ReportPlus

James R. Jacobs

COLLABORATORS			
	TITLE : ReportPlus		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	James R. Jacobs	August 13, 2022	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

ReportPlus

1.1 Report+

```
                    #*======================*#
               #|     R E P O R T +    |#
               #|     Version 5.51a    |#
               #|    Wed 29 May 2002   |#
               #|                      |#
               #|  by James R. Jacobs  |#
               #|  of Amigan Software  |#
               #*======================*#
```


1.2 Overview

Report+ is a freeware ReAction/GadTools-based utility with twelve functions:

1. It is an enhanced, reverse-engineered, 100% byte-compatible replacement for the official Commodore bug reporting tool (40.2).

2. It can generate/edit Aminet-style readmes.

3. It can administer the Amiga Certified Software Engineer test.

4.

```
                It can generate C-style autodocs.

                5.
                It can access the official manufacturer and product ID registries ↩
                   .

                6.
                It can view IFF FORMs and their component chunks.

                7.
                It can convert between various end-of-line (EOL) formats,  ↩
                   optionally
    also detabulating and/or unwrapping.

                8.
                It can show directory byte usage for any path, optionally also
    showing duplicate files.

                9.
                It can load/save/edit A3000/A4000-type battery-backed memory.

               10.
                It can check your OS3.9 installation (or any other path) for  ↩
                   missing,
    surplus and/or outdated files.

               11.
                It can generate/edit Amiga Games Database (AGDB) reviews.

               12.
                It can perform batch processing on icons.
```

1.3 New Features

```
Since 5.51:
    . Bug fixes.

Since 5.5:
    . Bug report editor, ACSE test: converted to ReAction.
```

1.4 Usage

```
                System Requirements

                Workbench ToolTypes

                CLI arguments
               No installation is required. These are the twelve functions:

    Edit:
```

```
                Bug report

                Aminet readme

                Autodoc

                Battery-backed RAM

                AGDB review
                   Conduct:

                ACSE test
                   View:

                Hardware IDs

                IFF FORMs
                   Process:

                EOLs/tabs

                Icons
                   Report:

                Path size

                System files
```

1.5 CLI Arguments

```
                Command Information

                                ReportPlus

Format:     ReportPlus [-f=FILL] [PUBSCREEN <pubscreen>] [-n=NOLOGO]
            [[FUNCTION] <function> [ICONTYPE DISK|DRAWER|TOOL|PROJECT|
            TRASHCAN|DEVICE|KICKSTART|APPICON] [RESET [TIMEOUT] [LUNS]
            [SYNC_XFER] [SLOW_SYNC] [TAG_QUEUES] [HOST_ID <host_id>]]
            [[FILE] <file(s)...>]]\n",

Template:   -F=FILL/S,PUBSCREEN/K,-N=NOLOGO/S,FUNCTION/N,ICONTYPE/K,
            RESET/S,TIMEOUT/S,LUNS/S,SYNC_XFER/S,SLOW_SYNC/S,TAG_QUEUES/S,
            HOST_ID/K/N,FILE/F

Purpose:    To run the Report+ utility.

Specification:

    -f:     fills the backgrounds of GadTools windows.

    pubscreen=<pubscreen>:
            the name of a public screen to open on (otherwise the default
            public screen is used).

    -n:     don't display Report+ logo.
```

```
    function=1|2|3|4|5|6|7|8|9|10|11|12:
           the ordinal number of the function you wish to jump to. These
           correspond as follows:

            1  Bug report editor
            2  Aminet readme editor
            3  ACSE administrator
            4  Autodoc editor
            5  Hardware ID database
            6  IFF FORM registry
            7  EOL/tab converter
            8  Path size report
            9  Battery-backed RAM editor
           10  System files report
           11  AGDB review editor
           12  Icon processor

    -i=DISK|DRAWER|TOOL|PROJECT|TRASHCAN|DEVICE|KICKSTART|APPICON:
           the type of icon you want to convert to, if <function> is
           12. (Defaults to DRAWER).

    reset: this switch will cause the battery-backed RAM to be
           overwritten. Of course, <function> must be 9. What is written
           to it depends on the TIMEOUT, LUNS, SYNC_XFER, SLOW_SYNC,
           TAG_QUEUES, and HOST_ID arguments.

    timeout, luns, sync_xfer, slow_sync, tag_queues, host_id <host_id>
           these modify the behaviour of the RESET switch. See
                here
                           for more information.

    file=<files>:
           a list of files for the EOL converter or icon processor to
           process, if <function> is 7 or 12, or...

    file=<file>:
           a file to be opened by the bug report editor, Aminet
           readme editor, autodoc editor, IFF FORM viewer, or AGDB review
           editor, if <function> is 1, 2, 4, 6, or 11, respectively.

           (Functions 3, 5, 8, 9 and 10 do not accept CLI filenames).
```

1.6 Workbench ToolTypes

```
                The following options can be specified in the utility's .info file ↩
                    :

FILL
PUBSCREEN=<pubscreen>
NOLOGO
ICONTYPE=DISK|DRAWER|TOOL|PROJECT|TRASHCAN|DEVICE|KICKSTART|APPICON
FUNCTION=<function>

These are equivalent to the relevant
```

```
                CLI arguments
                . You
cannot give a <file> or <files> argument through Workbench ToolTypes,
nor set the battery-backed RAM.
```

