ZIP AND UNZIP Dilwyn Jones

Zip is a utility program for packing a set of files into a single file, known as an "archive", with the capability of compressing files. The term "compressing" means reducing the space taken by those files, although zip can store files uncompressed in an archive if it thinks that attempting to compress those files may result in a file which is actually larger than the original.

In simple terms, what Zip does is to try to reduce the storage space needed for a set of files and to combine them into a single large file. This is useful for distributing programs via websites and bulletin boards, for example, because downloading a single file is more convenient than downloading several individual files, and if the archive is smaller than the sum of the individual file sizes, then of course you are online for shorter periods and so your telephone bill benefits too.

Zip is the program which does the packing and compressing.

Unzip is the program which decompresses and unpacks the files back into their original form.

INFO-ZIP

This is the name of the organisation which co-ordinates the Zip and Unzip rograms for various computing platforms such as Linux, Windows, RISC OS, Amiga, Atari and of course QDOS. Info-Zip is a diverse, internet based workgroup of about 20 primary authors and over 100 beta-testers, originally formed in 1990 in the USA. The Info-Zip programs are basically free to use and copy, subject to the terms of the licence (basically inclusion of Info-Zip copyright notices within redistributions), and you can get hold of the Zip/Unzip source code files if you wish to see how the software works.

INFO-ZIP FOR QDOS

One of the "team of twenty" is our own Jonathan Hudson, who ported the version for QDOS and SMSQ/E systems. At the time of writing, the version numbers for the official QDOS versions were Unzip version 5.41 and Zip version 2.3. You may also come across version 5.32 of Unzip which is a perfectly usable version in my experience.

It is important that you use the Info-Zip versions of Zip and Unzip. Older versions by various authors exist, but these have limitations such as inability to handle level 2 directories, being slower than the current version and the possibility of creating zipped files which work only on the system on which that file was created. Info-Zip is a cross-platform system which means that for people like Jonathan who create files on one system and move them to other systems they stand a good chance of working on that other system. So stick to the Jonathan Hudson versions of the QL Zip and Unzip programs.

OTHER ARCHIVERS

There are other archiving programs out there for various systems, including QDOS. You may come across LHA, Zoo, and RAR for example - versions of these by various authors exist for the QL. You can get most of these programs from my website at www.dilwyn.me.uk/arch/index.html

Broadly speaking, Zip is the most common archiver for the QL these days, as you will see if you visit any website or bulletin board offering QL software for download. I suggest you start off using Zip and Unzip and explore the others later as your confidence in using such programs grows.

Such archivers are generally "portable" - files compressed on Windows, say, can generally be decompressed on a QL. The only areas where you may encounter some difficulty are with QL executable programs when they are unzipped on a non-QL system causing loss of the job headers, or with filename extension separators. The QL prefers '_' between parts of filenames, other systems generally prefer '.' - although QL Zip and Unzip cope automatically with any conversion requirements where needed.

COMMAND LINES

Zip (and Unzip) is what is generally known as a "command line" utility. In other words, yuo type in commands to make it do what you want. Modern programs are generally menu driven or pointer driven to make them easier to use, and Zip on the QL is no exception. You can get "front end" programs like Archivers Control Panel to make Zip and Unzip (as well as other archiving programs) easier to use and we'll look at these in the next part of this series.

OBTAINING COPIES OF ZIP AND UNZIP

Since Info-Zip is freely copyable within the terms of the licence, you can get hold of copies on disk or CD from PD libraries, or by downloading copies from websites. Usually, the package will include everything - the programs themselves, the documentation files and sometimes the source code files (usually the sources are available separately). The list of files is quite large and you may not be interested in the source code files, for example, so you will need to know which parts you need to keep.

