes the MACRO source modules for all the routines, a user's manual, a cross reference chart, an indirect command file to build the library and the implemented library.

Changes and Improvements: The Library has been updated to include all EMTs through TSX-PLUS V5.0.

Documentation on magnetic media.

Media (Service Charge Code): Floppy Diskette (KA),
600' Magtape (MA)

Format: RT-11

RXP2

TXTWRT: An RT-11 Text Formatting Program

Version: V1.2, July 1983

Author: Stephen Cribbs, Atomic Energy of Canada Limited, Pinawa,
Canada

Operating System: RT-11 V4 or later

Source Language: MACRO-11, FORTRAN IV

Memory Required: 11.3KW

TXTWRT is an RT-11 text formatting program written to help programmers create effective text and graphics displays on today's "smart" terminals. In addition, full support of the ANSI standard escape sequence control strings used by Digital Equipment's video and printing terminals is provided. The output generated may be directly routed to the terminal, or to an ASCII stream file, or TXTWRT will create an output file suitably formatted as either FORTRAN IV or MACRO-11 source statements.

Documentation on magnetic media.

Media (Service Charge Code): Floppy Diskette (KA),
600' Magtape (MA)

Format: RT-11

VLISP-11

Version: March 1984

Author: Paul Mennen, Palo Alto, CA

Operating System: RT-11 V4 or later, TSX-PLUS

Source Language: MACRO-11mr

Memory Required: 16KW to 30KW

VLISP-11 is a complete LISP programming environment which will run on any model PDP-11 or LSI-11. The VLISP interpreter is written in MACRO assembly language and implements 160 LISP functions including several I/O functions providing access to the RT-11 file structure. The rest of the package is written in VLISP and includes a pretty-printer (with parenthesis matching and commenting facilities), a save utility, a list oriented editor, a debugger, a file transfer utility, and an infinite precision floating point package. Additional versions are included which utilize the EIS or EIS/FIS instructions.

Changes and Improvements: Deleted magnetic media documentation from distribution. Added and improved hard-copy manual. Fixed bug under RT-11FB, modifications for TSX-PLUS compatibility, fixed other minor bugs. Improved the infinite precision floating point package.

Documentation is not included on the magnetic media.

Media (Service Charge Code): Manual (EB), Floppy Diskettes (KC),
600' Magtape (MA)

Format: RT-11

Whetstone Benchmark

Version: February 1984

Author: Ethan Van Matre, Real Time Control, Portland, OR

Operating System: RSX-11M V3.2, V4, RT-11 V4

Source Language: FORTRAN IV

Other Software Required: FORTRAN Compiler

Special Hardware Required: Can be run without any extra math hardware. Requires line clock.

The Whetstone Benchmarks are well known and are accepted as well as any method of comparing computer performance. These FORTRAN programs are compute bound and heavy on the floating point. Both single and double precision Whetstone Benchmarks are included.

Documentation on magnetic media.

Media (Service Charge Code): Floppy Diskette (KA),
600' Magtape (MA)

Format: RT-11

WORD: Document Spelling Checker/Corrector

Version: June 1983

Author: R. R. DiMarco, South East Old Electricity Board,
Brisbane, Australia

Operating System: RSTS/E, RT-11, TSX-PLUS

Source Language: MACRO-11

Memory Required: 28KB

Special Hardware Required: EIS

The WORD package consists of a 15000 word dictionary file and a number of simple programs that allow the dictionary to be used to in the correction of spelling errors in documents. The major components of the kit are:

WORDS.DIC...dictionary file
WORDS.SAV...flags possible spelling errors
WORDE.SAV...corrects spelling errors
SPELL.SAV...finds correct spelling from abbreviation
SOUND.SAV...finds correct spelling from sound

Documentation on magnetic media.

Media (Service Charge Code): Floppy Diskettes (KB),
600' Magtape (MA)

Format: RT 11

9 FETCH.B

Version: January 1984

Author: Wayne Levine, Owatonna High School, Owatonna, MN

Operating System: RT-11

Source Language: MU-BASIC

Memory Required: 1500 Words

This program takes the place of an editor while you are in MU-BASIC. It can be used on data files, FORTRAN source files, text files and configuration files. This program actually writes a second program. The second program can be edited like a normal basic program, and when you run it your original files is recreated.

