GETTING STARTED WITH BASIC/RT-11 (V01B)

DEC-11-LBCLA-C-D

Order additional copies as directed on the Software Information page at the back of this document.

digital equipment corporation · maynard. massachusetts

First Printing, April, 1974
Revised, June, 1975

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this manual.

The software described in this document is furnished to the purchaser under a license for use on a single computer system and can be copied (with inclusion of DIGITAL's copyright notice) only for use in such system, except as may otherwise be provided in writing by DIGITAL.

Digital Equipment Corporation assumes no responsibility for the use or reliability of its software on equipment that is not supplied by DIGITAL.

The HOW TO OBTAIN SOFTWARE INFORMATION page, located at the back of this document, explains the various services available to DIGITAL software users.

The postage prepaid READER'S COMMENTS form on the last page of this document requests the user's critical evaluation to assist us in preparing future documentation.

The following are trademarks of Digital Equipment Corporation:

CDP	DIGITAL	INDAC	PS/8
COMPUTER LAB	DNC	KA10	QUICKPOINT
COMSYST	EDGRIN	LAB-8	RAD-8
COMTEX	EDUSYSTEM	LAB-8/e	RSTS
DDT	FLIP CHIP	LAB-K	RSX
DEC	FOCAL	OMNIBUS	RTM
DECCOMM	GLC-8	OS/8	RT-11
DECTAPE	IDAC	PDP	SABR
DIBOL	IDACS	PHA	TYPESET 8
			UNIBUS

CONTENTS

INTRODUCTION

This document introduces the BASIC/RT-11 V01B kit to users receiving it for the first time; it should be read completely before any action is taken with the BASIC/RT-11 kit, since it points out facts necessary for smooth installation of the software.

Chapter 1 explains the contents of the BASIC/RT-11 kit. Chapter 2 contains capsule descriptions for some of the software services available from DIGITAL. Chapter 3 contains step-by-step instructions for building BASIC/RT-11 from the distribution kit and for executing a simple demonstration program. Chapter 4 describes differences between BASIC/RT-11 V01B and BASIC/RT-11 V01-05. Finally, Chapter 5 contains release notes, documentation corrections, software corrections, and restrictions known at the time of system release.

Immediately upon receiving the BASIC/RT-11 software kit, this document and the BASIC/RT-11 Language Reference Manual (DEC-11-LBACA-D-D) should be read thoroughly. Once the contents of these manuals are understood, the BASIC/RT-11 Demonstration Package in Chapter 3 should be exercised. This demonstration will serve to build a working BASIC system from the distribution kits and will ease familiarity with BASIC's characteristics, since execution of a simple program is part of the procedure.

Included in this kit is the Software Performance Summary and recent editions of the Digital Software News; both documents list known software problems and solutions (if any) for PDP-11 software. These documents should be inspected carefully. The user should make the recommended modifications to BASIC/RT-11 and its documents immediately; failure to do so may cause difficulty which could have been avoided by early correction of known problems.

CHAPTER 1

OVERVIEW OF BASIC/RT-11 SOFTWARE KIT

BASIC/RT-11 is distributed on five media: DECpack, DECtape, Floppy Disk, cassette and paper tape. Each kit contains the BASIC/RT-11 Language Reference Manual and all the materials necessary to build BASIC/RT-11. The components of each package are inventoried on checklists attached to the outside of the kit. It is recommended that the user verify the contents of the Software Package with the checklist and report any discrepancies to DIGITAL's Software Distribution Center.

The BASIC/RT-11 DECpack, DECtape, Floppy disk, and Cassette kits all contain the following "ready-to-run" versions of BASIC:

BASIC.SAV	Complete BASIC, nonoverlaid; includes string support.
BAS8K.SAV	Smallest possible version of BASIC; overlaid, no string support.
BASGT.SAV	BASIC with VT11 graphics functions included; nonoverlaid.
BASGTO.SAV	BASIC with VT11 graphics functions included; overlaid.
BASLPS.SAV	BASIC with LPS-11 support functions included; nonoverlaid.
BGTLPS.SAV	BASIC with both VT11 and LPS-11 support functions included; nonoverlaid.
BGTLPO.SAV	BASIC with both VT11 and LPS-11 support functions included; overlaid.

All these versions of BASIC are ready to run (under RT-11 V02B); once placed on the system device, they can be started as described in Chapter 1 of the BASIC/RT-11 Language Reference Manual. The nonoverlaid versions (BASIC.SAV, BASGT.SAV, BASLPS.SAV, and BGTLPS.SAV) are linked

for maximum execution speed at the expense of the extra memory necessary to keep all components resident. The overlaid versions (BAS8K.SAV, BASGTO.SAV and BGTLPO.SAV) are linked for maximum memory efficiency, at a slight expense to execution time in response to certain commands.

Files with ".OBJ" extensions are relocatable object modules that can be linked together to form BASIC. They are used when customizing BASIC or when adding assembly language functions (paper tape users will use these files to build a running version of BASIC on the disk). BASICR.OBJ, BASICE.OBJ, and BASICX.OBJ are object modules that have been assembled to include the alphanumeric string capability. BASNSR.OBJ, BASNSE.OBJ and BASNSX.OBJ are the corresponding object modules assembled without the optional string support. To build a BASIC with all features included, use the linking instructions in Section F.2 of Appendix F in the BASIC/RT-11 Language Reference Manual. To build a version of BASIC without string capability, substitute BASNSR for BASICR, BASNSE for BASICE, and BASNSX for BASICX. FPMP.OBJ is the floating point package for BASIC. FPMP.EAE, FPMP.EIS, and FPMP.FPU are customized versions of that math package and support EAE, EIS and FPU options, respectively. To build a version of BASIC that uses one of these options, use the corresponding version of FPMP wherever FPMP is called for in the build instructions.

