QL Today
DEUTSCH
Jahrgang 5
Ausgabe 4
Nov/Dezember
2000
ISSN 1432-5446
Das Magazin über QL, QDOS,
Sinclair Computer, SMSQ...

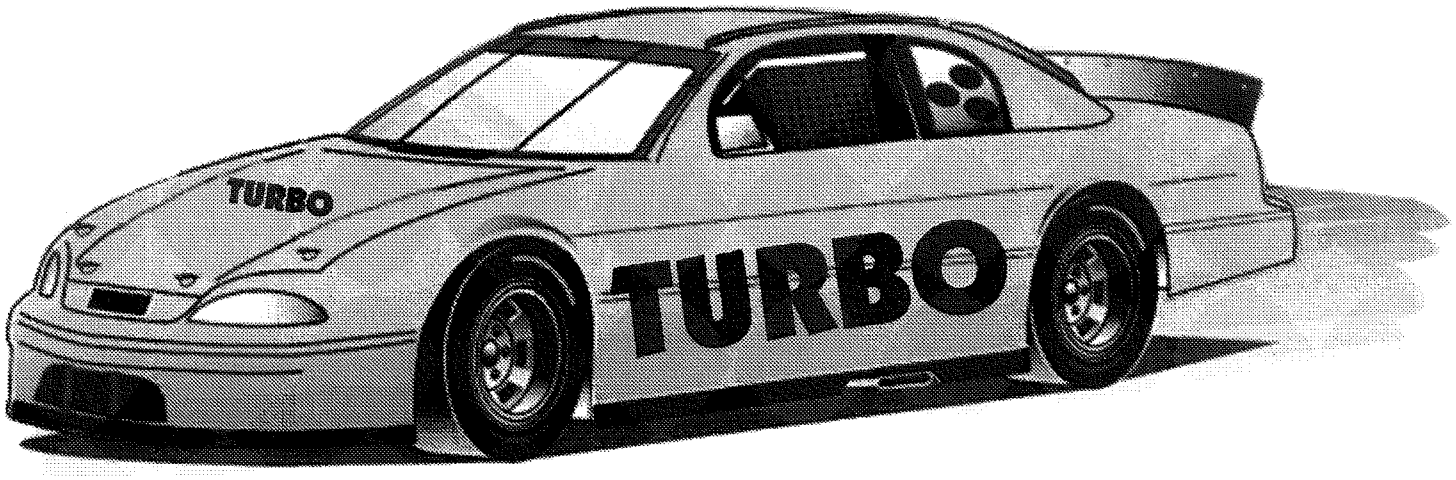
3 Wege zur Beschleunigung Ihrer BASIC Programme:

Hardware-Upgrade:
Q40

ODER

Software-Upgrade:
QPC2
Version 2

und/oder ...



... holen Sie sich den TURBO für IHR System!

Inhalt

Kleinanzeigen

QL Today ist die einzige, aktuelle Zeitschrift für QL-Neuigkeiten. Wir bieten auch Kleinanzeigen zum absoluten Sonderpreis an! Es gibt hier keinen Unterschied zwischen privaten und kommerziellen Anzeigen, nur mit dem QL und Drumherum sollte es schon etwas zu tun haben.

Bis zu 50 Worte im englischen oder deutschen Teil kosten DM 5,- (oder 3 Internationale Antwortscheine), bis zu 100 Worte kosten DM 10,- (oder 6 Internationale Antwortscheine). Soll die Anzeige im deutschen und englischen Teil erscheinen, verdoppelt sich der Preis.

Zu Verkaufen

SINCLAIR QL deutsch, komplett mit Bernsteinmonitor. TrumpCard 896Kb, FLP2 Level-Chip, 2*3,5Zoll-Floppy, 122Mb-HardDisk + Qubide. Preis 50,-DM

Andreas Schulz
Frankampstr. 170
45891 Gelsenkirchen
Tel.: 0209/799098

QL *Today* DEUTSCH

ISSN 1432-5446

Herausgeber:

Jochen Merz Software
Im stillen Winkel 12
47169 Duisburg
Deutschland

Tel. +49 203 502011
Fax +49 203 502012
Box1 +49 203 502013
Box2 +49 203 502014
Email: JMerz@j-m-s.com

QL *Today* erscheint alle zwei Monate, Erscheinungsdatum der ersten Ausgabe ist der 15. Mai. Das Abo beginnt mit der aktuellen Ausgabe zum Zeitpunkt der Bestellung. Preisinformationen und Antragsformulare sind bei Jochen Merz Software erhältlich.

Ihre Kommentare, Vorschläge und Artikel sind herzlich willkommen. SIE machen **QL *Today*** möglich. Wir verbessern das Magazin wo immer möglich, um Ihren Vorstellungen gerecht zu werden. Artikel sollten auf 3,5" Diskette (DD oder HD) eingeschickt werden. Das Format sollte ASCII, Quill oder Text87 (Druckertreiber angeben!) sein. Bilder sollten im _SCR-Format geschickt werden, GIF und TIF ist auch möglich. BITTE senden Sie auch einen Ausdruck der Bilder. Wenn ein Bild an einer bestimmten Stelle plaziert werden soll, geben Sie es bitte auch an. Natürlich können auch alle Bilder/Artikel auf elektronischem Weg übersandt werden, also in die Box oder per E-Mail.

Redaktionsschluß für Artikel und Werbung:

Ausgabe 1:	**30. April**
Ausgabe 2:	**30. Juni**
Ausgabe 3:	**30. August**
Ausgabe 4:	**30. Oktober**
Ausgabe 5:	**30. Dezember**
Ausgabe 6:	**28. Februar**

QL *Today* behält sich vor, eingeschicktes Material nicht zu veröffentlichen. **QL *Today*** ist unter keinen Umständen für die Richtigkeit der abgedruckten Artikel und Programmen haftbar, ebenso nicht für aus fehlerhaftem Material hervorgerufene Datenverluste, Unbenutzbarkeit oder ähnliche Probleme, die aus Artikeln in **QL *Today*** herrühren könnten. Die Meinung in diesem Magazin entspricht der des jeweiligen Autors und nicht notwendigerweise der des Herausgebers.

Editorial

Jochen Merz

Liebe Leser,

ein turbulenter Oktober mit vielen QL-Treffen liegt hinter uns.

Nach Österreich konnte ich dieses Jahr leider nicht fahren, aber Gerhard Plavec hat eine kleine Zusammenfassung geschickt. Hoffen wir, daß es nächstes Mal wieder was wird - und hoffentlich etwas näher zu Salzburg, damit mehr QLer zum Treffen finden können.
Das Treffen in Italien fand nach einigen Jahren wieder statt. Ich glaube, das letzte Treffen gab es vor vier Jahren. Da keine deutschsprachigen Besucher anwesend waren, gibt's einen Bericht auch nur in der englischen Ausgabe.
Highlight des Oktobers, des Jahres 2000 und sicher auch der letzten Jahre war QL 2000 in England. Eine tolle 2-Tages-Veranstaltung, wie wir sie schon lange nicht mehr erlebt haben. Dank an die Veranstalter und an alle Besucher. Dieses Ereignis war sicherlich förderlich für die Zukunft des QLs - alle QL-Akteure sahen Bestätigung im Interesse der QLer, und die Besucher konnten sich der Verbundenheit der Händler zum QL bestätigt sehen. Besucher aus allen möglichen Ländern Europas und auch den USA waren anwesend und trugen zum Erfolg des Treffens bei.
Eine Woche später gab's dann noch das QL-Treffen in Paris. Besser besucht als eigentlich erwartet (wir dachten, daß QL 2000 Besucher abzieht) war es auch ein Erfolg und soll im nächsten Jahr am gleichen Ort wieder stattfinden.

Nun ja, auch wenn die Treffen positiv verliefen, bin ich doch froh, daß nun für einige Monate keine QL-Reisen anstehen. Eindhoven ist ja ein Katzensprung von Duisburg, und dann kommen die nächsten QL-Treffen erst wieder im nächsten Jahr. So habt ihr auch Zeit zum Verschnaufen und könnt nach dem Weihnachts- und Neujahrs-Trubel wieder Pläne für zukünftige Treffen in den Terminkalender eintragen.

Die Q40 Situation hat sich bei mir verbessert: Die Mailbox läuft seit Monaten stabil. Es gibt zwar noch Wünsche meinerseits, aber daran wird gearbeitet. Die Geschwindigkeit ist nach wie vor ein Traum.

Die Drucker-Situation entspannt sich leider nicht wie gehofft. Der in der letzten Ausgabe gepriesene EPSON Stylus Color 880 scheint doch nicht QL-kompatibel zu sein. Ich schrieb ja, daß es mir "spanisch" vorkommt, daß so wenige Schriften vorhanden sind. Die Engländer haben von EPSON bestätigt bekommen, daß es auch nur ein Raster-Drucker ist. Komischerweise steht auf der deutschen Website von EPSON auch heute noch, daß es sich um einen ESC/P2 Drucker handelt. So auch in den herunterladbaren Prospekten. Naja, wer unentschlossen ist, sollte erst bei EPSON nachfragen bzw. sich vom Verkäufer beim Kauf schriftlich geben lassen, daß die Fähigkeit gegeben ist. Ein Blick in die Anleitung des Druckers könnte vielleicht auch Aufschluß geben. Wer mehr weiß, läßt es uns bitte auch wissen.

Bleibt nach wie vor nur EPSONs Top-Modell, der Stylus Color 900. Ich habe noch einen preiswert erstanden, da er momentan durch den 980 ersetzt wird. Der Drucker ist toll, sehr schnell, hat große Tintentanks und emuliert auch IBM-Drucker und "kann" natürlich ESC/P2. Wer ihn für unter 500,- DM noch bekommen kann sollte zuschlagen, ein echtes Schnäppchen. Vielleicht als Weihnachts-Geschenk...?

Auch wenn mit der vorliegenden Ausgabe eine Rekord-Ausgabe vor Euch liegt (meistens läuft es ja auf 18 Seiten hinaus), so ist es trotz alledem kein Grund, keine Artikel zu schicken! Wo bleiben die Reaktionen auf Peta's Artikel? Bislang ist noch überhaupt nichts dazu hier eingetroffen, von den "treuen" Schreibern mal abgesehen. Kein Thema, keine Wünsche, keine Vorstellungen? Kann ich nicht glauben! Will ich nicht glauben! Beweist mir doch das Gegenteil!

Bleibt mir nur noch, frohe Weihnachten und alles Gute für "den Rutsch" zu wünschen,

Jochen Merz

Warum immer ich!? ... oder Glück im Unglück - so geht's weiter!

