SMSQ/E for QPC

Manual revision 2.10

Introduction

This manual was written on the release of QPC2 version 3.00. It's the first version of the manual that doesn't explain QPC1 (the DOS version) anymore, which simplifies matters immensely.

QPC2 is a moving target, therefore the version you have might already incorporate some features that are not mentioned in this manual. Have a look at the supplied version2.txt file for the list of changes.

Up-to-date information regarding QPC (current version numbers, new manual revision etc.) can be obtained first hand from my web site at http://www.kilgus.net/.

Requirements

QPC2 requires at (the very) least a PC with a 486DX CPU, 8MB of RAM, VGA graphics, a sound card (if you want the beeper sound) and any of the 32bit Windows operating systems (i.e. 95, 98, ME, NT4, 2000 or XP). Furthermore, it needs DirectX 3 or later. DirectX is incorporated into service pack 3 (or higher) of NT4 and can be obtained separately for Windows 95 from the Microsoft web site. All other systems come with DirectX installed, but in case of problems, it might be a good idea to check whether a newer version is available for your system.

Note that those are *minimum* requirements; running QPC2 on a 486 is not really recommend. However, PCs not suitable for QPC are dying out very quickly. I currently have a 400Mhz PC (which could be considered "stone-age" by now) and that's quite ok. A decent graphics card is recommended, too, which means anything from about the age of an Nvidia Riva128 ("prehistoric") onwards. Only some laptop chips can still cause problems.

QPC2 Installation

QPC2 does not require any special installation procedure; it's ready to run straight from the install disk (just start "QPC2.EXE"). You can also copy it to wherever you want, just make sure that all three files (QPC2.EXE, SMSQE.BIN and REGISTER.KEY) are in the same directory.

You can also use the installation program supplied, but it doesn't really do more than copying the files and adding an uninstall entry to Windows' "software" facility.

QPC2 Configuration

When running QPC for the first time (and, unless you disable it, also later) you are presented with a configuration dialog from where you can adjust almost all the settings.

The settings are saved within a standard Config Level 2 block within the SMSQE.BIN file; therefore, you can also use SMSQ/E configuration utilities like MenuConfig from inside the emulation. This is especially useful as MenuConfig is capable of updating a new SMSQE.BIN file with your old settings.

The dialogs are supplied in three languages: English, German and French. Depending on the system language, your Windows system will automatically decide which one to show.

Main dialog

The main dialog is divided into 3 sections: screen, emulation and general.

The screen part adjusts anything that affects the QPC window and the SMSQ/E screen emulation. The screen driver option is only useful if you've more than one graphics card installed. In that case you can choose which one QPC2 should run on when in fullscreen mode. The resolutions offered to the left are the ones your graphics driver thinks it can support in fullscreen mode.

Since version 2, QPC2 can run within a window on your desktop instead of just fullscreen. New for version 3 is that you can also chose an initial window size (right below the "window mode" option). This can be used to "zoom into" the SMSQ/E screen. Note that this may not work unless the graphics card/graphics driver you're using supports it.

The "Keep aspect ratio" option makes sure that the X/Y ratio of the QPC window equals the X/Y ratio of the SMSQ/E resolution. This option can also be altered later by using the system menu (left click on the QPC logo in the upper left hand corner, or right click on the QPC button on the task bar). The same applies to the "Always on top" option.

The emulation part adjusts the behaviour of QPC2 towards other applications. You can set the priority QPC2 demands when running in the fore- or background. Don't alter these settings unless you need to.

Furthermore, you can specify the "power saving mode". This mode detects when SMSQ/E is running idle, in which case it case throttles the processor emulation. The option is mainly useful for laptop computers to conserve battery power but it is also recommend on desktop PCs, as it leaves more processor power for other processes.

Finally, in the general section you can specify how much memory is allocated to SMSQ/E. Please note that although the drop-down list only offers some selected values, you can manually enter anything up to 128 (MB!) into the edit box.

The country code sets the default language for the SMSQ/E keyboard layout and error messages. Possible values, based on the international telephone country codes, are:

1	U.S.A.
33	French
44	English
1	German

The AltGr key (for those who don't have one, this is a second Alt-key located to the right of the space bar) setting decides which code should be sent to SMSQ/E when this key is pressed. Recent SMSQ/E versions support the AltGr key natively, therefore you can now choose between Alt, Ctrl and AltGr itself.

At the bottom of the dialog, you can tell QPC to not show the dialog again on the next start-up unless you're holding down the shift key. This is the only option that is saved to the SMSQE.BIN file without hitting the "Save" button. If you made any changes, other than this option, and you want to retain them, always hit the "Save" button before continuing, using "OK".

Attention: the "save" button and the "don't show on next start-up" option are only available if QPC can write to the SMSQE.BIN file. Please make sure it is not set to read-only.

Devices dialog

The "devices" dialog unifies all the settings that somehow affect the storage devices WIN, DOS and FLP.

