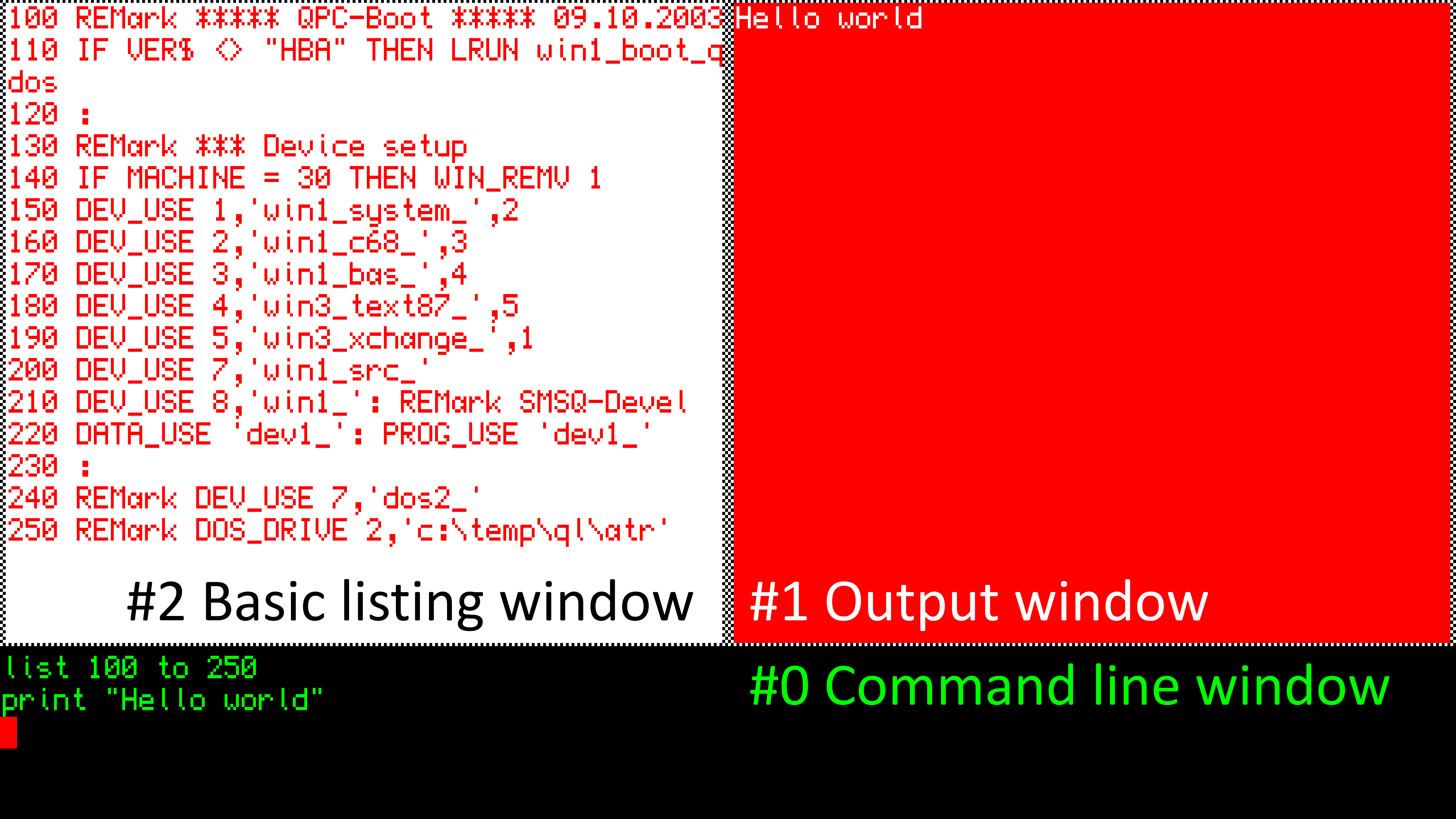
- ✓Personal introduction
- ✓History of QDOS
- ✓SMSQ/E
- ✓QPC emulator
- ✓OS internals/API
- ✓MiSTer FPGA core