Documentation on magnetic media.

Media (Service Charge Code): Floppy Diskette (KA),
600' Magtape (MA)

Format: RT-11

Spring, 1984
Cincinnati, OH

Annotated Directory

IMPORTANT

Read the file, README.1ST, first.

README.1ST 11 15-Jul-84 SIG tape copy instructions
 and new information.

XX

David Stagg
Dept of Pharmacology
Yale University Medical School
333 Cedar Street
New Haven, CT 06510
(203) 436-2151

This is a subdevice handler. (necessary for RT-11 Version 4), used to read the .DSK files on this tape. See README.1ST for an explanation and instructions.

XD	.SYS	2	05-May-81	SJ/FB driver for V4
XDATCH	.SAV	21	17-Apr-81	XDATCH utility
DUP	.SXD	41	21-Sep-81	DUP V4.0K patched for XD
XD	.DOC	10	16-Nov-81	Description document
XD	.SYS	2	05-Apr-81	XM driver for V4
DUP	.COM	1	16-Mar-82	Patch file for V4 DUP.SAV

XD .DEV 55 06-Apr-83 VIRTUAL DEVICE for rebuilding XD,
if necessary.

[illegible]

VIRTUL - Subdevice retriever for RSTS.

E.F.Beadel, Jr., Manager
CAUSE Instructional Computer Center
SUNY at Oswego
Oswego, NY 13126
(315) 341-3055

This program allows RSTS/E users to break down the subdevice files from this tape after they have been copied to disk. See README.1ST for details.

VIRTUL.BAS 1 File, 28 Blocks

SPRING 1984 RT SIG TAPE DIRECTORY

DIR2 - Annotated tape directories, part 2.

N. A. Bourgeois, Jr.
Sandia National Laboratories
Division 5238
P. O. Box 5800
Albuquerque, NM 87185
(505) 844-8088

R. W. Barnard
Sandia National Laboratories
Division 7523
P. O. Box 5800
Albuquerque, NM 87185
(505) 844-5115

Annotated directories of the DECUS Symposia RT-11 tapes from the Fall of 1981 through the Spring of 1984 (this symposium). Previous tapes have DIR1.DSK, which contains annotated directories from Spring 1978 through Spring 1981.

DIR2.DSK 6 Files, 241 Blocks

BENCH* - FORTRAN-77 Benchmark Programs

Adam Bridge
Multiware, Inc.
139 G Street, Suite 161
Davis CA 95616
(916) 756-3281

These are a series of benchmark programs for RT-11 using the new FORTRAN-77 compiler. There are versions linked using F77, F4 with EIS, F4 with FPU, and F4 with no hardware support. The developers of RT FORTRAN-77 would be interested in knowing the results of benchmark tests on as many machines as possible. If you would like to comment, please contact the person listed above.

BENCH1.DSK 14 Files, 83 Blocks
BENCH2.DSK 13 Files, 441 Blocks
BENCH3.DSK 14 Files, 432 Blocks
BENCH4.DSK 14 Files, 403 Blocks
BENCH5.DSK 13 Files, 425 Blocks

BUPRES - Read Individual Files from a BUP Magtape

Michael L. Johnson
Susan G. Frasier
Dept. of Pharmacology
University of Virginia
Charlottesville, VA 22908
(804) 924-2496

BUPRES is a FORTRAN IV program which reads individual files from a magnetic tape (MT:) created by the RT-11 version 5 BACKUP/DEVICE command. The program supports only single volume tapes. The .SAV file is built for TSX-plus Version 4.1, and FORTRAN IV version 2.5, but can be rebuilt for other systems, if desired.

BUPRES.DSK 10 Files, 206 Blocks

SPRING 1984 RT SIG TAPE DIRECTORY

DSKLB* - Librarian for multiple disks.

Joel Berez
Infocom
64 Jacqueline Rd.
Waltham, MA 02154
(617) 492-1031

Disk Librarian (DSKLIB) is a program to maintain a master catalog of multiple disk directories. Disks are given unique numbers and names, and then placed in the master catalog. A powerful Find command is then used to locate a particular file. Full wild-cards and multiple switches are supported. The Archive Handler is included here for those persons using RT-11 Version 4 who need such a feature. It has capabilities which complement DSKLIB. The source to DSKLIB is included in this submission.

