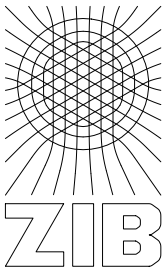
REDUCE

User's Guide

for Personal Computers

Winfried Neun



Konrad-Zuse-Zentrum Berlin 2004

REDUCE User's Guide for Personal Computers

Version 3.8

by

Winfried Neun
ZIB
D-14195 Berlin-Dahlem

Federal Republic of Germany

June 2004

Abstract

This document describes operating procedures for running REDUCE specific to personal computers running MS Windows NT, MS Windows 95/98, Windows 2000/XP or higher. There is no support for a MS-DOS, OS/2, or Windows 3.1 version.

Contents

1 Introduction

This document describes operating procedures for running REDUCE on personal computers under the operating systems

- **MS-DOS** [1], for computers based on Intel [2] processors (currently NOT supported),
- **MS Windows 95/98/2000/XP**
- **MS Windows NT** for Intel and Digital Equipment (Compaq) Alpha processors.

It supplements the REDUCE User's Manual, describing features, extensions and limitations specific to this implementation of REDUCE. For systems of the Unix family a separate User's Guide is available.

The Intel 32 bit versions of REDUCE have a lot of features in common. Therefore, in the following, we denote by Windows 95/98/2000 one of the operating systems Windows 95, Windows 98 or Windows 2000. There are a few special features in the Windows NT version.

1.1 REDUCE Versions

There are two versions of REDUCE available, the "professional" version and the "personal" version. Compared to the professional version, the following items are missing in the personal version:

- REDUCE sources
- rebuild scripts

This guide can be used for both versions. Differences are marked by notes $^{(prof)}$ for features which are available only in the professional version.

1.2 REDUCE Environment Variables and Directories

The files that form the REDUCE system are stored under the REDUCE root directory. Please ensure that some environment variables are set to the appropriate directory. The best strategy is to include a corresponding command in your initialization procedure (`autoexec.bat`) or under Windows NT and Windows 2000/XP use the $Settings > ControlPanel > System > Environment$ feature, e.g.

```
set reduce=c:\reduce
set helpdir=%reduce%\help
set machine=<win32 or dos386 or alphant see below>
set gnuplot=%reduce%\wutil\%MACHINE%
set lisp=psl
```

The environment variables should have been installed during the software installation process.

The following system names are used in this document and in the file system structure:

- **win32**: MS Windows 95/98/2000/XP and MS Windows NT for Intel processors,
- **alphant**: MS Windows NT for DEC (Compaq) Alpha processors.

[1] MS, MS-DOS, Windows , Windows 95/98, Windows 2000, Windows XP, and Windows NT are registered trademarks of Microsoft Corporation, OS/2 is a registered trademark of International Business Machines

[2] Throughout this document "Intel processors" mean the 32 bit processors of the Intel x86 family, namely i386(SX and DX) (optionally with i387 coprocessor), i486 and Pentium or compatibles. This does not necessarily mean that the processor has to be manufactured by the Intel Corporation.

The operating system dependent binaries reside in the subdirectories lisp\%MACHINE%\psl and lisp\%MACHINE%\red respectively. The directory ..\red holds the REDUCE load modules, while the compiler$^{(prof)}$ is kept in ..\PSL.

Directories of interest to the general user include DOC, containing relevant documents mostly in TEX syntax, DOCEXTRA, also containing relevant documents ready for printing, folder PACKAGES containing the specific package documentation and various examples of the use of REDUCE.

The more serious user may also be interested in the REDUCE sources stored in the PACKAGES folders in the .red files $^{(prof)}$. The directories HELP, WUTIL and PLOT contain additional modules for REDUCE runtime support.

1.3 Operating System Interface

REDUCE has been implemented for the microprocessors based on the full 32 bit or 64 bit architecture offered by the processor families.

Under all operating systems there is a rather small binary executable, the PSL [3] loader, which loads the REDUCE kernel preserved in a binary file, called the "image" file with suffix **.img**. The loader organizes all system contacts (e.g. input/output, memory management). The Image file structure is system dependent. E.g. an image created under Windows 95 cannot be used under Windows NT. In such a case a new image has to be created (see MKREDUCE in the Installation Manual).