AGENDA



```
100 REMark ***** QPC-Boot ***** 09.10.2003
110 IF VER$ <> "HBA" THEN LRUN win1_boot_q
dos
120 :
130 REMark *** Device setup
140 IF MACHINE = 30 THEN WIN_REMV 1
150 DEV_USE 1,'win1_system_',2
160 DEV_USE 2,'win1_c68_',3
170 DEV_USE 3,'win1_bas_',4
180 DEV_USE 4,'win3_text87_',5
190 DEV_USE 5,'win3_xchange_',1
200 DEV_USE 7,'win1_src_'
210 DEV_USE 8,'win1_': REMark SMSQ-Devel
220 DATA_USE 'dev1_': PROG_USE 'dev1_'
230 :
240 REMark DEV_USE 7,'dos2_'
250 REMark DOS_DRIVE 2,'c:\temp\ql\atr'
```

#2 Basic listing window

```
Hello world
```

#1 Output window

```
list 100 to 250
print "Hello world"
```

#0 Command line window

## PERSONAL INTRODUCTION

| | |
|---|---|
| Name: | Marcel Kilgus |
| Occupation: | Senior software guy @ Festo |
| First computer: | Sinclair ZX81 at age 7 |
| Second computer: | Sinclair QL at age 10 or so |
| Third computer: | 486DX33 – had such a guilty concience over it that in 1993 I started developing a 68000 emulator called QPC for it (age 14). Still going on! |
| Hobbies: | Reverse engineering of old stuff |
| | Hardware tinkering (MCU, FPGA) |
| | …and other stuff |

## SINCLAIR QL OPERATING SYSTEM

- Originally GST was tasked to write the OS for the QL, named 68KOS
- Another OS code-named "Domesdos" was developed in-house as a backup
- The Domesdos spec was "A version of UNIX that works"
- Ultimately "Domesdos" was selected and renamed "QDOS"
- 68KOS was still released as an add-on board, but never gained any traction

- QDOS author Tony Tebby later wrote that he could only take the technological risks he did because he knew that GST would produce an OS and so he was under no pressure to actually deliver
- First QLs were delivered with a "kludge" EPROM hanging out the back and pre-production firmware. TT claims that this was a ploy for the press to blame the software while in reality the hardware was far from finished

|  |  |
|---|---|
| Author kernel: | Tony Tebby (quit Sinclair in protest and formed QJUMP to continue writing software for the QL, now retired) |
| Author SuperBasic: | Jan Jones (now writes romantic novels) |
| Size: | 48kB ROM |
| Main features: | • True preemptive multitasking (design goal was to support up to 100 jobs)<br>• Extensible I/O system<br>• Very structured BASIC variant |
| Successors: | Minerva, SMSQ/E |

## QDOS SHORT FACTS

## QDOS VERSIONS

- Versions are identified by 2-3 letters. Rumour has those are either initials of Sinclair employees or Taxi drivers:
  - FB (1.00): April 1984
  - PM (1.01): Faster and improved MDV
  - AH (1.02): June 1984. First mass production (shipped as EPROMS)
  - JM (1.03): JM late 1984, now in ROMs
  - JS/JSU (1.10): Early 1985. Last version for UK market, JSU for US market
  - MGx (1.13): International ROMs, “x” denoting the language (e.g. MGG = German)

- Minerva is “JSL1” after the creators Jonathan Oakley, Stuart McKnight and Laurence Reeves. Versions 1.6x to 1.98
- SMSQ/E has the version “HBA”. It started with v2.00 and is now at v3.37
- -> Minerva cannot extend past 1.99 due to compatibility problems

- MUCH improved 3rd party QDOS written by Lau Reeves et al. He truly did miracles in the allotted 48kB, every trick to save a byte was employed.
- Open source (GPL). 240 source files with ~36000 LOC
- Last version 1.98a1 from me. Suffix "a1" because 1.99 is the last version before running into "2.00" compatibility problems!

- Many fixed bugs
- Much faster
- Only operating system supporting 2nd screen (which clashes with system variables)
- MultiBasic, allows several Basic interpreters to run at the same time
- Many advanced Basic commands
- Keyboard soft reset ☺
- Recommended OS for basic QDOS systems

# MINERVA█

## Toolkit II

SuperBasic as it was supposed to be (=you really NEED this!).

