

CONTENTS

1	Installing ARP.....
1	1. How compatible are the ARP Commands?.....
1	Wildcards.....
2	Environment Variables.....
3	2. ARP Command Enhancements.....
4	Assign and Mount speed up your startup sequence.....
4	Let ARP Copy Quick.....
5	Rename-ing, Copy-ing, and Move-ing files.....
5	AShell: Three commands in one.....
6	Type, Join and Protect.....
6	Search and Sort.....
6	CD and PATH.....
7	ASH.....
7	3. Disclaimer, warranty and distribution.....
8	4. CREDITS.....
8	

Overview of the 1.3 ARP Release

AmigaDOS Resource Project

c/o Microsmiths, Inc.
PO Box 561
Cambridge, MA 02140

Electronic Mail:
BIX, People Link: cheath
CIS: 76004,1766

All software, manuals, install programs and associated material are Copyright (c) 1987,88,89 by Arp Authors. All Rights Reserved.

The ARP Installation program and accompanying documentation may be freely distributed provided no changes or additions to the materials are made.

AmigaDOS is a trademark of Commodore-Amiga, Inc.
Unix is a trademark of AT&T Information Systems.

Installing ARP

To install the new 1.3 ARP commands on your system you use the ArpInstall program. This program can be run from either the Workbench (by clicking on its Icon) or if you prefer, from the CLI. This program uses a simple mouse driven interface to install the ARP CLI style commands on your system. You can select the commands you want, and where you want them placed on your disk, or you can elect to accept the default setup.

Normally, you will want to install the ARP commands on your bootdisk or in the appropriate drawers on your harddisk. However, if you wish to install the ARP commands on another disk, please insert this disk before you begin answering the questions that ArpInstall will be asking you.

1. How compatible are the ARP Commands?

Every effort has been made to make the ARP commands 100% compatible with the AmigaDOS V1.3 commands. You can use the

documentation in the AmigaDOS Users Manual and the AmigaDOS V1.3 enhancer manual with the ARP commands, and read further for a brief discussion of the more important enhancements made to each command.

Wildcards

The standard AmigaDOS pattern matching abilities are present in ARP commands as described in the AmigaDOS Users Manual. In addition to these basic capabilities, ARP provides the following enhancements in pattern matching:

- Use of the star ('*') to match any pattern. This corresponds to the AmigaDOS pattern '#?'. This use of the star character is so common as to be almost a de facto industry standard. Note that the AmigaDOS pattern matching is implemented as well: use the one that you like the best.

- The ability to match a class of characters. You specify the class of characters to be matched by enclosing them within square brackets. For example, the pattern 'file.[ch]' will match file.c or file.h, but not file.o or file.ch (you match only one of the characters in the class). To specify a range of characters, you can use [a-z], which will match any character between a and z inclusive.

- The ability to match 'anything but' a pattern. To do this, precede your pattern with the tilde('~') character. For example, to list all files except those which end in .info, use this pattern: '~*.info' or '~#?.info'.

- Smart 'tick' matching. The tick (single quote) is used by AmigaDOS to remove the special meaning of any wildcard character. For example, to refer to a literal, actual question mark in a filename, you would use '"?'. The AmigaDOS (and ARP) commands will interpret that two character sequence as a single question mark. Problems arise with filenames that contain a single quote, quite a common occurrence. ARP commands try to be clever about when to tick and when not to tick. For example, "Mike's Drawer" will be understood by the ARP commands to be "Mike's Drawer", but an AmigaDOS command will think it is "Mikes Drawer". To do this under AmigaDOS (and it works with ARP as well) you can use "Mike"'s Drawer".

While these extra features add additional power welcomed by most users, they are also a potential compatibility issue, since they are not supported by the current AmigaDOS commands. Fortunately, the problems are minor, and rarely occur. Here are some of the potential problems, and their solutions, roughly in order of frequency of occurrence:

(The ARP Eval command uses C-language Printf format, such as "%lx", rather than the BCPL "%X" formatting.)