The WIN device represents the native hard drive format for SMSQ/E. You do not need to partition your PC hard drive to use the WIN device, for the PC it is only one big file on one of the ordinary PC drives. In this dialog you can set where this file is located for each WIN drive (WIN1-WIN8). The default values correspond to the scheme SMSQ for the QXL always uses. Note that if you're not using a QXL card you're not restricted to calling all those files "QXL.WIN", any name is fine.

Next you can alter the settings of the DOS device. The DOS device grants SMSQ/E direct access to all the drives Windows can see. One limitation however is that the maximum file name is still restricted to 36 characters. This means that some deeper directory levels may not be accessible. To rectify this situation every DOS drive has a configurable base directory, so a "DIR dos1_" might actually access "C:\Documents\Important\Long\". You can connect a DOS drive to any directory in the system and even to shares on remote PCs, like for example

C:\WINDOWS
\\REMOTEPC\SOMESHARE

Lastly you can choose on which device SMSQ should look for the boot program on start up. Here only WIN and FLP is available as an option.

Sound dialog

This dialog got introduced in version 3.20. The volume of the beeper and the sampled sound system ("SSSS") can be adjusted here. Also you can set whether sound is enabled, disabled or in automatic mode. See the "beeper" chapter below for more details on this.

SER/PAR dialog

Here, you can adjust the emulation's SER -> COM binding. When enabling the "Leave ports open" option, COM ports, once opened, aren't closed until QPC is exited. This was only included by special request; most people won't need it.

In the parallel section, you can set the destination for each PAR port. For traditional parallel printers the LPT1-LPT3 options might be sufficient, for all others (including USB and network printers) choose the "printer" option and select the printer in the drop-down list. One speciality in the list is the "Default printer" option which always corresponds to Windows's default printer.

The “use filter” option is for future third party enhancements of the printer functionality. It is only available when “printer” is selected and a corresponding plug-in is found.

Command line

QPC has a few options that can already be supplied on startup. Those are:

- -? or -h: Show a short summary of the options available.
- -os file: Tell QPC where to look for the operating system. As the OS also includes the complete QPC configuration, you can have as many differently configured QPCs in one directory as you want.
- -cmdline "Text”: The contents of "Text" can be read using the QPC_CMDLINE$ function. Other than that, it has no effect on QPC.

Some technical explanations

Processor emulation

- As of version 3.33, QPC emulates the complete 68020 instruction set, except the CALLM/RETM instructions.
- All illegal instructions and all addressing modes are trapped.
- The F-line emulator works as usual, the A-line emulator is only used for communication with the PC-hardware.
- Unlike earlier QPC versions, instructions on odd addresses do cause an address fault exception.

Screen emulation

- The screen is not updated periodically. QPC checks any write access for screen operations and acts accordingly. This is not true for instructions that access the stack explicitly (JSR, BSR, PEA...).
- The hardware register at address $18063, is also kept up-to-date by SMSQ/E, but changes to it have no effect.
- As one is likely to be using higher resolutions than the original 512x256 most of the time, the base of screen memory will differ from the original $20000. This might cause problems for old programs (although not as often as before, see also next point). However, if you set the resolution to exactly 512x256, the base of system memory will again be at $20000.
- Version 3 introduces a special compatibility feature: QPC can map the original video memory into the first 512x256 pixels of a higher resolution screen, and even into the 16bit colour mode. See the description of the QPC_QLSCREMU command for details.

Real time clock

The calculation PC date -> SMSQ/E date is quite a time consuming operation, therefore it would be wasteful to do it every second. Instead the date of the last Windows boot process is converted. Windows can supply how many seconds have passed since the last boot, so this value is just added to the calculated “base” and there it is - a valid SMSQ/E date. This mechanism may stop working when Windows is put to sleep in some power saving modes. Therefore, the “base” is re-calculated once per minute, which is the maximum duration of the wrong SMSQ/E time.

Beeper

Since version 3.02, the beeper is no longer emulated through the PC speaker (this is only possible under Win9x/ME anyway) but through the soundcard by using the DirectSound interface. This way all Windows systems can benefit from the emulation. The emulation has been completely rewritten. The sound is now quite similar to the original beep.

From version QPC2 version 3.03 on, sound emulation can be configured to be enabled (default), disabled, or enabled automatically. The reason this item is optional, is that under some Windows versions DirectSound applications can work simultaneously with other DirectSound programs but not with old waveform applications. This means that after starting QPC, some applications can't emit any more sounds, even when QPC is not beeping. Now the emulation can be completely switched off or, if you still want the occasional beep and have other applications work correctly, you can set it to "automatic" mode. In this mode, DirectSound is enabled before every beep and released again immediately. The drawback is that this procedure needs some time (there's some "latency") thus short beeps may simply get lost (i.e. the sound is switched off again before the soundcard has time react).