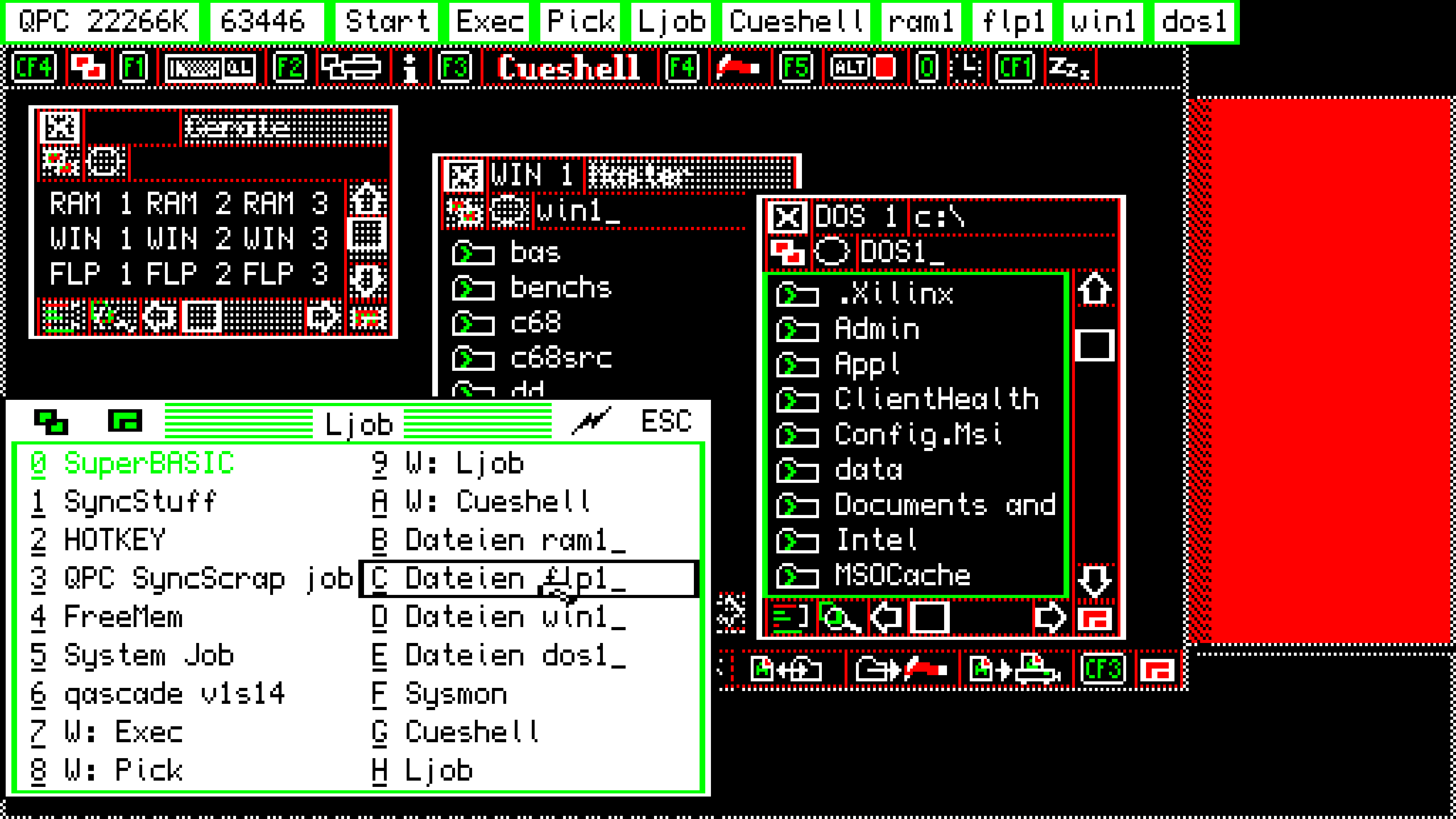
Written by QJUMP. Adds many essential Basic commands and features, including a proper screen editor or the simple feature of asking whether a file should be overwritten (previously: DELETE/SAVE necessary).