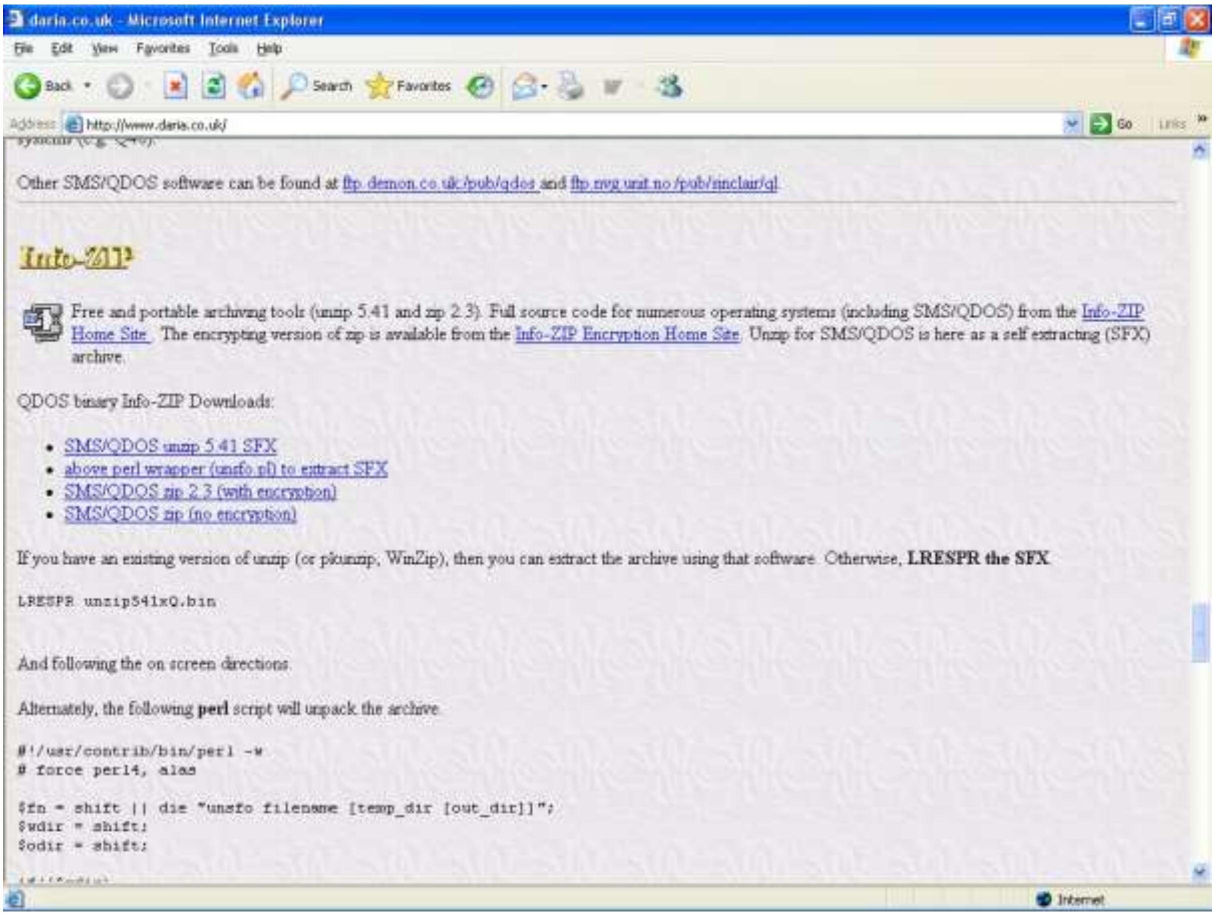
If you are downloading copies from the web, for example, you will generally find that Zip and Unzip themselves come to you in a zipped format! This means you have to have a copy of Unzip already to decompress them, thus putting you in a bit of a sticky situation if you do not already own a copy! Fear not, Jonathan Hudson has kindly made the distribution of Unzip available in a format which will "self-extract" as long as you follow the instructions to the letter.

Here are some web page addresses from where you can get copies of Zip and Unzip. Note that Zip comes in two forms - you can get a version with file encryption facilities and a version which does not. The latter is generally smaller and is good enough for most purposes unless you specifically need encryption facilities.

Jonathan Hudson's website:

http://www.daria.co.uk

On this site, scroll way down the page until you reach the section for Zip and Unzip (see Figure 1) and click on the underlined link for the package you wish to download.



daria.co.uk - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://www.daria.co.uk/
Go Links
Internet

Other SMS/QDOS software can be found at ftp.demon.co.uk/pub/qdos and ftp.nvg.unit.no/pub/sinclair/ql

Info-ZIP

Free and portable archiving tools (unzip 5.41 and zip 2.3). Full source code for numerous operating systems (including SMS/QDOS) from the Info-ZIP Home Site. The encrypting version of zip is available from the Info-ZIP Encryption Home Site. Unzip for SMS/QDOS is here as a self extracting (SFX) archive.