1.7 Bug Report Generator

```
This function replicates the functionality of the official bug report
generator, with a GadTools rather than console interface, and also with
automatic loading and saving of configuration and sender details,
and loading/editing/saving of preexisting bug reports.
  You can use Ctrl-Up and Ctrl-Down to scroll to the extremes of the
listview.
  You can use the 'New', 'Open...' and 'Save'/'Save As...' menu items when
you are on the 'Bug Report Details' page.
  S:Report.sender and S:Report.config are used for the sender and
configuration data, respectively, in the same format as used by Report.
  S:Report.ks and S:Report.wb are not used, as Report+ extracts version
data directly from the relevant KICKSTART and WORKBENCH environment
variables.

(Taken from official Report documentation, and from 3.0 NDK)

Summary:
    Please make sure the initial one-line bug description you provide
    in each of your reports is as explicit as possible. Engineering's bug
    summary reports and bug processing tools often list only the one-line
    description for each bug. "Bug in Intuition" is a useless title
    for an Intuition subsystem bug. "Pixel trash when window dragged left"
    is a much better title.
Debug tools and wedges:                     eg. Enforcer,MungWall
    Comma-separated names of debug tools and wedges you were using
Company Name: Without Inc., Corp., etc.     eg. Amigan Software
Phone: (AreaCode) Phone-Number              eg. (12) 345-6789
EMail: Your best electronic mail address.
    If UUCP, enter address                  eg. jsmith@vendor.com
    If other, enter address (SYSTEM)        eg. jsmith (BIX)
    RETURN if none

All bug reports, compatibility problem reports, and enhancement
requests must be generated with the latest Amiga "Report" program,
or must be compatible with the output of the Amiga Report program.
This makes automatic submission and routing of bug reports possible.

Please do not fill in any restricted value fields by hand editing! Only
the Report and Report+ programs can validate these fields.

Mail bug reports to your support manager unless your support manager says
to mail directly to Amiga.
```

1.8 Aminet Readme Generator

The 'Uploader' field is, by default, taken from the S:Report.sender file, if it exists. It is shared with the 'Submission' field of the AGDB review generator.
  You can use the 'New', 'Open...' and 'Save'/'Save As...' menu items when you are on the 'Aminet Readme Details' page.
  You can use Ctrl-Up and Ctrl-Down to scroll to the extremes of the 'Directory' listview.

(Taken from official Aminet submission documentation)

Output pathname: eg. RAM:ReportPlus.readme
Don't use version numbers in filenames if possible.
   If you want to make sure your file is not renamed on the CD, only use letters, digits and _ in the name and make sure the first 8 characters are unique.
   The maximum file name length is 18 characters including the archiver suffix (.lha, .lzh). Mixed case is OK, but it should be mainly lowercase. If your file name is generic (ls, pipe), append your initials (pipe-JU).
   Please do not upload in any other file format than .lha or .lzh, except that .jpg and .mpg files can be uploaded without putting them in archives.

Short:
The only mandatory field. It will be seen in INDEX and RECENT so everyone can easily learn about your upload. Don't repeat the file name here, but if several versions of your archive exist, specify the version number here. Try to explain what the program *does*. Music should specify the style and author. Don't boast or use much uppercase.
    If your upload requires a language other than English, please mention that language here.
    Version numbers are better here than in the filename.

Uploader: eg. umueller@amiga.icu.net.ch (Urban Mueller)
Please always provide it. It lets you indicate your email address so we can contact you if something goes wrong (and this happens more often than you think).

Author: eg. amigansoftware@abime.net (James R. Jacobs)
You can indicate who created the piece of software you uploaded.

Type: eg. games/misc
You can propose a directory [and subdirectory] where the file should be moved to. Check the file TREE for possible subdirs. If you want a new dir, read info/start/newdir.txt

Replaces: eg. biz/patch/PageStreamPatch*
Lets you specify files that are superseded by your upload. Give full path. Not needed if you overwrite an earlier file with same name.
    You can overwrite old versions of your uploads by uploading again using the same file name. This is the preferred way to do updates. However don't update within 10 days of the previous upload.

Requires: eg. util/misc/ReportPlus.lha, OS2.04+, 4Mb RAM, AGA
Other archives that your upload needs to work, with full path. Also name OS, mem and chipset requirements here.

Version: eg. 3.42b

The version number of your upload. Don't use version numbers in filenames if possible.

Distribution: eg. NoCD, Aminet
Lets you specify where your upload is OK to distribute.
If you specify 'NoCD', your upload will not appear on the CDs made of this site.
If you specify 'Aminet', you only give Aminet the distribution permission.

Description:
After a blank line, you may add a longer description, that could for example be the README found inside the archive. Don't rely on people downloading the .readme file; the information found there should be in the archive, as well.
If your upload is shareware, restricted or just a demo version, mention that here.

1.9 Amiga Certified Software Engineer (ACSE) Administrator

Report+ is the official testing tool for the Amiga Certified Software Engineer (ACSE) qualifications, currently at revision 1.4.

Objectives

This qualification assists in the recognition of Amiga developers as having important skills. It is similar in concept to certain other qualifications on other platforms.

The qualification has been developed by developers for developers and all input regarding it is welcome. The qualification is officially endorsed and approved by the Industry Council Open Amiga, and was created by Amigan Software, the moderators of the Amiga Qualifications Working Group.

At the successful conclusion of study and testing, you will be able to write high-quality software for the Amiga, and will have certification to prove it.

We do not seek to deny anyone the opportunity to develop for the Amiga. These qualifications are not compulsory by any means, and are unlikely to be required for participation on various Amiga projects.

Prerequisites

You need a working knowledge of fundamental computing concepts, including microcomputer platform environments and local operating system concepts (AmigaOS) prior to entering the qualifications program.

The qualification is free of charge, and a Certification Agreement is not required.