## Extended (/Pointer) environment

Also written by QJUMP. Only possible due to the extensive nature of the QDOS I/O system.

1. PTR_GEN replaces/extends the screen driver ("CON") and in this way adds non-destructible windows and mouse support
2. WMAN adds a standard windowing system (data structures, APIs)
3. HOT_REXT adds an extensible hotkey system

# ESSENTIAL ADD_ONS



```
RAM 1 RAM 2 RAM 3
WIN 1 WIN 2 WIN 3
FLP 1 FLP 2 FLP 3
```

```
WIN 1
win1_
bas
benchs
c68
c68src
dd
```

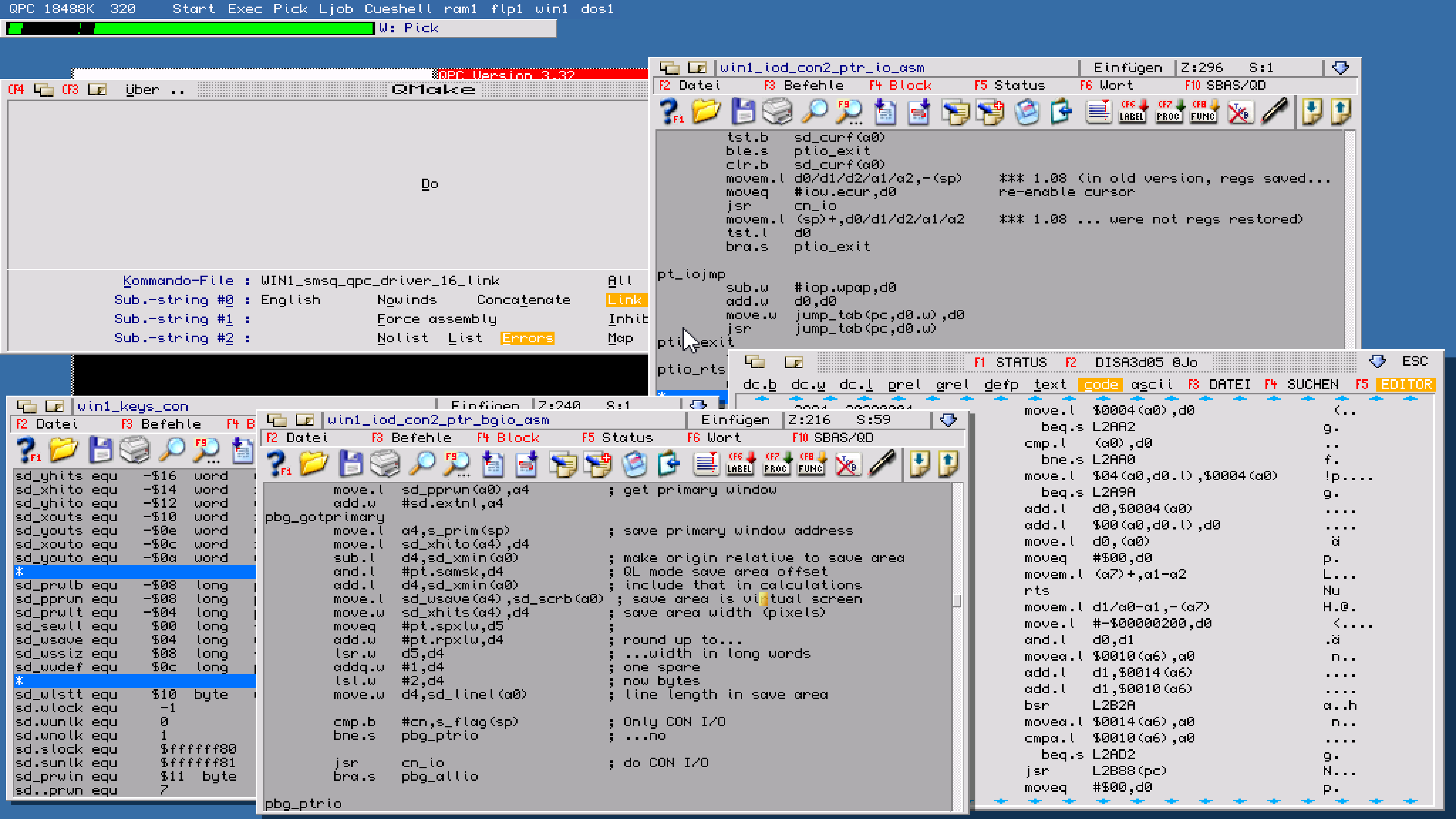
```
DOS 1 c:\
DOS1_
.Xilinx
Admin
Appl
ClientHealth
Config.Msi
data
Documents and
Intel
MSOCache
```

```
Ljob
0 SuperBASIC              9 W: Ljob
1 SyncStuff               A W: Cueshell
2 HOTKEY                  B Dateien ram1_
3 QPC SyncScrap job       C Dateien flp1_
4 FreeMem                 D Dateien win1_
5 System Job              E Dateien dos1_
6 qascade v1s14           F Sysmon
7 W: Exec                 G Cueshell
8 W: Pick                 H Ljob
```