1.3.1 32 bit Operating Systems

Under the operating systems Windows NT and Windows 95/98/2000/XP REDUCE can run in generic 32 bit mode on the Intel processors and in 64 bit mode on the DEC (Compaq) Alpha systems. The basic loader program is named BPSL.EXE - it executes REDUCE in a terminal window.

For advanced windows-based input/output facilities: under Windows NT and Windows 95/98/2000/XP there is a loader BPSLW.EXE which supports advanced window I/O and mouse support. These can be installed in the usual way as icons on the desktop. Special provision has been taken for running REDUCE in script mode in a batch style in the background.

During the Setup Process a group of icons has been created which allow the user to start the Windows interface by double click.

1.4 Memory Usage

All versions run better the more memory is available. The easiest way to adapt the resource usage of REDUCE to your computer is by editing the memory size (the td parameter) in the REDUCE 3.8 group which is automatically generated at setup time. A memory size value of **128 M Bytes** is the upper limit for the td parameter on a 32 bit system.

You should have enough swap space (free disk space) available. Otherwise your application might run into problems even if you have a computer with a big memory.

[3] Portable Standard LISP

2 REDUCE under Windows 32

This chapter describes the operation of REDUCE under Windows 32, i.e. Windows 95, Windows 98, Windows 2000, Windows XP, or Windows NT.

2.1 Starting REDUCE from the Taskbar

To execute REDUCE, first check that the environment variable *reduce* is set to the appropriate directory. We assume here that one or more icons have been installed in a group of your Windows environment by the Setup process. Please consult the REDUCE Installation Guide for a detailed description of the installation options. To run REDUCE, please activate a REDUCE icon, e.g. by double-clicking with the mouse. REDUCE will then open a window. Please wait a moment until the loading process is complete. During loading (and during all lengthy operations) the graphic cursor has the shape of an hour glass.

After loading REDUCE will respond with a banner line and then prompt for the first line of input:

```
REDUCE 3.8, 15-Apr-2004 ...

1:
```

You can now begin to enter commands.

2.2 Starting REDUCE in a Terminal Window

During installation a script `REDUCE.BAT` has been generated in the root directory of REDUCE. This script starts REDUCE when called, e.g. from a terminal window. It also can serve as specimen for other scripts. The output is then directed to the window. For activating REDUCE please enter

```
reduce
```

after which a REDUCE DOS type window will come up. After some load time REDUCE will respond with a banner line and then prompt for the first line of input:

```
REDUCE 3.8, 15-Apr-2004 ...

1:
```

You can now begin to enter commands. Alternatively you can call the script *reduce* with a parameter which specifies the memory size as decimal number of bytes, e.g.

```
reduce 9000000
```

If the parameter is omitted a standard value (encoded in the script file) is used.

2.3 The REDUCE Window

The REDUCE window has the following components:

- Frame: The outer frame can be used to resize the window with the mouse. There are buttons for maximizing, minimizing (iconify) etc. The window system informs its clients about any changes

of the window parameters and REDUCE will adjust its `linelength` to the current window size automatically.

- Title bar: as long as the REDUCE window is active, the title bar displays the start directory. If inactive (or iconic) the title is void (just LISP). In script mode the name of the current script is presented here.
- Vertical scroll bar: this is used to scroll around in the buffered input and output lines. You can move the thumb with the mouse directly, pick into the scroll bar above or below the thumb (= movement of one page in the desired direction) or you can pick into the arrow buttons (= movement of one line per pick).
- Command bar: there are four entries described below. For the current version the **Modes** entry can be ignored. It is introduced for further extensions of the interface:

File	Edit	Help
Dribble	CopyText	
Print	PasteText	
CopyBitmap	Cut	
Copy Metafile	CopyBitmap	
Graphics	AutoCopy	
Exit	Font	
	Page mode	
	Display/Set F1–F12	

These are described in detail in the following sections.