QDOS binary Info-ZIP Downloads:

- SMS/QDOS unzip 5.41 SFX
- above perl wrapper (unsfo.pl) to extract SFX
- SMS/QDOS zip 2.3 (with encryption)
- SMS/QDOS zip (no encryption)

If you have an existing version of unzip (or pkunzip, WinZip), then you can extract the archive using that software. Otherwise, **LRESPR the SFX**

```
LRESPR unzip541xQ.bin
```

And following the on screen directions.

Alternately, the following **perl** script will unpack the archive.

```
#!/usr/contrib/bin/perl -w
# force perl4, alas

$fn = shift || die "unsfo filename [temp_dir [out_dir]]";
$wdir = shift;
$odir = shift;
```

Figure 1 - Zip/Unzip on Jonathan's website (jrhzip01.jpg)

The first link is for downloading a copy of Unzip. This will download a file called UNZIP541xQ.BIN which a self-extracting version of QDOS Unzip v5.41, which is basically a zipped copy of the Unzip files, but with a small amount of extra code to help it extract the files without needing an existing copy of Unzip.

The third and fourth links are to download whichever version of the Zip package you prefer to use. With these you will get files called ZCR23xQ.ZIP (for the version with encryption) or ZIP23xQ.ZIP. Once you have a copy of Unzip, you can use it to decompress either of these packages.

If for any reason you cannot get copies from Jonathan's website, you can get copies from the official Info-zip website at www.info-zip.org or you can use their FTP (File Transfer Protocol) site at:

ftp://ftp.info-zip.org/pub/infozip/

If all else fails, try my website's Archivers page at:

www.dilwyn.me.uk/arch/index.html

INSTALLING ZIP AND UNZIP

Assuming you downloaded these packages on a PC, transfer the archives to a QL or to your emulator and I'll explain how to unpack them ready for use.

The first one to unpack is Unzip, for obvious reasons - you'll need it to unpack the Zip package!

As explained above, UNZIP541xQ.BIN is a Self Extracting file for the QL. Ensure no other programs are running (not even the hotkey job if you are using pointer environment - a HOT_STOP command will stop that if needed). You will need Toolkit 2 active on your system - if you are using SMSQ/E the commands needed are already built into your system, and most QL systems with disk drives these days have Toolkit 2 built in. Some interfaces such as a Gold Card may need a command such as TK2_EXT to activate the toolkit commands, if your boot program does not already do this.

Your system needs to have expanded memory too, at least 384K of RAM, but the more the merrier! It means that you cannot unpack on a demo version of QemuLator with just 128K of RAM, for example.

Ideally, your system will have a ramdisk (most systems do these days).

Version 5.41 of Unzip will state that it needs something called 'signal extensions' which is a small toolkit of job communication extensions by Richard Zidlicky. You can usually find this in PD libraries or on the web in a zip file called SigExt30_zip. If you have these extensions, fine, but Unzip will work perfectly well without them, although version 5.41 of Unzip will issue a message saying that the extensions are missing. Ignore the message.

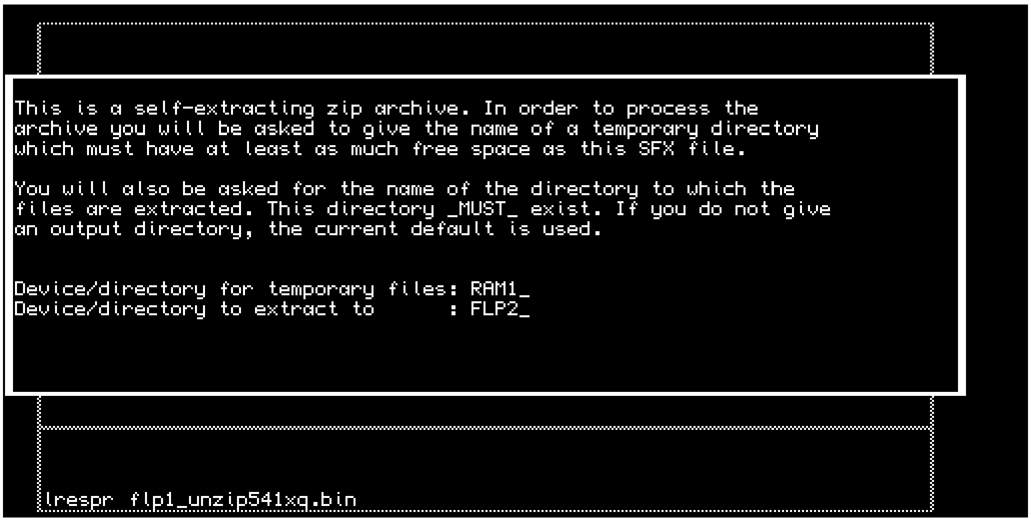
I'll assume that the UNZIP541xQ_BIN file is on a disk in FLP1_ and that you will be decompressing it to FLP2_ and using RAM1_ for temporary files created during the unpacking process. If you only have a single disk drive, you can unzip to a ramdisk temporarily and copy the files to an empty floppy disk later.