Furthermore, an option was added to version 3.03 that sets the default volume of the beeper emulation. This is useful because an emulated beep can be much louder than normal sounds.

SMSQ/E Samples Sound System (SSSS)

Since version 3.20 QPC can play sampled audio data. This has been implemented compatibly to the SMSQ/E Sampled Sound System (sometimes also called QLSSS – QL Sampled Sound System), which had been developed for the Q40. In principle every program written to support the SSSS should also work with QPC. I'd especially like to point out Simon Goodwin's SOUND device, it's worth a try.

Fast memory

One problem with earlier SMSQ/E implementations is the so-called "slaving" mechanism. Slaving means that the system uses unoccupied memory to cache accesses to slow disk drives, such as microdrives. On the original QL, hardware slaving was a good and clever idea. However, as memory sizes increased, it became increasingly inefficient to scan through so many slave blocks in search of the required information. With QPC it gets even worse, there access to a virtual hard disk is always faster than the slaving cache, as Windows already caches such accesses. This resulted in a slowing down of disk operations the more memory QPC was set to use.

It is not easy to disable the slaving mechanism, or to limit it to a certain maximum memory size. Therefore, a different approach was chosen based on so-called called "fast memory". Fast memory originates from the Atari systems, which could have a region of memory faster than the ordinary system memory, but which was also unsuitable for slaving (because it couldn't be directly read/written by the disk drives). Furthermore, there was a gap between the slow and fast memory areas.

What was done in the end was to let the whole SMSQ/E memory map (including the "common heap", the area where the common memory allocations take place) remain in the slow memory area, while the fast memory was included with the TPA (transient program area, originally the area where programs executed by "EX" etc. are loaded). If a program now allocates memory, this is served from the TPA first (instead of the common heap as in traditional systems) and the common heap is only used when this becomes full. This means that slaving is limited to the area set aside as slow memory.

With the release of QPC2 v3.03 a very similar approach is used. Essentially QPC just declares the first megabyte of RAM as "slow" memory and the rest as "fast". This way slaving only takes place in a small memory area and is thus speeded up considerably. All other memory allocations are served out of the huge "fast" pool. One immediate effect is that programs that try to determine how much free memory the system has will only return the free space within the "slow" memory, i.e. only about 800 kilobytes although there are some more megabytes waiting in the "fast" area. The FREE_MEM function has already been modified to return the right value, other applications might still have to be adapted (for programmers: sms.frtp returns the value you're looking for, don't try to calculate the free memory yourself).

Furthermore, you might experience some strange behaviour from the Sysmon program, and the QPAC2 "Jobs" window can show "Memory corrupt". These programs have been updated to work correctly again.

Mouse

In version 3, the mouse wheel is supported for the first time. The implementation is transparent to the applications. For every tick of the mouse wheel several Alt + up/down key combinations are stuffed into the keyboard buffer. Since v3.02, the amount depends on the global Windows setting. As Alt + up/down is the standard key-combination for scrolling a window, many applications can immediately profit from the wheel.

Unfortunately, Windows 95 implements the mouse differently to all other Windows versions, therefore wheel mice are not supported under this system. Furthermore, we came across some cheap mice with drivers that don't meet the Windows standards, so they may also not work.

MOUSE_SPEED

Syntax: MOUSE_SPEED [#ch,] acceleration, wakeup

Adjusts the mouse acceleration and wake up factor. From QPC2 version 2 on, the acceleration is no longer used as the mouse position is adapted from Windows. The wakeup values, however, may still be set. They range from 1 to 9, with 1 being the most sensitive.

MOUSE_STUFF

Syntax: MOUSE_STUFF [#ch,] hot$

Adjusts the string that is stuffed into the keyboard queue when the middle mouse button is pressed (or both left and right buttons are pressed simultaneously). The string cannot be longer than two characters, but this is enough to trigger any hotkey, which in turn, can do almost anything.

E.g.: MOUSE_STUFF '.' **Generates a dot if middle mouse button is pressed**
MOUSE_STUFF CHR$(255)&'.' **Generates hotkey Alt + .**

Machine type

There are two standard functions to determine the machine type

MACHINE

Syntax: mach% = MACHINE

Returns the machine type SMSQ/E is running on; 30 for QPC.

PROCESSOR

Syntax: proc% = PROCESSOR

Returns the 680x0 family type; 20 for QPC (10 for versions below 3.33).

E.g.: IF MACHINE=30 AND PROCESSOR<>20:PRINT 'This can hardly be QPC!'

QPC control

QPC2 reacts to two special key combinations:

SHIFT + CTRL + ScrLock: This key combination terminates the emulation. Unlike the QXL, which has its own memory and can resume a session later on, QPC is *really terminated* and the memory is given up. Therefore you should make sure that any data in your SMSQ/E applications is saved, that all files are closed etc. For safety, you might want to wait one or two seconds in case something is still in the drive caches.
In windowed mode, you can also use the standard ways of quitting a Windows application, e.g. the "X" button in the title bar, or go via the system menu.