Other files in the BASIC/RT-11 kit constitute the CALL functions for special purpose devices supported by BASIC/RT-11. For example, files labeled LPSx.xxx are the LPS function components; those labeled GTxx.xxx are used when VT11 support is desired. For additional information, consult Appendix I (for LPS support) and Appendix J (for VT11 support) in the BASIC/RT-11 Language Reference Manual.

Instructions for building BASIC from the distribution kit can be found in Chapter 3 of this document.

Users of previous releases of BASIC or RT-11 should note that the use of BASIC V01B requires RT-11 V02B for proper operation.

Users of the optional LV-11 package for BASIC should note the changes (documented in Chapter 5 of this document) that are necessary for proper operation.

Finally, in addition to the BASIC kit described above, BASIC/RT-11 Source Kits and Listing Kits are available to assist in system development or modification. These may be ordered from the Software Distribution Center.

CHAPTER 2

SERVICES

Training

A variety of hardware and software courses are offered by DIGITAL's Educational Services Groups as detailed in the Educational Courses Catalog (available from the Software Distribution Center). These courses are excellent vehicles for learning about both basic PDP-11 programming and the use of PDP-11 software. "Hands on" training using PDP-11 family systems is a particularly valuable feature of most course and seminars.

SPR System

The SPR (Software Performance Report) system is the mechanism by which BASIC/RT-11 users can report software problems, inadequacies, and suggestions for improvements. (Documentation errors and inadequacies should be reported on the READER'S COMMENTS page at the end of each manual.) SPRs are acknowledged when received in Maynard, and an individual answer is returned to the sender as soon as possible. If the SPR reports a software problem, the answer will include a patch or alternate procedure if possible.

Before sending an SPR to DIGITAL, the user should make certain that the problem is reproducible and that a correction for the problem has not already been published in the Digital Software News or Software Performance Summary. If the problem is new, fill out a Software Performance Report and send it to:

Software Communications
Post Office Box F
Maynard, Massachusetts 01754

The SPR should include as much documentation as possible to help describe and isolate the problem. It must include configuration information, software version numbers, and any examples, tapes, and listings that may be necessary for us to investigate a problem or suggested change. In general, the response time is shortened by additional information provided.

In addition to software problems, SPRs are useful for reporting suggestions and comments on BASIC/RT-11. SPRs are monitored by DIGITAL management, and are considered by the development groups when BASIC/RT-11 changes are made.

Blank SPR forms are included in software kits, and additional forms are available from the Software Distribution Center. Replacement forms are included with each answer.

Digital Software News for the PDP-11

Announcements of new and revised software as well as programming notes, software problems and proposed solutions, and documentation corrections are published monthly in the Digital Software News. Filling out the BASIC/RT-11 registration form included in the BASIC/RT-11 kit assures the user of receiving this publication for one year.

Software and Document Distribution

The PDP-11 Software Price List contains a complete list of programs and documents currently available. Item(s) may be ordered directly from the Software Distribution Center by using the Software Order Form enclosed in the Price List. As noted previously, new and revised software is announced via the Digital Software News.

DECUS

Digital Equipment Computers Users Society (DECUS) was established to advance the effective use of Digital Equipment Corporation's computers and peripheral equipment. It is a voluntary, non-profit users group supported by DIGITAL, whose objectives are to:

1. Advance the art of computation through mutual education and interchange of ideas and information,
2. Establish standards and provide channels to facilitate the free exchange of computer programs among members,

3. Provide feedback to the manufacturer on equipment and programming needs.

The Society sponsors technical symposia twice a year (Spring and Fall) in the U.S., and once a year in Europe, Canada, and Australia. It maintains a Program Library, publishes a library catalog, proceedings of symposia, and a periodic newsletter: DECUSCOPE.

A DECUS-Europe organization was formed in 1970 to assist in the servicing of European members.

Users interested in joining DECUS must obtain and complete a registration form. Forms can be obtained from the nearest DIGITAL sales office or by writing the appropriate Administrative office.

The main Administrative office is located at Digital Equipment Corporation, Maynard, Massachusetts 01754, and all correspondence should be directed to the attention of the DECUS Executive Director.

The European Regional Administrative office is located at:

DECUS EUROPE
Digital Equipment Corporation International (Europe)
P. O. Box 340
1211 Geneva 26
Switzerland

Software Consulting Services

DIGITAL maintains a staff of programmers and consultants whose services are available to DIGITAL customers for a fee. Through DIGITAL's Software Consulting Services, customers have been able to reduce development costs and still obtain quality customized software. Areas of expertise include process control, data communications, data analysis, information retrieval, numerical control, direct digital control, typesetting, simulation, commercial data processing, and special purpose timesharing.

Registration

By completing and returning the registration form included in the kit, the user is eligible to order new updates of this software at the prevailing BASIC/RT-11 Update Kit prices plus any handling or shipping

charges. The user must register to be eligible. Complete and mail the form to:

Digital Equipment Corporation
Software Distribution Center
Building 11-3
146 Main Street
Maynard, Massachusetts 01754

CHAPTER 3
BASIC/RT-11 SYSTEM BUILD AND DEMONSTRATION PACKAGE

This document provides the step-by-step information needed to perform a simple demonstration of BASIC/RT-11. The steps described are intended to provide a user who is familiar with RT-11, but unfamiliar with BASIC, enough information to build and exercise BASIC.

This procedure assumes familiarity with the RT-11 Operating System; the user should also be familiar with the contents of the RT-11 System Reference Manual (DEC-11-ORUGA-C-D) and the BASIC/RT-11 Language Reference Manual. New users should exercise the RT-11 System Demonstration package before exercising this BASIC/RT-11 Demonstration package.

The program required in the demonstration of BASIC/RT-11 is labeled DEMO.BAS and is supplied as part of the BASIC/RT-11 kit.