To start with, issue an LRESPR command on the file:

```
LRESPR "FLP1_UNZIP541xQ_BIN"
```

If it has not been renamed to a QL standard filename with '_' characters in it, the filename may be FLP1_UNZIP541xQ.BIN, it does not really matter but if it ends with '.' and it is being processed on a QDOS system, you should put the filename in quotes as QDOS will not allow '.' characters in unquoted filenames.

The screen shown in figure 2 will now pop up, asking you to enter the temporary files device and the device to extract to. Enter RAM1_ for the first question and FLP2_ for the second (assuming these are the drives being used of course).



```
This is a self-extracting zip archive. In order to process the
archive you will be asked to give the name of a temporary directory
which must have at least as much free space as this SFX file.

You will also be asked for the name of the directory to which the
files are extracted. This directory _MUST_ exist. If you do not give
an output directory, the current default is used.


Device/directory for temporary files: RAM1_
Device/directory to extract to      : FLP2_
```

```
lrespr flp1_unzip541xq.bin
```

Figure 2 - The 'SFX' screen (sfx.gif)

The program will now create the temporary files on RAM1_ and issue the following message:

```
[ To extract the files run the command LRUN RAM1_SFX_bas ]
[ Press any key to exit.                                 ]
```

Press a key for this part of the program to finish, then as prompted type this command into BASIC:

```
LRUN RAM1_SFX_bas
```

In most cases, this will automatically complete the unzipping process by itself. On some systems (and my QPC2 v3.32 system seems to be one of these) it fails to run properly and gives an error message, in which case you'll have to intervene manually and type in these commands (which is essentially what the SFX_bas program contained) yourself:

```
EW RAM1_SFX_EXE;'-d FLP2_'
delete ram1_SFX_exe
delete ram1_SFX_dat
delete ram1_SFX_bas
```

Hopefully you will now have the required files on FLP2_ and the temporary files will have been cleared out of the ramdisk by the delete commands.

If you cannot get the "self-extract" procedure to work at all even with the above help, you will need to obtain a copy of unzip on disk from somewhere, or unzip the files in Windows or Linux and copy them to a QL disk. This is not recommended as the executable file header will be lost (Unzip will give the error Bad Parameter when you try to execute it) and some heavy duty programming needed to restore them, like this.

```
100 fl = FLEN(flp2_unzip")
110 base = ALCHP(fl)
120 LBYTES FLP2_Unzip,base
130 DELETE flp2_unzip
140 SEXEC Flp2_Unzip,base,fl,51810
```

What this little program does is to load what is left of the program and uses the SEXEC command to save the executable, adding a dataspace value of 51810 to 'repair' the program.

FILES NEEDED

Basically, you need all the programs and documentation files. The main programs are the files called Zip and Unzip. There are various other utilities and all sorts of documentation files - read these when you get a chance although some may appear a bit too technical at this stage for less experienced users.

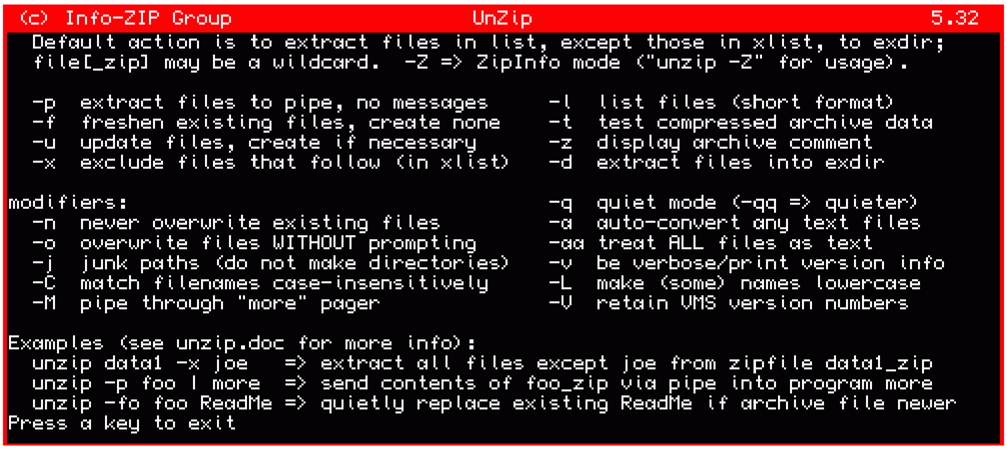
On my system, I've thrown all the unzip package files into a directory called win1_unzip_, all of the Zip package files into win1_zip_ and put copies of Zip and Unzip into the directory where I keep copies of all my QL programs, win1_exec_

