ESSENTIAL SUPERBASIC TOOLS – LIBERATION SOFTWARE

QLOAD Quick Loader for SuperBASIC programs (v1.9)
QREF Resident Cross Reference Utility (v1.3)

Getting Started

The distribution microdrive/disk contains the following files:

QLOAD_BIN	containing the procedures for QuickLoading SuperBasic programs.
QL_BIN	as QLOAD_BIN, but minus the QSAVE procedure, for use as part of a software package. It may be distributed freely with your own programs to enable users to get the benefit of fast loading, providing you credit Liberation Software in any documentation.
QREF_BIN	containing procedures for producing cross reference lists.
QLOADREF_BIN	containing all the QLOAD and QREF procedures in one file.
BOOT	A bootstrap program which loads QLOADREF_BIN to the resident procedure area. It also contains examples of boot routines for QLOAD_BIN and QREF_BIN.

We recommend tha you make a copy of the files to another medium as soon as possible, and retain the master supplied as a backup, but remember that the QLOAD and QREF software is protected by copyright. You are free to make as many copies as you wish for your own use but you must not give away or sell any part of it without the prior written consent of Liberation Software.

To experiment with the new commands, simply insert the microdrive in mdv1_, reset the QL and press F1 or F2. When the file has successfully loaded a copyright message is printed.

QLOAD

QSAVE, QLOAD and QLRUN are procedures which allow SuperBASIC programs to be loaded in a fraction of the time normally taken. Savings can be dramatic for large programs, eg 55 minutes reduced to 55 seconds! The procedures are all easy to use, being replacements for th efamiliar SAVE, LOAD and LRUN.

To understand how these short loading times can be achieved, consider how LOAD and SAVE work. When a SuperBASIC program is SAVEd, it is stored on microdrive or disk as a pure text file. When loading such a program, the interpreter has to process the program text character by character to create an internal tokenised form suitable for interpretation. It is this process and the associated memory restructuring which it involves, that makes LOADing so slow.

QSAVE short circuits these operations by storing the program in its internal form, with sufficient additional information for QLOAD to recreate it in memory precisely as it was when QSAVEd. Only useful data is stored in a QSAVE file; redundant entries in the name table are ignored.

Note that QSAVE does not produce a simple ‘memory snapshot’. The act of QLOADing a file affects only the SuperBASIC interpreter. Other jobs which may be active in the system continue to run. Furthermore, QSAEd programs are portable across different memory sizes, ROM versions and toolkit variations.

ESSENTIAL SUPERBASIC TOOLS – LIBERATION SOFTWARE

The QSAVE Procedure

Syntax: QSAVE filename

QSAVE stores the SuperBASIC program currently resident in memory in a file on disk or microdrive. The name of the file is a name you specify, with the extension '_sav' appended.

eg QSAVE MDVI_MYPROG produces a file named MDV1_MYPROG_sav.

If the file already exists, then QSAVE will prompt with the message "File already exists - Overwrite Y/N". Entering 'Y' (or 'y') will delete the old file then save the new version. Any other entry will abort the operation.

A QSAVEd tile is normally slightly larger than its equivalent text file. QSAVE takes much the same time to do its job as SAVE.

The QLOAD Procedure

Syntax: QLOAD filename

QLOAD can only load files which have been produced by QSAVE, ie files with the extension '_sav'. As with QSAVE, there is no need to type the extension when entering a file name since QLOAD adds it automatically. (If you do, QLOAD will still work.) With Super Toolkit 2, QLOAD uses the PROG_USE default directory.

For example QLOAD MDV1_MYPROG would load the program saved above.

in contrast to LOAD's stop start style of loading from microdrive, QLOAD loads an entire QSAVEd file into memory in one fast operation. It then proceeds to recreate the program in the form required by the interpreter. This step will take a few seconds for average programs and may be noticeable as a delay between the drive stopping and the cursor reappearing in channel 0. During this time QSAVE also checks that all machine code procedures and functions used by the program are present. If they are not, QLOAD will print the missing names on channel 0.

For example, if you QSAVE a program which uses some assembler extensions, then later try to QLOAD it without those extensions being present. QLOAD will warn you with a message. Subsequently those names will be treated as variables (LOAD does the same). If the program is later QSAVEd without correcting this error, the next time it is QLOADed, there will be no message. QLOAD now thinks the names really are variables.

Once a program has been QLOADed, you can work with it as you wish. QLOAD imposes no restrictions.

The QLRUN Procedure

Syntax: QLRUN filename

QLRUN behaves as QLOAD, but automatically runs the program after loading it.

MERGING

Files can be MERGEd with a program which has been QLOADed, but such files must of course have been produced by SAVE, not QSAVE. There is no QMERGE procedure for two reasons. Firstly, because the performance of MERGE is normally adequate for modest programs and secondly, because its a considerably more complex operation. It may be implemented later if there is a demand.

Using QLOAD with Q_Liberator

QLOAD and Q LIBERATOR were designed to complement each other. Between them they totally transform SuperBASIC programming. QLOAD eases the development of large programs by ensuring short loading times, and Q_LIBERATOR can compile the finished product so that It not only loads faster but runs faster and multltasks. Furthermore. release 3.0 of Q_LIBERATOR, can compile a program in 'background' mode directly from a QSAVEd file, while you continue to work with the interpreter. Refer to the QLIBERATOR instructions for further details.