DSKLB1.DSK 6 Files, 475 Blocks
DSKLB2.DSK 11 Files, 40 Blocks

HGRAF* - Plotting Package for RT-11.

Dennis V. Jensen
Ames Laboratory ISU/USDOE
310 Metallurgy
Ames, IA 50011
(515) 294-4823

This is the second DECUS release of HGraph. It was coded entirely in DEC Fortran-66 compatible code. Included here is a new WRTSTR (extracted as WRTST2.*) which will generate characters as vectors; it is not dependent on the Visual 550 character generator. Also, there is a cubic spline interpolation line drawing routine (extracted as SMOOTH.*) which is compatible with the DASHLN calling convention. Release two is coded in FLECS. An RT-11 FLECS translator is also included for reference. PLOT3D.FOR is included as a separate file.

HGRAF1.DSK 4 Files, 267 Blocks
HGRAF2.DSK 3 Files, 356 Blocks
HGRAF3.DSK 9 Files, 180 Blocks

IMAGE* - RT-11 Natural Image Display Program

David Fingerhut
Digital Equipment Corporation
146 Main Street
Maynard, MA 01754
(617) 223-3342

This program allows the display of photographic (or other natural) images on the PRO PC350 computer under RT-11. Included in the submission are the source code to access the PRO graphics display, as well as several files of images.

SPRING 1984 RT SIG TAPE DIRECTORY

IMAGE1.DSK 6 Files, 221 Blocks
IMAGE3.DSK 2 Files, 360 Blocks
IMAGE4.DSK 2 Files, 360 Blocks
IMAGE5.DSK 2 Files, 360 Blocks
IMAGE6.DSK 2 Files, 360 Blocks
IMAGE7.DSK 2 Files, 360 Blocks
IMAGE8.DSK 2 Files, 360 Blocks
IMAGE9.DSK 2 Files, 360 Blocks
IMAGEA.DSK 2 Files, 360 Blocks
IMAGEB.DSK 2 Files, 360 Blocks

LA100G - Vector-to-Raster Translator for LA100 Graphics

Robert Hamilton
Multiware, Inc.
139 G St., Suite 161
Davis, CA 95616
(916) 756-3291

LA100G is a program that can read a file of formatted generic graphics instructions, translate the instructions into a disk-based "bitmap" file, and then create a hardcopy image on one of Digital's Letterprinter or Letterwriter 100 serial printers. The program features a dot resolution of up to 1024 by 1024, simulated hardware character generation, dotted and dashed line generation, and a flexible means for scaling and rotating an image. The input file format is easily user-generated by "post-processing" device or metafile output from most any popular graphics package. It can be read with standard FORTRAN READ and FORMAT statements.

LA100G.DSK 13 Files, 282 Blocks

MODEM - Files for Control of DZ(V)11 I/O Lines

Thomas J. Shinal
General Scientific Corp.
1681 East Gude Dr.
Rockville, MD 20850
(301) 340-2773

These files allow a DIBOL user to control the characteristics of the DZ11 and DZV11 serial interface boards. These include modem-controlling functions, such as DTR, etc. The routines are written in MACRO and DIBOL.

MODEM.DSK 7 Files, 25 Blocks

RESEQ - FORTRAN Program Line Number Resequencer

R. W. Barnard
Sandia National Laboratories
Division 7523
Albuquerque, NM 87185
(505) 844-5115

SPRING 1984 RT SIG TAPE DIRECTORY

This program is an enhancement and modification of one originally submitted to the RT-11 SIG tape in 1980 by Larry Morton. It will renumber the statements of FORTRAN-IV and FORTRAN-77 programs to give them a nice order. The starting statement number and increment are specifiable.

Also included are some SYSLIB routines for easier conversion to FORTRAN-77. SYSLIB has the string-handling routines INDEX and LEN, for byte variables. FORTRAN-77 has two intrinsic routines of the same names for character variables. To aid in the conversion of FORTRAN-IV routines to F-77, two routines (INDEXB and LENB), can be added to SYSLIB which have the same functionality as the SYSLIB routines INDEX and LEN, so all references in FORTRAN-IV programs can be changed to these new calls.