GETTING STARTED

Before we make any real use of either program, I need to point out that the programs have a somewhat limited screen of help information built in - to see this, just execute the program with no parameters:

```
EXEC FLP1_UNZIP
```

This will display the help screen shown in figure 3.



```
 (c) Info-ZIP Group                     UnZip                                5.32
  Default action is to extract files in list, except those in xlist, to exdir;
  file[_zip] may be a wildcard.  -Z => ZipInfo mode ("unzip -Z" for usage).

  -p  extract files to pipe, no messages     -l  list files (short format)
  -f  freshen existing files, create none    -t  test compressed archive data
  -u  update files, create if necessary      -z  display archive comment
  -x  exclude files that follow (in xlist)   -d  extract files into exdir

modifiers:                                   -q  quiet mode (-qq => quieter)
  -n  never overwrite existing files         -a  auto-convert any text files
  -o  overwrite files WITHOUT prompting      -aa treat ALL files as text
  -j  junk paths (do not make directories)   -v  be verbose/print version info
  -C  match filenames case-insensitively     -L  make (some) names lowercase
  -M  pipe through "more" pager              -V  retain VMS version numbers

Examples (see unzip.doc for more info):
  unzip data1 -x joe   => extract all files except joe from zipfile data1_zip
  unzip -p foo | more  => send contents of foo_zip via pipe into program more
  unzip -fo foo ReadMe => quietly replace existing ReadMe if archive file newer
Press a key to exit
```

Figure 3 - Unzip's built in help screen (unzip.gif)

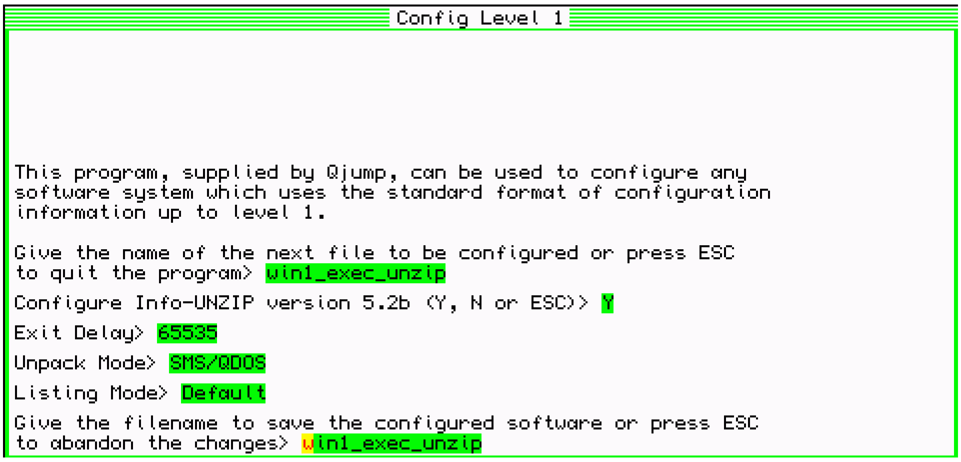
It is also important to know that Unzip has a configuration block built in, which lets you configure some aspects of how the program behaves. 3 options can be set for Unzip:

1. Timeout for the 'press any key' display before Unzip finishes. This is expressed as 65535 (which means wait forever for a keypress), 0 (which means don't wait at all) or a value from 1 to 32,767 in units of 1/50th of a second to wait for a keypress before giving up and finishing anyway. A value of 50 means 1 second, while the highest value of 32,767 allows for a delay of about 660 seconds.