Jochen Merz

Ja, mittlerweile hat sich Q40 bei mir als Mailbox- und Fax-Empfangs-Rechner etabliert.
Nach einigen Einstell-Schwierigkeiten mit den Interrupts (ich benötige 4 serielle Schnittstellen) habe ich nun auch 4 serielle Schnittstellen, die fast perfekt funktionieren. Dazu besitzt der Rechner auch noch 3 Parallel-Schnittstellen, die auch alle genutzt werden können. Wow!
Wie ich schon schrieb, ist die Geschwindigkeit atemberaubend, das merkt man insbesondere bei der Decodierung und Darstellung der Faxe.
Aber wie es meistens ist, fällt einem bei der Arbeit mit einem System auch die eine oder andere Schwachstelle auf. Größtes Manko für mich ist die Tatsache, daß ich keinen Mode 4 in der Auflösung 1024x512 anwählen kann. text87, eine meiner wichtigsten Anwendungen neben QD und QSpread, kann nur mit 512x256 genutzt werden. Naja, und in der Auflösung kann von "Nutzen" nicht mehr die Rede sein. Gut genug zum Drucken ist es gerade noch, zum Arbeiten aber nicht (ich bin ja schon seit vielen, vielen Jahren höhere Auflösungen gewöhnt - angefangen mit dem guten alten Extended MODE 4 Emulator für den ATARI - hier gab's schon 60% mehr Auflösung!). Peter weiß von dem Problem und ich hoffe, er findet dafür eine Lösung.
Zudem, und das ist ein generelles Problem, das mit der Einführung von "mehr Farben" kommen mußte, kann man gar nicht genug Speicher haben! Wer gewohnt ist, viele Programme gleichzeitig zu nutzen, braucht viel Speicher, denn jedes abgespeicherte Bildschirmpixel eines jeden Fensters braucht acht mal soviel Speicherplatz wie im MODE 4.
QXL-User sollten sich auch überlegen, ob für sie der Hi-Color Mode Sinn macht; eine 8MB QXL ist eigentlich unabdingbar. Glücklicherweise kann man bei der QXL jedoch die Farbtiefe wählen (also MODE 4, wenig Speicherverbauch, wenig Farben), oder Hi-Color, viele Farben, und viel Speicherverbauch) - in jeder Auflösung! Das wünsche ich mir noch sehnlichst für Q40.
Was die Kompatibilität angeht, auch hier hab' ich in den Monaten Erfahrungen gesammelt: Sehr gute Nachrichten (für mich): Alles, was ich an Software nutze, läuft!
Und was soll ich sagen: Die "vielen" Farben sind nun verfügbar, aber ich habe bis heute leider immer noch nicht die Zeit gefunden, damit "mal richtig" was anzufangen. Bis auf Probieren auf den Treffen gab's einfach keine Zeit dazu, den Streß in den vergangenen Monaten, jeden zweiten Monat QL Today, und die vielen Treffen - da bleibt nicht viel übrig. Ich freu' mich schon auf den Dezember...

Heidenreichstein 2000

Gerhard Plavec

Das diesjährige Treffen in Heidenreichstein kann ich diesmal leider keineswegs als international bezeichnen, denn es kamen ausschließlich Österreicher. Und auch diese hatten keinen einzigen "echten" QL, sondern nur wenige PCs - allerdings mit QPC aufgerüstet - dabei. Was war los? Hatten wir zu wenig Werbung gemacht? Oder war daran Schuld, daß das Treffen zum dritten Mal in Heidenreichstein stattfand? Oder sind einfach die PCs endlich - nach 15 Jahren - so verläßlich geworden, daß die letzten QLer auch umgestiegen sind? (Obwohl, verläßlich waren die PCs ja schon immer, man konnte sich stets darauf verlassen, daß sie Abstürzen :)
Man würde ja erwarten, daß Treffen, die etwa zur gleichen Jahreszeit und jeweils am gleichen Ort stattfinden, nicht immer wieder eine ausgedehnten Werbecampagne benötigen... Wieso klappt das scheinbar in Eindhofen, nicht jedoch in Solms oder eben Heidenreichstein?
Auf der anderen Seite war es auch wieder ein Glück, daß wir nicht so viele waren, denn der Wirt konnte uns den Saal erst Samstag Nachmittag geben, da sich ein vollbeladener Autobus zum Mittagessen angesagt hatte... Jedenfalls war es das erste Treffen, bei dem alle das Alternativprogramm - Besichtigung der Wasserburg Heidenreichstein, Besuch eines Wakkelsteinfeldes, etc - zur Gänze mitmachten. Wie es nun weitergehen soll, weiß ich auch nicht. Ich hoffe auf Entzugserscheinungen bei einigen Qlern, so daß der Wunsch nach weiteren Treffen wieder stärker wird...

Ist der Windows-PC die Zukunft des QL?

Peter Graf

Als ich in der letzten QL Today den Artikel von Peta Jäger gelesen habe, war ich verwundert, mit welcher Selbstverständlichkeit dort Windows als Zukunft des QL propagiert wird. (QPC2 ist Windows-Software, die SMSQ/E emuliert, es ist kein QL, es ist auch kein QL-Emulator.) Und dies soll nicht nur eine Möglichkeit unter vielen sein, sondern auch noch die „einzig wahre Antwort"?

Leider werden auch noch der Q40, seine Software-Programmierer und seine Anwender in ein schlechtes Licht gerückt. Petas Artikel wendet sich auch gegen QL-Hardware im allgemeinen, wovon auch GoldCard, SuperGoldCard usw. betroffen sind. Es folgen Antworten zu einigen Punkten:

„Auf der anderen Seite bin ich jedoch über die mangelnde Weitsicht Tony Tebbys hinsichtlich einer wirklichen QL-Plattform (in Soft- oder Hardware) über alle Maßen enttäuscht."
Der QL hat nun mal ein 68K Betriebssystem und es ist aufgrund der Assembler-Implementierung stark an die 68K CPU-Architektur gekoppelt. Beklagenswert ist die mangelne Weitsicht Motorolas, die 68K CPUs auf einen moderneren Stand zu bringen. Aber nicht eine mangelne Weitsicht Tony Tebbys, der in keinster Weise Motorolas Versäumnisse zu verantworten hat.

„Zunächst einmal ist es ein Schlag ins Gesicht derjenigen die den QL seit Jahren unterstützen, neue Hard- (QXL) und Software kaufen, ... wenn dann die Farbtreiber zuerst für Q40 programmiert werden und danach erst für die QXL und den QPC."
Haben die Q40 User früher den QL nicht unterstützt? Haben die Q40 User keine QL Hardware gekauft? Haben die Q40 User ihre QL Software und SMSQ/E geklaut? Gibt es sonst Gründe, warum Q40 User schlechter behandelt werden sollen als andere QL User? Nein? Dann ist es unfair, zu verlangen, dass Tony Tebby neue Software erst für andere Plattformen fertigstellen muss, obwohl die Reihenfolge so sinnvoller war! Hätte er die fertigen Farbtreiber des Q40 erst zurückhalten sollen, bis Marcel Kilgus sie an den QPC2 angepasst hat? Nur damit die Q40 User genauso lange warten müssen?
Bei alledem solltest du nicht vergessen, dass Tony Tebby für die Q40 Entwicklung einschließlich Q40 Farbtreiber von mir persönlich bezahlt worden ist. Der Q40 hat Tony Tebby für die Fertigstellung der Farbtreiber motiviert und ihm eine gute Entwicklungsplattform gegeben. Ohne Q40 könntest du heute immer noch ohne Farbtreiber dastehen. Du solltest dem Q40 und seinen Hard- und Softwareentwicklern sehr dankbar sein und uns nicht mit Ausdrücken wie „Schlag ins Gesicht" beschimpfen.

„Immerhin haben viele Leute die QXL für viel Geld gekauft und die meisten haben dann noch mal für SMSQE bezahlt."
Dasselbe gilt auch für den Q40 und seine User. Die QXL war zwar damals teurer als der Q40, aber das kann doch kein Argument sein, dass Tony Tebby eine bestimmte Reihenfolge einhalten muss.

„Dass QPC(2) dann allerdings so völlig unbeachtet als letzter bedient wird, haben Marcel Kilgus (der Programmierer) und alle begeisterten QPC-Benutzer wahrlich nicht verdient!"
Die Farbtreiber für QPC2 waren zwar als Letzte fertig, sie werden aber keineswegs völlig unbeachtet bedient. Sie werden mit hohem Werbeaufwand wie auffälligen ganzseitigen Anzeigen in Quanta und QL Today in Szene gesetzt! Im Vergleich dazu musste der Q40 bislang ohne derartige Anzeigen-Unterstützung auskommen.

„Die von Microsoft entwickelten Betriebssysteme (oder was MS als solche bezeichnet) haben sich leider überall auf diesem Planeten (vermutlich auch sonstwo) verbreitet. Wir konnten das nicht aufhalten."
Leider? Konnten das nicht aufhalten? Also wer Windows nebst Emulator als einzig wahre Zukunft des QL vorschlägt, der braucht doch nicht so zu tun als hätte er Windows aufhalten wollen. Entschuldigung, aber das ist doch unglaubwürdig!

„Zu 1. Abwärts-Kompatibilität, was Software betrifft. Q40 als einzig verfügbare (nicht emulierte) QL-Alternative bietet kein bisschen mehr Kompatibilität als QPC2, eher weniger im Moment."
Eine ziemlich haltlose Behauptung. Der Q40 bietet mit QDOS Classic ja sogar die Möglich-

keit, Software laufen zu lassen, die noch ein echtes QDOS benötigt und mit SMSQ/E nicht richtig läuft. Und beim Q40 können die Programme sogar noch direkt in den QL Bildschirm-Speicherbereich schreiben. Außerdem muss man sehen, dass QPC2 immer auf MS Windows mit all seinen Problemen angewiesen ist. Stürzt da z.B. ein instabiler Windows-Grafiktreiber ab, was bei mir oft vorkam, dann hat auch das SMSQ/E Programm im QPC-Fenster gelitten.

„Der Druckerport unter QPC wird korrekt angesprochen, die seriellen Schnittstellen funktionieren mit bis zu 115k Baud ohne erkennbare Probleme (u.a. im Mailboxbetrieb getestet). Floppyzugriff funktioniert mit DOS- und QDOS-Disketten"
Das kann der Q40 selbstverständlich auch.

„Festplattenzugriff über das QXL-WIN File hat bislang noch nie Probleme gemacht."
Der Q40 kann unter SMSQ/E natürlich auch auf IDE-Festplatten zugreifen. Er kann sogar richtige SMSQ/E Partitionen dort anlegen und braucht nicht den Umweg über Microsoft Filesysteme zu gehen.

„Sind allerdings zwei DOS-Rechner mit installiertem QPC vernetzt, dann kann QPC natürlich auch auf den zweiten PC zugreifen und verfügt damit über das schnellste Netzwerk unter allen QL-Varianten"
Falls damit SERNET gemeint sein soll: Das geht beim Q40 natürlich auch.
[Anmerkung des Editors: nein, das war nicht gemeint - es geht um den Zugriff auf QXL.WIN-Dateien, die auf anderen Rechnern liegen und beispielsweise über Ethernet angesprochen werden können.]

„Q40 läuft zwar parallel zu WINDOWS, allerdings nur wenn man sich noch einen zweiten PC ins Zimmer stellt. Keine wahnsinnig aufregende und vor allem eine teure Variante!"
Hier wird ja schon wie selbstverständlich vorausgesetzt, dass jeder QL User bereits einen PC braucht und kauft und Windows benutzen will! Ganz so weit ist es noch nicht, und das finde ich auch gut so. Gerade der Q40 bietet ja die Möglichkeit, auf Windows zu verzichten, weil hier erstmalig auf einer QL-Hardware auch Linux läuft. Q40 Linux bietet soviel Software, dass man als QL User möglicherweise dankend auf einen zusätzlichen Windows PC verzichten kann.

„Bei Q40 ist das auf Grund des Designs wohl kaum machbar. Allenfalls parallel oder seriell angeschlossene Geräte sind möglich."
Eine Unwahrheit, die nur wieder den Q40 schlecht macht. Beispielsweise läuft beim Q40 bereits jetzt Sampled Sound, Ethernet, CDROM und CD-Recorder. Über den Q40 Extension Bus kann man außerdem ISA-Karten aus nahezu allen Bereichen einsetzen. Für diese Dinge fehlen dem Q40 die QDOS Treiber, aber das gilt genauso auch für alle anderen QLs und auch Emulatoren wie QPC2. Gerade auf Grund seines Designs hat der Q40 im Gegensatz zu QPC2 bereits jetzt Treiber und Hardware für Sampled Sound. Damit kann man Sprache und Musik in Stereo abspielen, sowohl unter QDOS als auch SMSQ/E. Und auch der normale QL BEEP funktioniert. Auch die Farbtreiber-Entwicklung wurde auf Grund des Q40 Designs erleichtert.

„Unterstützt QPC den USB-Port in der Zukunft, dann wird man praktisch keine weiteren Treiber mehr schreiben müssen, da der USB-Treiber für SMSQE ausreicht, um alle aktuelle Hardware zu betreiben."
Stimmt nicht, ein eigener Treiber für jedes Gerät müsste entwickelt werden. Und USB-Treiber sind erheblich komplizierter als das meiste was uns sowieso schon fehlt. (Mit einigem Aufwand kann man übrigens USB-Hardware auch am Q40 betreiben, aber das wäre in Anbetracht der momentanen QDOS Treibersituation wohl eher lächerlich.)