In some situations you will be asked for file names, memory sizes, etc. by additional smaller windows. In all theses cases you can accept the proposed value by pushing the return key or picking the OK button, you can enter new values, or you can reject the selection completely by pushing the CANCEL button (or press the tab key and then the return key).

2.4 Input Control

2.4.1 Keyboard Input

You enter commands and data for REDUCE by typing on your keyboard. The current position of input is marked in the windows by a blinking cursor. As long as you don't press the return key, the input line is available for editing:

- $\leftarrow$ and $\rightarrow$: move the cursor one place left or right.
- home and end: move the cursor to the beginning or the end respectively of the current input line.
- del: delete the character at the current position.
- esc: delete the complete input line.
- backspace: delete the character left from the cursor.
- insert: switch *insert mode* off or on (toggle). after any output from REDUCE insert mode is automatically switched on again. During active insert mode a grey cursor is used, otherwise the cursor is black.

- return: send the current line to REDUCE. It is unnecessary to move the cursor to the end of the line before.

The last 30 input lines are kept in an internal buffer. You can call them back to the actual input line by using the keys

- ↑ fetch the next older input line,
- ↓ fetch the next younger input line.

An old input line fetched in that style is handled just like fresh input from the keyboard: you can edit it until you press return.

2.4.2 Using the Mouse

You can use the mouse for data transfer in the same style as with most Windows programs:

- set cursor: when the mouse points into the current input line and you click the left button, the text cursor is set to this position;
- mark: move the mouse to the first character of the text to be copied, press the left button and keep it pressed, move the mouse with pressed left button to the last character of the text and release the button. During the selection the text currently marked is represented in inverse color;
- Copy the marked text is copied to the internal clip board buffer. As long as *autocopy* is enabled, every marked text is copied to the clip buffer immediately after marking - usage of the Copy menu entry then is superfluous inside the REDUCE window;
- Cut the marked text is copied and removed from its current position – this function is enabled only if a portion of the current input line has been marked before.
- Paste the text in the internal clipboard buffer is copied to the actual input. Alternately you can us the middle or right mouse button for Paste. If only one line of the buffer is copied, you have to repeat the paste operation.
- Autocopy if you don't like the automatic copy as a side effect of mark, you can switch it off by disabling the Autocopy entry in the Edit menu. A check mark displays its present state.

A text entered by Paste is handled just like ordinary keyboard input: the last (or single line) which is not yet completed by return can be edited using the above facilities.

Copy and paste are not limited to the actual REDUCE window: you can move data to and from other windows (e.g. a window which currently displays a REDUCE source text) and you can copy data from previous window lines (see below).

Extended copy/paste services are

- CopyBitmap the marked text is copied to the internal clip board buffer as bit map (true snapshot of the current output window). Such a bitmap can be used in other programs with graphic facilities, e.g. windows paintbrush or MS Word. A bitmap cannot be reentered to REDUCE.
- print similar to CopyBitmap, but the bitmap is sent to a printer. If nothing has been selected, the complete window is sent after confirmation. This works under Windows NT only.

2.4.3 Function Keys

Each of the keys F1 – F9, F11 and F12 can be used as abbreviation for frequently used text input (F10 is used by Windows). The text is assigned to a key by selecting the menu entry set F1–F12. If the text should contain one or more carriage returns, you must use the hash mark instead because the key Return terminates the text input. The assigned texts are saved in the file *win.ini* such that they will be available again in following sessions.

2.5 Output Control

2.5.1 Fonts

By default, text is displayed using the system default fixed width font, which is a bold Helvetica character set. The *Fonts* entry offers some more fonts and font sizes. Here mainly True Type fonts are used, which are scalable to any size. For the standard input and output REDUCE needs fixed width fonts. Therefore the highly proportional fonts such as Times Roman, Arial are accompanied by fixed or almost fixed width fonts. In such cases the proportional font is used only for output in typesetting style.

2.5.2 Paged output

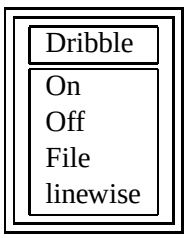
Normally the REDUCE output is scrolled up automatically. Often the output is too fast for reading. You can slow down the output by setting the toggle Page mode: the output is then halted after each window and a small window pops up which waits for a continuation confirmation.