The course

The course itself is the textbooks listed. We thought it unnecessary to write yet another manual on Amiga programming. Everything that is tested is taken from the textbooks. The official textbooks include the definitive sources for Amiga programming information.

The test itself is basically to ensure that you know the material covered in the textbooks. Of course these textbooks are only the recommended ones; third-party books do exist which cover similar material and may be useful.

Official textbooks

Amiga Developer CD 2.1 (Haage and Partner)
Amiga ROM Kernel Reference Manuals, 3rd Edition:
Hardware, Libraries, Devices, and Includes and Autodocs
(Addison-Wesley) (on Amiga Developer CD 2.1)
User Interface Style Guide (Addison-Wesley)
(available in AmigaGuide format from
http://www.users.bigpond.com/james.jacobs/amigan.html)
The AmigaDOS Manual (Bantam)
(available in AmigaGuide format from
http://www.users.bigpond.com/james.jacobs/amigan.html)

Test

This is an 'open-book' test: there is no prohibition on using the textbooks during the test itself, and such a prohibition would be unenforceable anyway.

However, the test requires you to answer the questions reasonably quickly. You either have to know the answers yourself or be able to look them up quickly. After all, it is not expected that you memorize every word in every textbook. The test is designed to measure real-world programming skills and knowledge in a random manner.

A thorough understanding of the course and test objectives is recommended prior to taking the test. The test should only be attempted once at most per day. 25 minutes are provided in which to answer 50 questions, so you can take about 30 seconds per question, on average.

There are two levels of ACSE qualification available: 90-94% (45-47 correct answers) is a 'credit' level pass, and 95-100% (48-50 correct answers) is a 'distinction' level pass. The ACSE is a respected qualification as it is not easily achieved.

The questions themselves and their possible answers are shuffled at random. Four types of gadget are used for test questions, depending on the nature of the question and the answer required. Questions may be 'passed' (deferred) until later in the test.

The test covers various Amiga programming subsystems, including some relating to AGA and OS3.5+.

Is there a way for me to help develop Amiga certification exams?

Yes! Contact Amigan Software. We rely on the feedback of professionals and enthusiasts who have expertise and experience with Amiga products and technologies.

```
Other qualifications, such as the proposed Amiga Certified Hardware
Engineer (ACHE) qualification, may be created in the future, given enough
interest.

Certification

If you are successful, an encrypted keyfile will be generated. This
should be sent to Amigan Software. In return we will send you your
personalized certificate in IFF ILBM format, and you will be registered
on the central ACSE database. This is all normally done by email;
however, it can be done by
                snail mail
                 if necessary.
```

1.10 Autodoc Generator

```
The purpose of this is to generate standard C-style autodocs, which may
then be incorporated into your source code. These autodocs may then be
processed via another utility, such as Autodoc, to extract and convert
them to readable ASCII, AmigaGuide, etc.

(Taken from Autodoc Style Guide)

When referring to a function, the standard format is FunctionName().

Capitalization should be correct. Here are some guidelines:

    1> The words Amiga, Exec, Workbench, Autoconfig, AmigaDOS,
           Kickstart, Commodore, Commodore-Amiga, etc. are all trademarks,
           and must be capitalized.

    2> Names of "things" are as defined.  For example, "OpenWindow()",
           and "a Window structure". "fiddles with your window" does not
       refer to the structure, and should not be capitalized.

modulename.type:                              eg. financial.library
FunctionName:                                 eg. StealMoney
Minimum version:                              eg. 77
Description:
    eg. Steal money from the Federal Reserve Bank.
    A one line description of what it does.
    Real sentences with periods are preferred.
Synopsis:
                     Type                      Name            Register
    eg. Return code: BYTE                      error           D0,Z
         Argument 1: STRPTR                    userName        D0
         Argument 2: UWORD                     amount          D1.W
         Argument 3: struct AccountSpec *      destAccount     A0
         Argument 4: struct falseTrail *       falseTrail      [A1]
    becomes:
        error = StealMoney(userName, amount, destAccount, falseTrail)
        D0,Z               D0        D1.W    A0           [A1]

        BYTE StealMoney(STRPTR, UWORD, struct AccountSpec *,
```

```
        struct falseTrail *);

    This has three parts:

        1> The C calling convention, where you name the parameters and
           return values.

        2> The assembly registers. Do not indicate that the library base
           goes in A6, unless there is something special about your module.
           If parts of a register are ignored, note that next to the
           register number. The standard form is the register number
           followed by the number of bits (D0:16). Only specify this if
           the upper bits are, and always will be, ignored.

        3> The ANSI standard function prototype. This must be a ready-to-
           compile indication of the function's types. Do *not* use the
           base types, use the "types.h" file. This line must compile!

Base type               Typedef    Notes

--untyped pointer--     void *     void* AllocMem(ULONG, ULONG);
--no parameter/return-- void       void RemakeDisplay(void);
--function pointer--    ?          unresolved issue

unsigned char *         STRPTR     "char *" is not acceptable!
short                   WORD
unsigned short          UWORD
unsigned short *        UWORD *    word-aligned pointer
unsigned long *         ULONG *    word-aligned pointer to ULONG object
                        BPTR       BCPL pointer

    If any of these lines are too long, exert your individuality and
    word-wrap it!
Function:
    Describe what your function does in generally accepted English.
    Keep jargon to a minimum, but don't sacrifice clarity and accuracy.
    You may even take the radical step of using a spelling checker.
    You can refer to parameters and return values by name.
Inputs:
    Describe the range and domain of each input parameter. Use the same
    name token used in the first SYNOPSIS line (so the user can match
    inputs to the descriptions). Don't forget to note the actions taken
    for NULL pointers!

    The suggestion has been made to standardize on:
    TheToken - If the description is long, then indent the second
            line by 4 spaces. Many modules currently use whatever number
            of spaces looks good.
Example:
    An optional short example of how your function is called. This
    must be tested. Write, test, then remove lines if needed to
    shorten the example. Use "..." to indicate removed sections. Do not
    edit the example after creation (unless you retest).

    Sadly some compilers do not allow nested C comments. Instead we will
    reverse the \, and have Autodoc magically fix things up.
```

```
    \* write this in your autodoc *\
    /* and autodoc will convert to the standard form */
Result:
    Describe the range and domain of each output.
    Describe which abnormal conditions produce each error output.
Notes:
    Helpful hints, warnings, tricks, traps, etc. (optional)
Bugs:
    If there are any, describe the bug, and how it can be avoided.
    List versions, workarounds, etc.
See also:
    eg. CreateAccountSpec(), security.device/SCMD_DESTROY_EVIDENCE,
        financial/misc.h
    If there are other functions which help describe the data structures,
    or are otherwise related to this function, place their names here.
    Note include files, where appropriate (it is acceptable to list
    just the ".h" file, and assume the assembly user will find the ".i").

    Functions in this module are simply listed, with () to
    indicate they are a function. Functions from other modules
    are preceded by the module name.
```