„Günstig ist eigentlich keiner von beiden. Wenn man jedoch die geringe Verbreitung von SMSQ in Rechnung zieht, kann man mit dem Preis für QPC oder Q40 leben, wobei QPC bei Vorhandensein eines PCs doch deutlich zu bevorzugen ist."
Im Vergleich mit anderer QL-Hardware ist der Q40 bei seiner Leistung sehr günstig. QL-Hardware kann man nicht mit Windows-Software vergleichen.

„Die jetzt verfügbaren und bald standardmäßig erhältlichen PCs kommen mit einer Taktfrequenz von 1000MHz und mehr. Damit wird QPC(2) bald auch Q40 überflügeln."
Ich gebe zu, dass irgendwann ein aufgebohrter PC mit einer irrsinnig hohen Taktfrequenz nach Durchorgeln von Megabytes zum Laden eines aufgeblähten Windows Betriebssystems und dem Start eines Emulators, die Rechenleistung des Q40 erreicht (weil Motorola leider keine schnellere 68040 baut).

Die Frage ist nur, was ein solcher PC noch mit dem QL zu tun hat. Für mich gehört zum Wesen des QL, dass er ein überschaubares und effizientes System ist. Der Q40 z.B. erbringt seine hohe Leistung mit einer möglichst einfach zu programmierenden Hardware.
Außerdem gibt es die neusten und schnellsten PCs auch nicht umsonst. Auch PCs kosten Geld und die Hardware veraltet so schnell, dass man sehr oft einen neuen PC braucht. Und Microsoft bittet für die aktuellen Windows Versionen auch nicht gerade selten und billig zur Kasse.

„Selbst ein eventueller Q60 wird, abgesehen von den hohen Kosten, nicht sehr viel mehr bieten können"
Schau mal in ein Datenblatt der 68060 CPU bevor du ein Produkt, das du nicht kennst, negativ darstellst. Warum ist eigentlich der Q60 nur *eventuell*, während die zukünftigen verbesserten Emulatoren als sicher angenommen werden?

„Es gibt auch wenig Entwicklungstools, zumal auch alle Programmierer zum PC abgewandert sind."
Aha! Also immer weiter den Windows PC fördern und QL kompatible Hardware schlechtmachen. Dann wird bestimmt alles besser.

„Die QXL hat es uns doch eigentlich gezeigt, dass man in der heutigen Zeit im Hardwarebereich nicht mehr mithalten kann."
In der Tat hat leider nach dem Thor keiner mehr die Entwicklung eines kompletten QL kompatiblen Systems gewagt. Erst beim Q40 war dies wieder der Fall. Hätte Miracle damals den Mut und den Entwicklungsaufwand erbracht, um ein QL-Komplettsystem wie den Q40 zu bauen, könnte die QL-Welt heute besser aussehen. Ich gebe zu, der Q40 kommt leider sehr spät, fast zu spät. (Nebenbei erwähnt: Das Konzept hatte ich schon zur Zeit der QXL, aber als Student kein Geld, um es zu verwirklichen.)

„Der Q40 hingegen ist und bleibt eine Nische und hat geringe Zukunftsaussichten. Ein Problem ist z.B. - die QXL-Fans können ein Lied davon singen - wenn die Entwickler mal keine Lust mehr auf Q40 haben (z.B. weil die Verkaufszahlen rückläufig sind)."
Wegen der Verkaufszahlen hat ganz sicher niemand für den Q40 entwickelt. Und was ist eigentlich, falls Marcel Kilgus mal keine Zeit mehr hat, um Fehlerbehebungen und Anpassungen an neue Windows Versionen zu machen?

„Dann ist es für alle, die sich die Hardware gekauft haben und z.B. Ersatzteile benötigen, aus und vorbei!"
Also erstmal: Das ist eine reine Unterstellung zu Ungunsten des Q40 und kein Argument. Und was ist denn dann mit den Lebenszyklen eines Windows PC? Die sind doch erfahrungsgemäß noch viel kürzer als die von QL Hardware. PC-Chips, die es noch vor einem halben Jahr gab, sind heute teilweise nicht mehr zu bekommen. Das sind Fakten, die jeder Elektronik-Distributor bestätigen kann. Übrigens bei einem Software-Emulator garantiert auch niemand, daß er ewig unterstützt wird.

„Eine PCI-Grafikkarte, wie sie für den Milan vorgesehen ist, ist jetzt schon im Fachhandel kaum noch zu bekommen."
Milan ist aber nicht Q40. Genau diesen Fehler habe ich eben nicht gemacht. Auch wenn es ein hoher Aufwand war, habe ich für den Q40 eine eigene, QL-kompatible Grafik entwickelt. Und die Chips dafür sind bestens verfügbar.

„ISA-Steckplätze gibt es beim normalen PC auch bald nicht mehr, PC99 Standard! Achtung QXL-Fans: Besorgt Euch vorsorglich schon mal ein Ersatzboard..."
Ja, das sind typische PC Probleme. Microsoft verbietet einfach bestimmte Hardware für einen Standard-PC. Der Q40 User hat es da etwas besser.

„PS/2 EDORAMS sind auch immer schwieriger zu beschaffen (Q40)."
Davon weiß ich nichts. PS2 Module werden außer zum Aufrüsten von PCs auch im Embedded-Bereich eingesetzt und sind gut verfügbar. Vor einigen Wochen waren PS2-Module bei großen Distributoren sogar mal wieder billiger und besser verfügbar als die SDRAM-Module, bei gleicher Größe.

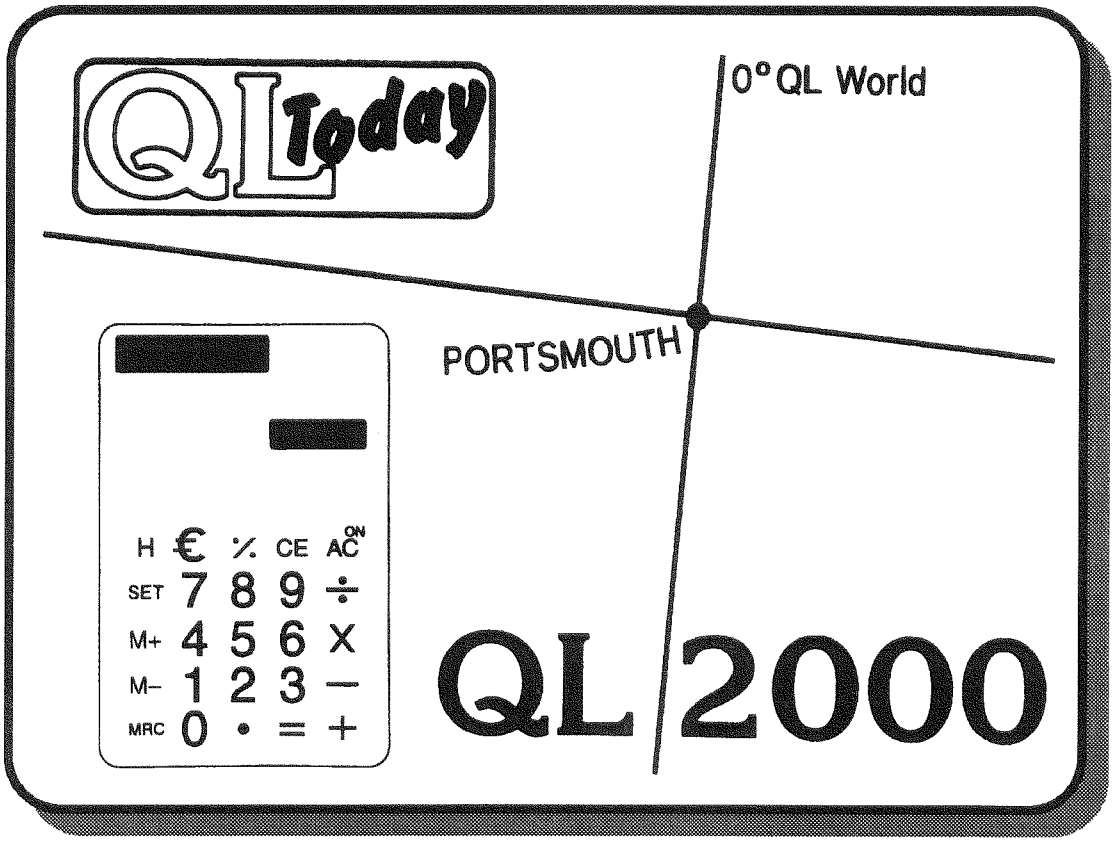
„Für den Bastler und QL-Hardcore Fan mag der Q40 denn auch die einzige Alternative sein, für alle anderen jedoch kann die Wahl eigentlich nur QPC sein!"
Es war sicher nicht so gemeint, aber hier könnte der Eindruck entstehen, dass der Q40 ein gebasteltes System ist. Dem muß ich entgegenhalten, dass der Q40 ein absolut professionelles Mainboard hat, das (auch im Gegensatz zu einigen PC-Mainboards) keine Drähtchen oder Hardwarefehler enthält. Und Q40 User sind nicht unbedingt bemitleidenswerte Bastler oder altertümliche QL-Schwärmer.

Insgesamt hinkt der Vergleich zwischen einem Windows PC und dem Q40 an allen Ecken und Enden. Denn der Q40 ist 68K Hardware und nur mit solcher ist ein fairer Vergleich möglich. Dennoch habe ich mich teilweise auf diesen Vergleich eingelassen, um den Q40 und seine Benutzer etwas gegen negative Behauptungen zu verteidigen. Ich finde, das einzig wahre System gibt es nicht, und jeder QL Benutzer wird seine eigene Entscheidung treffen müssen. Zum Schluss noch meine persönliche Antwort auf die Frage in der Überschrift: Ist der Windows PC die Zukunft des QL? Nein, danke.

Wer will noch mal, wer hat noch nicht...?

Jochen Merz

Wer nicht zu QL2000 gekommen ist, kann's nicht wissen: Das versprochene Präsent für alle QL-Today-Abholer war ein schnuckeliges, hochwertiges Mousepad, ein bißchen kleiner als DIN-A4-Größe, mit QL-2000 Logo bedruckt (zur Erinnerung) und - der Clou - mit eingebautem, solarbetriebenem Taschenrechner und Währungsumrechner (z.B. für EURO). Sowohl Display als auch Solarfeld sind ins Pad eingebettet, die Tasten sind ebenfalls Bestandteil der Mousepad-Oberfläche. Eine tolle Sache, die ich sofort als Präsent bestimmt hatte als ich sie sah.
Nun, ich ließ einige mehr anfertigen als ich Abhol-Voranmeldungen hatte. Trotzdem gingen alle an die Abholer - wie üblich hatte nicht jeder den Abschnitt zurück geschickt. Es ging genau auf "Null" auf, eines verblieb bei mir.
Jetzt habe ich noch Anfragen von QL-2000 Besuchern, die noch gerne Mousepads nachgekauft hätten. Das Problem ist: Ich muß mindestens 40 Stück gleichzeitig abnehmen, das ist die minimale Bestellmenge. Da Roy von QBranch und Tony Firshman auch noch gerne welche für ihrne Kunden hätten, stellt sich jetzt die große Frage: Bekommen wir 40 Stück zusammen?
Wer Interesse hat: Ich sammle Bestellungen bis zum Ende Dezember. Wenn genügend zusammen kommen, lasse ich nachdrucken, ansonsten ist es leider hinfällig. Wäre schön, wenn sich genügend Interessenten finden. Der Preis hängt von der Anzahl ab, wird jedoch maximal 19,- DM/Stück kosten. Wie gesagt, ist nicht nur ein "QL Collectors Item" sondern auch unglaublich praktisch, den Rechner immer direkt zur Hand zu haben.
Nachfolgend eine verkleinerte Abbildung, das Mousepad ist weiß. Zu bemerken sei, daß der Umriß und die "Tasten" des Taschenrechners nur aufgedruckt sind und keinerlei Kante oder dergleichen darstellen.