SHIFT + CTRL + F12: This combination is used to switch between window- and fullscreen mode.

QPC-specific commands

QPC_CMDLINE$

Syntax: cmd$ = QPC_CMDLINE$

Returns the argument that was supplied to QPC after the -cmdline command line argument. Can be used to do different actions depending on the way QPC was started.

QPC_EXEC

Syntax: QPC_EXEC command$[, parameter$]

This command can be used to call an external DOS or Windows program. The name of the executable file is given in the first parameter. Optionally, you can also supply a second parameter, which is then passed to the executed program as its command line arguments.

Furthermore, you can supply a data file as the first parameter. In this case, the associated application for this file type is executed.

E.g.: QPC_EXEC 'notepad','c:\text.txt' **Start notepad and load the c:\text file**
QPC_EXEC 'c:\text.txt' **Start the default viewer for .txt files**

QPC_EXIT

Quits QPC and returns to Windows.

QPC_HOSTOS

Syntax: os% = QPC_HOSTOS

This function returns the host operating system under which QPC was started. Possible return codes are:

0 = DOS (QPC1)
1 = Win9x/ME (QPC2)
2 = WinNT/2000/XP (QPC2)

QPC_MAXIMIZE, QPC_MINIMIZE, QPC_RESTORE

Maximises, minimises or restores the QPC window

QPC_MSPEED

Syntax: QPC_MSPEED x_accel, y_accel

This command is used on QPC1 to change the mouse acceleration. It has no effect on QPC2.

QPC_NETNAME$

Syntax: name$ = QPC_NETNAME$

This function returns the current network name of your PC (the one you supplied upon installation of Windows). This command can be used to distinguish between different PCs (e.g. in a BOOT program).

QPC_QLSCREMU

Syntax: QPC_QLSCREMU value

Enables or disables the original QL screen emulation. When emulating the original screen, all memory write accesses to the area $20000-$207FFF are intercepted and translated into writes to the first 512x256 pixels of the big screen area. If the screen is in high colour mode, additional colour conversion is done.

Possible values are:
-1: automatic mode
0: disabled (default)
4: force to 4-colour mode
8: force to 8-colour mode

When in QL colour mode, the emulation just transfers the written bytes to the larger screen memory, i.e. when the big mode is in 4-colour mode, the original screen area is also treated as 4-colour mode. In high colour mode however, the colour conversion can do both modes. In this case, you can pre-select the emulated mode (parameter = 4 or 8) or let the last issued MODE call decide (automatic mode). Please note that that automatic mode does not work on a per-job basis, so any job that issues a MODE command changes the behaviour globally.

Please also note that this transition is one-way only, i.e. bytes written legally to the first 512x256 pixels are not transferred back to the original QL screen (in the case of a high colour screens this would hardly be possible anyway). Unfortunately, this also means that not all old programs will run perfectly with this type of emulation. If you experience problems, start the misbehaving application in 512x256 mode.

QPC_SYNCSCRAP

In order to rapidly exchange text passages between Windows and SMSQ/E the Syncscrap functionality was introduced. The equivalent of the Windows clipboard is the scrap extension of the menu extensions. After loading the menu extensions you can call this command, which creates a job that periodically checks for changes in either the scrap or the Windows clipboard, and synchronises their contents if necessary. Please note that only text data is supported. The character conversion between the QL character set and the Windows ANSI set is done automatically. The line terminators (LF or LF+CR) are converted too.

QPC_VER$

Syntax: v$ = QPC_VER$

Returns the current QPC version.

E.g. PRINT QPC_VER$ will print 3.00 or higher.

QPC_WINDOWSIZE

Syntax: QPC_WINDOWSIZE x, y

Sets the size of the client area (the part that displays SMSQ/E) of the QPC window. It does NOT alter the resolution SMSQ/E runs with, so the pixels are effectively zoomed. It is equivalent to the "window size" option in the main configuration window. If QPC is currently in fullscreen mode it will switch to windowed mode. Window size cannot be set smaller than the SMSQ/E resolution or bigger than the desktop resolution.

E.g. DISP_SIZE 512,256
QPC_WINDOWSIZE 1024,512 do a 200% zoom of the QPC window

QPC_WINDOWTITLE

Syntax: QPC_WINDOWTITLE title$

Amends the title line which you can see when QPC runs in windowed mode. This can be used to easily distinguish between several QPC instances.

E.g. QPC_WINDOWTITLE "Accounting" sets the title to "Accounting – QPC II [...]"

Serial (COM) Ports

Unlike the basic SMSQ serial port drivers, the SMSQ/E serial port drivers are dynamically buffered. There is therefore no need to use the PRT device.