-Conflicts with the use of '*' as a wildcard and as a filename or as an escape character. AmigaDOS uses the star (*) as a filename (referring to the current window) and as an escape character. This is used most commonly with the Type and Copy commands. If you run across an old script that does this, or if you wish to use this yourself, you can issue the command "SET BCPL TRUE", which will cause Type and Copy to revert to the older behavior. Conflicts with '*' as an escape character are less likely, but possible. We suggest you add the statement "SET ESCAPE \ BCPL FALSE" in your Startup-Sequence, and use the "\" character as the escape character.

-OR patterns in ARP follow the AmigaDOS documentation. OR patterns are AmigaDOS patterns which look like this: (File1|File2|File3). This will match any one of File1 or File2 or File3. AmigaDOS commands allow you to leave out the parenthesis under certain circumstances, ARP commands implement the specification in the AmigaDOS manual: You must use parenthesis with the OR patterns in ARP commands.

-Funny characters in filenames: The AmigaDOS commands don't recognize the extended pattern matching characters of ARP (*[]~) as special. This means that a non-ARP using friend could create a file called "[LIST]", for example, using an AmigaDOS command that could cause you difficulty. The short term solution to this is to use the tick (') character in front of these special characters. So to delete the "[LIST]" file, you could type: Delete '[LIST]'. The long term solution is to give your friend a copy of the ArpInstall program.

Environment Variables

Before Commodore released 1.3, the only environment variables available on the Amiga were the MANX/Rokicki variables, which former ARP releases were compatible with. With AmigaDOS 1.3, Commodore introduced environment variables officially, but unfortunately not compatibly. The current ARP implementation is designed to help ease the conversion to the new implementation of environment variables as endorsed by Commodore.

Environment variables using ARP read both the old style environment, and the new ENV:, however, we write only the older format. This means, essentially, that you can use any format you prefer with ARP programs, and they will be able to find the

value of the environment variable you assigned.

Some programs (for example, older Manx programs) do not use the ARP library function calls and so will work only with the older style variables.

2. ARP Command Enhancements

Most of the ARP supplied commands have added capabilities when compared with their AmigaDOS cousins. In this section, we present a brief overview of the more significant enhancements, for a full description of each command, consult the complete ARP documentation (distributed separately at a later date as ARPDO3.ZOO).

One thing common to all the ARP commands is an extended help template. As you may already know, the AmigaDOS commands provide help in the form of a 'command template' in response to a question mark as an argument. ARP also provides these commands, and, in addition, provides an extended help feature if you type another question mark once you are presented with the template. For example, typing:

Search ?

brings up the normal command template, which in this case is:

From/a,Search,ALL/s,NUM/s,QUIET/s,QUICK/s,FILE/s,CASE/s:

If you were to now enter another "?" in response to the template prompt, you would get more information, in this case:

*Usage: Search <wildcards | STDIN> [Searchstring] [ALL]
[QUIET | QUICK] [FILE] [CASE]*

This extended help often prevents trips to the manual.

Assign and Mount speed up your startup sequence

ARP's Assign and Mount commands accept multiple assignments or Mounts on one command line. By taking advantage of this to specify all your assignments or Mounts in one stroke, you can avoid running each command several times in your startup-sequence, thus speeding things up. Here are two example command lines:

Mount dh0: dh1: dh2: pip:

Assign C: dh0:c LIBS: dh0:libs DEVS: dh0:Devs Fonts: dh0:Fonts

Let ARP Copy Quick

The new ARP Copy command has an option which will automatically skip a copy when the source and destination files are identical. The Copy command considers files identical when they have the same date and length. It will also compare Filenotes if you have enabled Filenote copying. This can dramatically speed up directory or disk copies from place to place. You can use it on request by using the QUICK keyword on Copy's command line, or you can make it a default by setting the "copyflags" environment variable. The copyflags variable lets you specify many default actions for copy, see the ARP manual page for Copy for complete details. As an example, to have Copy always use the QUICK option and also to copy the protection bits and date you can use:

```
Set copyflags=CQ
```

To specify a copy to or from the current directory, you can use a single dot (.) character. This is similar to the convention found on other operating systems, such as Unix. (Search and CMP also allow this usage).