TCP/IP tritt in die Testphase ein (erste Testfreigabe)

Jon Dent

Zuerst vielen Dank an Jochen für die Übersetzung meines letzten Beitrags. Es war sicher nicht einfach, meine Wortspiele aus dem Englischen zu übersetzen. Diesmal schreibe ich zuerst in Deutsch.
Dieser Bericht ist ziemlich speziell, weil es sicherlich der erste ist, der von einer Nativ-QDOS-Maschine an Jochen per E-Mail gesendet worden ist. Meine Vision, dass unser Computer auch im Internet zuhause wird, ist Wahrheit geworden. Jonathan Hudson hat ein Programm geschrieben, welches es erlaubt, E-Mails zu senden, bis jetzt lief es aber nur auf UQLX. UQLX ist Richard Zidliky's QL-Emulator für UNIX Rechner. Jonathans Programm, es heisst übrigens "qlmailer", braucht TCP/IP von UNIX um mit dem Internet in Verbindung zu kommen. Ich habe qlmailer abgeändert, so dass es meinen TCP/IP Stack benützen kann. Das neue Programm heißt soqlmailer. Soqlmailer ist kompiliert mit der SOQLlib_a Funktionsbibliothek für C68. Es kann so mit der "soqlEngine" arbeiten. Die "soqlEngine" ist ein Programm, das Daten zu und von Anwenderprogrammen wie soqlmailer weiterleitet und ihnen erlaubt, mit Serverprogrammen im Internet Verbindung aufzunehmen.
Soqlmailer baut eine Verbindung mit einem SMTP-Server im Internet auf. SMTP steht für Simple Mail Transfer Protocol. Ist die Verbindung hergestellt, dann nimmt soqlmailer E-Mails, welche mit QD oder QUILL oder was auch immer geschrieben wurden und liefert diese an den Server. Der Server leitet die E-Mails weiter. Wenn der Empfänger die Mail erhält, kann es sein, dass er zurückschreiben möchte. Falls er das macht, dann kommt die E-Mail beim POP3-Server an. Um die E-Mails auf den heimischen QL zu holen, gibt es ein POP3-Clientprogram.

Was braucht man nun, um im Internet mit dem QL E-Mails auszutauschen?

Hardware

Ich habe verschiedene Versuche mit QL-naher Hardware gemacht. Für den Original-QL liegt das Problem bei den Serial-Ports. Zuverlässige Übertragung ist nur bis 1200 Baud möglich. Für höhere Baudraten muss es möglich sein, den Datenfluss in Richtung QL zu bremsen. Normalerweise verwendet der QL die Leitungen DTR auf Ser2 (oder CTS auf Ser1), um die Bremsfunktion zu bewirken. Um eine von Computer zu Computer Verbindung zu machen, kann eines dieser Signale benützt werden, denn beide Computer können den Datenfluss auf Verlangen des anderen bremsen. Den Datenfluss von einem Modem kann man aber nicht bremsen; normalerweise dient DTR dazu, die Verbindung abzubrechen. Das Modem bei meinem Internet-Service-Provider verweigert Verbindungen unter 9600 Baud. Somit ist leider mit dem unveränderten QL kein Zugriff möglich.
Mit SuperHermes von TF Services kann Ser2 Daten bis mindestens 9600 Baud übertragen. Für schnelleres Übertragen ist SuperHermes Ser3 wohl notwendig. Das bedingt etwas zusätzlichere Verdrahtung, aber bei mir läuft es mit 57600 Baud.
Mein Rechner mit SuperHermes hat auch eine Super Gold-Karte. Es gilt nun zu testen, was mit anderen Kombinationen möglich ist.
Schließlich werden ein Telefonanschluss und ein handelsübliches Modem mit bis zu 57600 Baud benötigt.

Software

Die ganze Software in heutiger Auflage passt spielend auf eine DD-Diskette. Die Software läuft auf Minerva- und Sinclair- ROM-Betriebssystemen und auch auf SMSQ/E läuft sie einwandfrei. Die TK2-Erweiterungen werden auch benützt. Die jetzige Version ist eine Testversion und ist bis heute erst auf meinen eigenen Maschinen gelaufen.

Internetzugang

Die jetzige Version benötigt ein Internetzugang via SLIP. SLIP steht für "Serial Line Internet Protocol." Es stellt das einfachste Internet Protokoll für Modems auf Telefonleitungen dar. Heutzutage ist zwar PPP häufiger, besonders bei Gratis-Provider. PPP ist "Point to Point Protocol" und bringt Vorteile vor allem für große Provider. Wer unbedingt mit PPP arbeiten will, muss sich etwas gedulden, da die PPP-Version für den SOQL-Stack noch in der Entwicklung steckt. Wer zu den ungeduldigen Pionieren gehört, findet warscheinlich bei einem lokalen Provider den notwendigen SLIP Zugang. Man kriegt wohl ein Abonnement nicht gratis, aber dafür ist diese Version der Software für Testwillige umsonst. Ein Abonnement mit begrenzter Zeit ist

wohl ratsam, wenn man nicht zusätzlich auch mit einem PC browsen will. Mit etwas Glück wird vor Ablauf des Abonnement die PPP-Version fertig sein. (Und vielleicht ist auch das Browsen mit dem QL bis dann möglich).

Wer will Mit-Testen?

Wer zu den QL-Internet-Pionieren gehören will, kann beim Testen mitmachen. Erstens sucht man sich ein Provider mit SLIP-Zugang. Beim Provider wendet man sich vielleicht besser an einen Techniker als an einen Verkäufer. Berichte über Fund eines Providers würden die QL-Today Leser sicher gerne erfahren. Die Software kann bei mir bezogen werden. Ich werde jedem, der mir eine Diskette mit einer an sich adressierten Etikette und Rückporto zusendet, die Software schicken. Für das Rückporto gibt es bei der Post spezielle internationale Coupons. Ich kann es auch per E-Mail senden. Wer Interesse hat, aber noch unsicher ist, kann auch noch etwas zuwarten, darf sich aber trotzdem bei mir melden, so dass ich das Interesse abschätzen kann.

Jon. Dent
Brüölring 1A
6415 Arth am See
Schweiz
E-Mail: **jondent@crosswinds.net**

ProWesS in SBASIC programmieren - Teil 2

Wolfgang Lenerz

In dem zweiten Teil dieser (Mini-)Serie zeige ich, wie man Prowessobjekte aktivieren kann. Im ersten Teil dieser Serie sahen wir, dass ein Prowessfenster aus "Objekten" besteht, welche mit der PWcreate Funktion von Prowess erstellt werden. Das erste erstellte Objekt sollte immer der Fensterumriss sein. Dieser Umriss ist nur so eine Art Behälter, dessen Zweck es ist, alle anderen Objekte (wovon einige sehr mächtig sind) zu beinhalten (oder zu "besitzen").
Nachdem der Umriss erstellt wurde, kann man andere Objekte für diesen Umriss erzeugen, die auch von diesem Umriss enthalten werden. Der ganze Set dieser Objekte ist ein System. Wurde das System erzeugt, so kann man es aktivieren, d.h. das Fenster wird auf dem Bildschirm dargestellt, und Prowess kümmert sich dann um die Mausklicks und Tastatureingaben.

Activation

Normalerweise wird nur ein Umrissobjekt aktiviert - man sollte nie versuchen, andere Objekte zu aktivieren (es sei denn, die Dokumentation erlaubt es) sonst entsteht Chaos. Das ist eigentlich auch logisch, denn, wie schon gesagt, enthält das Umrissobjekt alle anderen Objekte des Systems.
Wird das Umrissobjekt mit diesem Befehl aktiviert, so erhält Prowess die Programmkontrolle. Dann (und nur dann) erstellt Prowess das Fenster auf dem Bildschirm, mit allen Objekten, die sie per PWcreate erzeugt haben. Nachdem das Fenster erstellt wurde, passt Prowess auf, was mit dem Zeiger in dem Fenster passiert, und auch, ob der Benutzer auf spezielle Tasten drückt, usw... Genau wie in der Pointerumgebung, kann der Benutzer ein Objekt, z.B. einen Menüposten, anklicken ("HIT") oder ausführen ("DO"). Es gibt aber auch noch andere Ereignisse ("Events"), so z.B. Schlafen gehen, Zeiger kommt ins Fenster, Drag, und noch viele andere). Prowess benutzt den generellen Ausdruck ***"anzeigen"*** ("indicate") um anzugeben, dass ein Objekt irgendwie ausgewählt wurde etwas zu tun. Wir sehen später noch, dass ein Objekt angezeigt wird, wenn es die Programmkontrolle bekommt, z.B. wenn der Benutzer einen Menüposten ausführt oder anwählt.
Nachdem ein Objekt angezeigt wurde, kommt die Programmkontrolle zu Prowess zurück, und Prowess zeigt dann ihrem Programm an, warum es die Kontrolle zurück bekam, d.h. was passierte (z.B. der Benutzer klickte auf einen Menüposten). Falls sie die Pointerumgebung kennen sollten, ist das nicht viel anders als die Read Pointer Schleife unter QPTR.
Es ist keine überraschung, dass ein Objekt aktiviert wird indem man die ***PWactivate*** Funkion benutzt. Deren Syntaxe ist wie folgt:

```
memzeiger = PWactivate (objekt,memzeiger,
anz_objekt,add_info,ereignis%)
```

Die Parameter dieser Funktion haben folgenden Sinn:

- ***objekt*** ist das Umrissobjekt, das aktiviert werden soll, es wurde mit der PWcreate Funktion erzeugt.
- ***memzeiger*** ist das Resultat der Funktion, wird aber auch als Eingangsparameter der Funktion gebraucht. Es ist ein Zeiger auf einen speziellen Speicherbereich, aber der wird ganz von dem Prowess Sbasic Interface verwaltet, sie brauchen sich also gar nicht darum zu kümmern. Sie müssen nur wissen, dass dieser PARAMETER GLEICH NULL SEIN MUSS, WENN DIE FUNKTION ZUM ERSTEN MAL für

dieses Objekt benutzt wird. Danach wird diese Variable vom System verwaltet, und Sie sollten sie ganz in Ruhe lassen (oder zu mindest den Wert dierser Variablen nicht mehr ändern). Wenn jedoch der Benutzer auf den QUIT Posten des Fensters klickt, und also das Fenster ganz verlassen will, wird der Wert dieser Variablen wieder auf 0 gesetzt, was anzeigt, dass der Benutzer dieses Fester nicht mehr haben will. Dann sollte man das Umrissobjekt ganz entfernen (und, wenn es dem Hauptfenster entsprach, vielleicht das Programm ganz stoppen).

- **anz_objekt** zeigt an, welches Objekt der Benutzer ***angeklickt*** (oder ***ausgeführt***, oder sonstwie ***angezeigt***) hat, und welches Objekt also Prowess gezwungen hat, die Kontrolle wieder an Ihr Programm zurückzugeben. Mit einem SELECT aller Objekte, die der Benutzer anzeigen kann, können Sie dann den Programmfluss dahin dirigieren, wo Sie dieses Objekt behandeln können.
- **add_info** enthält additionelle Information über das Objekt welches angezeigt wurde. Genau was diese zusätzliche Information ist, kommt zum ersten auf das angezeigte Objekt an, und zum zweiten auf den Grund für die Rückkehr vom Aufruf der ***PWactivate*** Funktion. Für manche Objekte (und sogar in den meisten Fällen), enthält dieser Parameter keine verwertbare Information... Der eventuelle Inhalt dieses Parameters kann also nur später erklärt werden, wenn wir die verschiedenen Objekte und ihre Tags besprechen. Wenn so ein Tag es erzeugt, dass ein Objekt verwertbare Informationen über diesen Paramter zurückgibt, wird dies beschrieben, wenn dieser Tag für dieses Objekt behandelt wird. In allen anderen Fällen, können Sie diese Variable einfach ignorieren.
- **ereignis%** (eine Ganzzahlvariable) enthält Information, ob der Benutzer das Objekt ***ausgeführt*** hat, oder in welcher Weise dieses Objekt ***angezeigt*** wurde, denn das könnte ja wichtig sein. Zum Beispiel ist es möglich, dass das Programm anders reagieren soll, je nachdem ob ein Menüposten ***ausgeführt*** oder nur ***angewählt*** wurde. Ist der Wert dieser Variable 0, dann wurder das Objekt ***angewählt*** (linker Mausknopf, oder Leertaste). Ist der Wert 1, wurde das Objekt ***ausgeführt*** (rechter Mausknopf, oder ENTER). Diese Variable kann noch viel mehr Werte haben, das hängt aber von den Aktionsroutinen ab, welche Sie gesetzt haben, als das Objekt erzeugt wurde. Mehr davon später.