The Baud rates supported by SMSQ/E on QPC are

115200
57600
38400
19200
9600
4800
2400
1200
600
300

The BAUD command works as in SMSQ/E on the Atari: If the port number is omitted, only SER1 is affected:

BAUD 19200 **set SER1 to 19200**
BAUD 3,115200 **set SER3 to 115200 baud**

SER_GETPORT$

Syntax: com$ = SER_GETPORT$(port%)

Returns the device the SER port is connected to, for example “COM1”.

SER_SETPORT

Syntax: SER_SETPORT port%, com$

Sets the COM port a SER port should be connected with. The change will take effect on the next open of the specified serial port.

SER_SETPORT 4,”COM32” **Associates SER4 with COM32**

Printer support (PAR)

The PAR device can be linked to a printer port or to a Windows printer queue. By default, output is dynamically buffered: the PRT device is not required.

Please note that no translation of the printer data is done, the data sent to the device is directly piped into the printer itself. The only exception is the “filter” option in the configuration dialog. A filter gets the raw data and can process it however it wants. We're currently working on an ESC P/2 emulation filter named QPCPrint. It may already be available as you read this.

PAR_DEFAULTPRINTER$

Syntax: name$ = PAR_DEFAULTPRINTER$

Returns the name of Windows' default printer. This name can later be used with PAR_SETPRINTER for example.

PAR_GETPRINTER$

Syntax: name$ = PAR_GETPRINTER$(port%)

Returns the PAR port setting: Returns "LPT1", "LPT2" or "LPT3" if it isn't linked to a printer but directly to a printer port.

PAR_SETPRINTER

Syntax: PAR_SETPRINTER port%, name$

Connects the PAR port either to a hardware port (e.g. name$ is "LPT1") or to the printer spooler (name$ is one of the names returned by PAR_PRINTERNAME$).

PAR_GETFILTER

Syntax: state% = PAR_GETFILTER(port%)

Returns whether the printer filter is enabled for the specified port.

PAR_SETFILTER

Syntax: PAR_SETFILTER port%, state%

Enables (state% = 1) or disables (state% = 0) the printer filter for the specified port. If the printer should be enabled although none is available a “not found” error is returned.

PAR_PRINTERCOUNT

Syntax: n% = PAR_PRINTERCOUNT

Returns the number of printers available on this system.

PAR_PRINTERNAME$

Syntax: name$ = PAR_PRINTERNAME$(n)

Returns the name of printer number *n* (counted from 1 to PAR_PRINTERCOUNT).

TCP/IP

From version 3.30 on QPC possesses TCP/IP capabilities in form of the TCP, UDP and SCK devices. For this it relies on the Windows TCP/IP stack, so if applications like the Internet Explorer can access the net QPC can do it, too.

This manual won't go into the details of the interface, however, as the interface is mostly compatible to the uQLx implementation all documentation for that is also valid for QPC.

Attention WIN95 users: QPC2 now needs a Winsock2 capable system to work. If you don't have it already please download and install the Winsock2 update for Win95 from the Microsoft homepage.

PC Floppy Disks

SMSQ/E accesses the PC floppy disks via low-level Windows calls. Read accesses are buffered internally by QPC and should be quite fast. This is not true for write accesses; depending on your Windows version, these might be quite slow.

Floppy Disk Driver Name

The default name of the floppy disk driver is FLP. A: is FLP1 and B: is FLP2.

FLP_USE

FLP_USE may be used to set the name of the FLP device. The name should be three characters long, in upper or lower case.

FLP_USE mdv	**The FLP device is renamed MDV**
FLP_USE FLP	**The FLP device is restored to FLP**
FLP_USE	**The FLP device is restored to FLP**

FORMAT FLP

The SMSQ/E FLP driver will usually format a diskette to the highest density the medium supports. The density may however be overridden using the FLP_DENSITY command, or by adding a special code to the end of the medium name in the FORMAT command.

FLP_DENSITY

The SMSQ/E format routines will usually attempt to format a disk to the highest density the medium supports. The FLP_DENSITY (code) is used to specify a particular density during format. The density codes are "S" for single sided (double density), "D" for double density and "H" for high density.

FLP_DENSITY S	**Set the default format to single sided**
FLP_DENSITY H	**Set the default format to high density**
FLP_DENSITY	**Reset to automatic density selection**

The same code letters may be added (after a *) to the end of the medium name to force a particular density format. (For compatibility with older drivers, if the code letter is omitted after the *, single sided format is assumed.

FORMAT 'FLP1_Disk23'	**Format at highest density or as specified by FLP_DENSITY**
FORMAT 'FLP1_Disk24*'	**Format single sided**
FORMAT 'FLP1_Disk25*S'	**Format single sided**
FORMAT 'FLP1_Disk25*D'	**Format double sided, double density**

FLP Control Commands

FLP_SEC, FLP_START and FLP_STEP

QPC has no influence over how the Windows disk driver works; therefore these commands are ignored.