Read the following general instructions, then proceed to Step 1.

NOTE

No RT-11 program ever HALTs with the expectation that the CONTINUE switch can be pressed to resume operation after corrective action has been taken. If the computer HALTs, a significant error has occurred and the entire package should be repeated from the beginning.

In case of errors which are not explained in this document, please refer to Appendix P in the RT-11 System Reference Manual.

If user errors occur within a section, go back to the beginning of that particular section.

User typing errors may be corrected using the standard RT-11 input editing techniques (RUBOUT and CTRL U).

Conventions, Abbreviations and Standards

1. All numbers are listed in decimal unless otherwise indicated.

2. The following abbreviations are used:

   CTRL - CONTROL key
   CR - RETURN key
   LF - LINE FEED key

3. <CR> or <LF> indicates that the RETURN or LINE FEED key should be typed at that place in the dialogue.

4. <ALT> indicates that the ALTMODE (or ESCAPE) key should be typed at that place in the dialogue.

5. Text enclosed in square brackets, [], is a comment; do not type such text.

6. CTRL x indicates that the CONTROL key should be pressed and held down while another key, "x", is also pressed.

7. <TAB> indicates that a horizontal tab should be typed.

8. On ASR33 and ASR35 Teletype[1] terminals, special characters that are produced by holding down one key and depressing another are:

^	SHIFT N
\	SHIFT L
[	SHIFT K
]	SHIFT M
<TAB>	CTRL I

9. The sample terminal dialogue provided in this document contains version numbers where they would normally appear. The version numbers given include "xx's" in those fields that may vary from installation to installation. The exact contents of these fields are not of interest as long as appropriate digits appear in the area indicated in this document. The same is true for "FREE CORE" messages printed by any of the system programs and for "FREE BLOCKS" messages included in device directories.

Following are instructions for placing running versions of BASIC on the system device, then loading and executing the example program DEMO. DEMO is a simple program to calculate a Fibonacci Series (i.e., each term is the sum of the preceding two terms) based on user inputs.

---

[1]Teletype is a registered trademark of the Teletype Corporation.

NOTE

In the following procedures, type the indicated command on the same line as the previous computer response.

1. Bootstrap an RT-11 V02B system and enter the date. If the BASIC/RT-11 kit was supplied on DECtape, DECpack, or Floppy disk, proceed to Step 4.

   If the BASIC/RT-11 kit was supplied on paper tape, proceed to Step 2.

   WRITE PROTECT the BASIC/RT-11 master cassettes by placing the orange tabs so that the holes are uncovered. Insert the BASIC/RT-11 Cassette 1 of 6 (DEC-11-LBACA-D-TC1) into Unit 0.

   Press the rewind button on each cassette drive.

   ```
   Type:      R PIP<CR>
   Response:  *

   Type:      *.*=CTØ:*.*/X/M:1<CR>
   Response:  *
   ```

   Dismount the master cassette and store it in a safe place.

   ```
   Type:      CTRL C
   Response:  ^C
              .
   ```

   Go to Step 6.

2. Place the paper tape labeled 'DEMO.BAS' (DEC-11-LBACA-D-PA1) into the paper tape reader and press the FEED button until blank leader is over the tape head. The reader should be turned on.

   [For the following steps, a ?CHK SUM? message indicates that an input error occurred during the reading of the paper tape. Retry the operation.]

3. ```
   Type:      R PIP<CR>
   Response:  *

   Type:      DEMO.BAS=PR:/A<CR>
   Response:  [Paper tape will be read.]
              *
   ```

   Place the paper tape labeled "BASICR.OBJ" (DEC-11-LBACA-D-PR1) in the reader as in Step 2 above.

   ```
   Type:      BASICR.OBJ=PR:/B<CR>
   Response:  [Paper tape will be read.]
              *
   ```
   ```

Place the paper tape labeled "BASICE.OBJ" (DEC-11-LBACA-D-PR2) in the reader as in Step 2 above.

```
Type:       BASICE.OBJ=PR:/B<CR>
Response:   [Paper tape will be read.]
            *
```

Place the paper tape labeled "BASICX.OBJ" (DEC-11-LBACA-D-PR3) in the reader as in Step 2 above.

```
Type:       BASICX.OBJ=PR:/B<CR>
Response:   [Paper tape will be read.]
            *
```

Place the paper tape labeled "BASNSR.OBJ" (DEC-11-LBACA-D-PR4) in the reader as in Step 2 above.

```
Type:       BASNSR.OBJ=PR:/B<CR>
Response:   [Paper tape will be read.]
            *
```

Place the paper tape labeled "BASNSE.OBJ" (DEC-11-LBACA-D-PR5) in the reader as in Step 2 above.

```
Type:       BASNSE.OBJ=PR:/B<CR>
Response:   [Paper tape will be read.]
            *
```

Place the paper tape labeled "BASNSX.OBJ" (DEC-11-LBACA-D-PR6) in the reader as in Step 2 above.

```
Type:       BASNSX.OBJ=PR:/B<CR>
Response:   [Paper tape will be read.]
            *
```

Place the paper tape labeled "FPMP.OBJ" (DEC-11-LBACA-D-PR7) in the reader as in Step 2 above.

```
Type:       FPMP.OBJ=PR:/B<CR>
Response:   [Paper tape will be read.]
            *
```

Place the paper tape labeled "BASICH.OBJ" (DEC-11-LBACA-D-PR8) in the reader as in Step 2 above.

```
Type:       BASICH.OBJ=PR:/B<CR>
Response:   [Paper tape will be read.]
            *

Type:       CTRL C
Response:   ^C
            .

Type:       R LINK<CR>
Response:   *

Type:       BAS8K=BASNSR,FPMP/T/B:400/C<CR>
Response:   TRANSFER ADDRESS =
```

```
Type:       GO<CR>
Response:   *

Type:       BASNSE/O:1/C<CR>
Response:   *

Type:       BASNSX/O:1/C<CR>
Response:   *

Type:       BASICH/O:2<CR>
Response:   *

Type:       BASIC=BASICR,FPMP,BASICE,BASICX/B:5ØØ/C<CR>
Response:   *

Type:       BASICH<CR>
Response:   *

Type:       CTRL C
Response:   ^C
            .
```

Go to Step 6.