So, nachdem Sie alle Objekte erzeugt haben, wird Ihr Programm zu einer simplen Schleife, ungefähr so:

(...Hier die Definition Ihrer Objekte - es wird angenommen, Sie haben ein Objekt namens "outline" erzeugt, und ein anderes namens Item1, das ist ein loser Menüposten...)

```
mem=0:anz_objekt=0:ereignis%=0:add_info=0
REPeat loop%
  mem=PWactivate(outline,mem,anz_objekt,
  add_info,ereignis%)
  SELect ON mem
    =0      : PWremove outline:STOP
            : rem user pressed QUIT
    =Item1 : Dieses_Objekt_behandeln
    (...)
  END SELect
END REPeat loop%
```

Das ist natürlich eine ganz einfach Programmstruktur - eine einfache Schleife, die beendet wird wenn der Wert 0 in memzeiger zurückgegeben wird. Wir wissen ja jetzt, dass dieser Parameter den Wert 0 bekommt, wenn der Benutzer auf den QUIT Posten gedrückt hat, und also dieses Fenster verlassen will. Ist das Fenster auch das Hauptfenster, will der Benutzer das Programm insgesammt verlassen, also schmeißen wir auch den ganzen Umriss weg (mit PWremove, das wird auch später noch erklärt), und STOPpen das Programm. Hat der Benutzer jedoch das Item1 angezeigt, wird die Prozedur aufgerufen, die das tut, was dieser Posten auch immer tun soll.

Rückkehrobjekte definieren

Also, Prowess erstellt das Fenster, und kümmert sich dann um den Rest, d.h. überprüfen, wo sich der Zeiger befindet, ob ein Objekt angezeigt wurde, usw... Das hieße aber, dass der ***PWactivate***-Aufruf nie zurückkommt (oder nur wenn der Benutzer den QUIT-Posten anzeigte). Es muss also einen Weg geben, Prowess zu zwingen, die Kontrolle wieder an SBASIC zurückzugeben. Das passiert automatisch wenn der Benutzer das Programm verlässt (mit QUIT), in dem Fall gibt die ***PWactivate*** FunCtion 0 zurück, wie wir schon sahen.

Was jetzt noch erklärt werden muss, ist wie man Prowess zwingen kann, auch aus anderen Gründen von ***PWactivate*** zurückzukommen (z.B. wenn ein Menüposten angezeigt wurde). Das macht man, wann immer man ein Objekt erzeugt (manchmal auch wenn man es ändert). Wie schon erklärt wurde, haben die Objekttypen alle Tags, die das Verhalten des Objekts in einer bestimm-

ten Weise verändern. So hat zum Beispiel ein Umrissobjekt den Tag ***"PW('outline_quit')"***. Wenn Sie dieses Tag beim Erzeugen eines Umrissobjekts einzetzen, dann bekommt das Objekt automatisch einen QUIT Menüposten. Setzen Sie dieses Tag nicht ein, dann hat der Umriss keinen solchen Menüposten (und wie verlässt man dann das Fenster???).
Wenn Sie jetzt die Tags eines bestimmten Objekttyps anschauen (Sie finden sie im Handbuch des Prowess Sbasic Interfaces) werden Sie sehen, dass es manche Objekte erlauben, "DO routinen", "Hit routinen" oder auch andere Aktionsroutinentags beizustellen. Ein Beispiel ist das Tag PW("OUTLINE_ACTION_DO"), welches einem Umrissobjekt beigefügt werden kann. Dadurch bekommt der das Umrissobjekt auch einen DO Menüposten. Wenn Sie so ein Tag benutzen, müssen Sie auch einen zusätzlichen Parameter angeben, nämlich welche Routine aufgerufen werden soll, wenn dieses DO Objekt angezeigt wird.
Das heißt, nachdem Prowess das Fenster erstellt hat, ist Prowess selber auch nur noch eine Schleife, die überprüft was der Benutzer mit der Maus ober der Tastatur macht - er produziert dann ein "Ereignis" (oft "event" genannt). Diese Schleife, dieses Stück von Prowess ist der Prowess Event Handler (Prowess Ereignisbehandler). Wenn Sie also den DO Posten anzeigen, prüft der Prowess Event Handler, ob da auch eine Routine ist, zu der er springen sollte (nämlich die Routine, die Sie angegeben haben). Existiert so eine Routine, springt der Handler auch sofort zu dieser Routine (die in Maschinencode geschrieben sein muss). Diese Routine tut dann, was auch immer sie zu tun hat, und gibt dann die Kontrolle wieder an den Prowess Envent Handler zurück.
Für den Basic Porgrammierer wäre es natürlich am Besten, wenn man Prowess den Namen einer bestimmten SBasic Prozedur oder Funktion übergeben könnte, die dann sofort vom Prowess Event Handler aufgerufen würde. Leider ist das aber nicht möglich, da es keinen sicheren Weg gibt, eine SBasic Prozedur innerhalb einer Maschinencode Routine (wie es der Prowess Event handler ist) aufzurufen. Um dies dennoch möglich zu machen, gibt es mehrere neue Basic-Befehle, die, wenn sie vom Prowess Event Hadler aufgerufen werden, dann die Kontrolle an SBasic zurückgeben: Das sind die Befehle ***HIT_ROUTINE, DO_ROUTINE*** und noch viele andere Aktionsroutinen, welche an einer anderen Stelle erklärt werden (siehe auch das Handbuch). Sie geben einfach diese Befehle (es sind Funktionen) als Parameter an, wenn ein Tag eine Aktionsroutine braucht. Für diejenigen, die technisch mehr interessiert sind, hier im nächsten Absatz ist eine etwas vollständigere Erklärung:

Eine falsche Rückkehr

Wann immer man diese neuen Befehle einzetzt um anzuzeigen, dass Prowess zu Sbasic zurückkehren soll wenn ein bestimmtes Objekt angezeigt wurde, geschieht dies auch: Der Aufruf der ***PWactivate*** Funktion kommt zurück mit einem bestimmten Wert in der Variablen ***memzeiger*** (die dann nicht gleich 0 ist). Das ist aber eine ***falsche*** Rückkehr: Der Prowess Event Hanlder weiß gar nicht, dass die Kontrolle an SBasic gegeben wurde. Er denkt immer noch, dass das Programm in der Aktionsroutine steckt, die von ihm aufgerufen wurde, da diese Aktionsroutinen wieder nach SBasic zurückkehren ohne den Event Handler richtig zu verlassen. Wie schon oben beschrieben, sind Sbasic-Programme in Prowess nur eine Schleife, in der die ***PWactivate*** Funktion immer wieder aufgerufen wird. Beim ersten Aufruf ist der Wert des ***memzeiger*** Parameters 0 - das zeigt dem Prowess Sbasic Interface, dass es sich um eine "richtige" Aktivation handelt. Wenn man jetzt von der ***PWactivate*** Funktion zurück kommt, und diese Rückkehr wurde durch eine der Aktionsroutinen veranlasst (man ist also in Wirklichkeit immer noch im Prowess Event Handler), dann ist der Wert von Mempointer nicht 0, und der Wert dieses Parameters ist beim nächsten Aufruf von PWactivate also auch nicht null. Dann weiß das Prowess Sbasic Interface, dass es sich nicht um eine richtige Aktivation handelt, sondern um die Rückkehr von einer Aktionsroutine. Also kehrt es zum Prowess Event Handler zurück, anstatt das Objekt wirklich neu zu aktivieren.
Es ist also sehr wichtig, dass man die ***memzeiger*** Variable, die von ***PWactivate*** zurück gegeben wird, überprüft (aber nicht ändert). Ist dieser Wert 0, dann wurde ein "richtige" Rückkehr unternommen - das sieht man auch daran, dass das Fenster verschwunden ist: Wann imer eine richtige Rückkehr erfolgt, wird das Fenster welches aktiviert wurde (über seinen Umriss) zuerst entfernt.
Ist der Wert von ***memzeiger*** nicht 0, dann wurde eine falsche Rückkehr erzeugt, sie müssen dann PWactivate noch einmal aufrufen, damit eine Rückkehr von dem Event Handler gemacht werden kann. Dann sollten Sie auf keinen Fall den Wert der memzeiger Variablen ändern!
Diese (etwas umständliche) Art mit Ereignissen umzugehen erklärt auch die Schleifenstruktur im obigen Beispiel eines Sbasicprogrammes unter

Prowess: Eine einfache Schleife, worin wiederholt PWactivate angerufen wird, bis diese Funktion den Wert 0 (in memzeiger) zurück gibt. Tun Sie dies nicht, sondern rufen Sie ***PWactivate*** immer wieder neu auf (also mit memzeiger = 0 bei jedem Aufruf), dann verlieren Sie immer mehr Speicher (mindestens 4 KB bei jedem Aufruf, das ist das garantierte Minimum, in Wirklichkeit wird es aber (viel) mehr sein). Außerdem wird Ihr Fenster dann etwas komisch. Es besteht auch die (sehr große) Möglichkeit, dass das Programm, und/oder Prowess und/oder Ihre ganze Maschine abstürzt. Also, lassen Sie es!
Aus Grüden, die später erklärt werden, ist es auch keine gute Idee, die Variablen die die Objekte beinhalten, und auch memzeiger, zu LOCalen Variablen zu machen. Diese sollten immer globale Variablen sein.
Ein letzter Punkt muss noch erklärt werden, bevor wir zum ersten 'richtigen' Beispielprogramm übergehen können - das ist die Frage des Umrisses einen Fensters. Dieser Umriss kann mit PWoutln gesetzt werden. Bitte beachten Sie, dass der Umriss eines Fensters und das Prowess Umrissobjekt zwei sehr verschiedene Dingen sind!

