FOR THE ARCHIMEDES

Computer Concepts

SPELL MASTER

For The Archimedes

First Archimedes Edition 1988.

Registration and technical support

A registration card is supplied with every legitimate Spell-Master package. Please complete the card and return it (post free within the U.K) as soon as possible. The card holds a unique serial number, which is duplicated on the inside front cover of this manual. Technical support is only available to owners of legitimate packages when quoting their name and serial number.

Spell-Master is supplied as a ROM chip mounted on a small carrier board. Please note that the chip and carrier unit must be treated as a single unit and never separated. Note also that six of the steel pins are deliberately removed from the carrier board during manufacture; this should not be mistakenly identified as damage.

Contents

1. Installation

The Spell-Master ROM

Spell-Master is based around an in-built dictionary (word list) containing approximately 60,000 words. The complete program and dictionary are supplied in ROM form, so that less of the computer's main memory need be occupied by a huge list of words.

In ROM form, Spell-Master is always available in the machine, requiring only one simple command in order to activate it. This avoids the need to have Spell-Master on disc with every program that requires it.

Installing Spell-Master

Installing the Spell-Master ROM is a simple matter, once a suitable backplane and ROM Podule are fitted inside the computer.

Follow the instructions supplied with the appropriate ROM Podule for fitting a standard ROM. Although Spell-Master is actually a 128K EPROM, the integral carrier board makes it appear to the ROM Podule as a simple 16K device. Therefore, when Spell-Master has been installed it must be configured as a 16K EPROM. On the Computer Concepts ROM/RAM Podule this can be achieved with one command:-

***ROMPOD <podule Nº> <Socket Nº> 27128**

This is described in detail in the ROM/RAM Podule manual. Once Spell-Master is installed and configured, cataloguing the ROM Filing system should list one new file named simply "Spell".

Initialising Spell-Master

It is necessary to initialise Spell-Master whenever the computer is turned on (and after Ctrl-Break), before it may be used in any way. Issuing Spell-Master commands without prior initialisation will simply give "Bad Command" or similar errors.

Spell-Master requires an amount of memory to be allocated during initialisation. However, due to the somewhat archaic memory management of the Archimedes, this may require prior configuration of the RMA size.

Archimedes memory management

Whilst not essential, background knowledge of the way in which the Archimedes handles its memory is advantageous.

There are two main areas of memory in the Archimedes, the Application area and the RMA (Relocatable Module Area). All programs designed afresh for the Archimedes should be in the form of an RM (Relocatable Module). All RMs are capable of being positioned anywhere in memory and occupy only as much memory as they need. There can be as many RM programs and utilities in memory simultaneously as will fit.

A program which uses the application area is more of a problem. It severely restricts the use of available memory for other programs whenever it is present. It claims one huge amount of memory and refuses to relinquish any for other programs, even when it doesn't need it. Neither is it possible to have more than one program simultaneously using the application area. Programs which use the application area are generally undesirable for these reasons.

If it were possible to completely avoid programs which use the application area, the memory management and therefore initialising Spell-Master, would be simple. However, BASIC uses the application area as described, so it is frequently the case that memory is difficult to allocate.

There is a large area of memory available for programs and data. On a machine with 1M Byte of RAM (310 or 410) the available memory will usually be about 600K (the remainder being used for low-level buffers, the screen display, system modules, etc.). One special value stored in CMOS RAM even when the machine is switched off is used to determine how much of this area is to be allocated as the RMA.

Assuming that this value is configured very small, the memory arrangement would be as in Fig.1.1 when the machine starts in BASIC.

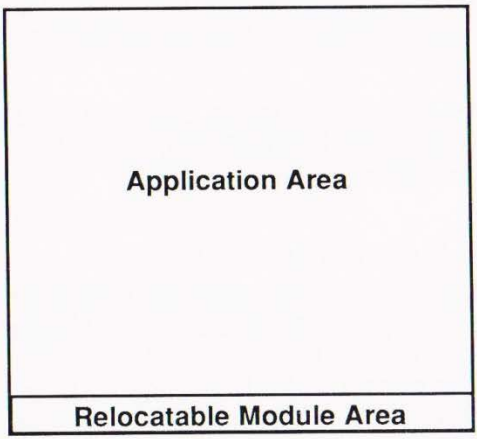


Fig. 1.1 Memory arrangement with small RMA

In this state, an attempt from BASIC to load an RM which is larger than the configured RMA size will cause a "No room in RMA" error. The only way around this is to enter a command to increase the configured RMA size, e.g. *CONFIGURE RMASIZE 10.

The number given in the RMASIZE configuration specifies the number of 'blocks' of memory to reserve for the RMA. In an Archimedes model 305, 310 and 410 a memory block is 8K Bytes, whilst it is 32K Bytes on a model 440. Thus, *CONFIGURE RMASIZE 10 actually reserves 80K Bytes on a 305, 310 and 410, but 320K Bytes on a 440.

Whatever memory is allocated as RMA, it is inaccessible to BASIC. Some BASIC programs require a relatively large amount of memory and cannot work with even an 80K Byte RMA size. In such a case, there is no choice but to re-configure the RMASIZE and pay the penalty (i.e. Spell-Master may not be able to run).

To make matters worse, any change to the RMA size will only take effect when the machine is completely reset, using Ctrl-Break. This in itself will mean that Spell-Master and all other RM programs will have to be re-initialised, but any data they currently have in memory will be lost - so each will have to be saved before Ctrl-Break.