4. If the BASIC/RT-11 system was received on DECtape, mount the BASIC/RT-11 System DECtape 1 (DEC-11-LBACA-D-UC1) on Unit 7, WRITE LOCKed.

```
Type:       ASS DT7 A<CR>
Response:   .
```

Proceed to Step 5.

If the BASIC/RT-11 system was received on DECpack, mount the BASIC/RT-11 System Disk on Unit 1, WRITE PROTECTed.

```
Type:       ASS RK1 A<CR>
Response:   .
```

If the BASIC/RT-11 system was received on Floppy disk, mount the BASIC/RT-11 System Disk 1 of 2 (DEC-11-LBACA-D-YC1) on Unit 1.

```
Type:       ASS DX1 A<CR>
Response:   .
```

5.
```
Type:       R PIP<CR>
Response:   *

Type:       *.*=A:BASIC.SAV,BAS8K.SAV,DEMO.BAS/X<CR>
Response:   *
```

Dismount the master disk or DECtape and store it in a safe place.

```
Type:       CTRL C
Response:   ^C
            .
```

6. The following instructions are for running the simple BASIC program, DEMO.BAS.

```
Type:       R BAS8K<CR>
Response:   BASIC Vxx-xx
            *

Type:       <CR>
Response:   READY
```

Load the demonstration program into memory.

```
Type:       OLD "DEMO.BAS"<CR>
Response:   READY
```

List the program.

```
Type:       LIST<CR>
Response:
```

```
DEMO    02-APR-75  BASIC V01B-01

10 REM BASIC PROGRAM TO GENERATE N TERMS OF A FIBONACCI SERIES,
20 REM THE FIRST TWO TERMS OF WHICH ARE SPECIFIED BY THE USER.
30 REM
40 REM PRINT IDENTIFYING MESSAGE
50 PRINT "PROGRAM TO GENERATE A FIBONACCI SERIES"
60 REM
70 REM GET THE LENGTH AND FIRST TWO TERMS OF THE SERIES
80 PRINT "HOW MANY TERMS DO YOU WANT GENERATED";
90 INPUT L
100 IF L<>0 THEN 130
110 REM IF HE REQUESTS 0 TERMS,TERMINATE EXECUTION
120 STOP
130 PRINT "WHAT IS THE FIRST TERM";
140 INPUT T1
150 PRINT "WHAT IS THE SECOND TERM";
160 INPUT T2
170 REM MAKE SURE L IS NOT NEGATIVE OR TOO LARGE
180 IF L<3 THEN 200
190 IF L<50 THEN 220
200 PRINT L;"TERMS DOES NOT REALLY MAKE SENSE."
210 GO TO 80
220 REM PRINT THE FIRST TWO TERMS OF THE SERIES
230 PRINT "THE REQUESTED SERIES IS"
240 PRINT T1
250 PRINT T2
260 L=L-2
270 REM CALCULATE NEXT TERM AND PRINT IT
280 N=T1+T2
290 T1=T2
300 T2=N
310 PRINT N
320 REM DETERMINE IF SERIES IS FINISHED. IF SO,DO NEXT ONE.
330 L=L-1
340 IF L<=0 THEN 80
350 GO TO 280
360 END

READY
```

Execute the program.

```
Type:       RUN<CR>
Response:   DEMO DD-MON-YR  BASIC Vxx-xx
            PROGRAM TO GENERATE A FIBONACCI SERIES
            HOW MANY TERMS DO YOU WANT GENERATED?

Type:       1Ø<CR>
Response:   WHAT IS THE FIRST TERM?

Type:       2<CR>
Response:   WHAT IS THE SECOND TERM?

Type:       4<CR>
Response    THE REQUESTED SERIES IS:
            2
            4
            6
            1Ø
            16
            26
            42
            68
            11Ø
            178
            HOW MANY TERMS DO YOU WANT GENERATED?

Type:       CTRL C
Response:   ^C
            .

Type:       RE<CR>
Response:   READY
```

BASIC/RT-11 may be run in immediate mode, described in Chapter 4 of the BASIC/RT-11 Language Reference Manual. Statements without line numbers are executed as soon as they are entered.

```
Type:       FOR I=1 TO 5\PRINT I,SQR(I)\NEXT I<CR>
Response:   1     1
            2     1.41421
            3     1.732Ø5
            4     2
            5     2.236Ø7

Type:       CTRL C
Response:   ^C
            .
```

The demonstration is complete.

Further instructions for building and using other versions of BASIC (including those with LPS-11 and VT11 functions) can be found in the BASIC/RT-11 Language Reference Manual. Before using BASIC, the user should carefully read Chapter 5 of this document and note the restrictions and clarifications for BASIC/RT-11 V01B.

CHAPTER 4

DIFFERENCES BETWEEN BASIC/RT-11 V01B AND BASIC/RT-11 V01-05

Outlined below are the major differences between BASIC/RT-11 V01B and BASIC/RT-11 V01-05; users upgrading from V01-05 to V01B should be aware of these differences. The differences fall into two categories:

1. Problems with BASIC/RT-11 V01-05 for which an article and/or patch was published in the Digital Software News.
2. Problems with BASIC/RT-11 V01-05 for which no article was ever published.

Note that in all cases in both categories, the problems have been corrected in BASIC/RT-11 V01B; published patches should not be applied to BASIC/RT-11 V01B.