Minerva Integer Tokenisation

This version (as of v1.8) has been tested with Minerva version 1.80 which includes integer tokenisation, a feature which confused earlier releases. Note that _sav files produced with integere tokenisation switched on can only be compiled with Q_Liberator version 3.32 or higher and cannot be QLOADed into systems with earlier Minerva or Sinclair ROMs. If distributing programs in QLOAD format always LOAD programs with integere tokenisation switched off before QSAVEing the program. This will maintain compatibility with earlier ROMs

QREF

QREF and the related procedures desribed below form a comprehensive, resident cross reference utility for SuperBASIC programs. BASIC programmers will find it indispensible for finding their way round a program during development, while assembler programmers can use QREF to locate the start address of their SuperBASIC extensions In memory.

Given a name to work with, these procedures print details of that name derived from the Interpreter's name table, followed by a cross reference list showing every line number in which a reference to that name appears.

The information printed for each possible type is as follows:

Variable	The variable's type le string, float or integer. If the variable is a FOR or REPeat index, this is shown.
Array	The array type. eg integer array, string array.
SuperBASIC procedure	The line number where it is DEFined.
SuperBASlC function	The function type and line number where it is DEFined.
Machine code procedure	The hexadecimal address in memory where it starts.
Machine code function	The function type and hexadecimal start address.

A machine code routine is either one of the standard SuperBASIC routines in the QL ROM or an extension present in an additional ROM or loaded into the resident procedure area.

ESSENTIAL SUPERBASIC TOOLS – LIBERATION SOFTWARE

The QREF Procedure

Syntax: QREF [#channel,] “name” [,“name” ...]

QREF prints a cross reference for 1 or more names, which must be specified in full. Each name can either be a variable name or a string containing a procedure, function or variable name. QREF normally prints to channel 1, but you can produce cross references on a printer or to a file by including a channel number In the QREF command. QREF automatically adjusts the formatting of its output to suit the device in use. QREF_L was added to QREF in version 1.2

Some examples of QREF:

QREF apples	cross reference of the variable ‘apples’ to #1
QREF “INPUT”,”INKEY$”	cross reference of 2 procedures
QREF #4,x,y,z	cross reference of 3 variables to channel 4

QREF_V, QREF_P, QREF_M, QREF_L & QREF_A

Syntax: QREF_A [#channel,] [wildcard_name]
QREF_L [#channel,] [line_number [, line_number]]

If you need a more global picture than QREF provides, these procedures produce a sorted cross reference list for a whole program or, given a wildcard name, for a subset of the names In a program. The list is printed In ascending alphabetic order. The procedures share a common syntax, and differ only In the type of name which they select for cross reference.

QREF_V produces a cross reference of all used Variables.

QREF_P produces a cross reference of all SuperBASIC Procedures and functions In the current program.

QREF_M produces a cross reference of those Machine code routines which are actually used in the current program.

QREF_A produces a cross reference of ALL name table types. Entries are printed regardless of whether they are actually referenced in the current program. To avoid listing Irrelevant entries, it is useful to type CLEAR prior to using QREF_A.

QREF_L As of version 1.2 (April 1991), QREF has been enhanced to give a line number cross reference facility. If one line number parameter is present then all references to this line number will be output. If two line numbers are given then all references to this range of line numbers will be output. If no line number parameters are given, QREF_L will cross reference the whole program.

If a wildcard name Is present, then only names whose leading characters match the wiidcard name will be printed. Case is not important. If no name or a null string is passed, all names will match. Note that with large programs these procedures might need a few seconds ‘thinking time’ before printing starts. They can be paused by F5 and interrupted by CTRL Space.

Some examples:

QREF_P	Cross reference all procedures
QREF_V “old”	Cross reference all variables beginning with “old”
QREF_M “q”	Print all machine code extensions beginning with “Q”

OPEN #3.ser1
QREF_A #3 Cross reference entire ROM to printer

QREF_L 100,200 Cross reference all lines between 100 and 200

The QFIND Function

Syntax: QFIND [(name)]

This function is closely related to QREF. It makes it easy to edit all occurences of a given name in a program. The first time QFIND is called, It should be given a name to search for. If this name is a SuperBASIC procedure or function then QFIND will return the line number on which it is defined. If the name refers to a machine code routine then the start address is returned (In decimal). If the name is a variable then QFIND returns the line number on which it is first referenced.

Subsequent calls to QFIND without a parameter will return the next line number from the cross reference list. Note That QFIND will only return a line number once even when there are multiple occurences of a name within a line.

Consider this small example program.

```
10 Square 1
20 Square 2
30 DEFIne PROCedure Square(s)
40   PRINT aa
50 END DEFIne
```

Typing EDIT QFIND("square") would put you Into the SuperBASIC editor at lIne 30 the line on which 'square' is defined. Now typing EDIT QFIND would edit line 10, the first occurence of 'square' in the program. Repeating this command would edit line 20. Toolkit 2 users can also use this technique with the full screen editor, ED.

QREF and SMSQ/E

QREF does not recognise the new binary and hexadecimal constants in SBASIC, ie values preceded by % for binary and $ for hexadecimal.

So, a patch was made available with SMSQ/E to alter QREF (and QLOADREF), included here.

Within the archive there are three files preceded by 'smsqe_':

Smsqe_QREF_TXT Some notes on the patch
Smsqe_QREF_BAS The patch program, written in SuperBASIC
Smsqe_QREF_BIN A (notionally) pre-patched version.

If this pre-patched version doesn't work on your system, make a **copy** of the main QREF_BIN or QLOADREF_BIN and run the smsqe_QREF_bas to patch it on your system.

In theory, you should only need to patch QREF_BIN (not QLOADREF_BIN), as the QLOAD extensions are already built into SBASIC.