ReportPlus

James R. Jacobs

COLLABORATORS

	TITLE : ReportPlus		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	James R. Jacobs	August 9, 2022	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

ReportPlus

1.1 Report+

```
                                #*======================*#
                                #|     R E P O R T +    |#
                                #|      Version 3.5     |#
                                #|    Wed 11 Oct 2000   |#
                                #|                      |#
                                #|  by James R. Jacobs  |#
                                #*======================*#
```

1.2 Overview

Report+ is a freeware ReAction/GadTools-based utility with nine distinct 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 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 access the official IFF FORM registry.
7. It can convert between various end-of-line (EOL) formats, optionally also detabulating.
8. It can show directory byte usage for any path.
9. It can edit A3000-type battery-backed memory.

1.3 New Features

```
. EOL converter: expanded to optionally also detabulate.
. Path size reporter: root and parent gadgets, and you can now click
    on any directory to go into that directory.
. It now normally opens as a window on the default public screen.
. The About... requester is now always available.
. Bug reporter, Aminet readme generator and autodoc generator now
    have 'Menu' buttons.
. ACSE: updated to 1.12 (improved parsing of string input).
. Miscellaneous bugfixes.
```

1.4 Usage

System Requirements

Workbench ToolTypes

CLI arguments

No installation is required.

If diskfont.library and WormWars.font can be opened, Report+ will use WormWars.font (an 8x8 monospaced bitmap font included with the game

Worm Wars

); otherwise, Topaz 8 will be used.

Report+ opens its windows on the default public screen (generally Workbench), unless the PAL option is specified, in which case it opens a custom 640x256 screen with a backdrop window.

Bug report generator

Aminet readme generator

ACSE administrator

Autodoc generator

Hardware ID database

IFF FORM registry

EOL converter

Path size report

Battery-backed RAM

1.5 CLI Arguments

```
Command Information

                                  ReportPlus

Format:         ReportPlus [-p=PAL] [FUNCTION <function>]

Template:       REPORTPLUS -P=PAL/S,FUNCTION/N

Purpose:        To run the Report+ utility.

Specification:

    -p:         if this is specified, the utility creates a 640x256
                custom PAL screen. If it is not specified, the utility
                opens its windows on the default public screen.

    function=<function>:
                a number from 1 to 9, representing the ordinal number of
                the function you wish to jump to.
```

1.6 Workbench ToolTypes

```
                The following options can be specified in the utility’s .info file ←
                    :

PAL
FUNCTION=<function>

These are equivalent to the relevant
                CLI arguments
                .
```

1.7 Bug Report Generator

```
If you are running OS3.5+, you use the texteditor.gadget for completing
multi-line text fields, otherwise you use your specified editor.

(Taken from official Report documentation)

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@endor.COM
    If other, enter address (SYSTEM)         eg. jsmith (BIX)
    RETURN if none

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 from environment variables.

T:ReportPlus.temp is used as a temporary file.

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.

The original contact details suggested by Report 40.2 are, of course,
no longer valid.
```

1.8 Aminet Readme Generator

```
If you are running OS3.5+, you use the texteditor.gadget for completing
multi-line text fields, otherwise you use your specified editor.

(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. james_jacobs@altavista.net (James R. Jacobs)
You can indicate who created the piece of software you uploaded.

Type:                eg. games/misc
You can propose a directory 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 ACSE Administrator

Report+ is the official testing tool for the ACSE 1.12 qualifications.

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.

Obviously you will also need an OS2.04+ Amiga with which to run Report+.

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 (Addison-Wesley)
The AmigaDOS Manual (Bantam)

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.

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. 10 minutes are provided in which to answer 30 questions, so you can take about 20 seconds per questions, on average.

There are now two levels of ACSE qualification available: 90-94% (27-28 correct answers) is a ‘credit’ level pass, and 95-100% (29-30 correct answers) is a ‘distinction’ level pass. The ACSE is a respected qualification as it is not easily achieved.

Both 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 ACSEs database.

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.

Any instances of '!' in the output file are placeholders to indicate where you should enter text. (Replace them.)

(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

You can access the official Commodore Applications and Technical Support registries of hardware manufacturers and products. The registry used covers 64 manufacturers and 175 products.

Cards are browsed one at a time. You can navigate forwards and backwards through them.

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.

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 and click the query button. The manufacturer name, product name and product description are displayed on the left, if found. Clicking the query button updates the information display, based on the ID numbers typed by the user.

1.12 IFF FORM Registry

You can access the official Commodore/Electronic Arts registry of IFF FORMs. The registry used covers 90 IFF FORMs.

IFF FORM names can be found by various utilities. Generally, they are located in the second longword of the file.

You type an IFF FORM ID code in the FORM ID gadget and press ENTER, RETURN, Help, Tab or Shift-Tab. The IFF FORM description and contributor, and other information, are displayed below, if found. Pressing ENTER, RETURN, Help, Tab or Shift-Tab updates the information display, based on the ID string typed by the user.

The following boolean display gadgets deserve explanation:

CD-ROM: 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.

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

1.13 EOL 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 multi-select files using the ASL requester.

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.

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.
  The 'Root' and 'Parent' directories allow you to easily move upwards 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 you enable `Log to file?', then whenever you do a path size report, the report will be appended to the specified file.

1.15 Battery-backed RAM

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

Set bits are indicated by '#'. Clear bits are indicated by '-'. Changes which have been made, but not yet written to the battery-backed memory, will be shown in white; bits which have not been modified are shown in black.

'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 specific meanings, as described below. These are linked to the relevant checkbox gadgets; toggling the gadget will toggle the relevant bit, and vice versa.

(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 Requirements