1. Problems in V01-05 Which Have Been Corrected in V01B

The following are problems that existed in BASIC/RT-11 V01-05 for which an article and/or patch was published in the Digital Software News.

Problem/Article Name	D.S.N. Sequence #
1. Problem with "RTS" Command in LPS Support	1
2. Problem with "DSAV" in More Than 16K of Memory	2
3. Problem with Calls to "APUT" or "FPUT" on "FIGR" Array	4
4. Erroneous Results When Using Virtual Files	5
5. Overlay Command Produces "?ULN" Error	6
6. Overlay Appears to Scratch Data Area (Note that it is no longer necessary to follow overlay statements with a RESTORE statement.)	7

Problem/Article Name	D.S.N. Sequence #
7. Destruction of String Array Elements by BASGT Routine "FIX"	8
8. Termination of BASIC Compiler When Call to BASGT Subroutine "FIX" is not Preceded by Calls to "INIT" and "FREE"	9
9. Reference to 128-Character String Virtual File Causes BASIC to HALT	10, 11
10. "?ETC" from Long Programs Running in Machines Greater Than 16K	12
11. LPS Array Pointers Relocated Incorrectly in BASGT and BGTLPS	14

2. Unpublished Problems in V01-05 Which Have Been Corrected in V01B

The following is a list of problems that existed in BASIC/RT-11 V01-05 for which no article or patch was ever published. Where applicable, a page number from the BASIC/RT-11 Language Reference Manual is given for reference.

1. BASIC virtual memory files, as produced by BASIC/RT-11 V01B, are completely compatible with FORTRAN/RT-11 direct access files.

2. The BASIC "POS (X$,Y$,Z)" function now produces correct results for all values of Z. (6-15)

3. The "OVERLAY" command no longer changes the current program name.

4. BASIC now handles input lines up to 132 characters.

5. The user-defined function evaluation algorithm has been changed to correct a problem that occasionally generated incorrect results. The problem occurred when the user-defined function used more than one argument, and the same argument names were used in the model statement (DEF) and the executable statement. For example:

```
10 DEF FNA(X,Y)=X+Y
20 X=2
30 PRINT FNA(10,X)
```

   These statements caused 20, rather than 12, to be printed.

6. The power fail support in BASIC, which was minimal, has been removed. (1-3, 9-3, E-3)

7. A problem that occurred in overlaid versions of BASIC, due to the initial value of the stack pointer, has been corrected. The problem caused an area of memory, from approximately 23706 to 23776, to be overwritten since it was used by the overlay processor as stack space.

8. When input from the high speed reader is used, no "^" prompt character is typed. (5-9, 7-3)

9. The BASGT routine, DSAV, no longer causes syntax errors in overlaid versions of BASIC.

10. A problem that caused BASGT ERAS and DSAV loops to hang the processor has been corrected.

11. BASIC with VT11 support can be run while the monitor is using the scope (i.e., the monitor command "GT ON" is in effect). Note that the program GTON.SAV distributed with RT-11 V01-15 is obsolete and should not be used with BASIC with VT11 support. (J-2)

12. The modules PERPAR.MAC, PERVEC.MAC, and FTBL.MAC have been updated to include the required code for LV-11 support. These new modules should be used instead of the modules LVPAR.MAC, LVVEC.MAC, and LVTBL.MAC, where the latter are indicated in the LV-11 document.

13. The command "OLD 'PR:'" no longer sets the program name to "PR:". BASIC assumes "NONAME" for the program name. (7-3)

14. A problem with the LPS display routines "FSH", "DIS", and "DXY", causing BASIC to trap to location 4 if an even value (0,2, and so on) was specified for the first data point to display, has been fixed.

15. A problem with the BASIC function INT(X), which caused incorrect results for certain values of X, has been fixed. The problem caused large negative values of X to be rounded in the wrong direction; negative powers of 2 were also incorrectly integerized.

16. A problem in the BASIC/RT-11 file processing, which could cause string storage overflow or buffer storage overflow, has been fixed. The problem generally occurred when a program that did a large number of file OPENs and CLOSEs ran for a long time.

17. A problem in the LPS routine "SETR", which caused one clock status register to be set incorrectly, has been fixed. The problem caused interrupts from Schmitt trigger 1 to be ignored.

18. Several problems in the LPS "WAIT" routine have been fixed. These problems were connected with the "SETR" problem (17 above) and caused the "WAIT" routine to ignore interrupts from Schmitt trigger 1, if the Schmitt trigger was initialized by a "SETR" command.

CHAPTER 5

RELEASE NOTES AND RESTRICTIONS

BASIC/RT-11 users should always keep abreast of BASIC-related notices published by DIGITAL. Changes published in the Software Performance Summary need be made only once and should be made immediately. Changes published in the Digital Software News should be made as soon as possible to systems in use. In addition to continued surveillance of the above documents, the user should be aware of the following release notes and restrictions at installation time.

Patches made with RT-11 PATCH, as described in this chapter, assume a non-overlaid version of BASIC is being patched. The following procedure is used when patching overlaid versions of BASIC (computer output is underlined):

```
.R PATCH
PATCH Version number
FILE NAME --                    [use BAS8K or name of overlaid ver-
*BAS8K.SAV/O                    sion being patched]
?BOTTOM ADDR WRONG?
*4ØØ;B                          [use 400 or bottom address specified
                                to LINK with /B if different]
*nnnnnn;ØR                      [nnnnnn is BASICR address from link
                                map]
*  .                            [rest of procedure is same as for
   .                            non-overlaid versions]
   .
```

The error message ?BOTTOM ADDR WRONG? is expected. It is generated as a result of the way BASIC initializes the stack pointer in overlaid versions.

VT11 Support Functions

1. The size of BASIC and the size of BASIC's VT11 support has increased. The net effect of these increases and S/J monitor increases is to decrease the size of the normal display buffer under the S/J monitor by about 40 words.