On an Archimedes 305 this situation can be quite impossible to live with. It is usually acceptable on a 310 and 410 machine, and is very unlikely to be a problem on a model 440.

In order for Spell-Master to be initialised (when a program such as BASIC or the 6502 emulator is running) it is necessary to have an RMA size of at least 5 (or 2 on a 440), with nothing else using it; i.e. this is the size required just to switch on the computer and initialise Spell-Master. However, the 6502 emulator, INTER-WORD, and Spell-Master and Spell-Master extension dictionaries all use RMA space. In order to use Spell-Master effectively in INTER-WORD, the RMA size will have to be configured to approximately 20. (Note: each extension dictionary occupies 16K Bytes of RMA, therefore a minimum of four 8K blocks is recommended for this.)

In summary, it is advisable to configure the RMA size to at least 20 (5 on a 440). If this causes a problem for other programs, try decreasing the amount; if this still causes a problem, it will be necessary to re-configure back and forth. Obviously this is unsatisfactory, but it is the best compromise available, given the design of Archimedes memory management.

***CONFIGURE RMASIZE 20**

...then press Ctrl-Break, or switch the machine off and back on again.

The initialisation command

The file named "Spell" present on the ROM Podule is a transient utility program which initialises Spell-Master. The exact command used to execute this program depends upon the type of podule in use. With a Computer Concepts ROM/RAM podule use the command:-

***RUN RFS:SPELL**

Note: If an error "No room in RMA" occurs, it will be necessary to increase the configured size of the RMA (see above), then try again.

On an Acorn ROM Podule, the command would be:-

***RUN ROM:Spell**

The reason for needing different commands is that the filing systems have different names - RFS and ROM.

Users of the Computer Concepts ROM/RAM Podule with at least one battery-backed RAM chip fitted may prefer to add the initialisation command to their !BOOT startup file. Since this is executed whilst RFS is the current filing system, the command may be shortened to just *SPELL. Even if this is not added to the !BOOT file, assuming that the Run$Path system variable has been set to include RFS: (as described in the ROM/RAM Podule manual), the command may be abbreviated to just *SPELL in almost all instances.

The 6502 Emulator (65Arthur)

Please note that in order to initialise Spell-Master whilst already in the 6502 emulator (65Arthur), it is necessary to issue the *SPELL command in full, e.g. *RUN RFS:SPELL.

2. Introduction

Spell-Master is a completely general-purpose spelling checker. It offers a range of star commands which can be entered by the user from almost any program. More important still, it contains a low-level interface available for interaction with any program. This means that you will probably be using Spell-Master in a way decided by another program, using the low-level interface to make Spell-Master do the hard work of checking words and dealing with dictionary extensions.

This manual documents all of the standard features of Spell-Master, namely the initialisation (dealt with in the previous chapter), dictionary extensions and all of the general purpose star commands.

Each application which interfaces to Spell-Master will provide its own specific documentation as an accompaniment to this manual. The exception to this is the documentation for the INTER-WORD word processor which was completed prior to this manual.

Pitfalls of a spelling checker

A spelling checker is a program which checks text against a list of correct words and indicates any words in the text which are not recognised. This does not necessarily indicate that the word is spelt incorrectly, merely that the spelling checker does not recognise it. Consequently if vital words are missing from the spelling checker's 'dictionary' then they will be a constant source of irritation. Most spelling checkers will therefore allow the user to add words in some way, as does Spell-Master.

Because a spelling checker only compares each word against its dictionary, it will not be able to spot mistakes in which a word is spelt correctly but used in the wrong context. For example, the sentence:

"The whether is sunny today."

uses "whether" when the correct word should be "weather". This is

a common mistake in written English, but because the word used does exist, a spelling checker will not spot the mistake.

A similar problem can arise from a genuine typing error, for example:

"I didn't eat my super last night."

in which a letter 'p' has been accidentally omitted from the intended word "supper". Once again, because the resulting word "super" does exist, a spelling checker would not spot the mistake.

Describing the pitfalls of a spelling checker may seem a peculiar way to introduce Spell-Master, but it is important to be aware of the limitations in order to appreciate the benefits. Of course the majority of mistakes are caused by pressing the wrong key, rather than an inability to spell correctly. This sort of error is regularly picked up by Spell-Master. Transposition is the most common typing error, e.g. typing "exmaple" instead of "example". Spell-Master is very good at spotting such mistakes.

The advantages of Spell-Master

Spelling checkers usually suffer from one, or sometimes all of these major problems:-

- **Limited dictionary size**
- **Slow checking speed**
- **Inconvenient operation**
- **Large memory requirement**

Spell-Master is designed to overcome all of these problems. Addressing each of these points in turn will provide a good overview of its facilities.

Dictionary size

The on-board dictionary contains almost 60,000 words, treating each possible ending as a different word. As a rough guide, a base dictionary of about 10,000 words would be adequate for most people, but it would need to be increased to hold words specific to their particular interests.

Spell-Master's dictionary should not therefore show any glaring omissions. One problem is that different professions and even hobbies include words of little use to outsiders. Computer programmers use words like EPROM, FIRMWARE, PIXELS, etc. which would probably be quite useless to 99% of the population. However it is vital to be able to add such words to the dictionary for future use in order to prevent them from continually being indicated as possible errors.