- Originally developed by Tony Tebby. Since ~1996 also by me
- Main (sometimes only) OS for most advanced QL compatible platforms (QPC, QXL, Q40, Q68, Atari, …)
- Consists of QDOS compatible version of SMS2 + SBasic + TK2 + Extended Environment + Level 3 device drivers
- SBasic is extended SuperBasic with a compiler architecture
- Only operating system to support high colour display modes (8/16-bit)

- Open source since 2002 (proprietary license, re-licensed to BSD in 2013)
- Over 222000 LOC in 2000 source files
- 200-350kB, depending on platform
- Currently mainly developed by Wolfgang Lenerz (author of Java based SMSQmulator) and me
- I develop SMSQ/E within SMSQ/E
- Unfortunately, development coordination is still pretty-old school via eMail or forum, no Git etc. (yet)

## SMSQ/E


QPC 18488K 320 Start Exec Pick Ljob Cueshell ram1 flp1 win1 dos1
W: Pick
QPC Version 3.32
CF4 CF3 Über .. QMake
Do
win1_iod_con2_ptr_io_asm Einfügen Z:296 S:1
F2 Datei F3 Befehle F4 Block F5 Status F6 Wort F10 SBAS/QD
F1 STATUS F2 DISA3d05 0Jo ESC
dc.b dc.w dc.l prel grel defp text code ascii F3 DATEI F4 SUCHEN F5 EDITOR
win1_keys_con Einfügen Z:240 S:1
win1_iod_con2_ptr_bgio_asm Einfügen Z:216 S:59

```
Kommando-File : WIN1_smsq_qpc_driver_16_link       All
Sub.-string #0 : English     Nowinds    Concatenate  Link
Sub.-string #1 :             Force assembly          Inhib
Sub.-string #2 :             Nolist   List  Errors   Map
```

```
        tst.b    sd_curf(a0)
        ble.s    ptio_exit
        clr.b    sd_curf(a0)
        movem.l  d0/d1/d2/a1/a2,-(sp)    *** 1.08 (in old version, regs saved...
        moveq    #iow.ecur,d0            re-enable cursor
        jsr      cn_io
        movem.l  (sp)+,d0/d1/d2/a1/a2    *** 1.08 ... were not regs restored)
        tst.l    d0
        bra.s    ptio_exit

pt_iojmp
        sub.w    #iop.wpap,d0
        add.w    d0,d0
        move.w   jump_tab(pc,d0.w),d0
        jsr      jump_tab(pc,d0.w)
ptio_exit

ptio_rts
```

```
sd_yhits equ  -$16  word
sd_xhito equ  -$14  word
sd_yhito equ  -$12  word
sd_xouts equ  -$10  word
sd_youts equ  -$0e  word
sd_xouto equ  -$0c  word
sd_youto equ  -$0a  word
*
sd_prwlb equ  -$08  long
sd_pprwn equ  -$08  long
sd_prwlt equ  -$04  long
sd_sewll equ  $00   long
sd_wsave equ  $04   long
sd_wssiz equ  $08   long
sd_wwdef equ  $0c   long
*
sd_wlstt equ  $10   byte
sd.wlock equ   -1
sd.wunlk equ   0
sd.wnolk equ   1
sd.slock equ  $ffffff80
sd.sunlk equ  $ffffff81
sd_prwin equ  $11   byte
sd..prwn equ  7
```