2. If GT ON is in effect and a BASIC program is to do graphics, the BASIC program must CALL "INIT" before any PRINT statements are executed. Failure to do so may cause the BASIC program to be stopped and the screen to go blank. An alternative solution is to delay any CALL "INIT" after a PRINT statement by doing

    a. An INPUT statement

    b. A long expression evaluation

    c. A CALL "TIME" of sufficient duration.

    Should the problem occur, a CTRL O or CTRL C and REENTER will restore control to BASIC.

3. If a display which used graphics arrays in BASIC is running, and an OLD, SCR, NEW, or CLEAR command is issued to BASIC, the display deteriorates as BASIC automatically re-initializes the graphics array.

    The problem can be avoided by a

    ```
    CALL "INIT"
    ```

    before issuing the OLD command.

    Should the problem occur, a CTRL C will clear up the screen and restore control to the scroller if it was active.

4. For the Single-Job Monitor only, if a display is running in BASIC, and a CTRL C is used to return to the Keyboard Monitor, the display may deteriorate as the USR or Keyboard Monitor is swapped over the display file. This is a harmless occurrence, and the display can be restored by re-entering BASIC. An alternative is to clear the display by typing an INIT monitor keyboard command. The problem can be avoided by stopping the display with a CALL "INIT" or CALL "DSTP" before leaving BASIC.

5. A CALL "DSAV" following a CALL "DSTP" will turn on the display. A CALL "DSTP" is not necessary before the CALL "DSAV", so to compress the file and stop the display, use the sequence CALL "DSAV", CALL "DSTP".

Syntax Errors in Multiple Statement GOSUB or in DEF Statement

If a syntax error appears in a multiple statement line after a GOSUB, the line number reported in the error message is not the line number containing the error. Instead, it is the line number of the RETURN statement which returned control after the GOSUB. For example,

```
1Ø  GOSUB 3Ø \ ERROR
2Ø  STOP
3Ø  RETURN
```

When RUN, the message ?SYN AT LINE 30 will appear instead of ?SYN AT LINE 10.

Similarly, if a syntax error appears in the DEF statement for a user-defined function, the line number reported in the error message is not the line number containing the error. Instead, it is the line number of the statement which referenced the user-defined function. For example:

```
1Ø  DEF FNA(X)=ERROR
2Ø  A=FNA(1Ø)
3Ø  STOP
```

When RUN, the message "?SYN AT LINE 20" will be printed instead of "SYN AT LINE 10".

LPS Support

The LPS routine "WAIT" does not set up any of the LPS devices, it merely waits for a specific event to occur (e.g., clock overflow, firing of Schmitt trigger). Unless the device being waited for has previously been set up by another LPS command, doing a "WAIT" for that device will never return control to BASIC.

SAVE Command to Magtape and Cassette

When doing a SAVE to magtape or cassette, BASIC/RT-11 does not check to see if there is already a file on the magtape or cassette that has the same name as the file being saved. If a SAVE is done to disk or DECtape, BASIC checks for such a file and issues a "?RPL" message, indicating that the REPLACE command must be used. For magtape or cassette, the SAVE always deletes the old file and puts the new file on the tape.

Interrupt Level Device Support

When an assembly language routine for BASIC includes interrupt-driven device service that directly accesses BASIC's data space (arrays, variables, etc.), the user must ensure that the interrupt processing is turned off when BASIC returns to edit mode. The user's BASIC program could be corrupted if an interrupt level routine is still accessing variables after BASIC has begun executing a CHAIN or OVERLAY statement.

Additionally, if interrupt level device service is continuing when the user's program terminates and the user immediately begins issuing editing commands, his BASIC program could be corrupted. The user should ensure that device service is always stopped before he begins issuing editing commands. This could be accomplished by a special user subroutine that unconditionally stops device service.

The LPS support routines for BASIC include such interrupt level routines, but there is no special routine to abort device service. Therefore, LPS users must be careful to ensure that interrupt level access to BASIC's data space has stopped before a CHAIN or OVERLAY is executed, or before editing commands are issued.

If the interrupt level device service has its own buffers and does not directly modify BASIC's data space, the problem does not occur.

Bottom Address When Linking BASIC

All distributed versions of BASIC/RT-11 V01B, except BAS8K.SAV, are linked with a bottom address of 500_8; this is because the area below 500_8 can be used for device interrupt vectors. If the user's configuration has no devices with vectors above 400_8, he can link any of the BASIC versions with a bottom address of 400_8. The distributed version of BAS8K.SAV is standardly linked with a bottom address of 400_8 to allow maximum user area in small machines. Users with 8K, whose configurations have interrupt vectors above 400_8, should relink BAS8K.SAV with a bottom address of 500_8.

Clock Interrupt Enable for SETR and SETC

LPS sampling rates are limited when the clock is enabled to interrupt. Both the SETR and SETC commands always enable the clock interrupt,

although the interrupt enable is not always necessary. To achieve faster sampling rates, users may wish to disable the clock interrupt for some SETR and SETC commands. The clock interrupt enable and start bits for the SETR and SETC commands are contained in location CLKIES in the module LPS2. This location initially contains 101_8, which enables interrupts. To disable clock interrupts, this location must be changed to contain 1. The user may wish to write an assembly language routine that accesses location CLKIES (a global), so that interrupt enable/disable can be dynamically selected at run-time.

Column Width

BASIC/RT-11 V01-05 had a fixed-column size of 72 characters for all devices. For BASIC/RT-11 V01B this column size may be changed for all devices by patching a single location. Attempting to PRINT on any device past the highest column number, as specified by this location, causes BASIC to print an automatic carriage return/line feed and continue output on the next line.