2.5.3 Scrolling in Previous Output

REDUCE saves the last 300 window lines in an internal buffer. You can access them either by using the vertical scroll bar or by using the keys Page↑ and Page↓. The scrollbar thumb represents the position of the current display relative to the total available information. You can copy from these lines, but not more than one windows size. The display immediately jumps down to the most recent section as soon as you enter information by keyboard of by paste.

2.6 Dribble File

A dribble file is a file to which all input and output of the REDUCE session is echoed. You can create and control a dribble file by clicking Dribble in File, which generates a submenu with the entries



where On and Off allow to activate or deactivate the dribble file arbitrarily. The present state is annotated by a check mark at either On or Off. Upon the first activation you are asked for a file name - the default entry is $reduce.lst$. If the selected file exists already you are asked whether to overwrite or extend it. Later you can change the dribble file by selecting File - then a new alert box for a file name appears. If you select the same file name as before you can overwrite it from its beginning. If the file name does not begin with a drive character and a colon (absolute path), the actual directory is prefixed internally.

In contrast to other REDUCE output files the dribble file is reopened and closed for every interaction. This mode of operation causes some overhead. On the other hand it guarantees that

- the data for this file are fairly safe even if the session breaks down,
- you can inspect the file by other Windows programs like the NotePad even if the REDUCE session is still active. In that operation mode the dribble file can be viewed as an extension of the internal window buffer.

If the option linewise is activated, REDUCE will open and close the dribble file for every output line. This option causes a high overhead and should be used only in hard system failure cases, e.g. if the system breaks down before the last relevant piece of output has arrived safely in the dribble file.

2.6.1 Graphics mode

If you activate the toggle Graphics the standard formula output represented by ASCII characters is replaced by a type setting form using conventional mathematical notation whenever possible. If toggling the Graphics button does not work properly, please enter `on fancy;` with the keyboard.

For controlling the output in this mode there are some additional facilities:

- the share variable `fancy_print_df` can be set to one of the values PARTIAL, TOTAL or INDEXED. This influences the printing of differential expressions. E.g. $df(f, x, 2, y)$ will be printed as
  - PARTIAL: $\frac{\partial^3 f}{\partial x^2 \partial y}$
  - TOTAL: $\frac{d^3 f}{dx^2 dy}$
  - INDEXED: f_{xxy}
- An operator can be declared `print_indexed`. Its arguments will then be printed as indexes. E.g.

  ```
  print_indexed a;
  ```

  then $a(i, 2)$ will be printed as a_{i2}
- When the switch `fancy_tex` is set ON the internal form is displayed instead of the typesetting form.

2.6.2 Postprocessing of Dribble Files

A dribble file produced under active Graphics contains REDUCE output in an internal encoding. There is a tool `log_latex` in the module `logutil` for tanslating such a dribble file into a true LaTeX

file. E.g. for translating a dribble file *reduce.log* to a LaTeX file *log.tex* type

```
load_package logutil;
log_latex("reduce.log","log.tex");
```

2.7 Directory control

REDUCE maintains its own current directory independent from the Windows setting:

- Display: the startup current directory is permanently visible in the title bar of the REDUCE window as long as it is active;
- Change Directory: you can use the REDUCE command `CD` for changing the actual directory. This command has one parameter which must be a string containing an absolute or relative directory path in the syntax of the operating system. In an extension to the logic of the operating system the command `CD` also changes the current drive if the string starts with a drive character followed by a colon. If successful, a value *T* is returned.
- Current Directory: The command *PWD* returns the current directory as a string.

When you start a new REDUCE session, a saved directory path is selected after an eventually existing initialization file `reduce.rc` has been processed. The resource file is executed with the directory specified as "working directory" in the icon box, and a `CD` in the resource file has only local effect.

2.8 Interrupt

A REDUCE evaluation can be interrupted by pressing the Break key. However, REDUCE can identify a buffered break request only if it contacts the operating system for a service, e.g. an output or an input. Additionally REDUCE will need some time to recover from the interrupt, so you might have to wait for a moment if REDUCE is in a silent computation phase. In order to avoid unbreakable endless loops, at minimum the garbage collector will inspect the keyboard input queue, even if no output or input is ongoing. Thus only computations which do not consume memory at all are unbreakable. Such sessions have to be canceled using the task manager.