1.11 Hardware ID Database

Report+ itself knows 273 manufacturers and 175 products (the internal database). It can also make use of the boards.library (the library database), if you have it installed.

Report+ will first search one database, then if unsuccessful it will search the other one. You can adjust which is first and which is second by using the ‘Precedence’ gadget. This is like getting a second, independent opinion.

On startup, your expansion hardware is queried and the cards found are then available to browse through. After the final detected card is a ‘query card’ for displaying information about any arbitrary card. Cards are browsed one at a time. You can navigate forwards and backwards through them.

Various extra information is displayed about the cards:

```
Serial Number: unique serial number for every card (most cards don't make
               use of this).
ConfigDev    : address of the ConfigDev structure associated with this
               card.
Driver       : address of the relevant node of the driver.
Board Address: address where the card was placed.
Board Size   : size of the card in bytes.
Slot Address : which slot number.
Slot Size    : number of slots.
```

The ‘CPU’ and ‘FPU’ gadgets show the detected CPU/FPU of your Amiga.

The ‘Internal’ checkbox indicates whether the displayed information for the current card came from the internal database (ticked), or from the

library database (unticked).

The 'Official' checkbox indicates whether the relevant manufacturer number was properly allocated to that manufacturer by Amiga; that is, whether the manufacturer number is officially registered. This checkbox is only meaningful when the 'Internal' checkbox is ticked, as only the internal database contains information regarding the official registration status of the numbers. There are certain manufacturers that have both officially registered and unofficial numbers. For example, ReadySoft have the officially registered number 2100 and also the unofficial number 8448. It's worth noting that several manufacturers (notably Phase 5 and GVP) have reused the same product number for different products.

The library database also lacks product descriptions; these are only available in the internal database.

In the query card, you type a manufacturer ID number in the manufacturer ID gadget and (optionally) a product ID number in the product ID gadget. The manufacturer name, product name and product description are displayed on the left, if found.

The most current version of the official Hardware Manufacturer ID Registry is available at http://www.users.bigpond.com/james.jacobs/reg/manuf.html.

1.12 IFF FORM Viewer

You can load and browse IFF FORMs. You can display the contents of certain chunks. Click on the chunk you are interested in. If it is one of the supported chunk types, relevant information will appear in the 'chunk contents' listview.

You can also view generic information about any FORM type by entering the FORM ID in the 'FORM ID' string gadget.

Report+ knows cursory information regarding 90 IFF FORM types. It also understands over 50 chunk types. The known chunk types are from the 8SVX, ACBM, AIFF, ANIM, FTXT, HEAD, ILBM, PREF, SMUS and WORD FORM types, plus the defined generic chunks (eg. ANNO).

Any value which is out of bounds, according to the official specification of that FORM, will be appended with a '!', or replaced with '?', as appropriate.

If formatted display of the chunk is not possible, or if 'Raw view only?' if on, a hexadecimal/ASCII view of the chunk is displayed instead.

The documentation checkbox display gadgets deserve explanation:

ADCD 2.1: This FORM is documented on the Amiga Developer CD 2.1, in the Extras/IFF/IFF_FORMs directory.

RKM: This FORM is documented in the Amiga ROM Kernel Reference Manual: Devices (3rd Edition), Appendix A.

The 'Official Status' gadgets indicate the current known status of the relevant format.

Standard: This FORM is one of the original four types specified in the original EA 85 IFF standard (8SVX, FTXT, ILBM and SMUS).

The most current version of the official IFF FORM and Chunk Registry is available at http://www.users.bigpond.com/james.jacobs/reg/iff.html.

1.13 EOL/Tab Converter

Various platforms have differing ways of encoding EOL (end-of-line) sequences. You can convert between the three most important EOL formats. This is done as a fix-in-place operation; ie. the input file is overwritten with the output file. You can convert a list of files one after the other in one operation.

You can type multiple arguments, separated by spaces. Any pathnames which themselves contain spaces must be enclosed within quotes ("). You can also of course multiselect files using the ASL requester. Wildcards are allowed.

You can also simultaneously convert tabs in the file into their equivalent number of spaces. The source tab size defaults to 8 but may be changed.

Note that although the default EOL conversion setting is IBM-PC input and Amiga output, those settings are fine for detabulation of Amiga files. Also note that the Atari ST uses the same EOL format (CR+LF) as the IBM-PC.