The following example shows how to change the column width to 132 *for all devices* (including terminal).

```
.R PATCH

PATCH Version number

FILE NAME --
*BASIC.SAV
*5ØØ;ØR                    [use 500 or BASICR address from link
                           map, if different]
*Ø,1Ø/  11Ø   2Ø4<CR>      [column size in octal]
*E
```

The column size is stored in location COLSIZ, which is a global and appears in the link map. It is at an offset of 10 (octal) from the beginning of BASICR. Note that once the patch is made, any BASIC program that depends on BASIC's printing an automatic carriage return/line feed at column 72 will generate different output. COLSIZ initially contains 72. If its contents are changed to 'n', the TAB function is then done in modulo n rather than modulo 72.

Stack Size

BASIC/RT-11 uses part of the user area for its working stack. A portion of this stack is reserved for interrupt level routines (device

handlers) that require stack space. The remainder is used by BASIC and user assembly language routines. It is possible that some user-written modules, or interrupt level routines, will require more stack space than is currently reserved for them. Therefore, the following procedure can be used to increase the total amount of stack space, or the portion of that total reserved for interrupt level routines, or both. The two locations involved are STKEXT and STKSIZ. The first, STKEXT, is the number of bytes (must be even) of stack space reserved for interrupt level routines; the second, STKSIZ, is the *total* number of bytes (must be even) of stack space that may be used by BASIC, by interrupt level routines, and by user routines (includes STKEXT).

Initially, the value of STKEXT is 40_8 and STKSIZ 200_8. Since STKSIZ includes STKEXT, this means that the amount of stack space available for BASIC and user assembly language routines is 140_8. To increase the amount of stack space available to interrupt level routines, both values should be increased by the desired number of bytes of stack space to be added. Locations STKEXT and STKSIZ are globals and will show up in the link map. They are at an offset of 14_8 and 12_8, respectively, from the start of BASICR.

The following example shows how to increase the stack space available to interrupt level routines (by 20_8 bytes) by adding 20_8 to the current values stored in STKSIZ and STKEXT:

```
.R PATCH

PATCH Version number

FILE NAME --
*BASIC.SAV
*500;0R                    [use 500 or BASICR address from link
                           map, if different]
*0,12/   200   220<CR>     [STKSIZ]
*0,14/    40    60<CR>     [STKEXT]
*E
```

If the user has the sources of BASIC/RT-11, these patches can be made by defining the variables:

```
$STKSZ=220
$STKEX=60
```

in an assembly parameter module and reassembling BASIC with this parameter module as described in Appendix F of the BASIC/RT-11 manual.

NOTE

If the value of STKSIZ is increased by 20_8 bytes and the value of STKEXT is not changed, no additional space is reserved for interrupt level routines, but an additional 20_8 bytes of stack space are reserved for BASIC and user assembly language routines.

String Storage Garbage Collection Pause

Versions of BASIC/RT-11 with string support periodically appear to hang for some time. This is usually noticed when using assembly language routines to drive some type of graphics display. This hang results from the fact that each call of an assembly language routine causes a new string to be created (consisting of the name of the assembly language routine). When string storage becomes filled, BASIC performs a garbage collect operation, causing it to become internally compute-bound for a period of time.

This problem affects programs doing real-time data acquisition, since BASIC can service interrupts and collect data during the garbage collect operation, but it can neither process the data collected nor write the data out to a file until the garbage collect is completed. For very fast sampling rates, the interrupt service must be able to buffer a large amount of data during a garbage collect operation in order to avoid data loss.

The problem can be eliminated by linking the assembly language routines with the versions of BASIC that do not support strings.

If string support is required, the following patch can be made to eliminate the creation of a new string each time an assembly language routine is called. Note, however, that *with this patch installed*, string variables and string expressions are not allowed as the name of an assembly language routine in a CALL statement. Only a literal string, enclosed in quotes, may be used for the name. In addition, to prevent BASIC from hanging, users should ensure that no string constants, string variables, or string expressions appear inside a time-critical data collection loop (except for the name of the assembly language routine in the CALL statement). The patch is made to location STRFND in BASICR, which is a global and appears in the link map. The contents of this location are changed from the address of

subroutine FNDSTR to that of subroutine FNDSTL. Both these routines are globals and their addresses appear in the link map to facilitate changes back and forth.

The patch is as follows:

```
.R PATCH

PATCH Version number

FILE NAME --
*BASIC.SAV
*500;0R                          [use 500 or BASICR address from link
                                 map, if different]
*0,16/ 35160  3044<CR>           [use 3044 or FNDSTL address from
                                 link map, if different]
*E
```

Note that the old contents of location STRFND may vary with the version of BASIC being patched; 35160 is used for this example only.

Size of VT11 Screen

The VT11 support for BASIC/RT-11 V01B assumes a 12-inch display screen. Users with a 17-inch screen should use RT-11 PATCH to change the screen size. The screen size is stored in location $TOP, which is a global and will appear in the link map. For 12-inch screens, $TOP contains 1350_8; for 17-inch screens, $TOP contains 1750_8. To implement this patch, first link the desired version of BASIC with GT support and obtain a link map. Then use PATCH as follows:

```
.R PATCH

PATCH Version number

FILE NAME --
*BASGT.SAV                       [or name of BASIC version just cre-
                                 ated]
*nnnnnn/1350  1750<CR>           [nnnnnn is $TOP address from link
*E                               map]
```

.DEVICE List for LPS Support (F/B Users Only)

The LPS support routines for BASIC contain .DEVICE EMTs for turning off the various LPS devices when BASIC execution is terminated using CTRL C under the F/B Monitor. These EMTs assume a full complement of LPS hardware. If the user's LPS does not include either an analog to digital converter, a real-time clock, or digital I/O, the .DEVICE list

should be patched to reflect that fact. Failure to do so will cause a monitor trap under the F/B Monitor when attempting to stop the non-existent device at CTRL C time. The .DEVICE list is located in the module LPSØ at location SDLST1, which is a global and appears in the link map. The contents of this list are as follows:

Location	Contents
SDLST1+4	Address of Analog to Digital Converter Status Register
$SDLST1+10_8$	Address of Real-Time Clock Status Register
$SDLST1+14_8$	Address of Digital I/O Status Register

To eliminate digital I/O from the list, for example, use PATCH as follows:

```
.R PATCH

PATCH Version number

FILE NAME --
*BASLPS.SAV
*nnnnnn;ØR              [nnnnnn is the address of SDLST1 from
                         link map]
*Ø,14/17Ø41Ø  Ø,16<CR>  [replace address of Digital I/O stat-
                         us register with address of next word
*E                       in list]
```

LV-11 Support

Version 1 of the LV-11 support package for BASIC (DEC-11-LBLVA-A) must be patched so that it will operate correctly with the revised VT11 support in BASIC/RT-11 V01B. The following patch should be made to the module PLOT.OBJ using the RT-11 PATCHO program (a copy of the original PLOT.OBJ should be saved as PLOT.OLD):

```
.R PIP
*PLOT.OLD=PLOT.OBJ
*↑C

.R PATCHO
*OPEN<CR>
ENTER INPUT FILE *PLOT.OBJ<CR>
ENTER OUTPUT FILE *PLOT.OBJ<CR>
*WORD +21Ø=SCAL.F+Ø<CR>
*EXIT<CR>
ENTER CHECKSUM: 1Ø7534

STOP --

.
```

The following patch should be made to the module LVSUBS.OBJ using the RT-11 PATCHO program (a copy of the original LVSUBS.OBJ should be saved as LVSUBS.OLD):

```
.R PIP
*LVSUBS.OLD=LVSUBS.OBJ
*^C
_

.R PATCHO
*OPEN<CR>
ENTER INPUT FILE *LVSUBS.OBJ<CR>
ENTER OUTPUT FILE *LVSUBS.OBJ<CR>
*WORD ._ABS.+42=LVSUBS+13522<CR>
*EXIT<CR>
ENTER CHECKSUM: 5Ø554

STOP --

.
```

Note that if PATCHO issues the ?BAD PATCH? message, the user should do the following:

```
.R PIP
*xxxxxx.OBJ=xxxxxx.OLD<CR>
*^C
_
```

and then repeat the patch.

LV-11 Assembly Errors

If the LV-11 source module RTDSK.MAC is assembled using the RT-11 V02B SYSMAC file, several assembly errors result. These errors occur because V02B system macro expansions do not default to RT-11 Version 1 format. The problem can be avoided by editing the module RTDSK.MAC so that it forces Version 1 expansions as follows (a copy of the RTDSK.MAC file is first saved as RTDSK.OLD):

```
.R PIP
*RTDSK.OLD=RTDSK.MAC
*^C
_

.R EDIT
*EBRTDSK.MAC$RF.GLOBL$ØA$$     [$ represents ALTMODE]
*I.MCALL ..V1..
_..V1..
$$
*EX$$
_
```

Module Addresses in Distributed SAV Files

The following table listing the addresses of important modules in the distributed versions of BASIC.

.SAV File	Bottom Address	BASICR	FPMP	BASICE	BASICX	BASICH
BAS8K	400	1136	6072	12374	12374	24212
BASIC	500	500	5776	12276	24666	37242
BASGT	500	500	5776	12276	24666	54704
BASLPS	500	500	5776	12276	24666	45752
BGTLPS	500	500	5776	12276	24666	63410
BGTLPO	500	1414	6712	33644	33644	46236
BASGTO	500	1414	6712	25140	25140	37532

LPS Display with F/B Monitor

When using BASIC with LPS support in the background under the F/B Monitor, the LPS display is not refreshed when BASIC is waiting for keyboard input. This is because BASIC issues a .TTINR request and if no input is available, BASIC refreshes the display. This is acceptable under S/J, but under F/B, control does not return from the .TTINR until a character is available, so BASIC cannot refresh the display. This problem can be avoided by setting bit 6 of the Job Status Word (location 44) for BASIC. This causes .TTINR under F/B to act as it does under the S/J Monitor.

For example:

```
.R PATCH

PATCH Version number

FILE NAME--
*BASLPS.SAV<CR>
*44/      Ø      1ØØ<CR>
*E
.
```

For overlaid versions:

```
.R PATCH

PATCH Version number

FILE NAME--
*BGTLPO.SAV/O<CR>
?BOTTOM ADDR WRONG?
*5ØØ;B
*44/    1ØØØ     11ØØ<CR>
*E
.
```

Please cut along this line.

READER'S COMMENTS

NOTE: This form is for document comments only. Problems with software should be reported on a Software Problem Report (SPR) form (see the HOW TO OBTAIN SOFTWARE INFORMATION page).

Did you find errors in this manual? If so, specify by page.

Did you find this manual understandable, usable, and well-organized? Please make suggestions for improvement.

Is there sufficient documentation on associated system programs required for use of the software described in this manual? If not, what material is missing and where should it be placed?

Please indicate the type of user/reader that you most nearly represent.

- ☐ Assembly language programmer
- ☐ Higher-level language programmer
- ☐ Occasional programmer (experienced)
- ☐ User with little programming experience
- ☐ Student programmer
- ☐ Non-programmer interested in computer concepts and capabilities

Name _____ Date _____

Organization _____

Street _____

City _____ State _____ Zip Code or Country _____

If you do not require a written reply, please check here. ☐

---------- Fold Here ----------

---------- Do Not Tear - Fold Here and Staple ----------

FIRST CLASS
PERMIT NO. 33
MAYNARD, MASS.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

Postage will be paid by:

digital

Software Communications
P. O. Box F
Maynard, Massachusetts 01754