```
Hardware             Required        about 256K free RAM
                     Recommended     Colour monitor
                                     Mouse
                                     Battery-backed clock
                                     A3000-style battery-backed memory
Firmware             Required        Kickstart R2.04+
                                         exec.library V36+
                                         dos.library V37+
                                         gadtools.library V37+
                                         intuition.library V37+
                     Recommended     Kickstart R3.1+
Software             Required        Workbench/CLI R2.04+
                                         asl.library V37+
                                         version.library
                     Recommended     MultiView
                                     AmigaOS 3.5
                                     Worm Wars 5.0+
                                     diskfont.library
                                     expansion.library
                                     window.class
                                     layout.gadget
                                     button.gadget
                                     texteditor.gadget
                                     label.image
```

1.17 Other Information

1.18 Contact Details

Licence

Report+ is freeware. It has been written as a service to the Amiga community. There are no limits on usage, distribution or modification, except that you are not allowed to modify and/or distribute it for commercial purposes or port it away from the Amiga without consent.

Permission is hereby granted to the Amiga community in general to use and distribute this program as they deem appropriate.

Bugs

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

Please contact us immediately if any bugs are found. Please use the supplied bug reporting tool :-D !

Contact details

EMail	james_jacobs@altavista.net
Website	http://users.interact.net.au/~cjaj/amigan.html
Mail	James Jacobs Amigan Software 11 Yate Gdns RIVETT ACT 2611 Australia
Voice	+61 (02) 6287 4917

1.19 Amiga Development System

Hardware	Virtual A1200 (14MHz MC68EC020+FPU on 133MHz Pentium) 1.2Gb 3.5" IDE hard disk (compressed to 2.4Gb)

```
            2Mb chip RAM
            4-8Mb fast RAM
            1Mb RTG RAM
            56Kbps Dynalink (Rockwell) modem
            Quickshot QS-209F Skyhawk joystick
            ESS MF-1868 (SoundBlaster Pro-compatible) sound card
            NEC MultiSync 2A colour monitor running as
                15Hz PAL (640*512)/uaegfx (640*480)
            Battery-backed clock
Firmware    Kickstart 3.1
Software    WinUAE 0.8.6 R6 and 0.8.14 R1
            Fellow 0.3.6
            OS3.5 with Boing Bag 1 and Installer 44.7
            SAS/C 6.58 with SLink and SAS/C Editor and CodeProbe
            Amiga Developer CD-ROM 2.1
            AmigaGuide Writer 1.02
            Amiga Lint 2.0b
            BlowUp
            CheckGuide
            CodeWatcher 1.4
            CXXC 1.4
            CygnusEd Professional 4.2
            Deluxe Paint 5
            Gguide2txt
            IFF 2 Source 1.0
            IO_Torture
            KingCon 1.3
            LhA 1.51
            MungWall
            OctaMED 5 and MEDPlayer Programmer's Sources
            PatchWork
            Picasso96 2.0
            PrintA
            Report+
            Sashimi
            SmartCrash
            SoundBox 2.9 beta
            StackCheck
            StackSnoop
            StackWatch
            VirusChecker ][ 2.2 (BF 2.18)
            Worm Wars 6.4
```

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

1.20 Source Code

This utility is written in SAS/C 6.58. Source code is not provided, to preserve the secrecy of the ACSE test. That is the only reason that this program is closed-source.

If you do not intend to take the test, or have already done so successfully, we can provide you with source code at your request.

1.21 History and Future

History

```
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.

If you have a copy of the manufacturer, product, developer or IFF registries, please contact us, so that Report+ can be updated accordingly.

There are also other enhancements which are possible. Contact us to suggest new features. But never MUI, because that would not be an enhancement! :-)

1.22 Other Software

Worm Wars 6.4

Worm Wars is an arcade game for 0-4 players. It combines the playability of its basic concepts with 32 interesting object types, 10 species of creature, and other enhancements, for more diverse and strategic gameplay.

One to four worms travel around a rectangular maze leaving a deadly trail behind them, competing and sometimes cooperating with other creatures, collecting letters to advance to the next level.

The integral field editor allows you to load, edit and save user fieldsets, for greater lasting attraction. You can shuffle the levels if desired. There is support for playing MED and IFF 8SVX files as music and sound effects respectively. Isometric 3D and overhead graphics are available. Custom fonts and backgrounds are used.

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