You can also unwrap words. That is, if a file contains EOLs in the middle of sentences, you can unwrap the words so that such EOLs are converted to spaces. EOLs will only preserved at the end of paragraphs.

It is possible to use the EOL conversion functions directly from the CLI, without invoking the Report+ GUI. This is done by specifying at least one <file> on the command line. You may specify several files if desired.

When operated from the CLI, Report+ always does IBM-PC to Amiga conversions, without detabulating or unwrapping.

1.14 Path Size Report

This is useful for quickly seeing which directories are taking up the most space. Directory sizes are calculated from the totals of their contents, including subdirectories. You can specify any path you like, so that you can specify either an entire drive/partition or any directory within it. You can also optionally also generate a list of all duplicate files (files with the same name) within the specified path.

The 'Root' and 'Parent' directories allow you to easily move through the directory tree of the current drive.

Directories are marked with the '*' symbol. If you click on one of these directories, you will go inside it.

If 'Include slack?' is ticked, the size of each file is calculated according to the actual space taken up on the disk, eg. the block size multiplied by the number of blocks used.

You can view the sizes of the files and directories in bytes, kilobytes, megabytes, or blocks. If you view in blocks it does not matter whether 'Include slack?' is ticked, slack is of course included.

The duplicate file list always shows sizes in bytes, and does not include slack.

If you enable 'Log to file?', then whenever you do a path size report, the report (including the duplicate file list, if enabled) will be appended to the specified file.

1.15 Battery-backed RAM editor

This allows you to view and alter the contents of the battery-backed memory, as found in the A3000 and A4000. On other models, this does not exist, and will thus contain zeroes and will not be really writable.

Changes which have been made, but not yet written to the battery-backed memory, will be shown in white. Bits which have defined meanings have a black background.

'Revert' will reread the contents of the battery-backed memory.

'Write' will write your changes to the battery-backed memory, and then automatically reread the battery-backed memory.

Some bits have defined meanings, as described below. These are linked to the relevant checkbox gadgets; toggling the gadget will toggle the relevant bit, and vice versa. It is not considered safe to edit undefined bits.

You can load and save the battery-backed RAM from and to a 12-byte file, using the Project menu.

You can set the contents of the battery-backed RAM from the CLI, without invoking the Report+ GUI. To do this, specify FUNCTION=9 and the RESET switch. The other available switches are TIMEOUT, LUNS, SYNC_XFER, SLOW_SYNC and TAG_QUEUES. The available integer argument is HOST_ID, which can be 0-7. The entire contents of the battery-backed RAM will be written to, according to the keywords specified and according to these rules:

The AMIGA_AMNESIA and SHARED_AMNESIA bits, which are inverted, are always set (1).

The TIMEOUT, LUNS, SYNC_XFER and TAG_QUEUES bits are set (1) if the appropriate switch is given, or cleared (0) otherwise.

The FAST_SYNC bit, which is inverted, is set (1) if the SLOW_SYNC switch is given, or cleared (0) otherwise.

The HOST_ID bits, which are inverted, are set according to the HOST_ID integer, or cleared (0) if it is omitted (which is the same as specifying HOST_ID=7).

All other bits are cleared (0).

This is provided mainly for users with faulty battery-backed RAM, which may forget its contents; this apparently can occur with certain expansion hardware. Calling Report+ in this manner from, for example, the S:User-startup will automatically refresh the contents of the battery-backed RAM according to your wishes, without the need for user intervention. If you don't want to see the 'Done.' message, redirect output to NIL:.

For example:

```
1> ReportPlus 9 RESET
Done.
```

This will set short timeouts, no LUNs support, a SCSI host ID of 7, synchronous transfer requests not initiated, fast synchronous transfer requests, and no tagged queueing. Or to put it another way, this will clear (0) every bit of the battery-backed RAM, except for the AMIGA_AMNESIA and SHARED_AMNESIA bits, which are set (1).

```
1> ReportPlus 9 RESET SLOW_SYNC SYNC_XFER TIMEOUT HOST_ID=4 >NIL:
```

You would give this command if you want long timeouts, no LUNs support, a SCSI host ID of 4, synchronous transfer requests initiated, slow synchronous transfer requests, and no tagged queueing. No output will be generated.

(Taken from RKRM: Devices)

The battery-backed memory (BattMem) preserves a small portion of Amiga memory while the system is powered off. Some of the information stored in this memory is used during the system boot sequence.

The system considers BattMem to be a set of bits rather than bytes. This is done to conserve the limited space available. All bits are reserved, and applications should not read, or write undefined bits. Writing bits should be done with extreme caution since the settings will survive power-down/power-up.

(Taken from resources/battmembits#?.h)

AMIGA_AMNESIA ($00): The battery-backed memory has had a memory loss. This bit is used as a flag that the user should be notified that all battery-backed bit have been reset and that some attention is required. Zero indicates that a memory loss has occured.

SHARED_AMNESIA ($40): The battery-backedup memory has had a memory loss.

This bit is used as a flag that the user should be notified that all battery-backed bit have been reset and that some attention is required. Zero indicates that a memory loss has occured.

SCSI_TIMEOUT ($01): Adjusts the timeout value for SCSI device selection. A value of 0 will produce short timeouts (128 ms) while a value of 1 produces long timeouts (2 sec). This is used for Seagate drives (and some Maxtors apparently) that don't respond to selection until they are fully spun up and intialised.

SCSI_LUNS ($02): Determines if the controller attempts to access logical units above 0 at any given SCSI address. This prevents problems with drives that respond to ALL LUN addresses (instead of only 0 like they should). Default value is 0 meaning don't support LUNs.