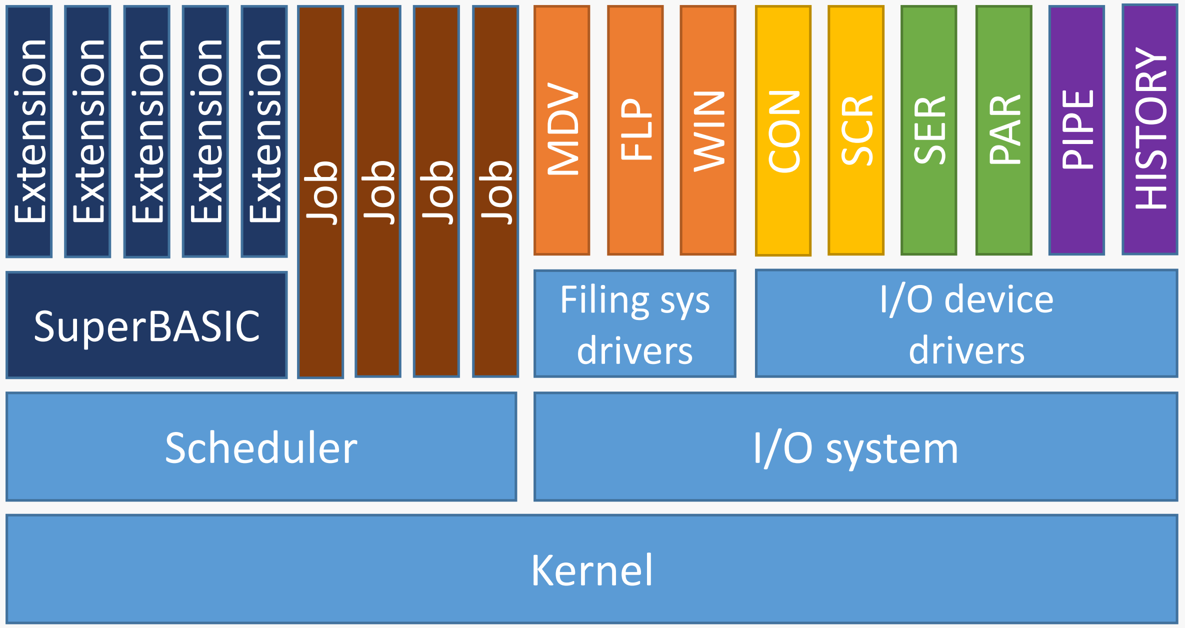
```
        move.l   sd_pprwn(a0),a4          ; get primary window
        add.w    #sd.extnl,a4
pbg_gotprimary
        move.l   a4,s_prim(sp)            ; save primary window address
        move.l   sd_xhito(a4),d4
        sub.l    d4,sd_xmin(a0)           ; make origin relative to save area
        and.l    #pt.samsk,d4             ; QL mode save area offset
        add.l    d4,sd_xmin(a0)           ; include that in calculations
        move.l   sd_wsave(a4),sd_scrb(a0) ; save area is virtual screen
        move.w   sd_xhits(a4),d4          ; save area width (pixels)
        moveq    #pt.spxlw,d5             ;
        add.w    #pt.rpxlw,d4             ; round up to...
        lsr.w    d5,d4                    ; ...width in long words
        addq.w   #1,d4                    ; one spare
        lsl.w    #2,d4                    ; now bytes
        move.w   d4,sd_linel(a0)          ; line length in save area

        cmp.b    #cn,s_flag(sp)           ; Only CON I/O
        bne.s    pbg_ptrio                ; ...no

        jsr      cn_io                    ; do CON I/O
        bra.s    pbg_allio

pbg_ptrio
```

```
move.l   $0004(a0),d0              (..
  beq.s  L2AA2                     g.
cmp.l    (a0),d0                   ..
  bne.s  L2AA0                     f.
move.l   $04(a0,d0.l),$0004(a0)    !p....
  beq.s  L2A9A                     g.
add.l    d0,$0004(a0)              ....
add.l    $00(a0,d0.l),d0           ....
move.l   d0,(a0)                    ä
moveq    #$00,d0                   p.
movem.l  (a7)+,a1-a2               L...
rts                                Nu
movem.l  d1/a0-a1,-(a7)            H.@.
move.l   #-$00000200,d0             <....
and.l    d0,d1                     .ä
movea.l  $0010(a6),a0               n..
add.l    d1,$0014(a6)              ....
add.l    d1,$0010(a6)              ....
bsr      L2B2A                     a..h
movea.l  $0014(a6),a0               n..
cmpa.l   $0010(a6),a0              ....
  beq.s  L2AD2                     g.
jsr      L2B88(pc)                 N...
moveq    #$00,d0                   p.
```