2. Unpack mode: this can be SMS/QDOS (filenames use '_' characters) or default value (probably non-QDOS - "." character in filenames)

3. Listing mode - can be default (non-QDOS) or specifically SMS/QDOS.

Figure 4 shows the configuration process. You can use either the Config or MenuConfig programs to alter these settings for Unzip. You will be able to set Zip as well once we have unpacked that program.



```
                                Config Level 1

This program, supplied by Qjump, can be used to configure any
software system which uses the standard format of configuration
information up to level 1.

Give the name of the next file to be configured or press ESC
to quit the program> win1_exec_unzip

Configure Info-UNZIP version 5.2b (Y, N or ESC)> Y

Exit Delay> 65535

Unpack Mode> SMS/QDOS

Listing Mode> Default

Give the filename to save the configured software or press ESC
to abandon the changes> win1_exec_unzip
```

Figure 4 - the configuration process (config.gif)

Hopefully, we now have a working copy of Unzip, so we can go about unpacking a working version of Zip.

You should have downloaded a file called either ZIP23xQ.ZIP or ZCR23xQ.ZIP, it doesn't really matter which you got, since the only difference is that the second is a version with file encryption or 'cloaking' facilities and is slightly larger.

To decompress the package you need to use the EW command to pass a small list of parameters to Unzip to tell it exactly what to do:

```
EW FLP1_UNZIP;'-d FLP2_ FLP1_ZIP23xQ.ZIP'
```

The -d FLP2_ part tells it to unzip the file called FLP1_ZIP23xQ.BIN to disk drive FLP2_. If, like me, you only have a single disk drive on your computer, you can use -d RAM1_ instead and have it unpacked to ramdisk RAM1_, then later copy all of the files to a floppy disk. If you have a hard disk system, you can extract the files direct to a directory on that:

```
MAKE_DIR WIN1_ZIP_
EW FLP1_UNZIP;'-d WIN1_ZIP_ FLP1_ZIP23xQ.ZIP'
```

You will have noticed how unwieldy these unzip commands can be - this is what I meant by describing the programs as "command line" utilities. You have to use not very memorable commands to achieve anything, which is why programs like Archivers Control Panel will come in useful later, although if you can't remember the commands, it's possible to write small BASIC programs to help you use Unzip and Zip.

Once Zip has been unpacked, you can use Config or MenuConfig programs to configure the defaults for Zip. There are only 2 options with Zip, one sets the timeout value as for Unzip, while the other tells Zip which is the file type number representing a directory. On most modern QL systems (99% of systems!) it will be 255 for level 2 devices, although older systems such as Thors may use type 3 or 4.

USING UNZIP

Now for the fun part, actually using Unzip. In all cases, I'll assume we'll be using it to unpack a file called FLP1_EXAMPLE_ZIP which we have downloaded from the web.

Unzip normally (unless you explicitly tell it otherwise) unpacks files to the DATA_USE default drive. On a floppy disk system, it is usually FLP1_ and on a hard disk system WIN1_, although of course you can set it to just about anything with a DATA_USE command, so if you want to unzip to RAM3_ just issue a DATA_USE RAM3_ command before the Unzip command:

```
100 DATA_USE RAM3_
110 EW FLP1_UNZIP;'FLP1_EXAMPLE_ZIP'
```

Line 100 sets where the unzipping will be sent to, and line 110 starts the unzip program, telling it to unzip FLP1_EXAMPLE_ZIP. Note how the filename is specified as a string after a semi-colon, this is how these option commands are passed to programs like Unzip using EX or EW commands. Note: although some versions of the EXEC and EXEC_W commands have been extended to allow use of these command parameters, it is normally better to use the Toolkit 2 EX and EW versions.

It is possibly to explicitly tell the program where to unzip to using a "-d destination" command. Zip has all sorts of these one letter commands available, preceded by a hyphen. -d states which drive/directory to unzip to:

```
EW FLP1_UNZIP;'-d RAM4_ FLP1_EXAMPLE_ZIP'
```

This will tell unzip to unpack EXAMPLE_ZIP from FLP1_ to ram4_.

It is also possible to view a list of what files are contained in EXAMPLE_ZIP, using a -l command (l for listing):

```
EW FLP1_UNZIP;'-l FLP1_EXAMPLE_ZIP'
```

This is useful if you think you may not need to unpack every file. You can get a list and write down those files you think you'll need and just extract those, by specifying the names after the name of the archive. For example, suppose we only want to extract prog1_bas and prog2_bas files from EXAMPLE_ZIP:

```
EW FLP1_UNZIP;'-d RAM4_ FLP1_EXAMPLE_ZIP prog1_bas prog2_bas'
```

Note that these names may need to match the case of those stored inside the archive. If it says they are called prog1_bas and you enter PROG1_BAS it may not work! There may also be problems such as filenames with names clashing with QDOS directory names, for example. Some of these filename clash problems can be hard to diagnose and not always easy to solve, so be aware of this as a possible cause of a mysterious problem you might run into at some stage!