SCSI_HOST_ID ($41-$43): A 3 bit field (0-7) that is stored in complemented form (this is so that default value of 0 really means 7). It's used to set the A3000 controller's SCSI ID (on reset).

SCSI_SYNC_XFER ($44): Determines if the driver should initiate synchronous transfer requests or leave it to the drive to send the first request. This supports drives that crash or otherwise get confused when presented with a sync xfer message. Default=0=sync xfer not initiated.

SCSI_FAST_SYNC ($45): Determines if the driver should initiate fast synchronous transfer requests (>5MB/s) instead of older <=5MB/s requests. Note that this has no effect if synchronous transfers are not negotiated by either side. Default=0=fast sync xfer used.

SCSI_TAG_QUEUES ($46): Determines if the driver should use SCSI-2 tagged queuing which allows the drive to accept and reorder multiple read and write requests. Default=0=tagged queuing NOT enabled.

AMIX ($20-$3F): See AMIX documentation for these bit definitions.

1.16 System File Report

This function makes lists ('snapshots') of the files in a certain path, including their versions, dates, times, and sizes, and then later can compare this snapshot against the current contents of the path, showing the differences.

The main use for this is to make a snapshot immediately after installing AmigaOS (and BoingBags). This will give you a snapshot of a clean AmigaOS installation. Then you can later compare your SYS: partition against this to determine which files have changed. This can be helpful in, for example, uninstalling software, or to check whether any system files have

been accidentally deleted, or accidentally overwritten with an old version, or infected with a linkvirus, etc.

It will tell you:

- files found which ARE in the snapshot AND which have correct versions/dates/times/sizes;
- files found which ARE in the snapshot BUT which have incorrect versions/dates/times/sizes;
- files found which are NOT in the snapshot ('3rd-party files');
- files NOT found which ARE in the snapshot ('missing files').

You can suppress reporting of any or all of these categories, using the appropriate 'Show' option.

If 'Log to file?' is selected, the report will be appended to the given file.

See below for Amiga's recommendations about what is deletable, although note that those recommendations seem to only cover components that were also present in OS3.1.

Sorting of the output into each category is possible using the AmigaDOS SORT command, if this is desired.

After the operation is completed, the log file (if any) will be written and you will see the report in a listbrowser gadget.

You can use Ctrl-Up/Down to scroll to the extremes of the listbrowser, Shift-Up/Down to scroll by pages, and Up/Down to scroll one line.

A sample snapshot is provided, consisting of an ideal 'clean vanilla' installation. This is provided primarily for example purposes: you are better to make your own snapshot immediately after you finish installing AmigaOS and the associated updates. This is because there are many configurations of AmigaOS which are valid, depending on what options are selected during installation.

The sample snapshot is of the following, in the order listed:

```
AmigaOS 3.9 including Internet, CD-ROM and PowerPC options
GenesisPrefs Update    ( 9 February 2001)
Boing Bag #1 for OS3.9 ( 4 April    2001)
Euro Update            (24 January  2002)
Boing Bag #2 for OS3.9 (20 March    2002)
Shell Update           (24 April    2002)
```

It assumes you have not copied/installed any Contributions or Manuals from the AmigaOS 3.9 CD and Boing Bags #1 and #2. Only American keymaps and English language are supported, but all printers are assumed to be installed. Generally the defaults suggested by the scripts have been accepted. No PowerPC or graphics card is assumed.

(Taken from OS3.9 official user documentation)

Files You Can Delete:

Delete files from your Workbench disk starting with the least critical files, such as the Clock program, and programs that do not apply to your system, such as C:MAGTAPE if you do not use a tape drive and System/NoFastMem if you do not have any Fast ("other") memory. Delete the AmigaDOS text editor C:EDIT to free approximately 18K of disk space. If you do not need to use CrossDOS, delete DEVS:mfm.device, L:CrossDOSFileSystem, and L:FileSystem_Trans, to free approximately 33K. Deleting everything in Rexxc, in addition to System/RexxMast, System/RexxMast.info, LIBS:rexxsyslib.library, and LIBS:rexxsupport.library frees almost 50K of disk space. However, we do not recommend this since many Amiga applications use ARexx and must call upon these files.

Files to Avoid Deleting:

Do not delete the following under any circumstances:

- Any directory normally found on the Workbench disk; delete only the files within directories
- DEVS:parallel.device
- DEVS:printer.device
- DEVS:serial.device
- LIBS:asl.library
- LIBS:commodities.library
- LIBS:diskfont.library
- LIBS:iffparse.library
- LIBS:locale.library (if your language and country are not English and united_states)
- S:Startup-sequence
- L:Port-handler
- Any non-Internal command that appears in the default Startup-sequence

Do not delete any file whose purpose you do not know. Do not delete more files than necessary to fit the new material you intend to add to the disk.

1.17 Amiga Games Database (AGDB) Review Generator

The Amiga Games Database (AGDB) is a website hosting user reviews of Amiga games. The AGDB review generator helps you to create reviews for submission to the AGDB. Report+ has been endorsed by the AGDB administrators as the recommended means of creating AGDB reviews.

The website can be found at:

http://www.angusm.demon.co.uk/AGDB

The ‘Submission’ field is, by default, taken from the S:Report.sender file, if it exists. It is shared with the ‘Uploader’ field of the Aminet readme generator.

You can use the ‘New’, ‘Open...’ and ‘Save’/‘Save As...’ menu items when you are on the ‘Amiga Games Database (AGDB) Review Details’ page.