A disc-based dictionary is very slow; too slow to be used by many of Spell-Master's facilities. In order to truly extend the dictionary it is necessary to store the new words in main memory. Spell-Master allows up to nine extension dictionaries - or user dictionaries as we call them - to be held in memory simultaneously. One of these may be a special 'Ignore dictionary', containing words which are to be ignored during checking, but which are not to be added to the main vocabulary. For example, the name of the person to whom a letter is being written is likely to be missing from the dictionary, but neither should it be added for later use. Instead it is added to the special ignore dictionary, which can be discarded when no longer needed, so that the name is not continually indicated as an error.

When a word is added to a user dictionary, it is added to that dictionary in main memory. At the end of the session, the user dictionary should be saved in a file on disc. This file can later be loaded back into memory for use on another occasion.

One user dictionary occupies 16K Bytes of main memory when it is present. This is sufficient space for approximately 3,000 words. With a capacity of up to eight user dictionaries (plus an ignore dictionary) it is possible to add approximately 24,000 words of your

own! However, it is far more likely that just one dictionary will be sufficient with, perhaps, others containing specialist vocabularies for use on specific occasions. Of course, it is possible to build even more than seven user dictionaries, saving each on disc. The only limitation is on the number present in the machine at any time.

With a Computer Concepts ROM/RAM Podule, user dictionaries can be saved in on-board RAM at the end of a session, then automatically re-loaded as part of the start-up routine whenever the machine is switched on. This makes the operation of user dictionaries completely transparent.

Checking speed

There are two commonly used measures of the performance of a spelling checker. These are the maximum number of words in the dictionary and the number of words per minute (WPM) at which it can check. The latter, as is often the case with comparative measures, is not a useful measure unless it is seen in context.

When it is necessary to check the spelling in a document, other spelling checkers will usually require that the text is stored on disc, the spelling checker is loaded, the text is checked and the errors marked in a new file, then the text has to be loaded back into the word processor for the user to make corrections. Obviously, the speed of actually checking the text is only part of the time (and inconvenience) to be considered.

Spell-Master works efficiently whichever way it is examined. In terms of plain checking speed, it averages up to 10,000 WPM. However, it is unusual for checking to proceed for more than a few seconds without an error being encountered. This means that if it worked at twice the speed, there would be very little difference to the time between each interactive prompt.

Convenience of use

Because Spell-Master is a totally general purpose spelling checker, the exact way in which its facilities are implemented in a particular

word processor is not governed by Spell-Master itself. The facilities offered are widespread and allow a whole range of checking options to be easily implemented by any program.

The simplest method for checking is 'continuous checking'. When turned on, it monitors each word being typed (without slowing down the typing rate) and simply 'bleeps' if an unrecognised word is entered. The user may ignore the warning if the word typed is acceptable, correct the error if the mistake is obvious, ask Spell-Master for the nearest match for the word, or browse through the dictionary, etc. Obviously immediate checking is only useful as text is being typed or edited and it is of no use at all when it comes to correcting previously written work.

It is a simple matter for a word processor to use Spell-Master to check text in-memory, rather than having to first store the document on disc. For example, when using Spell-Master with INTER-WORD, all of the current text can be checked without leaving the program. Once the check is initiated, Spell-Master moves to each unrecognised word in the text in turn, allowing any one of five actions: ignore the word (which adds it to the ignore dictionary, so that all subsequent occurrences are automatically ignored), correct the word by manually editing it, add it to a user dictionary (so that subsequent occurrences pass unnoticed), guess the correct spelling, or browse through the dictionary to find the correct word.

Finally, for dealing with text files created by a program which does not directly interface with Spell-Master, a facility is provided for checking a plain text file on disc.

Memory requirement

Spell-Master holds its main 60,000 word dictionary in ROM. This avoids the alternative of loading a large dictionary into main memory, occupying vital space which could otherwise be used for text etc.

3. Star commands

This section deals with the general purpose star commands available in Spell-Master. These can be entered at any star command prompt (*), or preceded by an asterisk as a statement on its own line in most programming languages, including BASIC.

*HELP

A help message is available for every Spell-Master star command, obtained by issuing the command *HELP followed by the name of the command. In addition, a single help topic is available for listing a summary of Spell-Master star commands. Entering the command:-

***HELP SpellMaster**

which can usually be abbreviated to:-

***HELP Spell.**

will list the following summary:-

==> Help on keyword SpellMaster

SpellMaster 1.00 (12 May 1988)

Commands available:

AddWord <word> [<dictionary name>]
Anagram <string>
Browse [<word>]
Check <word>
CheckFile <source file> <destination file>

DCreate <dictionary name>

DeleteWord <word> [<dictionary name>]

DList [<dictionary name>]

DLoad <dictionary name>

DRemove [<dictionary name>]

DSave <dictionary name>

FileToUser <filename> <dictionary name>

Fuzzy <word>

Typo <word>

UserToFile <filename> <dictionary name>

Optional parameters

Each of the command names in the help list is followed by parameters. Each individual parameter is enclosed within angle brackets < >. Parameters which are optional are further enclosed by square brackets [ ]. Thus, parameters enclosed in square brackets need not be specified, but can be specified in order to be explicit where required. Each parameter must be separated from the command name and from each other (when there is more than one) by a space.

Command syntax

The order of parameters, number of them and their separation is known as the command syntax. If a command is entered with incorrect syntax (e.g. too many parameters, too few parameters, no space between parameters, etc.) then a "Syntax" error will be generated. A syntax error in a star command causes the correct syntax to be shown. For example, entering *Check with *no* parameter would produce the error message:-

Syntax: Check <word>

Unfortunately, the user is left to work out exactly what the mistake is, since there are several ways in which the same syntax error can be caused.