PWoutln: Den Umriss eines Fensters setzen

Mit PWoutln setzen Sie den "Umriss" eines Fensters. In einer idealen Welt sollte dieser Befehl eigentlich nicht nötig sein, da der Umriss eines Fensters automatisch von Prowess gesetzt wird wann immer ein Umrissobjekt aktiviert wird. Dieser Befehl ist jedoch nötig, wenn man den Sbasic Interpreter benutzt (also das Programm nicht kompiliert ist).
Sie erinnern sich vielleicht, dass in der Einleitung erwähnt wurde, dass Prowess ein Window Manager ist, und also noch die Pointerumgebung (genauer gesagt, ptr_gen) braucht, denn ptr_gen zeigt den Programmen an, dass der Zeiger jetzt in Ihrem Fenster und was mit dem Zeiger passiert. Um die Pointerumgebung jedoch richtig zu benutzten, muss diese sich ***genau bewusst*** sein, wo das Fenster ist und wie groß es ist. Es braucht also bestimte Auskünfte über das Fenster.
Wenn der Umriss eines Fensters gesetzt wird, wird der Pointerumgebung diese Information mitgeteilt, und man zeigt auch an, dass dieses Fenster von der Pointerumgebung gemanagt werden soll. Normalerweise wird das automatisch über die ***PWActivate*** Funktion gemacht, sobald ein Umrissobjekt erzeugt wird, wir bräuchten uns also darüber nicht viel den Kopf zerbrechen. Eine der Eigenschaften der Pointerumgebung ist aber, dass ***der Umriss des allerersten Fensters eines Programms auf diese Weise gesetzt werden muss*** - ist das nicht der Fall, können die Umrisse der anderen Fenster des Programms auch nicht richtig gesetzt werden.
Das ist kein Problem, wenn Ihr Programm kompiliert ist ***(mit der 'WINDS' Option abgestellt)***- wir schauen uns Programmkompilation noch später an. Es ist auch kein Problem, wenn sie Ihr SBasic-Programm so ausgefüht haben (EXEC, das ist ja unter SBASIC mit Basicprogrammen möglich), dass keine Fenster beim Start auf sind. Aber im Sbasic Interpreter sind Fenster #0 ,#1 und #2 ja schon offen, bevor Sie Ihr Programm laufen lassen - und (normalerweise) ist der Umriss von Fenster #0 noch nicht gesetzt - Fenster#0 ist aber das erste Fenster des Programms. Dann benutzt man den Befehl ***PWoutln*** um den Umriss (auf englisch "outline") des Fensters #0 zu setzen. das muss für jede Kopie von Sbasic geschehen. Ist das gemacht, funktionert alles wieder gut.
Sie können das auch selber probieren, indem Sie das u.a. Beipsielprogramm laufen lassen. Setzten Sie ein REM vor den Teil, wo der Umriss des Fensters#0 gesetzt wird, und lassen Sie das Programm laufen. Sie werden sehen, dass die Fenster (mehr oder weniger richtig) erstellt werden, aber die Maus oder der Zeiger, funktioneiren nicht so ganz... In anderen Worten, zeigt Ihr Programm der Pointerumgebung nicht so richtig an, was mit dem Fenster los ist, eben weil der Umriss des ersten Fensters nicht richtig gesetzt wurde, es sei denn, sie benutzten ***PWoutln***.

Die Syntaxe diese Befehls ist wie folgt:

```
PWoutln [#chan%],x_size%,y_size%,x_orig%,y_orig%
```

und die Parameter dafür sind:

- ***#chan%*** ist die Kanal-ID des Fensters, dessen Umriss gesetzt werden muss - das sollte normalerweise Fenster#0 sein. Sie können diesen Parameter auch weglassen, dann wird Kanal#0 genommen.
- ***x_size%*** und ***y_size%*** sind die Grössen des Umrisses. VORSICHT: Alle Fenster für das Programm müssen in diesen Umriss passsen. Setzten Sie den Umriss des Fensters auf eine kleine Grösse (z.B. 50,50), können Sie kein Fenster grösser als 50,50 öffnen, Sie bekommen sonst eine Fehlermeldung 'Wert außerhalb Bereich' zurück!
- ***x_orig%*** und ***y_orig%*** sind die x und y Ausgangspunkte des (Umrisses des) Fensters.

Wenn Sie diesen Befehlt benützen, müssen Sie danach die Fenster für das Programm (Kanäle #0, #1 und #2) wieder neu definieren. Das Beispielprogramm benutzt die Prozedur "set_windows" dafür.

Das erste Programm

So, jetzt genug Theorie, wir schreiben mal das erste Prowess Programm in SBASIC. Hier ist es:

```
100 init
110 main
120 STOP
130 :
140 DEFine PROCedure init
150 REMark initialisieree/mache das Umrissobjekt
170   set_windows                  : REMark Fenster OK
180   text$='Prowess Sbasic test 1'&CHR$(0)
210   outline=PWcreate(0,PW('TYPE_OUTLINE'),PW('OUTLINE_TITLE_TEXT'),text$,
                       PW('OUTLINE_QUIT'),PW('OUTLINE_QUIT_KEYPRESS'),27 )
220 END DEFine init
230 :
240 DEFine PROCedure main
245 REMark Die Programmschleife - warte bis Ereignis
250 LOCal loop%,object,add_info,ereignis%
260   memzeiger=0:object=0:add_info=0:event%=0
270   REPeat loop%
280     memzeiger=PWactivate(outline,memzeiger,object,add_info,ereignis%)
290     IF NOT memzeiger:EXIT loop%
300   END REPeat loop%
310   PWremove outline
320 END DEFine main
330 :
340 DEFine PROCedure set_windows
350 LOCal not_compiled,upper%,xo%,yo%,xs%,ys%,ysize_0%
360   not_compiled=IS_OPEN(#0)      : REMark Fenster#0 offen?
370   IF not_compiled
380     xs%=0:ys%=0:xo%=0:yo%=0
390     PWscrsize#0,xs%,ys%,xo%,yo%
400     upper%=28:ysize_0%=50
410     PWoutln#0,xs%,ys%-upper%,xo%,upper%+yo%
420     WINDOW#0,xs%,ysize_0%,xo%,ys%-ysize_0%
430     WINDOW#1,xs% DIV 2,ys%-upper%-ysize_0%,xo%+(xs% DIV 2),yo%+upper%
440     WINDOW#2,xs% DIV 2,ys%-upper%-ysize_0%,xo%,yo%+upper%
450     BORDER#1,1,255:BORDER#2,1,255
460     PAPER#1,2:PAPER#2,7
470     INK#1,7:INK#2,2
480     CLS%0:CLS%1:CLS%2
490   END IF
500 END DEFine set_windows
```

So, was macht das Programm denn? Die ersten zwei Linien rufen die Prozeduren 'init' und 'main' auf. Es ist keine Überraschung, dass die Prozedur 'init' das Umrissobjekt initialisiert, und die Prozedur 'main' enthält die Hauptprogrammschleife. Danach hält das Programm einfach mit STOP an (der STOP Befehl ist nicht wirklich nötig, ich stelle ihn nur dahin um anzuzeigen, dass das wirklich das Programmende ist).

Die set_windows Prozedur

Die set_windows Prozedur ist das erste, was von 'init' aufgerufen wird. Set_windows ist eine Prozedur, die Sie auch in allen anderen Prowess SBasic Programmen benutzen können (und sollten). Sie stellt sicher, dass der Umriss des Fensters richtig gesetzt wird, wenn wir nicht in einem "kompilierten" Programme sind. Da es eine Prozedur ist, die in jedem Beispielsprogramm benutzt wird, werde ich sie etwas genauer erklären:
Wie Sie sehen, wird zuerst geprüft, ob Fenster#0 offen ist (Zeile 360). Dies wird mit der "IS_OPEN" Funktion erzeugt. Das ist eine neue Funktion, die im Prowess SBasic Interface erhalten ist, sie kann Ihnen anzeigen, ob ein Kanal offen ist oder nicht:

```
resultat%=IS_OPEN(#kanal%)
```

Hier ist resultat% = 1, wenn der Kanal offen ist, oder 0 wenn nicht. Bitte beachten Sie, dass das

"#" notwendig ist. Lassen Sie es aus oder geben Sie gar keine Kanalnummer an, dann wird Kanal #0 überprüft.
Es wird in set_windows wie folgt gemacht: Wenn Sie ein Programm im SBasic Interpreter normal laufen lassen ("RUN"), ist Kanal #0 das erste Fenster welches für diesen Job (d.h. den Interpreter) eröffnet wurde. Wir können also überprüfen, ob dieses Fenster offen ist. Wenn ja, müssen wir den Umriss davon setzen. Wenn nicht, wird halt angenommen, dass der Job noch gar kein Fenster auf hat. Das heißt, dass das erste Fenster das geöffnet wird, auch das sein wird, welches dem Umrissobjekt entspricht, sobald dieses Objekt mit PWactivate aktiviert wird - da braucht man dann den Umriss nicht noch zusätzlich zu setzen. Das funktioniert gut, sowohl im SBasic Interpreter (ob das Programm nun mit EXEC oder RUN gestartet wurde) als auch in mit QLiberator kompilierten Programmen.
Bitte beachten Sie, dass die set_windows Prozedur nicht richtig funktioniert, wenn Sie Kanal #0 zugemacht haben, aber noch andere Fenster aufhaben. Warum Sie das machen würden, wäre mir allerdings schleierhaft!
Ist Kanal #0 auf, dann müssen wir den Umriss für dieses Fenster setzen bevor das Umrissobjekt aktiviert wird. Aber, wie schon oben angegeben, müssen alle Fenster des Programms dann in den Umriss des ersten geöffneten Fensters passen (dieses Fenster wird oft das "primäre Fenster" genannt). Es hätte also keinen Sinn, den Umriss von Fenster#0 zu klein zu setzen, dann könnten ja allen anderen Fenster auch nur so klein sein.
Im Idealfall wäre es sogar richtig, den Umriss von Fenster #0 so gross wie den Bildschirm zu machen, dann passen allen anderen Fenster hinein. Bitte beachten Sie, dass, auch wenn der Umriss von Fenster #0 so groß wie der Bildschirm ist, Fenster #0 selber dann auch kleiner werden kann. Ich ziehe es aber immer vor, Fenster #0 nicht ganz so groß wie den Bildschirm zu machen, sondern eine Reihe von 28 Pixels am oberen Rand freizulassen, damit ich auch schön alle meine Buttons sehen kann. 28 Pixel sind genug für 2 Reihen Buttons, das reicht mir.
Also, wir wollen den Umriss von Fenster #0 so groß wie den Bildschirm machen, aber oben 28 Pixel frei halten. Hmm, und wie groß ist der Bildschirm? Heutzutage ist das gar nicht mehr so klarliegend. In der Frühzeit des QLs war der Bildschirm (in mode 4) immer 512*256 Pixel groß, aber seitdem es die QXL, Aurora, QPC und natürlich die verschiedenen Atari Emulatoren gibt, kann die Bildschirmgröse enormen Variationen unterliegen - dieser Text, z.B. wird auf einem System mit einem 800 x 600 Pixeln großen Bildschirm geschrieben.
Gut, so müssen wir zuerst herausfinden wie groß der Bildschirm ist. Man könnte natürlich voraussetzen, dass der immer eine fixe Grösse hat (nämlich die IHRES Systems), aber falls Ihr Programm auch auf anderen Systemen laufen soll, wäre das eine heikle Annahme. Es braucht also einen (universellen) Weg, um herauszufinden wie groß der Bilschirm ist. Das kann man mit dem PWscrsize Befehl erfahren, so wie es Zeile 390 des Beispielprogramms geschieht. Dieser Befehl zeigt Ihnen an, was die maximale Größe eines Fensters sein kann:

```
PWscrsize [#chan%,]x_size%,y_size%,x_orig%,y_orig%
```

Außer dem optionalen Kanalparameter (wenn er ausgelassen wird, wird Kanal #0 genommen) **werden die vier Parameter von dem PWscrsize Befehl ausgefüllt.** Sie können also einen beliebigen Wert haben, wenn der Befehl aufgerufen wird, aber sie müssen vom korrekten Typ sein (d.h. Ganzzahlvariablen).

- **#*chan%*** ist die Kanal-ID. Das würde normalerweise Kanal #0 sein. Wenn Sie den Parameter weglassen, wird automatisch Kanal #0 genommen.
- ***x_size%*** und ***y_size%*** werden auf die maximale x und y Grösse des Fensters gesetzt. Wird das für Fenster#0 erfragt, d.h. das erste Fenster des Jobs, sollte dies die Bildschirmgrösse sein. Für alle anderen Fenster, die für diesen Job dann geöffnet werden, wird es die Größe dem Umrisses von Fenster #0 sein (da ja, wie gesagt, kein Fenster größer sein kann als der Umriss von Fenster #0)
- ***x_orig%*** und ***y_orig%*** werden auf die Ausgangspunkte des Fensters gesetzt (müssten also 0 und 0 für das erste Fenster eines Jobs sein).