Unzip also supports '*' and '?' wildcard options to selectively extract and process files. * stands for any sequence of characters, and ? stands for any single character. So if you wanted to extract only files ending with _bas (BASIC programs) you could try this:

```
DATA_USE RAM4_
EW FLP1_UNZIP;'FLP1_EXAMPLE_ZIP *_bas'
```

Note: you could have used either _bas or .bas, it seems to recognise both.

And if there were several versions of a program, e.g. prog1_bas, prog2_bas, prog3_bas and so on, you could use prog?_bas as a wildcard to extract all of them.

```
EW FLP1_UNZIP;'FLP1_EXAMPLE_ZIP prog?_bas'
```

Difficult to grasp at first, but you get used to it after a while.

Normally, if a file already exists, you get a 'yes/no' prompt offering to overwrite it. It is possible to bypass this and overwrite without asking using a -o command to overwrite without asking - useful (if dangerous!) if you know you will need to overwrite a large number of files without having to answer yes or no every time.

As well as specifying which files are to be unzipped, you can also tell it to specifically exclude specified files using a -x command. For example, if you don't want the text files (those ending with _txt) you can use a command like this:

```
EW FLP1_UNZIP;'FLP1_EXAMPLE_ZIP -x *_txt'
```

A PROGRAM TO UNZIP

I mentioned you could write a BASIC program to control Unzip. This is a little easier than you might think. Here's an example to extract all files from a specified zip file:

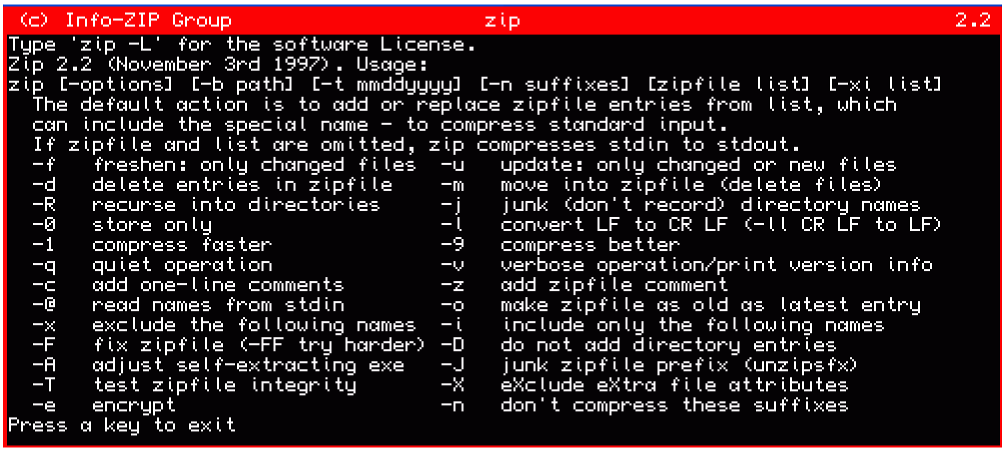
```
100 CLS : CLS #0
110 INPUT #0,'File to unzip > ';ip$
120 INPUT #0,'Unzip to > ';op$
130 INPUT #0,'Where is UNZIP program > ';uz$
140 EW uz$&'unzip';'-d '&op$&' '&ip$
```

ZIPPING FILES

The process of packing files into a single compressed file is called Zipping. Like unzipping, we can put all files from a single place into an archive, or use wildcards or specify a list of files to be included. If an archive already exists, files are added automatically to it, otherwise a new zip file is created from scratch.

```
EW FLP1_ZIP
```

entered by itself, this command displays Zip's built in help screen, see figure 5.



```
 (c) Info-ZIP Group                      zip                            2.2
Type 'zip -L' for the software License.
Zip 2.2 (November 3rd 1997). Usage:
zip [-options] [-b path] [-t mmddyyyy] [-n suffixes] [zipfile list] [-xi list]
  The default action is to add or replace zipfile entries from list, which
  can include the special name - to compress standard input.
  If zipfile and list are omitted, zip compresses stdin to stdout.
  -f   freshen: only changed files  -u   update: only changed or new files
  -d   delete entries in zipfile    -m   move into zipfile (delete files)
  -R   recurse into directories     -j   junk (don't record) directory names
  -0   store only                   -l   convert LF to CR LF (-ll CR LF to LF)
  -1   compress faster              -9   compress better
  -q   quiet operation              -v   verbose operation/print version info
  -c   add one-line comments        -z   add zipfile comment
  -@   read names from stdin        -o   make zipfile as old as latest entry
  -x   exclude the following names  -i   include only the following names
  -F   fix zipfile (-FF try harder) -D   do not add directory entries
  -A   adjust self-extracting exe   -J   junk zipfile prefix (unzipsfx)
  -T   test zipfile integrity       -X   eXclude eXtra file attributes
  -e   encrypt                      -n   don't compress these suffixes
Press a key to exit
```