PC Hard Disks

SMSQ hard disks for QPC are just large files on the host operating system's file system. The files usually have the suffix ". WIN" but anything else is fine, too. Name and directory can be configured separately for all drives. (See also the configuration section). SMSQ/E's FORMAT command creates the file.

Hard Disk Driver Name

The default name of the hard disk driver is WIN.

WIN_USE

WIN_USE may be used to set the name of the WIN device. The name should be three characters long, in upper or lower case.

WIN_USE mdv	**The WIN device is renamed MDV**
WIN_USE WIN	**The WIN device is restored to WIN**
WIN_USE	**The WIN device is restored to WIN**

FORMAT WIN

Formatting a WIN drive simply creates a large file on the PC's hard disk. The "name" of the WIN device should be the size required in megabytes.

Before you issue the FORMAT command, you have to allow the drive to be formatted. SMSQ/E has a two-level protection scheme to make sure you (or somebody else) cannot format your hard disk accidentally. All drives are protected by default, so you have to declare them to be formattable first.

Please note that the FORMAT command for a WIN drive should only be used from the console of job 0, i.e. the first SBASIC.

FORMAT will fail if there is not sufficient space left on the specified drive, if the medium is write-protected, or if the file *.WIN already exists and contains invalid information (e.g. a DOS-subdirectory).

WIN_FORMAT 1	**Allow WIN1_ to be formatted**
FORMAT WIN1_10	**Create a 10 Megabyte WIN device.**
	... You have to echo the two characters displayed...
WIN_FORMAT 1,0	**protect WIN1_ again against unwanted formatting**

Drive/filename assignment

Every WIN drive number is assigned to a file on your hard disk, which contains the complete contents of your WINx_. Default settings are C:\QXL.WIN for WIN1, D:\QXL.WIN for WIN2 etc. It is possible to change the filename of the file that is assigned to a WIN drive number while QPC is running:

WIN_DRIVE 2,"D:\QPC.WIN"	**now WIN2_ is assigned to the WIN file QPC.WIN**
PRINT WIN_DRIVE$(2)	**will tell you the current filename.**

Removable drives, such as ZIP or SyQuest, are now supported. If auto-detection fails use the WIN_REMV command:

WIN_REMV 2	**declares WIN2_ to be a removable drive.**
WIN_REMV 2,1	**does the same for WIN2_.**
WIN_REMV 2,0	**magic - WIN2_ is no longer removable!**

When a drive is declared removable, the .WIN file is closed after all SMSQ/E files on it are closed. This can also be used to allow a single .WIN file to be shared over a network. (Files on a remote computer are automatically set to be removable). As long as one instance of QPC has open files on the drive, no other instance can access it.

The DOS Device

The DOS device has been created to transfer data between the Windows and SMSQ/E environments. Using this device you can directly browse your PC hard disks (or network drives, or CD-ROMs or whatever), as well as read and write files.

Please note that the DOS device is NO replacement for the WIN device (it never was intended to be), all SMSQ header information gets lost on DOS drives; therefore, you cannot store executable code on them.

Drive/directory assignment

By default, DOS1_ corresponds to C:\, DOS2_ to D:\ and so on. However, from version 3 on you can freely choose the base used. From v3.03 on you can also read and change the setting while QPC is running:

```
DOS_DRIVE 2,"C:\WINDOWS"      assign DOS2_ to the windows directory
PRINT DOS_DRIVE$(2)           would now return "C:\WINDOWS"
```

Restrictions and some background information on the DOS device

You can use this device in the same way as any other QL directory device to access and exchange files between Windows and SMSQ/E. It is easier than ever before. The usual restrictions imposed by the general QDOS file naming convention apply, i.e. the length of the directory + filename is limited to 36 characters. Names longer than that won't show up in the directory lists! Therefore, it is a good idea to place files that you want to access from both SMSQ/E and Windows only one or two directory levels deep, or change the base of a DOS drive to one directly above the desired directories.

Many filenames that are valid under SMSQ are not valid under Windows. The offending characters (e.g. *, /, ? etc. or filenames with spaces at their end) are translated into other, valid ANSI characters. This conversion works quite well, but you are advised to only use valid filenames wherever possible.

One problem with the SMSQ way of accessing files is that the "_" separator can be a valid part of a name or a directory separator. Therefore, the relation SMSQ filename -> Windows filename is ambiguous. This can cause quite some problems:
Let's say you have two directories named C:\QL\STUFF\ and C:\QL\STUFF_NEW\ and you want to create a file called dos1_QL_STUFF_NEW_BRANDNEW.TXT. Where does that file belong? It could mean any of the following:

C:\QL_STUFF_NEW_BRANDNEW.TXT
C:\QL\STUFF_NEW_BRANDNEW.TXT
C:\QL\STUFF\NEW_BRANDNEW.TXT
C:\QL\STUFF_NEW\BRANDNEW.TXT

Your intention was probably the last one, but how should QPC now? The easy solution is not to use underscores in directory names. However, if you can't help it, it becomes essential to know how the DOS device works. Since v3.02, there is a new algorithm, which is based on the simple assumption that if you have a directory called "QL_STUFF" you won't also want to create "QL\STUFF".
Unlike the algorithm used in versions prior to 3.02, the new one is not easy to describe, but I'll try nonetheless. The basic principle is that the algorithm always searches for the longest consecutive part of the name. In the example above, QPC would begin by searching for any directory *starting* with "C:\QL". If none was found, the process completes and the result is simply "C:\QL_STUFF_NEW_BRANDNEW.TXT". Otherwise, it will look for any directory starting with "C:\QL_STUFF". If found, QPC will then try "C:\QL_STUFF_NEW", and so on. If not found, however, it will test whether the last successful part ("C:\QL_STUFF") is itself a directory. If it is, it is considered part of the filename and all future searches use this as the base (i.e. the next step would be "C:\QL_STUFF\NEW"). If not, the search terminates with the result again being "C:\QL_STUFF_NEW_BRANDNEW.TXT".
If this sounds too confusing or too badly explained (probably both) just remember one thing: never use "_" within directory names.

Finally, please note that you cannot use RENAME to rename files on a DOS drive. SMSQ/E allows you to rename files from one directory to another one, but this is not compatible with the DOS way of doing things. If you want to rename a file, you need to COPY it to the new location and DELETE the old file.

The QPC CD-Audio module

As a little extra bonus, QPC contains a module to play Audio CDs. There are 23 new BASIC commands for the complete control of all audio functions of a CD-ROM drive. A tiny CD-player comes as an example. The functions of the player are described at the end of this chapter.

First some terms of CD programming:

Track:	one title
Frame:	one sector of a CD. The sector length on Audio CDs is 2352 Bytes

REDBOOK-Format: a standard format for direct sector addressing. Sectors are addressed through a time index in the form of a longword formatted as $00MMSSFF. MM is the minute, SS the second and FF is the frame. One second has 44100(Hz)*2(Stereo)*2(16 Bit)/2352 (sector length) = 75 frames.

HSG-Format: another format to address a sector. Here they are only addressed sequentially. HSG=(minute*60+second)*75+frame

New Basic commands

As usual, all parameters in square brackets are optional.
Unless specified, all sectors are addressed in Redbook-Format.

CD_INIT

Syntax: CD_INIT ['name']

This command must be used before any other in order to initialise the CD drive for SMSQ. After the first call, the command is ignored on all subsequent calls. The string parameter is only there for compatibility with QPC1. It is ignored by QPC2.

CD_PLAY

Syntax: CD_PLAY [start[,end]]

This is the most important command. Without parameters the whole CD is played. An optional start and end track can be given. The command returns as soon as the CD starts playing. The parameters are given in tracks (bit 31 clear) or in sector units (bit 31 set).

E.g.: CD_PLAY 3 or with the same effect
CD_PLAY CD_TRACKSTART(3) + $80000000

CD_STOP

Pauses playing. If the driver was already in pause mode, a complete stop is performed (as if a new CD was inserted; restart from track 1 and so on)

CD_RESUME

Resumes playing from where it stopped.

CD_EJECT, CD_CLOSE

Opens/closes the drive tray.

CD_ISPLAYING, CD_ISCLOSED, CD_ISINSERTED, CD_ISPAUSED

Syntax: x% = CD_xxx

These functions return the current status according to the keyword. Please note that Windows cannot tell whether the tray is closed or not, therefore CD_ISCLOSED always returns the same result as CD_ISINSERTED when used on QPC2. An empty tray was obviously something the Microsoft geniuses could not imagine.

CD_TRACK

Syntax: track% = CD_TRACK

Returns the number of the track currently being played.

CD_TRACKTIME

Syntax: x = CD_TRACKTIME

Returns the elapsed time within the current track.

CD_ALLTIME

Syntax: x = CD_ALLTIME

Returns the total elapsed time of the CD.

CD_HSG2RED, CD_RED2HSG

Syntax: red=CD_HSG2RED hsg
hsg=CD_RED2HSG red

Converts an HSG address to Redbook and vice versa.

CD_TRACKSTART

Syntax: x = CD_TRACKSTART track

Returns the start sector of a track.

CD_TRACKLENGTH

Syntax: x = CD_TRACKLENGTH track

Returns the length of a track.
Attention: This is the only function that returns an HSG-number.

CD_FIRSTTRACK, CD_LASTTRACK

Syntax x% = CD_xxx

Returns the number of the first/last track.

CD_LENGTH

Syntax: x = CD_LENGTH

Returns the total length of the CD.

CD_HOUR, CD_MINUTE, CD_SECOND

Syntax: x% = CD_xxx Redbook

Returns the hour, minute or second of a Redbook address.