Rename-ing, Copy-ing, and Move-ing files

The new ARP Move command is an enhanced version of Rename. Move can do everything Rename can, and will also move a file from disk to disk. This is cleaner and easier than doing a Copy and then a Delete, especially when transferring multiple files. Both Rename and Move can use wildcards as a source pattern, so you can Rename or Move multiple files with one command. (Note: this is an enhancement: The Commodore Rename command will not accept a wildcard pattern.)

ARP's Rename, Move and Copy commands also allow you to specify a simple substitution in the destination name, for example:

```
Rename *.c *.cBAK
```

See the complete ARP documentation for more information on replacement patterns.

AShell: Three commands in one

AShell is a replacement for NewCLI, SYSTEM/CLI and NewSHELL, this one small command takes on the functions of all three. It's default action is to always provide you with the 'best' shell available. You can also request a CLI by using the CLI keyword, or by copying it or renaming it "NewCLI", in which case it will do its level best to act like the standard NewCLI command.

By editing the tool type for the Shell icon to refer to this file, you will also be able to eliminate the SYSTEM/CLI file from your disk. See the manual page for AShell for more information on this versatile little command.

Type, Join and Protect

Type, Join and Protect allow wildcard patterns unlike their AmigaDOS cousins. ARP Type also has two new options: B (for Banner) and F (for formfeed). B prints a small header at beginning of each file containing the name of the current file, F prints a formfeed (which clears the screen or advances the printer paper) at the end of the file. These options are most useful when typing multiple files. Specify these with the OPT keyword:

*Type *.c OPT BF*

If you do not provide a source filename to Type, it will read from the keyboard. This allows it to be used as part of a pipeline. If typing from the keyboard, use CONTROL-\ to exit type.

Search and Sort

Search allows you to specify a wildcard pattern as a search string, instead of the simple literal string which the AmigaDOS Search allows. You can also select to have a case sensitive search using the CASE keyword (default is to ignore case). The ARP Search command will store the last search string used in the environment variable "Search". Running Search again with no search string will use the value of the "Search" environment variable. Text editors or databases which use ARP can also examine this variable.

You can also use patterns to specify a range of files to search, you can specify a directory, or you can use dot (.) to mean the current directory (see Copy and CMP for another example of this use of dot).

ARP's Sort is quite safe, it will not crash, even with large files and the default stack, unlike AmigaDOS's Sort. Both Search and Sort can read from the keyboard instead of from a file, which allows them to be used in pipelines. To use Sort in a pipeline, just omit the input filename. To use Search, you must use the filename STDIN (and it must be capitalized, as shown here). For more information on pipelines, see the ASH (Arp's shell) manual.

The ARP Search command is three times faster than the AmigaDOS Search, and ARP's Sort is about five times faster than the AmigaDOS Sort, which are nice enhancements (compatible too!).

CD and PATH

Enhancements here are mainly allowing specification of directories using wildcards. This is a big help when typing long directory pathnames!

NOTE:Old friends of ARP should note that the %P support has been removed from CD. (New friends of ARP who don't understand this shouldn't worry.) It is now in the shell where it belongs.

ASH

ARP is now distributing a shell replacement for Shell-Seg, the 1.3 Commodore supplied shell. It is reasonably compatible with Shell-Seg, and also provides some very significant enhancements. The best way to get acquainted with Ash is to read the short users manual included in this zoo file.

One of the most significant features of ASH is it's use of the arp.library process functions and resident features. ARP's resident is superior to AmigaDOS's in terms of safety and memory usage. Other features include command substitution and piping, built-in batch language, and more!

3. Disclaimer, warranty and distribution

We make no warranty for fitness of use of any of the ARP commands, arp.library, Installation program (ArpInstall) or the accompanying documentation. The user assumes all responsibility related to his or her use of any portion of the ARP distribution. We have made considerable efforts to insure that ARP works reliably and as documented but cannot assume any liability for problems that may be related to any use of ARP.