Figure 5 - Zip help screen (zip.gif)

To pack files into an archive, use a command like this:

EW FLP1_ZIP;'FLP1_EXAMPLE_ZIP FLP2_*'

This puts all files it can find from FLP2_ into the zip file called FLP1_EXAMPLE_ZIP. If FLP1_ already existed, it added all the files to it, otherwise it created FLP1_EXAMPLE_ZIP and put zipped copies of the files on FLP2_ into it. Note - it does not delete the original files on flp2_, it puts compressed copies of them into the zip file (although there is a -m option to MOVE files into an archive instead!).

You can replace the FLP2_* wildcard with a list if you prefer:

EW FLP1_ZIP;'FLP1_EXAMPLE_ZIP FLP2_BOOT FLP2_MYPROG_BAS'

As with Unzip, you can use -x commands to specifically exclude some files if required.

If you need to, you can delete specified files from an archive, e.g. when you accidentally inserted a file you did not wish to be included:

EX FLP1_ZIP;'FLP1_EXAMPLE_ZIP -d myprog2_bas'

That command deleted myprog2_bas from the archive called FLP1_EXAMPLE_ZIP.

You can include a comment into an archive by using a -z command. This comment is just that - a comment among the list of files, useful for including short copyright notices, for example:

EX FLP1_ZIP;'FLP1_EXAMPLE_ZIP -z'

The -z command causes the zip program to ask you for a comment. It can be more than one line long. Enter each line and press ENTER. Simply enter a full stop on the last line to end the comment. So, using -z you could add a copyright notice including your name and address, ending the comment with a full stop on the last line (that line is not part of the comment):

The zip program prompts with the line:

```
enter new zip file comment (end with .):
```

You would then enter the comment:

```
This program is copyright Fred Bloggs 2006
123 The Street
Anytown
AB12 3CD
.
```

If the file already contains a comment, it shows you what it was so that you know what you are replacing. To see and test the comment you have just added, use a simple unzip listing:

```
EX FLP1_UNZIP;'-l FLP1_EXAMPLE_ZIP'
```

The comment appears before the list of files.

If you are in the habit of using sub-directories, a useful option is a -R command, which causes Zip to recurse into sub-directories, which in simple terms means include files contained in sub-directories as well as the current directory:

```
EX FLP1_ZIP;'FLP1_EXAMPLE_ZIP -R RAM4_*'
```

That command tries to add all files in RAM4_ and all sub-directory contents on that drive into FLP1_EXAMPLE_ZIP. Be careful, it is possible to include a huge number of files by mistake. For example,

```
EX FLP1_ZIP;'FLP1_EXAMPLE_ZIP -R WIN1_*'
```

would try to add my entire hard disk content into a zip file on FLP1_, some hope!

Zip normally needs to make some temporary files while it is working. By default it will create these on RAM1_ but you can make it create them on another drive with the -b option:

```
EX FLP1_ZIP;'-b WIN1_ FLP1_EXAMPLE_ZIP RAM2_*'
```

That command created temporary files on WIN1_ while it tried to add all files from RAM2_ into FLP1_EXAMPLE_ZIP

The commands -1 through to -9 offer a trade off between compression efficiency and speed. -9 maximises compression at the expense of speed (although on modern fast systems it might not make much difference to speed!).

There are all sorts of other options, but I won't over complicate matters here. At first, you'll just be using Zip and Unzip to compress and decompress all files on given drives to or from given zip files. Once you are confident with the simplest use, you can start to explore the more advanced options.

CREATING SELF-EXTRACTING ZIP FILES

There is a program called makesfx which adds some code to the start of a zip file to make it into a self-extracting file. Unfortunately, it is not particularly easy to use or very reliable. If you are still interested, read the QDOS-specific part of the Zip documentation.

In the next part of this article, we'll take a look at Archivers Control Panel and a few other front end systems for Zip and Unzip.