RESEQ.DSK 13 Files, 170 Blocks

RUNOF* - Bonner Lab RUNOFF, Version 5.1

John M. Clement
Bonner Lab, Rice University
P. O. Box 1892
Houston, TX 77401
(713) 527-4018

Bonner Lab Runoff is a text formatter which when used with your favorite editor makes a complete word processor. Its syntax is almost a complete emulation of DSR (Digital Standard Runoff) and it is very compatible with previous versions of Runoff. The intent of this program is to support complete scientific word processing to produce publication quality output.

This version allows complete control of any special printer available via user definable escape sequences. In addition a macro facility allows text or sequences of commands to be abbreviated to a single label. If the printer has the correct features then variable spacing, subscripting, superscripting, and equation formatting are possible. By properly defining escape sequences the user may support different printers in a transparent fashion.

Table of contents, indexing and sub-indexing are all supported. Multiple table of contents or multiple indexing may be constructed from the available commands. A variety of LAYOUT and STYLE commands may completely change the look of the pages. Table layout is simplified by right justified and 'decimal' justified tabs. Permanent margins which apply to the page headers are available. These simplify the setting of header and text margins. This runs under VMS, RSX, IAS, RT11, TSX, and probably RSTS.

RUNOF1.DSK 108 Files, 480 Blocks

RUNOF2.DSK 21 Files, 482 Blocks

RUNOF3.DSK 63 Files, 476 Blocks

RUNOF4.DSK 5 Files, 136 Blocks

SPRING 1984 RT SIG TAPE DIRECTORY

TSXLIB - Programmed Requests for TSX+

N. A. Bourgeois, Jr.
NAB Software Services, Inc.
P. O. Box 20009
Albuquerque, NM 87154
(505) 298-2346

This is an update of the TSX+ system services available via programmed requests. They are in the form of a library, available to the FORTRAN programmer. The library has been updated to include all EMT's through TSX+ Version 5.0.

TSXLIB.DSK 18 Files, 490 Blocks

UCLPLS - User Command Language (UCL) Program.

William K. Walker
Monsanto Research Corp.
P. O. Box 32 OS-123
Mammoth, OH 45342
(513) 865-3557

UCL+ is upward-compatible with the UCL distributed with RT-11, Version 5.01B. It contains a number of extensions, including chaining to additional UCL's, "run-by-name", path definition, display of command expansions, etc. Symbols are defined by entering a "symbol definition string" in the format:
symbol==definition.

UCLPLS.DSK 11 Files, 293 Blocks

UCLTSX - User Command Language (UCL) Program.

William Davidson
Sandia National Laboratories
Albuquerque, NM 87185

This is an implementation of a UCL which allows RT-11 command files to be treated in the same fashion as TSX+ command files. With this UCL, unrecognized keyboard entries are processed in the same order under RT as they would be under TSX+.

UCLTSX.DSK 15 Files, 100 Blocks

UNMAC - Object File to Source File Translator

Robert Walraven
University of California
Applied Science
Davis, CA 95616
(916) 752-3300

UNMAC is a FORTRAN program to convert an RT-11 object file into a source or listing file. Object files are much more useful

SPRING 1984 RT SIG TAPE DIRECTORY

to decode then SAV files because they contain information such as subroutine names, variable names, psect info, etc. This distribution contains complete sources and a command file for building UNMAC. The file UNMAC7.SAV is an executable version that was built with the FORTRAN-77 compiler.

UNMAC.DSK 14 Files, 297 Blocks

BIBLE*, BOOK** - The King James Version of the Bible

Submitted by:
Dean Lampman
STARFLO, Inc,
#5 Hickory Wood Drive
Cincinnati, OH 45140
(513) 683-4871

The files labeled BIBLE*.DSK are subdevices containing several books of the Bible. The files labeled BOOK**.LST are individual books. This was converted to magnetic media by an optical scanner. Users are cautioned that there may have been conversion errors.