Upper and lower case significance

Although commands are listed in a mixture of upper and lower case letters for clarity, no matching of upper and lower case takes place when a command is used. Therefore, the command listed as CheckFile could be entered as *CHECKFILE or as checkfile or as cHeCkFiLe etc.

Command name abbreviations

As with all star commands, these can be abbreviated to the minimum number of characters necessary to distinguish them from all other star commands. An abbreviated command must be followed immediately by a full stop. Thus, *CheckFile could usually be entered as:-

*CheckF.

However, it is recommended that the complete name is used in programs, because a new star command is always liable to be introduced at a later date with a similar name. For example, if the command name *CheckForSomething were added by another program, *CheckF. would no longer be sufficient.

Detailed command descriptions

There are two main categories into which Spell-Master commands can be divided: User dictionary related and general purpose. The commands related to user dictionaries require some background information and are described fully in the next chapter; a brief description of each is given here for completeness. The *HELP listing given above was in alphabetical order; the list is now repeated to show the division of the two categories:-

User dictionary related:

AddWord <word> [<dictionary name>]

DCreate <dictionary name>

DeleteWord <word> [<dictionary name>]

DList [<dictionary name>]

DLoad <dictionary name>

DRemove [<dictionary name>]

DSave <dictionary name>

FileToUser <filename> <dictionary name>

UserToFile <filename> <dictionary name>

General Purpose:

Anagram <string>

Browse [<word>]

Check <word>

CheckFile <source file> <destination file>

Fuzzy <word>

Typo <word>

All of the general purpose star commands will not only look at the main dictionary, but also any user dictionaries that are installed. Many of the star commands would not be used in the normal course of word processing, since they are equivalent to facilities which the word processor would offer directly (by using the low-level Spell-Master interface).

◗ AddWord <word> [<dictionary name>]

Adds a specified word to the current default user dictionary, or to a specific user dictionary if its name is specified.

◗ DCreate <dictionary name>

Creates a new user dictionary.

◗ DeleteWord <word> [<dictionary name>]

Deletes a word from the current default user dictionary, or from a specific user dictionary if its name is specified.

◗ DList [<dictionary name>]

With no parameter, this command lists all the names of the user dictionaries currently present in memory. The current default dictionary (to which words will be added unless a different dictionary is explicitly specified) is indicated. If a dictionary name is specified following *DLIST, the contents of that dictionary are listed on-screen.

◗ DLoad <dictionary name>

Loads a previously saved user dictionary, from a filing system, and places it in memory.

◗ DRemove [<dictionary name>]

Deletes a user dictionary from memory.

◗ DSave <dictionary name>

Saves a specified user dictionary from memory onto a filing system, e.g. onto disc. This user dictionary may subsequently be re-loaded using the *DLOAD command.

◗ FileToUser <filename> <dictionary name>

Converts a plain text file containing a list of words, and creates a user dictionary in memory. This may then be saved as a special format user dictionary with the standard *DSAVE command.

◗ UserToFile <filename> <dictionary name>

Converts an in-memory user dictionary to a plain text file. This may then be loaded into a word processor or dealt with by some other program. It may not be loaded with the *DLOAD command, but can instead be converted back with the *FileToUser command.

General Purpose star commands

◗ Anagram <string>

This expects a collection of characters to follow the command. It will then search through the entire ROM dictionary and user dictionaries (if any) and list all words that contain precisely these characters.

Example

***ANAGRAM AETL**

will give the result:-

LATE
TALE
TEAL

Unfortunately, anagrams in real life puzzles are often a combination of several words. Spell-Master cannot decode multiple word anagrams, since the number of possible combinations is too large and would take far too long.

◗ Browse [<word>]

This allows direct access to the Spell-Master browse window from BASIC or other applications. The <string> parameter is used to specify the start word in the dictionary. Once issued, a window will appear in the top left corner of the current screen, listing a group of words starting from the specified word. If no start point is specified, it starts at the first word in the dictionary. The list is always in alphabetical order (except that endings of the same word are not necessarily in order).

Usually the browse window will be used to look for a certain word, knowing approximately how it is spelt. This is unlikely to be as fast as using one of the commands such as *FUZZY or *TYPO, but requires less knowledge of the correct spelling.

The up and down cursor keys are used to move backwards and forwards through the dictionary. However, with about 60,000 words in the dictionary, it would take a very long time to reach the desired word if these were the only movement controls.

The up and down cursor keys in conjunction with Shift will move backwards and forwards to the first word which starts with the next or previous letter of the alphabet. This means that from a word starting with 'K', Shift down would move to the first word starting with 'L', whilst Shift up would move to the first word starting with 'J'. More succinctly we could describe this as moving to the start of the next or previous 1st letter category. Pressing this key combination 26 times (once for each first letter category) will cycle round to the same position.

For moving in smaller steps through the dictionary, Ctrl in conjunction with cursor up or cursor down will move to the start of the next or previous 2nd letter category. Thus, starting at a word beginning with the letters 'AB', Ctrl down would move to the first word beginning with 'AC', whereas Ctrl up would move backwards to the first word beginning with 'AA'.