An interrupt will always terminate only the current evaluation. If you had entered more than one command in an input line, REDUCE will proceed to the next evaluation. If input comes from a file, you are asked whether to continue reading or not.

2.9 Online Help

Using the Help menu you have access to various online documentation, e.g. a hypertext version of the REDUCE reference manual, and there is a summary of the key bindings of the REDUCE windows environment. There are a couple of other information available through the help button and of course through the Internet, e.g. at *http://www.zib.de/Optimzation/Software/Reduce* .

2.10 Resource Requirements

The distributed version of REDUCE requires a minimum of 4.5 megabytes for storage of the executable binary kernel file. At run time, it takes its execution size from the command line in the Windows program group entry, which has the form:

```
xxxx\BPSL.EXE -f yyyy\REDUCE.img -td mmmm
```

where `xxxx` and `yyyy` are installation specific directories. The parameter `mmmm` describes the number of bytes (as a decimal number) for the REDUCE run including code and data. A control of the stack size is unnecessary as the operating system extends the stack automatically.

You can omit `-td mmmm` and `ssss` - default values are then taken.

Thus it is good practice to have two REDUCE icons in the group, one with a standard (small) default memory size, and one with bigger memory size for special requirements.

2.11 Execution of Other Programs

The REDUCE function `SYSTEM` lets you execute other programs via a standard command line. The function has one argument which must be a string with the target command plus its parameters. The command to be started needs to be included in the PATH. A program started in that way will run as a separate task. This feature can be useful, e.g., for postprocessing of data produced by the REDUCE session. The REDUCE session will wait for the termination of the child task. If you do not want to wait, please start the child with the command *start* which terminates immediately, e.g.

```
system "start notepad myfile.red";

system "start command";   % starts the command interpreter.

system "del myfile.red";  % return immediately
```

2.12 Script Processing

The complete command line for calling a REDUCE window has two more optional parameter – value pairs:

```
xxxx\bpslw -f yyyy\reduce.img -td mmmm  -i iiii -o oooo
```

where

- `-i` followed by a file name containing the script input; if this parameter is given, an **in** command will be generated with the file name followed by a **bye** command.
- `-o` followed by a file name to receive the output of the session.
- `-m` indicates that REDUCE runs with an iconic (minimized) window.

For running scripts (e.g. in the background) you can use the REDUCE command line as described above in a command file (a $.BAT$). For redirection of input and output the parameters $-i$ and $-o$ must be used. If both are specified, REDUCE will create an output window but will not use it for the session output. So such a session runs silently without operating system overhead. The scripts of the directory `UTIL` are organized in that style.

You can control the inner operation of a REDUCE session by environment variables: the LISP function $getenv$ converts a name given as string to the corresponding value of the environment variable. If a Dollar sign is part of a file name the following characters are treated as environment variable and the file names is expanded by replacing the variable by its actual value. Example:

```
command level:
  set src=proj1\prog1.red
  set comp=yes

inside REDUCE:

  write "READING:"  ,  lisp getenv "src";
  if lisp getenv "comp" = "yes" then on comp else off comp;
  in "$src";
```

For programming direct interaction between the script driven session and the human partner there are LISP functions `yesp`, `askUser` and `tellUser`. All of them expect one parameter which must be a string. They create on the screen a little dialog box which displays the given string. Yesp then waits util you push one of the buttons [OK] or [CANCEL] and will return your choice as the logical value T or NIL. AskUser expects that you enter some characters and then push [OK] or [CANCEL]. In the case of OK these characters are returned as string to the program, NIL otherwise. Please do not enter string quotes in the text - these are generated automatically. The input is case sensitive. The function `tellUser` can be used to send a message to a user and waiting for acknowledgement.