(Taken from official AGDB submission documentation)

If you're a bit fuzzy about who released the game or its compatibility don't worry, I'll knock it into shape as best I can. The important thing is to say what you think about the game in your own way, and please try and give a general description of how the game is visually presented; for example, "The game is viewed from above and scrolls in such a way as to always keep the player's character in the centre of the screen."

One other thing. We're happy to have another review of a game that's already been done, but if we've already got three reviews of a game, then please don't send in a fourth. Three should definitely cover it. You can check which games to avoid reviewing on the Overkill list.

If you do submit a review, it becomes copyright of the AGDB, and I reserve the right to use it in any way I see fit, for example; attempt to correct the spelling, edit the length, submit it to a magazine etc.

Send your review to angus@angusm.demon.co.uk.

Below is an example review for your reference.

```
Title:           The Sentinel
Publisher:       Firebird (1988)
Game Type:       3D Combat Sim
Players:         1
HD Installable:  Patch on Aminet
Compatibility:   Not AGA
Submission:      angus@angusm.demon.co.uk
```

Review:
The Sentinel by Geoff Crammond was first released for the C64 and I believe the BBC computer. I remember reading the review in Zzap 64 and I think two out of three of the reviewers loved it to death, but Jazza Rignall was not so happy, and felt that it certainly wasn't everyone's cup of tea. Which I suppose is fair enough. What Sebntinel is though, is one of the most original games ever created. Obviously the Amiga version of the game is somewhat enhanced on the old 8-bit machines, but the basics remain pretty much the same.
The Sentinel is about energy management. You, the player, exist on a sort of island floating in space, and you must navigate your way to the highest point of the island. It's more complicated than that though. The islands, there are thousands of them, are rather like a chessboard, but have many different levels, effectively hills and valleys, of various sizes. The "floor" though is chequered. Your movement is performed by creating a replica of yourself and "beaming" across to it. You would then typically absorb the husk of your old body, thereby losing no energy. It is possible to ascend one level at a time, in this way, and so access the island's summit. But it's more complicated than this..... Each island has a Sentinel, that stands on the summit, and rotates 30 or 40 degrees every few seconds. If he can see the square on which you're standing, he will start absorbing your energy, which depletes rapidly. Now on the island, many of the squares have trees on them, a tree is the lowest unit of energy. You can make trees, although there's no particular point, you can also make boulders, which are equivalent in energy terms to two trees, or replicas of yourself (robots); worth three trees. The Sentinel (the bad guy, not the game) is programmed to start absorbing from anything it can see worth more than a tree, ie a boulder or a robot. This is bad news for

you, but the good news is you can absorb trees, and boulders, and ultimately, when you can see the square he's standing on, the Sentinel himself. Boulders are particuarly useful, because you can use them to make an artificial platform on which to stand. This game is largely about your elevation. Well, that's about it, as you can see the main problem is explaining what the game is about, rather than saying how much you like it. It is a superbly creative game, which can generate some very exciting moments, particuarly when the Sentinel starts using Sentries to help him. The Amiga version is much faster paced than the original and includes a useful option to view the landscape as a whole while your playing. There is also some cleverly composed background music that seems appropriate to the gameplay. A classic game.

1.18 Icon Processor

This lets you perform certain operations on an icon or list of icons. You can change the icon type (project, tool, etc.), and can also select whether or not to enable various options which reduce the size of the icon file(s). This is done as a fix-in-place operation; ie. the input file(s) is/are overwritten with the output file(s). You can convert a list of files one after the other in one operation.

An example use for this is to convert a list of drawer icons into project icons. Report+ allows you to automate such procedures.

You can type multiple arguments, separated by spaces. Any pathnames which themselves contain spaces must be enclosed within quotes ("). You can also of course multiselect files using the ASL requester. Wildcards are allowed.

It is possible to use the icon processing functions directly from the CLI, without invoking the Report+ GUI. This is done by specifying at least one <file> on the command line. You may specify several files if desired. If you don't specify an <icontype> argument, the default is DRAWER.

When operated from the CLI, Report+ always optimizes colours, and does not preserve planar data.

(Taken from V44 icon.library autodocs)

'Optimize colours': A palette-mapped icon image might not use the entire icon palette. If this checkbox is FALSE, icon.library does not bother about this, it expects the creator of the icon to choose its palette efficiently. But just in case, you can tell icon.library to attempt to identify how many colours are really in use and optimize its image compressor for them. This may take extra time and is not recommended for daily use.

'Preserve planar data': If you decide that the palette-mapped icon imagery is sufficient to represent an icon, you can tell icon.library not to store the original, planar icon image data with the icon file. To do this, set this checkbox to FALSE. Instead of the planar icon images, a single default image will be stored. This can result in space savings but may not look too pretty.

1.19 System Requirements

Hardware	Required	MC68020+
		about 512K free RAM
	Recommended	Colour monitor
		Mouse
		Battery-backed clock
		A3000/A4000-style battery-backed memory
		about 1Mb free RAM
Firmware	Required	Kickstart R3.1+
Software	Required	AmigaOS 3.9+
	Recommended	Worm Wars 6.81+
		boards.library (included)

1.20 Other Information

1.21 Contact Details

Credits

Report+ was designed and programmed by James Jacobs of Amigan Software.
Main menu graphics were drawn by Martin 'Mason' Merz.

Special thanks to Thomas Beck for his suggestions and bug reports.

Licence

Report+ is freeware. It has been written for the benefit of the Amiga
community. There are no limits on usage, distribution or modification,
except that you are not allowed to modify and/or distribute it (or
modified versions of it) for commercial purposes without consent.