Sample CD player

The program CDPLAYER.BAS which comes with QPC is a complete CD-player. It uses all the above commands, so you can see how they actually work.

It is possible to pass a command string to the program that is then passed to the CD_INIT command. As mentioned above, this is not needed on QPC2.

E.g. EX flp1_cdplayer.bas;'mscd001'

After initialisation, a small window appears in the middle of the screen. On the left side, the current status is shown. That is:

- the current track
- the relative position within the CD
- the remaining play time
- the elapsed time of the current track
- the remaining time of the track
- the drive status: whether a CD is being played, the tray is open or a CD is inserted.

On the right side are the control buttons in the form of the number pad of your keyboard (switch NumLock on!). The commands are:

7 Stop	8 Pause	9 Play
4 Previous Track	5 Replay	6 Next Track
1 Rewind	2 Open/Close Door	3 Forward
0 Quit		

Quitting the player does not affect the playing of the CD. When the player is restarted, it will resume with the current status.

Troubleshooting

When starting QPC2 I get the error "ws2_32.dll not found".

From version 3.30 on QPC2 needs a Winsock2 system to run. Win95 does not ship with Winsock2, however there is a free update on the Microsoft page that solves this problem.

When starting QPC2 I get the error "DINPUT.DLL not found".

DirectX is not properly installed. For NT4 apply the current service pack. For all other systems, please go to the Microsoft homepage and get the DirectX installation files (also available with many games, and on magazine cover CDs).

The display is distorted

First, make sure that it is not a QL program incompatibility issue, by starting QPC2 in the QL mode, not in high colour mode. If the problem persists, then it is likely that you need to update your computer's graphics card display driver and/or DirectX.

BASIC is always in the left hand area of the screen

This can easily be changed. For example, select a higher resolution (e.g. DISP_SIZE 640,480) and type
WMON ,50,50 - you see? Procedures like WINDOW with larger parameters than usual can use the higher resolutions!

Sometimes the display looks strange, especially with games

Some programs and games assume that the screen base is located at address $20000 (131072) and that the display size is 512x256. If you are using a different resolution this is no longer the case and the output no longer goes to the real display. Either switch back to 512x256 QL mode or try the QPC_QLSCREMU command.

I have problems accessing floppy disks.

If you are using aggressive anti virus software, try disabling it.

I get exception faults, especially when running QPC2 in high colour mode.

If you are using something like Norton Crash Guard, try disabling it.

I can't rename files on the DOS device

The DOS device does not support renaming files. Use COPY and DELETE instead.

The mouse leaves trails in fullscreen mode

Most likely the "mouse trail" option in the Windows mouse configuration has been set. This is often the case on laptop computers. Unfortunately, the way Windows handles mouse trails is incompatible with QPC's way of displaying the emulated screen, so you'll have to switch it off.

How can I switch between QPC2 and other Windows applications?

Use one of the standard Windows keystroke functions: hold the ALT key and press TAB once or several times until the application you want is highlighted. Alternatively, hold ALT and press ESC to cycle through the various Windows applications.

The "save" button on the QPC2 configuration dialog is disabled.

The button is automatically disabled when QPC detects that it can't write to the SMSQE.BIN file. Please make sure the file is not read-only and it's not used by any other application.

The middle mouse button doesn't do anything, though pressing both buttons does the mouse hotkey.

QPC can only recognize the middle mouse button if the mouse driver is set to use it. Go to the Windows mouse setup in Control Panel and set the middle button to "middle button".

Sysmon just makes noises after startup.

Update Sysmon to version 2.06 or higher.

QPAC2 "Jobs" just shows "corrupt memory"

Update QPAC2 to version 1.40 or higher.

Manual revision history

Revision 2.10

- CPU chapter reworked for new 68020 kernel.

Revision 2.09

- Explained PAR_GETFILTER and PAR_SETFILTER.
- Explained QPC_WINDOWSIZE
- Small chapter about TCP/IP added

Revision 2.08

- New SER functions explained.
- Explained PAR_DEFAULTPRINTER$.
- New command QPC_WINDOWTITLE.
- New configuration dialogs explained.
- Small chapter about SSSS added.

Revision 2.07

- Revised language and grammar (thanks to Per Witte)
- Removed out-of-date references (e.g. availability of certain software updates).

Revision 2.06

- New PAR functions explained.
- New command line explained, including new QPC_CMDLINE$ function.
- Separated syntax definition from titles. Makes editing this manual much easier.

Revision 2.05

- Deleted obsolete paragraph messing with “much memory slows down disk access”.
- Mentioned “left+right mouse button together” substitute for middle mouse button in MOUSE_STUFF.
- Added paragraph that WIN devices should only be formatted from within the console of job 0.
- Some new troubleshooting items

Revision 2.04

- Added new configuration options to beeper chapter
- Added chapter about fast memory
- Added DOS_DRIVE and DOS_DRIVE$ commands and updated the DOS device text
- Added manual revision history :-)