- Example of a QL/QDOS memory map
- Resident procedure area (Basic and OS extensions) is filled from top and locked once the first job is executed in the transient program area
- Basic area can move at any time! Start of area is always held in A6 and all addresses are relative to it
- SMSQ/E has lifted most of the restrictions plus has another heap that is not used for file cache (up to 256MB in size)

| | |
|---|---|
| SV_RESPR – SV_RAMT | Resident procedure area |
| SV_TRNSP – SV_RESPR | Transient program area (jobs) |
| SV_BASIC – SV_TRNSP | SuperBasic area |
| SV_FREE – SV_BASIC | Free (file cache “slave blocks”) |
| SV_HEAP – SYS_FREE | Common heap |
| $28000 – SV_HEAP | System variables and tables |
| $20000 - $27FFF | 32 KB screen |
| $18000 - $1FFFF | Internal I/O |
| $10000 - $17FFF | External I/O and ROMs |
| $0C000 - $0FFFF | ROM extension |
| $00000 - $0BFFF | QDOS |

## MEMORY MAP



- Scheduler runs at least 50 times per second (poll interrupt). More often if I/O calls are made

## ARCHITECTURE (ROUGHLY)

## I/O DRIVER FACTS █

- I/O devices parse their own names. E.g. “con_512x200a0x0_128” results in a 512x200 console window at 0x0 with 128 bytes keyboard buffer
- Inbuilt QDOS drivers: NET, SER, PIPE, CON, SCR
- SMSQ/E adds: HISTORY, NUL, SPP, QSOUND (new)
- I/O calls are given a timeout in D3:
  - -1 = “blocking”
  - 0 = “non-blocking”
  - >0 = block with timeout in 20ms steps.

- Timeout is handled by the OS, driver must return „err.nc“ (not complete) if it couldn‘t do something right away. Will be retried later until timeout runs out

- Filing system drivers are named by 3 characters. MDV = Microdrive, FLP = Floppy, WIN = Hard disc (Winchester)
- Filenames are formed as WIN1_test or FLP2_directory_file_txt
- Extension and directory separators are the same! ☹
- Support for directories was only added later (“Level 2 drivers”) and not too well (to stay compatible) ☹
- Filenames have a maximum of 36 characters (including directory!) ☹

- Every file has additional meta-data associated to it in the directory: Jobs (“EXE” files) have a special type and a “data space” value
- This makes interfacing with other file systems difficult. ZIPs can retain the values but MUST be unpacked within QDOS to restore them
- Some emulators solve this by pre-pending a header in-band and hiding it when viewed from within the system

## FILE SYSTEM FACTS

- Kernel uses TRAPs #0-3:
  - #0: Enter supervisor mode
  - #1: Kernel interface. Memory allocation, job handling, drivers…
  - #2: Open/close/delete files.
  - #3: File I/O. Delegated to driver
- Basic uses vectored routines ($00c0 - $013c) and TRAP #4. This makes address registers for next TRAP relative to A6
- No standard calling convention, interface is aimed at consumption from assembler

- Trap #4: Make address registers of next TRAP relative to A6
- System variables at address $28000 (optionally at $30000 on Minerva)
- Extension APIs often provided using the “THING system”, a standard way to find APIs and call them in a defined manner.
- Pointer environment implemented using TRAP #3 calls and further vectored routines