BOOK01.LST 409 Blocks
BOOK02.LST 351 Blocks
BOOK03.LST 262 Blocks
BOOK04.LST 365 Blocks
BOOK05.LST 303 Blocks
BIBLEA.DSK 3 Files, 438 Blocks
BOOK09.LST 267 Blocks
BIBLEB.DSK 2 Files, 482 Blocks
BIBLEC.DSK 2 Files, 481 Blocks
BIBLED.DSK 2 Files, 371 Blocks
BIBLEE.DSK 3 Files, 381 Blocks
BOOK19.LST 475 Blocks
BIBLEF.DSK 3 Files, 259 Blocks
BOOK23.LST 402 Blocks
BOOK24.LST 461 Blocks
BIBLEG.DSK 2 Files, 462 Blocks
BIBLEH.DSK 13 Files, 460 Blocks
BIBLEI.DSK 2 Files, 426 Blocks
BIBLEJ.DSK 2 Files, 486 Blocks
BIBLEK.DSK 3 Files, 478 Blocks
BIBLEL.DSK 19 Files, 476 Blocks
BOOK66.LST 129 Blocks

The Spring, 1984, RT SIG tape contains 66 Files, 19823 Blocks.
It was prepared by:

It is distributed by:

R. W. Barnard
Sandia National Laboratories
Division 7523
P. O. Box 5800
Albuquerque, NM 87185
15-July-1984

Thomas J. Shinal
General Scientific Corp.
1681 East Gude Dr.
Rockville MD 20850
(301) 340-2773

"I have 512 KB on my PRO but my application gets a 'KMON-F insufficient memory error.' How come? I only had 56 KB on my LSI-11/2 and it worked fine."

RT was implemented on the PRO in a short period of time, and what was implemented is referred to a good old RT, or GORT. Development on RT for the PRO continues. The recommended XM monitor is going to continue to be cleaned up, both for efficiency and to solve the "alleged" low-memory problem. In the short term, however, there are some solutions which will get most applications running on the PRO. It is the developers' intent to ultimately have 99% of RT applications running on the PRO without modification.

In the meantime, you will need to relink your programs. The reason is that the LSI-11 environment saw you (anybody got a pointer to lend me?) with 50-51 KB in low memory to run your programs. On the PRO, memory is divided into a high memory portion and a low memory portion. The lower portion of it, for compatibility reasons, is kept looking more or less the same as it does on the LSI-11. The one significant difference is that the amount of memory the monitor uses, because the XM monitor is simply much larger, leaves you with less user memory available. In the worst case scenario approximately 28 KB is left for use as

true user space. In addition, if you start installing foreground, spoolers, communications packages, single-use editor, the foreground resource reporter, a copy of the text editor, not bad for a single user system I might add, you then wind up being able to run programs that are about 16 KB (compared to 51 KB).

All that we need to do is relink the program such that part of the program resides in lower memory, and part in upper memory. RT supplies that mechanism, and it is not all that hard to accomplish.

All of the solutions carry some restrictions. There are some restrictions to using the XM rather than the FB monitor that you should be aware of. In particular, the PAR1 restriction. PAR1 is the area in memory from 20000 octal to 40000 octal, and it is mapped by a particular memory management register, and the monitor uses that area to go around looking at other jobs. As a result there are two data structures that the monitor uses which are allocated in the job's address space. Those are the queue elements and the channel areas. Almost nobody allocates channels, so that is not usually a problem, but quite a few people allocate queue elements because those are required to do a real-time program. In general, however, programs are larger than 16 KB, and when they are the queue elements are allocated clear of this area and you probably will never ever see this problem. There are, however, a few programs that indeed do have queue elements allocated in the 20000-40000 range and the program, therefore, will fail.

This is a simple problem to solve... (Dan, don't hold that [a microphone] up, when I see it, it makes me nervous. Hide it.) It is very easy to simply place these somewhere else and in the calls that do it, the emulator traps and the system lib functions the provision is there to place the queue elements wherever you like.

Device handlers in RT-11: you do not want them around all the time so we have fetchable handler support. Handlers are brought into memory as they are needed and then they are released. Under the XM monitor one of the things that can happen is that the logic of the program you are transporting over can cause a device handler to be fetched into extended memory. The logic of the program may accidentally try to fetch the handler into extended memory. As a short-term solution, you can simply load the handler.