Example of a script using an explicit prompt:

```
start:
  mod := lisp askuser("enter module name");
  if not lisp filep mod then
     if lisp yesp "file not found - retry?"
        then goto start else bye;
  in1 mod;
  ...
  bye;
```

3 REDUCE Documentation

REDUCE documents are kept in the directory DOC and DOCEXTRA. The descriptions of the packages can be found in the PACKAGES folder in the .tex files. For convenience, the documentation is also stored in a form ready for browsing or printing in a DOCEXTRA directory on the distribution CD. Most of the documentations are written for processing by LaTeX.

4 Customizing the REDUCE Environment

At load time REDUCE tries to read a file **REDUCE.RC** in the home directory (DOS or Windows95/98/2000/XP: please set the environment variable `HOME`). Under Windows NT the (predefined) variable HOMEPATH will be used. If this is not sucessful, the file is sought in the current directory. If the file exists, its contents is executed silently. The file can contain any REDUCE command in standard REDUCE syntax. It is evaluated in algebraic mode.

5 Internal Parameters

5.1 File Handling

The file names that appear in IN, OUT and SHUT statements follow normal system conventions. If the name contains special characters (*e.g.* , . \ $) it must be enclosed in double quotes ("FileName"). Internally all characters are translated to upper case equivalents. If a filename contains a dollar sign (UNIX style), the environment is searched for a corresponding variable and its value is substituted. Environment expansion for file names using the DOS style (percent signs) is not supported. If no such variable is found, an error occurs. No special or national characters are allowed in file names.

Under OS/2 and Windows NT, Windows 95/98/2000/XP, DOS and HPFS file names can be used. When a file is opened for input, at first an attempt with the full, case sensitive file name is made. If that fails, a second attempt with the file name converted to DOS (or FAT) conventions is taken. For new files the rules of the file system are applied by the operating system automatically.

5.2 Object Sizes

The maximum string and identifier lengths are almost unlimited. The current implementation allows several thousand characters in both identifiers and strings. However, we recommend that such names be limited to 24 characters or less for compatibility with other versions of REDUCE.

Arbitrary precision integer and real arithmetic are supported.

5.3 Special Characters and Interrupts

In its default mode REDUCE 3.8 is not case sensitive. The control of the character handling is now implemented by the operators `input_case` and `output_case`. Both operators accept arguments `lower, raise` or `NIL`. The former value of the case is returned, such that it can be restored easily.

input_case controls the handling of characters on input. The default is **lower**, i.e. characters are mapped to lower case on input unless this is inhibited by using an exclamation mark. `The command in-`

put_case NIL; will turn the system into a case sensitive mode. All REDUCE operators so far are stored in lowercase mode.
Examples:

```
1:    { aa, Bb, CC, !Dd ,
           if AA = aa then "not case_sensitive" else "case_sensitive"};

{aa,bb,cc,Dd,not case_sensitive}

2:    input_case NIL;

lower

3:    { aa, Bb, CC, !Dd ,
          if AA = aa then "not case_sensitive" else "case_sensitive"};

{aa,Bb,CC,Dd,case_sensitive}

4:    num(3/4);  % Standard operators are now available in lower-
case only.

3

5:     NUM(3/4);

Declare NUM operator ?  (Y or N)
```

output_case controls the handling of characters on output. The default is NIL, i.e. no change will be done. Setting the output_case to lower or raise will raise or lower all characters printed outside of a string. For example:

```
1:    input_case NIL;

lower

2:    { aa, Bb, CC, !Dd ,
          if AA = aa then "not case_sensitive" else "case_sensitive"};

{aa,Bb,CC,Dd,case_sensitive}

4:    output_case raise; % Value is nil and won't be printed.

5:    { aa, Bb, CC, !Dd ,
          if AA = aa then "not case_sensitive" else "case_sensitive"};

{AA,BB,CC,DD,case_sensitive}
```

The full 8 bit ASCII character set is supported with the exception of the character 0x80 (decimal 128).

A terminal interrupt Ctrl + C causes the REDUCE run to interrupt at the time of the next input–output action; REDUCE tries to continue, but a successful recovery is not possible in all cases.

Ctrl + G is used to terminate strings in the REDUCE interactive editor (ERGO: only when `PAGE` is ON).

5.4 Miscellaneous

The internal ordering on alphabetic characters is from A through Z followed by a through z.