ARP V1.3 may be freely redistributed in the form of the ZOO files which will be initially distributed by ARP Support onto BIX, Compuserve and People Link. These files may be placed on other BBS's without charge provided the original contents and organization of these ZOO files are not altered in any way.

You may make up to fifty (50) printed copies of the ARP documentation without specific permission from ARP Authors. ARP Authors reserve all commercial rights for printed versions of the documentation.

For users groups and other vendors of public-domain diskettes, we request that you contact us to get the complete ARP release diskette. The V1.3 ARP release diskette is not complete as of this writing, please send a SASE with your request and we will let you know when the complete V1.3 diskette is available.

For applications developers who would like to use arp.library or include the ARP commands with their commercial software packages: We encourage this use of arp.library and place only minimal restrictions on your distribution intended to help insure that you distribute verified copies of the latest version of the ARP library and command programs. If you would like to be registered as an ARP developer, please send a SASE to ARP Support and we will send you further details. You are also welcome to use arp.library without registering for support, but we request that you register with us if you want to include a copy of any part of the ARP distribution with your release.

4. CREDITS

ARP - The AmigaDOS Resource Project - is a cooperative effort by a group of Amiga developers to enhance the Amiga.

ARP has three main goals:

Provide enhanced commands for Amiga users.

-Provide a resource for Amiga developers to help build smaller, more consistent, more powerful applications using "arp.library"

-Make this work available to Commodore in order to encourage enhancements to AmigaDOS.

ARP V1.3 represents our third major release. The first release was made in October 1987 (called V1.0). The second release was in March 1988 (V1.1). With Version 1.3 of ARP, we have addressed the limitations and incompatibilities of the ARP V1.1 commands compared with Commodore's V1.3 enhancer release, also providing a much more complete command set, several new commands not found in the Commodore V1.3 set, and an ARP Shell. At this time we do not have any specific plans for the next release of ARP, and we welcome feedback from users and developers which will help us choose our future directions.

The ARP commands and arp.library were written in assembler, using Innovatronic's "CAPE" and Manx's "AS" for development. The ArpInstall program was written in Modula-2, using M2S Ltd's "M2Sprint".

ARP is not ShareWare. If you would like to support the ARP project, the best ways to do it are to spread the word about ARP to other Amiga users, to write and distribute programs which use ARP, and to support the commercial and shareware products which take advantage of ARP's capabilities, many of which are also written by members of the ARP team. These include (but are not limited to):

-the TxEEd Plus package from Microsmiths (this includes a printed version of the ARP manual).

-Cape, a 68010 macro assembler from Innovatronics.

-M2Sprint, an implementation of Modula-2 from M2S Ltd, which provides a full interface to arp.library.

In addition, you might want to investigate Bill Hawe's products. Bill has not been personally involved with ARP, but his work complements ARP, and he was also one of the first to distribute the ARP command set with his products.

There have been many people who have helped with the development and testing of ARP, and I am sure I will miss a few who should be mentioned here. The team has a rare quality of cooperation and excitement and it has been a joy to work with this group.

Arp 1.3 Overview

March 30, 1989

Charlie Heath, VP of Microsmiths, Inc. - ARP Coordinator
Scott Ballantyne - one of the original ARP hackers.

Martin Taillefer - wrote ArpInstall program (among other things).

Ken Salmon - programmer for portions of V1.1 of ARP

Willy Langeveld - developer and beta tester

Bill Barton - beta tester for arp.library

Les Noland - beta tester for ARP commands

Chuck McManis - provided prototypes for V1.0 of ARP

John Toebe - provided prototypes for arp.library

Wes Howe - programmer

Bill Hawes - contributed "LoadLib" for V1.3 of ARP

Steve Tibbett, Joanne Dow, Justin McCormick, Andy Levy,

Mike Scalora, John Spadafora, Jeff Blume,

Marvin Weinstein, Warren Block, Eric Haberfellner,

Michael Sinz, Paul Ockenden, Larry Phillips,

Brian Waters - beta testers for V1.3 of ARP.