At any time after scrolling through the dictionary, it is simple to move directly to any point without having to scroll. Pressing one letter will move to the first word which starts with that letter. Pressing another letter will move to the first word starting with the two letters entered, and so on. Scrolling in any way cancels the letters typed so far, and allows a new sequence to be entered. For example, pressing the letters 'COMPUT' will move to the first word starting with "COMPUT", which is COMPUTABLE. Pressing 'E' as the next letter moves to the first word beginning with 'COMPUTE', which, not surprisingly, is the word COMPUTE itself. The list in the browse window at this point will show:-

COMPUTE
COMPUTED
COMPUTES
COMPUTING
COMPUTER
COMPUTERS
COMPUTERISED
COMPUTERIZED

Pressing the delete key moves back to the position prior to the latest key press, effectively removing the most recent letter and allowing a new one to be entered.

The words in the browse window come from the main dictionary, plus any user dictionaries which are currently in the memory. Note that words from the Ignore dictionary are *not* present.

The browse window is accessible via the low-level Spell-Master interface so that it can be offered as an option during checking in a word processor. INTER-WORD does just this. In such a case, pressing the Return key is used to select the current word for entry into the text. At an even lower level, the facilities used by the browse command itself are accessible, permitting other programs to produce their own version of the browse window, perhaps operating in a different (but similar) manner.

◗ Check <word>

This command is used for listing words or groups of words which match a string containing wildcards. The <string> following the command can contain any known characters in their correct positions, mixed with one or more # wildcards (each representing a single character) and optionally a * wildcard (representing any series of characters).

Example

CHECK COMPUTE

would list all words beginning with COMPUTE, thereby showing all endings present in the dictionary:-

COMPUTE
COMPUTED
COMPUTES
COMPUTER
COMPUTERS
COMPUTERISED
COMPUTERIZED

The * wildcard can be used to represent characters within a word, as well as at the end. This could be used to find a correct spelling when only a few letters at the start and end are known, but the number of missing letters is not known.

Example

***CHECK ACCE*RATE**

will show the result ACCELERATE. This saves having to know whether ACCELERATE is spelt with one L or two.

Using # wildcards allows the correct spelling for a word to be determined when in doubt about some specific characters.

Examples

***CHECK COMPUT#R**

would list the correct spelling of COMPUTER.

CHECK FR#A

would list all words that start with **FR** and have **A** as their fourth letter. Any number of letters could follow the **A**, as represented by the * wildcard.

***CHECK RELEV#NT**

would be the command to enter when in doubt about whether the correct spelling is 'relevent' or 'relevant'.

If no wildcards are used then this will search the dictionary for an exact match, listing the word if present. Therefore this command has two uses; either to list one or more words which match a given wildcarded string, or to check whether a particular word is present in the dictionary.

Example

***CHECK COMPUTOR**

would generate the error "Word not found" since this is not the correct spelling.

See also the *Fuzzy command below for another way of finding words when the exact spelling is not known.

◗ CheckFile <source file> <destination file>

This command will perform a spelling check upon a specified file containing plain text. It is intended primarily for checking files saved in plain ASCII format from a word processor/editor which has not been interfaced directly with Spell-Master.

◗ Fuzzy <word>

This command performs a 'sound-alike' search through the dictionary, listing near matches to the specified word. Usually it is simplest to not attempt to specify the word as accurately as possible, but instead to spell it phonetically. Some examples will help to show the sort of word searches that *Fuzzy is best suited to.

Examples

***FUZZY MUDAL**

lists

MUDDLE
MUTUAL
MUTUALLY

***FUZZY THEASORUS**

lists

THEOCRACIES
THEOCRACY
THESAURUS

◗ Typo <word>

This command is used for correcting slight typing errors, such as transposition of two letters, one missing letter, one extra letter, etc. In fact, this accounts for the majority of typing errors. *Typo is therefore preferable to *Fuzzy. It assumes that just one error is present, since the number of matches found would otherwise be too large.

Examples

1. ***TYPO DICTOINARY**

lists

DICTIONARY

2. ***TYPO EXMAPLE**
lists
EXAMPLE

3. ***TYPO ADDDRESS**
lists
ADDRESS

4. ***TYPO FALIBLE**
lists
FALLIBLE

4.Dictionary extensions

An overview

Dictionary extensions, which we call user dictionaries, are a vital part of Spell-Master. The facility allows the user to add words to one or more separate dictionaries, which Spell-Master uses just as if they were part of the main dictionary.

Dictionary names

Although it would be unusual to have more than two user dictionaries in memory at the same time, Spell-Master can handle up to nine.

In order to identify a user dictionary, and to distinguish one from another, each is assigned a name. Spell-Master prevents two dictionaries with the same name from being present at the same time.

The dictionary name is used as its filename when saved, so that each dictionary name is retained permanently unless the user wishes to deliberately change it by *RENAMEing the file.

Whenever any action is performed upon a user dictionary, it is the dictionary name which is used to specify which user dictionary is to be affected.

The default user dictionary

As soon as Spell-Master is initialised, it creates a system variable containing the name of the user dictionary to which words should be added. The name of the variable is Spell$Dict and its initial value is always set to "UserDict".

When more than one user dictionary is present in memory, the Spell$Dict variable may be used to direct the adding of words to a particular dictionary. For example, if a user dictionary named "Dict2" is created, the command:-

***SET Spell$Dict Dict2**

would cause subsequent words to be added to Dict2. This variable may be changed at any time. It is advised that programs do not override the user's setting of this variable unless instructed by the user to do so. It is intended as a user defined preference, so the user must be left in control.