Bitte beachten Sie, dass die Parameter **GANZZAHLVARIABLEN** sein müssen.
In Zeile 390 bekommen wir also die Maximalgrösse des Fensters, und da angenommen wird, dass Fenster #0 das erste Fenster des Jobs ist, ist das tatsächlich die Bildschirmgrösse.
In Zeile 410 setzen wir dann den Umriss von Fenster #0 so, dass er den ganzen Bildschirm erfasst, minus 28 Pixels oben. Danach setzen wir Fenster #0, #1 und #2 auf ihre normale Grösse in SBasic. Bitte beachten Sie, dass der Umriss von Fenster#0 nicht durch die Grösse des Fensters #0 beeinträchtigt wird. Der Umriss des Fensters, und die Größe des Fensters sind zwei verschiedene Dinge: Die Fenstergröße ist das, was man

auf dem Bildschirm sieht, der Umriss ist das, was die Pointerumgebung "kennt" und "betrifft". Die Fenstergröße kann allerdings die Umrissgröße nicht übertreffen.

Die 'init' Prozedur

Jetzt haben wir also sicher gestellt, dass der Umriss des ersten Fensters, wenn nötig, gesetzt wurde, indem wir die set_windows Prozedur von der init Prozedur aufriefen (Zeile 190). Nun können wir weitergehen, und das Umrissobjekt des Fensters erstellen, mit folgendem Code in Zeile 210:

```
210 outline=PWcreate(0,PW('TYPE_OUTLINE'),
PW('OUTLINE_TITLE_TEXT'),text$,PW('OUTLINE_
QUIT'), PW('OUTLINE_QUIT_KEYPRESS'),27)
```

(das ist alles auf einer Zeile)!

Wir schauen es uns einmal etwas genauer an:
Zuerst geben wir der Funktion PWcreate einen Parameter 0. Das ist also der Besitzer das Objekts, und dass es das erste Objekt ist, das erzeugt wird, gibt es noch keinen Besitzer, also ist der Besitzer 0. Danach geben wir an, welchen Objekttyp wir erzeugen wollen mit dem Tag PW('TYPE_OUTLINE')), also eine "outline", d.h. ein Umrissobjekt.
Das nächste Tag ist spezifisch für Umrissobjekte mit dem Ziel dem Umrissobjekt beizubringen, dass es einen Titel haben soll, mit einem Text darin (PW('OUTLINE_TITLE_TEXT')), und der Text folgt dann (d.h. die Variable text$, die in Zeile 190 initialisiert wurde - bitte beachten Sie, wie da eine CHR$(0) ans Ende gehängt wurde. Das ist nötig, und wird später noch genauer erklärt).
Danach sagen wir dem Umrissobjekt, dass es einen QUIT Posten haben soll (PW('OUTLINE_QUIT')) und dass dieser Posten mit einem bestimmten Tastendruck, nämlich ESC, angezeigt werden soll - ESC ist natürlich CHR$(27).
Jetzt haben wir unser Umrissobjekt initialisiert und müssen es nur noch aktivieren. Das geschieht in der Prozedur "main".

Die 'main' Prozedur

Zuerst werden hier einige Variablen initialisiert (Zeile 260). Es ist der Hauptteil der Variablen, die dann an PWactivate übergeben werden. Diese Initiliasierung ist nicht unbedingt notwendig (da die meisten Variablen von PWactivate gesetzt werden), aber ich halte es für einen guten Programmier-Stil, und für die memzeiger Variable ist es ausdrücklich notwendig, um sicher zu sein, dass diese gleich 0 beim ersten Aufruf ist.

Dann wird das Umrissobjekt aktiviert (Zeile 280). Jetzt wird also Prowess aufgerufen. Prowess wird das Fenster zeichnen und die Ereignisse behandeln.
Da das Fenster hier mehr oder weniger leer ist (ausser dem Titel und dem Quit-Posten), passiert natürlich nicht viel. Sie können das Fenster über ESC verlassen.
Dann kommen wir zur Zeile 290. memzeiger ist dann 0, da wir dieses Fenster verlassen haben. Also gehen wir aus der Schleife raus und leeren das Umrissobjekt (da es sonst noch Speicher belegen würde).
Zugegebenermaßen tut das Programm (noch!) nicht viel. Später können wir aber darauf aufbauen, aber bevor wir das tun, müssen wir noch andere Konzepte besprechen - die Funktion "PW" und wie Zeichenketten unter Prowess behandelt werden. Ich schlage vor, wir schauen uns das mal in der nächsten Folge an.

Zusammenfassung QL Today Volume 5, Issue 3 Sept./Oktober 2000

Wolfgang Uhlig

Neuigkeiten

Erste QL TCP/IP Email verschickt!
Jonathan Dent hat die absolut erste E-Mail mit einem QL-System mit Aurora, SuperGoldKarte und SuperHermes verschickt. Herzlichen Glückwunsch, John!

PROGS ist umgezogen
Die neue Adresse ist:
Mechelbaan 344, 2580 Putte, Belgien
Die neue Telefonnummer ist: +32 15 22 23 26
Die E-Mail-Adresse ist gleich geblieben:
PROGS@triathlon98.com

Q40 unter Linux jetzt mit Ethernet Unterstützung
Der Q40 kann auf diese Weise Teil eines Netzwerks sein. Möglich geworden dank des neuen Linux Kernels 2.2.6-2. Mehr darüber unter **http://www.q40.de**

Web-Neuigkeiten und Updates

Norman Dunbar sammelt Dokumentationen über das Innenleben von QDOS und publiziert das unter: **www.Dunbar.cwc.net/qdos/qdos.html**

Thierry Godefroy hat einige neue und ge-update Software auf seiner Seite:
http://qdos.cjb.net/english/download.html
wie z.B. eine neue C68 Library, eine neue Micro Emacs Version (13.9.00) und bug fixes in FileInfo 3.41.

Dilwyn Jones hat diverse Updates von Programmen auf seiner Webseite:
www.soft.net.uk/dj/index.html
www.soft.net.uk/dj/software/other/other.html

Geoff Wicks hat einige Verbesserungen an seinem Stil-Checker vorgenommen. Zu finden auf seiner Webseite:
http://members.tripod.co.uk/geoffwicks/justwords.htm

Darran Branagh hat einen DOS 6.22 Klon ausfindig gemacht für diejenigen, die QPC1 auf einem PC installieren wollen ohne Microsoft unterstützen zu müssen. Zum Downloaden auf:
www.freedos.org

Richard Zidlicky berichtet, dass es ihm gelungen ist, gcc zum Crosscompilieren für QDOS zu nutzen. Nähere Informationen bei ihm persönlich
Richard.Zidlicky@stud.informatik.uni-erlangen.de

Händler's Reiseerfahrungen

von Tony Firshman

Tony berichtet in diesem Artikel über die Mühen und Sorgen, die man so haben kann, wenn man viel für QL-Shows und -Treffen unterwegs ist. Durch einen Autounfall in Belgien - auf dem Weg zum Kroatien-Treffen im schlimmsten Winter seit 50 Jahren - kam er in den Clinch mit seiner Versicherungsgesellschaft, die das Kleingedruckte anders interpretierte als er. In typisch trockenenglischer Art und mit viel Humor erzählt er, wie er sein Auto mit Hilfe auf dem Schrottplatz gefundener Teile selbst reparierte und letztlich nach anderthalb Jahren sogar einen richterlichen Ausspruch zu seinen Gunsten erwirkte. Das ist zumindest ein Trost für die Reise nach Kroatien, auf der er sage und schreibe eine QL Tastaturfolie verkaufte! Der Punkt ist, dass immer weniger Menschen zu den Treffen kommen und dass sie auf diese Weise einen stillen Tod sterben werden. Tonys Aufruf ist darum auch: Leute, kommt wieder einmal zu einem Treffen, es lohnt sich immer noch! Es ist ein soziales Geschehen, man kann sich in Ruhe unterhalten, Erfahrungen austauschen und vielleicht sogar gemütlich hinterher zusammen essen gehen.

Q40 Farben usw.

von George Gwilt

Der Autor beschreibt die Möglichkeiten des neuen Farbtreibers, die Informationen sind also auch für QXL- bzw. QPC-Benutzer relevant. Leider (zumindest für mich) bezieht sich das meiste auf Programmieren auf Maschinenebene, ist also vor allem für Leute gedacht, die in Assembler programmieren. Über S*BASIC findet sich nicht so viel.

You and Your Programs - Just good Friends?

Du und deine Programme - einfach nur gute Freunde?

Teil 10 - Kommerziell und Nicht-Kommerziell

In dieser letzten Folge seiner Reihe beschreibt **Goeff Wicks**, vor welchen Entscheidungen ein Händler steht, wenn es darum geht, Programme zu verkaufen. Da gibt es die eigentlich guten Programme, die jedoch kaum zu bedienen sind, oder die Programme mit guter Oberfläche, die voller Fehlern sind, Programmautoren, die hartnäckig jeden Änderungsvorschlag ablehnen, Copyrightverletzungen und viele Dinge mehr, die zu beachten sind. Sein Tipp an alle Menschen, die sich mit Programmieren beschäftigen: Wenn du ein Programm geschrieben hast, das möglicherweise auch Anderen gefallen könnte, zögere nicht, es vorzustellen, darüber zu diskutieren und flexibel zu sein. Die QL-Gemeinde ist voller Talent, das allen zugute kommen sollte. Geoffs Serie ist damit zu Ende. Er bedankt sich für das Interesse und die vielen Rückmeldungen, die er bekommen hat.

GEE Graphics! (On the QL?) Part 18

Herb Schaaf befasst sich dieses Mal mit Matrix-Operationen auf dem QL. Er hat dazu eine Reihe Funktionen und Prozeduren entwickelt, die in einem langen Listing aufgeführt werden. Es geht dieses Mal um klassische Mathematik. Angeblich kannte man in China schon vor tausenden von Jahren Matrices. Der Name "Matrix wurde jedoch erstmals 1858 erwähnt. James Joseph Sylvester und Arthur Caley entwickelten damals die Theorien, die heute noch gültig sind.

Programmieren von ProWesS in SBASIC - Teil 3

von Wolfgang Lehnerz

Da Wolfgang erfreulicherweise damit angefangen hat, seine Serie auch in der deutschen Ausgabe zu bringen, kann ich mir eine Beschreibung des dritten Teils hier ersparen.

Q40/Linux Magazin

von Tim Swenson

Der Autor hat beschlossen, ein Magazin (in elektronischer Form) herauszubringen für alle, die an Linux auf dem Q40 interessiert sind. Es wird auf seiner Website erscheinen und die Leute, die sich auf der Q40/Linux Mailing Liste befinden, bekommen Nachricht, wenn etwas Neues im Magazin erscheint. Um auf die Liste zu kommen, wende man sich an: **Bruce@qltoday.com**

Programmieren in Assembler - Teil 9

von Norman Dunbar

Der Artikel beginnt mit dem Satz: Sie können sich diesmal entspannen, dieser Teil ist ziemlich kurz! Das stimmt, es sind nur 3 Seiten, aber dafür diesmal mit einer richtigen Hausaufgabe!

Die Epson-Saga

von Jochen Merz

Jochen war immer einer derjenigen, die auf Epson-Drucker schworen, weil man als QL-Programmierer und -Benutzer prima mit den Epson-Druck-Codes, auch ESC-P2-Codes genannt, arbeiten konnte. In den letzten Jahren ist Epson jedoch immer mehr dazu übergegangen, das Druckmanagement dem Computer zu überlassen und die im Drucker eingebauten Möglichkeiten zu reduzieren. Trotzdem gibt es eine Möglichkeit für uns, diesen Engpass zu umgehen, indem wir uns mit der sogenannten Raster-Grafik-Sprache beschäftigen. Dass das gar nicht so schwer ist, will uns Jochen mit seinem Artikel zeigen. Ich hoffe doch, dass er ihn uns in Kürze auch auf Deutsch zur Verfügung stellen wird, nicht Jochen?