Chaining:

If you are making programs virtual, you have to really understand chaining and whether a particular program is chaining from a virtual job to a virtual job, a virtual job to a privileged job, and so on, because there is an addressing difference. In fact, the book says that we simply do not support chaining from a virtual job, and that is what we are going to tell you how to do. However, with a little proper placement of things, chaining from a virtual job can be made to work without too much difficulty.

Migrating a simple program:

The simplest kind of program is a program which is straight linked (linked such that program equals main with any external subroutine sources concatenated to the end). Normally, this program would be loaded and a routine called .SETTOP would set up a pointer to the free memory at the top of core. This is the typical sort of program and often it will not run when transported over directly, simply because either the program takes up too much memory or worse, because there is not enough scratch space for the job to run once loaded into memory.

This is an easy problem to solve. We can create pseudo-virtual overlays. Pseudo-virtual overlays are created first of all by creating a little dummy stub to use as the "main" program. Typically just a jump to the real main in MACRO. Under FORTRAN, for example, you simply make a call to the real main program, and then add a "CALL EXIT" just in case.

Instead of linking the program such that the files are concatenated, you link the program such that you create a "V:1" overlay:

```
.R LINK
*PROG=DUMMY/V/I/C
*MAIN/V:1/C

*SUB1/C
*SUB2/C
*SUB3
Name of module to include? $QBLK
```

The result is your program more or less as it existed in the previous case, with room for .SETTOP. The gaps in your virtual memory space are not a problem with physical memory utilization, but there is nothing we can do about it in your physical address space because it is a hardware limitation of the memory manager on your PDP-11. When this program is loaded, the dummy is loaded into low memory and the remainder is loaded into extended memory, as would the .SETTOP space be. By not using the V switch except for the MAIN specification, you specify that the trailing modules are to be concatenated to the "switched" module. If a "V:2" switch had been used in the above example, it would have forced common references into the root in low memory, and low memory would possibly become filled up. The problem is, if you have a big array or something, and that array is referenced by more than one virtual overlay, the big array is forced into the root and you lose a lot of your gain.

This solution creates a problem since fetched handlers are loaded into the .SETTOP area and we have just placed .SETTOP in extended memory. Fetched handlers cannot reside in extended memory. You can just load the handler and avoid all the problem, but that is not always pleasant for the end user. A small line of code has been added to the fetch code that says, "This address space is not available to you for use because it is, by default, not mapped. However, if we look at its physical equivalent we can guarantee that there will be nothing in memory from the dummy module up to the 4 K boundry. This memory is absolutely free, and you can find out the address of this memory by using the .GTJB programmed request. It will come back and it will tell you what the low memory high value is. What that means is, with just a few lines of logic that say, "Am I running under XM?" you have lots of room in there to fetch your device handlers. With languages which place their stack at the beginning of the program (and therefore in the root), the principle is still the same. You still create the dummy, there is simply less gain from the technique.

If you are using this technique with FORTRAN, you also have to link in the library module \$QBLK. If you are writing a MACRO program you would set the virtual bit in the job status word. If you are running Pascal or C, you can patch it in at 44:

```
.ASECT
.=44
.WORD 2000
.END
```

This last method will work with all RT-11 languages.



DECUS

DECUS SUBSCRIPTION SERVICE
DIGITAL EQUIPMENT COMPUTER SOCIETY
249 NORTHBORO ROAD, (BPO2)
MARLBORO, MA 01752

MOVING OR REPLACING A DELEGATE?

Please notify us immediately to guarantee continuing receipt of DECUS literature. Allow up to six weeks for change to take effect.

- ☐ Change of Address
☐ Delegate Replacement

DECUS Membership No.: _____

Name: _____

Company: _____

Address: _____

State/Country: _____

Zip/Postal Code: _____

Phone No.: _____

Mail to: **DECUS - Attn: Subscription Service**
249 Northboro Road, (BPO2)
Marlboro, MA 01752 USA

Bulk Rate
U.S. Postage
PAID
Permit No. 18
Leominster, MA
01453