The command:-

***SHOW Spell$Dict**

may be used to display the current setting of the Spell$Dict variable. Initially this will show the default setting, i.e.:-

Spell$Dict : type String, value : UserDict

The user dictionary specified in the Spell$Dict variable is known as the default dictionary, or the currently selected dictionary. Note that when using the *DLIST command (see below), the currently selected user dictionary will be indicated.

Adding a word

A simple star command, *AddWord, allows the user to specify a word to be added to the 'current' user dictionary. Similarly, a low-level interface call exists in Spell-Master, enabling a word processor to ask for a word to be added; the effect is the same.

When Spell-Master is instructed to add a word, it looks to see if the current user dictionary (defined by Spell$Dict, as above) is in memory and, if not, it automatically creates a new dictionary of that name. However, if the dictionary named in Spell$Dict does *not* currently exist, but another one does, the available dictionary is automatically selected as the default dictionary. The specified word is then added to the user dictionary in memory.

From this point onwards, when Spell-Master is asked to check a word, it examines the main dictionary and each of the user dictionaries in memory. If the word is found in any of the dictionaries then it is passed as correct.

Using an existing dictionary

Although the default dictionaries "Ignore" and "UserDict" can be created automatically by Spell-Master when needed, it is likely that after the initial sessions of using Spell-Master you will want to load a previously created user dictionary and, perhaps, an Ignore dictionary as well.

The actions taken when adding a word to a user dictionary are described above. The necessity for checking to see if the dictionary named as default actually exists - but using another one instead if present, is specifically to make it easy for a pre-loaded dictionary to be used automatically. In this way, a user dictionary can be loaded after initialising Spell-Master and it will automatically be used as the default dictionary when a request is made to add a word.

Saving user dictionaries

When words have been added to a user dictionary in memory, it is important to remember to save it. If a dictionary is not saved then it will be lost when the computer is switched off or when Ctrl-Break is pressed.

User dictionaries are usually saved on disc using the *DSave command. They can be loaded when next required, using the *DLoad command. *DSave uses the dictionary name as the file name, and similarly, the filename used for DLoad becomes the dictionary name.

Converting user dictionaries

A user dictionary is held in a special format whilst in memory. When it is saved with the normal *DSAVE command it retains exactly the same format. However, in this form the words cannot be loaded into a word processor or text editor, so a pair of conversion commands is provided. *UserToFile converts a user dictionary in memory to a plain text form and saves it as a file. This may then be examined and even edited in a word processor. A list of words in a plain text file can be converted into a user dictionary in memory with the reverse command, *FileToUser.

The ignore dictionary

One special variation upon user dictionaries is the ignore dictionary. It is frequently desirable to skip all occurrences of a word, but without having to add it to a user dictionary. Such words need to be ignored during checking, so a special 'ignore dictionary' can be created. This ignore dictionary can be saved and later re-loaded if necessary, though usually it would be destroyed upon completion of each document.

Although no star command exists for specifically adding a word to the user dictionary, it is easily achieved using the normal *AddWord command. Instead of allowing the specified word to be saved in the currently selected user dictionary, an overriding dictionary name can be specified in *ADDWORD. Thus, to add a word "SpellMaster" explicitly to the dictionary named "Ignore", use the command:-

***AddWord SpellMaster Ignore**

The ignore dictionary can only ever be called ignore. Spell-Master specifically checks for the name "Ignore" when a user dictionary is created or loaded by the user and treats that dictionary as the ignore dictionary. If an ignore dictionary is saved with *DSave, *RENAMEd, then re-loaded, it will simply be used as an ordinary user dictionary. This is useful for converting an ignore dictionary to a user dictionary and vice-versa, but equally should be noted as a casual warning.

File Types

The Archimedes provides a mechanism for setting and examining file types. Each program may use a unique type identity, so that it may easily recognise its own files. Spell-Master assigns a type named "UserDict" to user dictionary files saved with *DSave and a type named "Ignore" to ignore dictionaries saved with *DSave.

The DLoad command can only load files previously saved with *DSave. Therefore, *DLoad first checks the file type and, if it is neither "UserDict" nor "Ignore", it issues an error message "Not a user dictionary".

Creating a user dictionary

When it is necessary to explicitly create a new user dictionary, the command *DCreate should be used. This command is followed by the name of the dictionary to be created, which must not be the same as one already in memory. The command:-

***DCreate Dict2**

will attempt to create a new user dictionary named "Dict2". Note that there are several reasons why it may not be possible to create this dictionary, each of which will generate an error message or prompt.

If a user dictionary with the same name already exists in memory then a prompt message "Overwrite dictionary currently in memory ?" will be issued, offering the chance to replace the existing dictionary or cancel the action.

The most likely problem will be that there is insufficient room in the RMA (see explanation in chapter 1). Creating a new user dictionary requires 16K Bytes of RMA and, if this is not available, an error "No room in RMA" will be generated.

Probably the least likely cause for an error is an attempt to create a ninth dictionary, when only eight are permitted.

A full list of Spell-Master error messages is provided in Chapter 6.

5. Inter-Word

Inter-Word is a word processor converted from the original BBC Micro, which runs under the 6502 emulator. On the BBC Micro, it has an in-built interface which operates with the original BBC Micro version of Spell-Master. This section details the way in which the Archimedes version of Inter-Word operates in conjunction with the new Archimedes Spell-Master. Although the operation is fundamentally the same, new options have been added and existing ones extended.

The way in which Inter-Word makes use of the Spell-Master facilities without the user having to issue explicit star commands is recommended as the basis for designers of future word processors.