Bugs

Amiga development and style guidelines have been adhered to,
using the official Amiga Technical Reference Series as authoritive
reference.

There are no known bugs. Please contact us immediately if any bugs are
found. Please use the supplied
bug reporting tool

```
                :-D !

Contact details

EMail             amigansoftware@abime.net

Website           http://www.users.bigpond.com/james.jacobs/amigan.html

Mail              James Jacobs
                  Amigan Software
                  11 Yate Gdns
                  RIVETT ACT 2611
                  Australia

Phone             Australia (02) 6287 4917
```

1.22 Amiga Development System

```
Hardware    Virtual A1200 (MC68020+FPU on 233MHz Pentium)
            1.2Gb 3.5" IDE hard disk (2Gb compressed)
            2Mb chip RAM
            8Mb fast RAM
            1Mb RTG RAM
            56Kbps Dynalink (Rockwell) modem
            ESS MF-1868 (SoundBlaster Pro-compatible) sound card
            AcerView 56L digital colour monitor running as
                15Hz PAL (640*512)
            Battery-backed clock
Firmware    Kickstart 3.1 (R40.63)
Software    WinUAE 0.8.8 R9
            AmigaOS 3.9 with all updates
            SAS/C 6.58
            Amiga Developer CD 2.1
            OS3.9 NDK
            Amiga Lint 2.0b
            Autodoc 1.0
            BlowUp
            boards.library
            CheckGuide
            CodeWatcher 1.4
            CXXC 1.4
            CygnusEd Professional 4.2
            Deluxe Paint 5
            IFF 2 Source 1.0
            IO_Torture
            KingCon 1.3
            LhA 1.51
            MungWall
            PatchWork
            PrintA
            Report 40.2
            Sashimi
            SmartCrash
            StackSnoop
            VirusChecker ][ 2.2 (BF 2.18)
```

```
            Worm Wars 7.0

Thanks to all those whose software was used to create Report+.
```

1.23 Source Code

```
Source code is provided, with the deliberate omission of the f3.c (ACSE
test) module.

This utility is written for SAS/C 6.58, but should be portable to StormC
without major changes.
```

1.24 History and Future

```
History

5.51a: Wed 29 May 2002.
5.51:  Thu 16 May 2002.
5.5:   Wed  8 May 2002.
5.4:   Tue 16 Apr 2002.
5.31:  Sun 17 Mar 2002.
5.3:   Fri  8 Feb 2002.
5.22:  Sun 20 Jan 2002.
5.21:  Sun 23 Dec 2001.
5.2:   Sun 18 Nov 2001.
5.1a:  Sun  4 Nov 2001.
5.1:   Mon 15 Oct 2001.
5.06:  Thu  4 Oct 2001.
5.05:  Tue 25 Sep 2001.
5.04:  Thu 20 Sep 2001.
5.03:  Mon 10 Sep 2001.
5.02:  Tue 21 Aug 2001.
5.01:  Sat 11 Aug 2001.
5.0:   Sat 28 Jul 2001.
4.71   Sun  8 Jul 2001.
4.7:   Fri 29 Jun 2001.
4.64:  Thu  7 Jun 2001.
4.63b: Tue 29 May 2001.
4.63a: Thu 24 May 2001.
4.63:  Sat 19 May 2001.
4.62:  Sun 13 May 2001.
4.61:  Fri 11 May 2001.
4.6:   Tue  8 May 2001.
4.51:  Sat 28 Apr 2001.
4.5:   Sat 21 Apr 2001.
4.4a:  Mon 16 Apr 2001.
4.4:   Tue 27 Mar 2001.
4.32:  Thu  8 Mar 2001.
4.31:  Thu  1 Mar 2001.
4.3:   Wed 21 Feb 2001.
4.24:  Wed 14 Feb 2001.
4.23:  Thu  1 Feb 2001.
```

```
4.22:  Wed 24 Jan 2001.
4.21:  Wed 27 Dec 2000.
4.2:   Sat 16 Dec 2000.
4.11:  Sat  2 Dec 2000.
4.1:   Thu 16 Nov 2000.
4.0:   Wed  1 Nov 2000.
3.5:   Wed 11 Oct 2000.
3.4:   Sat 23 Sep 2000.
3.3:   Sat 26 Aug 2000.
3.22:  Sun 30 Jul 2000.
3.21:  Sun 16 Jul 2000.
3.2:   Tue  4 Jul 2000.
3.12:  Sun 25 Jun 2000.
3.11:  Sun 18 Jun 2000.
3.1:   Tue  2 May 2000.
3.0:   Wed 12 Apr 2000.
2.1:   Tue 22 Feb 2000.
2.01:  Fri 11 Feb 2000.
2.0:   Sat 29 Jan 2000.
1.1:   Sat 11 Dec 1999.
1.0:   Fri  3 Dec 1999.
```

Future

It is our intention to update Report+ as necessary so that it always matches the latest official revisions of the relevant file formats and qualifications.

There are also other enhancements which are possible. Contact us to suggest new features.

1.25 Worm Wars

Worm Wars 7.0

Worm Wars is an arcade game for up to 4 players, who travel around a series of rectangular maze leaving a deadly trail behind them, fighting 14 types of creature and collecting 32 varied object types.

The integral field editor allows you to load, edit and save user fieldsets, for greater lasting attraction. There is support for playing MED and IFF 8SVX files as music and sound effects respectively.

It is enjoyable either for one player, or for competitive multiplayer games, and demo mode is available. Amiga or human control can be specified for any worm. Two keyboard players, four joystick players, and/or four CD32 gamepad players are supported. It is system-friendly and style compliant. The Amiga version is freeware.