[Wenn Interesse besteht, werde ich diesen Artikel gerne übersetzen - laßt es mich wissen! - Editor]

Inside GoldFire - Teil 2

"Nasta" hat freundlicherweise die gegenwärtigen Spezifikationen seiner GoldFire zur Verfügung gestellt, so dass QL-User sich vorab informieren können über dieses doch immer wieder erwähnte Projekt. Der (bzw. die folgenden) Artikel ist sehr technisch und eingentlich nur für Hardware-Spezialisten wirklich verständlich. Eine Zusammenfassung ist mir (Wolfgang) jedenfalls nicht möglich.

Über Sprites

Eine Programmbeschreibung von Bruce Nicholls

Bruce hat sich das neue Programm von Jerome Grimbert angeschaut, mit dem man Sprites sowohl in den bekannten QL-Modi 4 und 8 als auch in den Farben erstellen kann, die uns der neue Farbtreiber zur Verfügung stellt. Man kann damit alte Sprites einlesen und sie in die neue Form konvertieren, man kann Programme laden und die Spritedefinitionen auslesen lassen. Neue fertige Sprites werden entweder als _pic Dateien gespeichert oder als Assemblerdatei. Die Bedienung ist einfach und übersichtlich und auch ohne Handbuch schnell zu lernen. (Leider ist nichts gesagt über die Verwendbarkeit in der EasyPtr-Suite) Erhältlich als Download unter

www.crosswinds.net/~grimbert/

Vorsicht: Bei mir bringt die Seite diverse Browser komplett zum Absturz!

Zum Titelbild

Dieses Heft ist deutlich voller geworden als geplant, doch will ich die Nachricht nicht vorenthalten: Ja, es gibt einen neuen TURBO-Compiler, der auch SMSQ/E-kompatibel ist. Dazu noch weitere Verbessungen und - das Beste von allem - er ist kostenlos!

Wir planen auch bei Interesse der nächsten Ausgabe eine weitere Cover-Disk beizulegen, auf welcher der neue Turbo und auch das neue (ebenfalls kompatible) Turbo-Toolkit enthalten sein wird.

Schön wäre, zu der Cover-Disk auch den (in der zeitgleich erscheinenden englischen Ausgabe 4) Turbo-Artikel von Dilwyn Jones und George Gwilt (letzterer hat den Compiler überarbeitet) auf deutsch übersetzt anbieten zu können (denn Leser, die nur die deutsche Ausgabe lesen, können ohne diesen Artikel bzw. die englische Anleitung nicht viel mit Turbo anfangen). Ich werd's nicht machen können, daher der Aufruf: Freiwillige vor! Ihr habt weit über einen Monat Zeit, sooo viel ist's nun wahrlich nicht. Bitte sagt mir Bescheid, daß sich niemand die Arbeit doppelt macht (obschon das wohl eher Wunschdenken ist): **JMerz@j-m-s.de**

Q60 Benchmarks

von Peter Graf

Wie von einigen geschwindigkeitshungrigen QLern gewünscht, habe ich einige Benchmarktests an meinem Q60 Prototypen durchgeführt. Die Ergebnisse sind in der Tat interessant, sodass ich sie veröffentliche, obwohl der Q60 noch nicht produziert wird. Die Frage, wann der Q60 zu kaufen sein wird, kann ich leider noch nicht beantworten. Es gibt noch Probleme mit SMSQ/E, unter anderem mit der Ansteuerung der 68060 Caches und dem bei der 68060 fehlenden MOVEP-Kommando. QDOS Classic läuft schon ziemlich gut auf dem Q60 und Linux nahezu problemlos.
Bei meinem getesteten Prototypen handelt es sich um einen Q60/80 mit einer 68LC060RC75 CPU, die mit 80 MHz betrieben wird. Außer der schnelleren CPU mit größeren Caches und verändertem Speicherinterface ist das Konzept weitgehend wie beim Q40. Das heißt, ein QL-kompatibles Mainboard mit Highcolor-Grafik, Stereo-Sound, Erweiterungsbus für ISA-Karten, Anschlüsse für die üblichen Schnittstellen, sowie Harddisk, Floppy und PC-Tastatur.
Ich habe die vier im QL Bereich wohl bekanntesten Benchmarks ablaufen lassen:
Bogomips: Eine Indikator, wieviele Instruktionen pro Sekunde verarbeitet werden können. Weltweit einer der bekanntesten Benchmarks. Verwendet wurde die Version 1.5 von Thierry Godefroy. Ergebnisse siehe Diagramm 1.

QSBB: Ein BASIC-Benchmark, der von Al Feng in QL Today Juli/August 1998 vorgestellt wurde. Die Werte für QL und QXL habe ich nicht selbst gemessen, sondern aus diesem Artikel übernommen. Achtung: Alle Werte wurden unter SMSQ/E gemessen, außer beim Original QL. Ist dem QL gegenüber etwas unfair, da BASIC ja bei QDOS viel langsamer ist als bei SMSQ/E. Ergebnisse siehe Diagramm 3.

TEST909: Ein Speed-Test für die Betriessysteme QDOS, SMSQ und SMSQ/E von Rolf Ritter. Beinhaltet Schleifen-Test, Mathe-Test, Text-Test, Sroll-Test und Grafik-Test. Angegeben ist jeweils der Gesamtwert als Faktor gegenüber dem Original QL. Achtung: Der Wert für den Q60 wurde noch bei abgeschaltetem Copyback-Cache ermittelt, ohne den Scroll-Test, da es hier noch Betriebssystem-Probleme gibt. Mit eingeschaltetem Copyback-Cache ist wahrscheinlich ein noch besserer Wert zu erwarten. Ergebnisse siehe Diagramm 4.

Folgende QL-kompatible Hardware wurde zum Vergleichen verwendet:
Original QL, 128 KB, JS ROM
GoldCard, SMSQ/E 2.89
SuperGoldCard, SMSQ/E 2.89
Q40, 16 MB, SMSQ/E 2.92
Q60, 80 MHz, 16 MB, SMSQ/E 2.92
/ QDOS Classic beta p

GoldCard	1,61
SuperGoldCard	5,78
Athlon 600 MHz (Windows NT) QPC 2	11,01
Q40	26,63
Q60/80	160,02

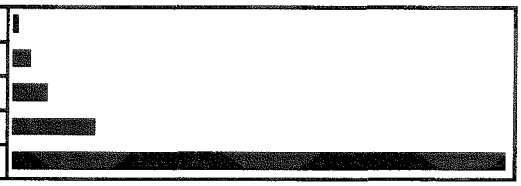


Diagramm 1: Bogomips

Dhrystones: Ein synthetischer Benchmark, der repräsentativ für die Integer-Systemperformance sein soll. Die Werte sind von der verwendeten Programmiersprache abhängig. Verwendet wurde die mit dem C-Compiler GCC komplierte Version 2.1 von Thierry Godefroy. Ergebnisse siehe Diagramm 2.

Außerdem kamen die folgenden Systeme mit Emulatoren zum Einsatz:
PC mit QXL, 20 MHz, Wert aus QL Today Juli/August 1998, näheres nicht bekannt
PC mit 600 MHz Athlon Prozessor, 196 MB RAM, Microsoft Windows NT 4.0, QPC II Demo Version 1.5, 512x256 Vollbildmodus

SuperGoldCard	6667
Athlon 600 MHz (Windows NT) QPC 2	17482
Q40	36443
Q60/80	101010

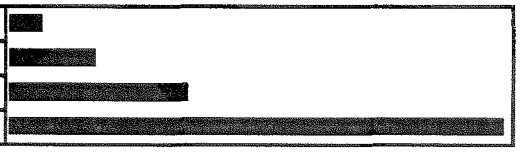


Diagramm 2: Dhrystones/s

	Print	Function	String
Sinclair QL (QDOS)	980	840	1100
GoldCard	3920	8480	20750
SuperGoldCard	7840	23700	59440
QXL	14300	27900	70640
Athlon 600 MHz (Windows NT) QPC 2	7460	65060	146580
Q40	53000	99320	249640
Q60/80	103040	317300	852400

Diagramm 3: QSBB

Unter Windows 95 konnte ich QPC II nicht testen, da der PC direkt nach dem Anzeigen des Begrüßungsbildschirms abstürzte. Eventuell ein Problem mit den Windows-Grafiktreibern, die ich kürzlich aktualisiert hatte. Zum Nachvollziehen der Benchmarks auf Q40 und Q60 gilt es zu beachten, dass ich SMSQ/E 2.92 verwendet habe. Bei aktuelleren Versionen von SMSQ/E scheint versehentlich der Copyback-Cache abgeschaltet zu sein, was hoffentlich bald behoben wird. Unter QDOS Classic kann man den Copyback Cache unter BASIC mit `COPYBACK` selbst einschalten.
Ich bin mir im klaren, dass Benchmarks nicht immer repräsentativ sind. Schließlich hat jeder User seine eigenen Anforderungen. Dennoch hat mich die gemessene Geschwindigkeit des Q60 sogar selbst überrascht. Der Q60 ist bei weitem der schnellste QL kompatible Computer aller Zeiten und auch eines der schnellsten 680x0 Systeme, die es überhaupt gibt. Auch wenn mein Test nur auf meinem eigenen Prototyp beruht, hoffe ich niemanden damit gelangweilt zu haben.

Sinclair QL (QDOS)	1,00
SuperGoldCard	11,82
Athlon 600 MHz (Windows NT) QPC 2	13,83
Q40	63,98
Q60/80 ***	171,81

Diagramm 4: TEST909

Die Postkarte

Alle QL-Händler haben im September eine Menge Geld und ihre Datenbanken zusammen geworfen und eine große Mail-Aktion an alle Kunden in Europa durchgeführt - hauptsächlich zur Einladung zu den vielen QL-Treffen im Oktober aber auch mit der Bitte, die beiliegende Postkarte an uns zurückzusenden, sofern noch Interesse an QL-Mailings besteht.
Die Rücklaufquote ist unerwartet gering, auch bei den Lesern von QL-Today. Die meisten Anzeichen lassen nicht auf Desinteresse schließen (bei vielen Personen ist es klar, **daß** Interesse weiterhin besteht), doch darum ging es nicht.
Wir hatten gehofft, mit der Rücksendung auch E-Mail-Adressen zu bekommen, um hier im Hinblick auf's Porto Kosten senken zu können.
Wir möchten eine Datenbank aufbauen von interessierten QL-Usern, die wir zu weiteren QL-Treffen einladen können. Damit dies möglichst kostengünstig geschehen kann, wäre eine E-Mail-Adresse natürlich sehr hilfreich.
Daher noch einmal die Bitte zum Zurücksenden der Postkarte - natürlich geht's auch per Fax. etc.

QL-Treffen Deutschland

Die Frage wird noch relativ häufig an mich gestellt: Wann gibt's das nächste QL-Treffen in Deutschland? Daher meine Frage zurück: besteht Interesse?
Wenn ja, wo soll es stattfinden? Hannover war ja schon öfter im Gespräch, und von der Lage her sicherlich nicht schlecht.
Organisations-Angebote gab's auch, doch möchte der/die Personen dies aus zeit- und anderen Gründen nicht alleine verantworten und durchführen.
Die Art, wie die Amerikaner Ihre Treffen durchführen (bei Voranmeldung reduzierter Eintrittspreis, mehr "Drumherum", z.B. gemeinsames abendliches Essen) - all dies fehlte bislang bei den meisten deutschen Treffen und war bislang in Amerika, England und auch in den vergangenen Jahren in Österreich erfolgreich. Und es könnte auch wieder mehr Besucher von weiter weg veranlassen zu erscheinen.
Naja, ich kann nur raten und vorschlagen, daher: Feedback, bitte!

WUNSCHZETTEL AN DEN QL-WEIHNACHTSMANN

QL TODAY NIMMT SIE GERNE ENTGEGEN UND LEITET - SOFERN MOEGLICH - AN DAS ZUSTAENDIGE CHRISTKIND WEITER!

Ich bedanke mich bei allen Autoren und Lesern für die Mitarbeit und baue auch fest in der Zukunft darauf - vielen Dank für die Treue!

Schöne Weihnachten, alles Gute und viel Erfolg für 2001,

wünscht Jochen Merz