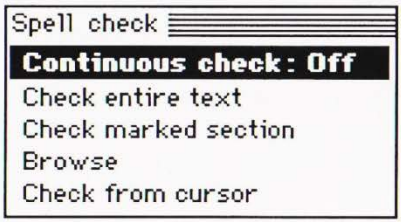
Spell-Master star commands

All of the Spell-Master star commands described in earlier chapters are available for use, should they be required, from the Inter-Word main menu.

The pull down menu

The following assumes that Inter-Word is running, and Spell-Master has already been initialised as described in Chapter 1.

Pressing Ctrl-f8 from within the Inter-Word edit mode will display the 'Spell check' menu. This displays five options which are not listed unless Spell-Master has been installed and initialised.



Any of these options may be selected in the normal INTER-WORD fashion by moving the highlight bar on to the required option with the cursor keys, then pressing Return to select the highlighted option.

Continuous check

The Continuous Check option may be turned on and off by using the right and left arrow keys whilst it is highlighted.

Like most INTER-WORD settings, the 'continuous check; on/off' setting is saved with the document when using menu option 1.

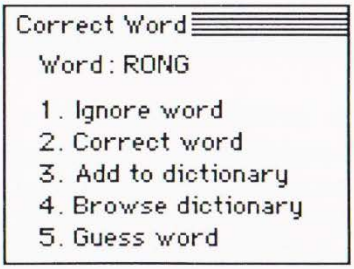
With continuous checking mode turned on, every word is checked as it is typed. Spell-Master monitors all key presses and normally only checks the word as the subsequent space is typed. If the word is not found in the main dictionary or any of the user dictionaries, then the computer will bleep to indicate that it thinks this word is faulty.

SPELL MASTER also checks a word if it is edited. In order to do this, it monitors all key presses and will check the word at the cursor after any alterations are made and an attempt is made to move off this word. Therefore it is quite possible to move to a word already entered, make changes to that word, and then move the cursor away. If the computer bleeps as you move off the word, this indicates the word is unacceptable.

It would be too disruptive for Spell-Master to attempt to correct the word, or force you to correct the word during continuous checking. Therefore this bleep serves as a warning. Normally, the user would make a mental note that it has bleeped, and go back and correct the word at the end of the current sentence, or after the current piece of text has been entered. Often the mistakes are obvious typing errors which can be corrected without further assistance from Spell-Master. If the mistake is not obvious then the best course of action is to move back onto the word and use Ctrl-C to re-check the word. This allows the word to be corrected from the dictionary, or added to a user dictionary. See below under 'Check word at cursor'.

Check entire text

Once selected, this option will start checking all the text currently in memory, starting at the top. Whenever it comes across a word it does not recognise, it moves the cursor to the word in question and displays a window over the text, showing the word in error and offering five options for dealing with it.



Pressing the Escape key will cancel the correction and the spelling check and continue with normal editing.

Pressing the appropriate number will select the corresponding menu option. In fact there is a short-cut to option two; if any alphabetic character is typed at this point it will assume that option two is required.

Each of the correction options is detailed below:-

1. Ignore word

The first option will cause Spell-Master to add the word to the current ignore dictionary. As a result, subsequent occurrences will be ignored during spell checking. Note that if an ignore dictionary does not exist in memory then Spell-Master will attempt to create one. It is possible therefore that an error will occur (such as "No room in RMA"), making it necessary to rectify the situation before re-commencing the check.

2. Correct word

This option will display another small window, this time showing just the incorrect word. A cursor is placed at the end of the word and the user is free to edit the word in the usual manner, using cursor and delete keys, inserting any additional characters necessary. When the word has been corrected, pressing the Return key will replace the erroneous word in the text with the corrected word. In order to avoid having to repeat the process for each similarly misspelt word in the document, Inter-Word automatically performs a search and replace operation from this point onward, replacing further occurrences with the corrected word.

After the word has been corrected, the check restarts immediately prior to the word. As a result, the word already corrected is being re-checked, but it does also mean that if the correction was itself an error, it will still be identified. Furthermore, if the correction created a word which is not in the dictionary, it creates an opportunity for it to be added.

Note that this correction option is particularly useful for splitting two words when a separating space is accidentally omitted. For example, if the space is omitted from between the words "NoSpace" then Spell-Master would be unable to guess the *two* words which should replace the mistake. In this case, the correct word option would be used to insert a space between the words.

3. Add to dictionary

The third option 'Add to dictionary' will add the word in question to a user dictionary, if present. If no dictionary is present in memory then Spell-Master will attempt to create one. It is possible that this will cause an error if there is no room to add the word to the current user dictionary, or if there is insufficient memory available to create another user dictionary.

Once a word has been added in this fashion, it immediately becomes part of the dictionary. This means that, should the word occur again in the text, it will not now be indicated as unrecognised.

4. Browse dictionary

The fourth option makes use of Spell-Master's browse window, described in Chapter 3. Basically this lets the user find the correct word in the dictionary by 'browsing' through any or all of the words in the main dictionary and user dictionaries. Pressing Return whilst in the browse window will cause Inter-Word to replace the misspelt word in the text with the selected word from the browse window. As with the correct word option, Inter-Word then performs a search and replace operation to similarly change any other occurrences of the misspelt word from this point onwards in the document.

Full details of how to control the browse window are given in Chapter 3.