To exit REDUCE type `bye;`. Under Windows alternatively you can use the system button, the Exit entry from the File menu, or combined Alt ∓ F4 to terminate the REDUCE session.

You can freeze the current state of a REDUCE session by entering the command

```
savesession "ffff";
```

where `ffff` is a file name. Then the complete REDUCE memory is written to the file `ffff`. Later you can continue the session by using this file instead of the normal REDUCE image file (the first parameter of PSLL, PSLLW or BPSL respectively) or by using the LoadSession entry from the File menu under Windows 3.1. Under Windows NT or Windows 95/98/2000/XP, please construct a new icon by copying the REDUCE icon in the REDUCE 3.8 group and edit the -f parameter.

6 Compiling of Modules $^{(prof)}$

The underlying PSL places code and some binary data in a special memory area, the BPS. That organization is normally irrelevant for the user of REDUCE, as this memory is administered in a dynamic and automatic manner. However, if load modules are produced (that is a process started with a command `FASLOUT` of `mkfasl`), the PSL compiler fixes the BPS size such that a memory overflow (which normally occurs with pretty large modules only) cannot be caught and a message:

```
BPS exhausted during FaslOut; output file too large
```

occurs. In these cases the following steps are necessary in order to succeed with the compilation:

1. Call:

   ```
   lisp gtwarray nil;
   ```

   This command returns the size of the currently free part of BPS in LISP items (1 item = 4 bytes).

2. Call:

   ```
   lisp set!-bps!-size nnnnn;
   ```

   in which `nnnnn` is a positive integer number, guarantees that at least `nnnnn` items in the BPS are free; if the current BPS is smaller, a part of the heap is taken as new BPS. So this operation is possible only if more than `nnnnn` items in heap are free.

As with a call to `SET!-BPS!-SIZE` the old free part of BPS is lost, it is highly recommended to load the compiler explicitly before enlarging the BPS. The compiler then goes to the old BPS. After a call to `SET!-BPS!-SIZE` no more calls to `SAVESYSTEM` resp. `SAVESESSION` are possible.

Unfortunately there is no way to predict the size of nnnnn, but taking a large value like 100000 after loading the compiler should be enough even for difficult cases.

So a session with a large faslout would start as:

```
REDUCE
load compiler;
lisp set!-bps!-size 100000;
 ....
 mkfasl "......";
```

7 Implementation Dependent Error Messages

A number of error messages from the underlying PSL system may be seen from time to time - please report them to Anthony C. Hearn or the REDUCE distributor if they are not caused by an attempt to build a program in symbolic mode. These include:

Unable to relocate the image ... This is caused by different address space layouts under various operating systems. A new image file `reduce.img` has to be produced. This can be done by double-clicking the `MKREDUCE` icon in the REDUCE group under Windows or by starting `..\util\mkreduce` under DOS.

Cannot start the image ... This is caused by different address space layouts under various operating systems. A new image file `reduce.img` has to be produced. This can be done by double-clicking the `MKREDUCE` icon in the REDUCE group under Windows or by starting `..\util\mkreduce` under DOS.

Zero Divide. Probably means a division by zero has been attempted.

Heap space low. This message indicates that your problem in its present form is too large for the currently available workspace. The actual heap size was determined by the command line parameters and limits imposed by your machine configuration.

Non-numeric argument in arithmetic. This means that a LISP arithmetic routine has been called with an invalid argument.

General protection fault. This indicates an illegal memory reference. It can arise from applying the LISP functions CAR and CDR to an atom in compiled code.

Segmentation violation. This indicates an illegal memory reference. It can arise from applying the LISP functions CAR and CDR to an atom in compiled code.

Stack overflow (ERGO only). This indicates a too deep (or endless) recursion. The current calculation is interrupted, but REDUCE continues to operate. Note that you can increase the stack size by modifying the command line parameters of REDUCE.

BIOS checksum does not match, please reconfigure. (ERGO only) This indicates that REDUCE does not run with the configuration for which it was installed. Unpredictable effects can occur. Please redo the installation steps. Please contact an authorized distributor if no installation material is available.