If the correct spelling of the word is known, simply type the letters. Each new letter typed will move through the dictionary, gradually getting closer to the required word. Once all the letters have been entered, the window should be showing the correct word and, as before, pressing Return at this point will put the word into the text in place of the one being corrected.

5. Guess word

This option will attempt to work out what the incorrect word was intended to be, using the *TYPO matching method. Since this relies upon there being only one error in the word, it is likely to be very accurate with character transpositions, one missing letter or one extra letter, but no other sorts of error. Used appropriately, it offers a quick solution.

A window similar to the browse window is displayed, listing one or more guesses (if any) for the word. Use the up and down cursor keys to select the correct word and press Return to replace the misspelt word in the text. As usual, Inter-Word will automatically replace all other occurrences of the same misspelt word. If no correct guess is shown in the window, pressing Escape will return to the five correction options so that some other means of correcting the error can be chosen.

Check marked section

This is similar to the previous option except it will only check text within the currently marked section. Details of how to mark sections of text are given in the Inter-Word manual.

Browse

This option provides access to the Spell-Master browse window, as with the correction option 4 above, and as described fully in Chapter 3.

If a word is selected from the browse window by pressing Return, that word will be inserted into the text at the current cursor position. Alternatively, pressing Escape will exit from the browse window and return to editing in Inter-Word as usual.

Check from cursor

Check from current cursor position. This is useful for checking the remainder of the text when all text above the cursor has already been checked. In all respects except the starting position, this behaves in exactly the same manner as 'Check entire text'.

Control-key access to Spell-Master

In addition to being able to access Spell-Master facilities via the menu described above, a range of Control keys (keys pressed in conjunction with Ctrl) are available. Note that the keys operate only whilst in Inter-Word edit mode, not from the main menu.

Ctrl-V

Check from current cursor position, in exactly the same way as the menu option described above.

Ctrl-C

Check word at cursor. This is directly equivalent to the menu option described above.

Ctrl-B

Browse through the dictionary. This provides access to the standard browse window, as described fully in Chapter 3.

6. Error messages

Spell-Master errors are relatively simple. Unlike errors generated by a programming language, the exact cause of the error is usually quite obvious.

Spell-Master has been allocated its own range of unique error numbers. It is recommended that when a program is intended to act upon a specific Spell-Master error message, it is the unique error number which is considered, rather than the message text. The error number is guaranteed to remain the same between one release of Spell-Master and the next, whilst no such assurance is offered for the message.

The list below gives the error number in hexadecimal, followed by the message text. A brief description of when and / or why the error will be generated is also provided.

&800F00 Word not found

Issued when *Check, *Anagram, *Typo or *Fuzzy are unable to find any suitable matches.

&800F01 Illegal character

Issued when an unacceptable character has been included within a word passed as a command parameter. Attempting to use wildcard characters in any command other than *Check will cause this error.

&800F02 String too short

Issued when a single character has been passed as a command parameter when a string of at least two characters is required.

&800F03 String too long

Issued when a string (or a word) of greater than 63 characters is passed as a parameter to a command.

&800F04 No such category

Issued as a result of using the low-level SWI interface. This error should not occur during normal use of Spell-Master.

&800F05 Word number too big

Issued as a result of using the low-level SWI interface. This error should not occur during normal use of Spell-Master.

&800F06 Dictionary not found

Issued when a dictionary name is specified, but that dictionary is not present in memory. Usually this simply indicates that a dictionary name has been misspelt.

&800F07 Dictionary name too long

Issued when a dictionary name of greater than ten characters in length has been specified in a *DCreate command.

&800F08 Not a user dictionary

Issued if an attempt is made to *DLoad a file which is neither a user dictionary nor an ignore dictionary. The *EX command will list a catalogue of files including their types. Only files with type names "UserDict" and "Ignore" can be loaded with *DLoad.

&800F09 Can't find master dictionary

This error will not occur in the normal use of Spell-Master. It means that there is a very fundamental problem. Attempting to remove the Spell-Master program module outside its ROM environment is one possible cause. If the error does occur

&800F0A Unable to add word

Issued when Spell-Master is requested to add a word (e.g. with *AddWord) but it has found insufficient available space remaining in the currently selected user dictionary.

&800F0B Unable to delete word

Issued when Spell-Master has been requested to delete a word (e.g. with *DeleteWord) but it has been unable to locate the word in the specified user dictionary. This is probably because the word specified was spelt incorrectly, or the word is in the main dictionary from which it may not be removed.

&800F0C Unknown SpellMaster operation

Issued as a result of using the low-level SWI interface. This error should not occur during normal use of Spell-Master.

&800F0D Unknown file type

Issued when a file specified for use with a Spell-Master command is of an unknown type. For example, attempting to convert a BASIC program into a user dictionary with *FileToUser would generate this error.

&800F0E Already eight dictionaries

Issued if an attempt is made to load or create a ninth user dictionary. Spell-Master permits only eight user dictionaries (plus an ignore dictionary) to be present in memory at the same time. It would be necessary to use *DRemove to delete one of the existing user dictionaries from memory in order to load or create another one.

&800F0F *Currently unused*

&800F10 *Currently unused*

&800F11 Escape

Issued if the Escape key is pressed during a Spell-Master operation, e.g. during a *Fuzzy command.

Index

Computer Concepts Ltd.
Gaddesden Place
Hemel Hempstead
Hertfordshire
HP2 6EX

Spell-Master Software by Dave De Vorchik and James Lynn
Manual by Rob Pickering