# OS API BASICS

- Many languages in the early days, but few were maintained long enough to still be actually usable
- Assembler:
  - GST assembler/linker or TT linker
  - GWASS (George Gwilt assembler)
  - QMake make utility
- SuperBasic/SBASIC
  - QLiberator compiler. Very compatible. Source missing, but is slowly being reconstructed.
  - Turbo compiler. Stricter than QLiberator, but also faster. By Simon N. Goodwin

- C:
  - C68: Open source compiler by Dave Walker. Non-optimizing, little C99 support and some annoying quirks. But most complete OS library
  - XTC68: Cross compiler version of C68
  - QDOS-GCC: Patch of old GCC 2.95, hard to get working these days. Uses C68 libraries
  - CLANG would be cool ;-)
- Pascal:
  - Karoly Balogh started FreePascal for QL for #QLvember 2020, which I helped to improve. But library is still very lacking

## DEVELOPMENT

- One of the (if not the) first available 68000 emulators for PCs (1996)
- Beta versions emulated a QL, but switched to SMSQ/E before release
- Does not emulate any actual hardware except the screen, all device drivers have been specifically adopted
- QPC1: 100% x86 asm (~15000 LOC)
  QPC2: 50:50 mix x86 asm and C
  QPC2 v5: rewrite of all parts except the 68000 core in C

- 68000 emulation constists of 8000-9000 lines of x86 assembler
- Only very few (5-10) bugs have been found over the last 25 years
- Pure interpreter, still was about 3 times as fast as a QL on a 486DX2-66
- v3.33 added 68020 support
- Line A emulator is OS interface

## QPC QL EMULATOR FACTS

| | |
|---|---|
| 1993: | Started development (as "PCQL") |
| 1995: | First public beta for DOS (QDOS based) |
| 1996: | First sale (SMSQ/E based) |
| 1999: | QPC2 v1 for Windows released |
| 2001: | QPC2 v2, adds windowed-mode, high-colour (16-bit) driver |
| 2002: | QPC2 v3, last major commercial upgrade |
| 2014: | QPC2 v4 became freeware |
| 2021: | QPC2 v5 major rewrite in C |

# QPC QL EMULATOR TIMELINE

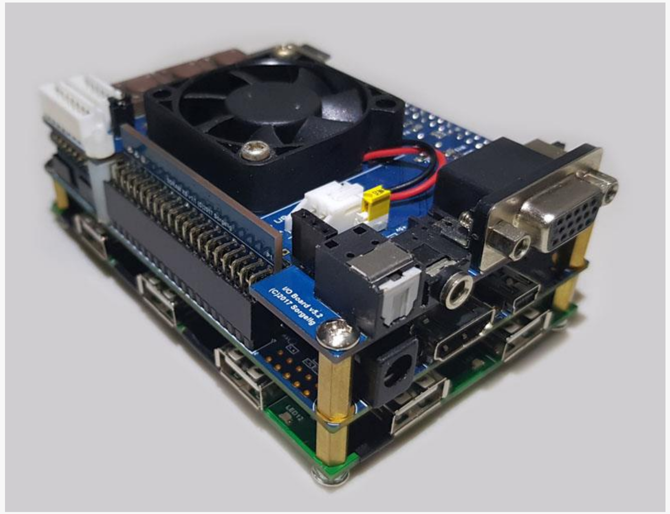
## OTHER EMULATORS

- QEmuLator: Windows and Mac based emulator of mostly standard QL, very good to play old games (free / commercial). QDOS, but can run a special version of SMSQ/E
- SMSQmulator: like QPC, but Java based and thus available on more platforms (open source). SMSQ/E only
- uQLx/sQLx: Unix based emulators written in C (open source). Emulates an extended QL. QDOS

- QLAY/QL2k: Open source emulators for DOS and WINDOWs. Limited functionality. QDOS
- ZEsarUX: Multi-platform emulator that now also supports the QL
- QDOS classic for Amiga. Open source, QDOS

MiSTER FPGA board, open source retro hardware based on the Terasic DE10-Nano board (Altera Cyclone V).

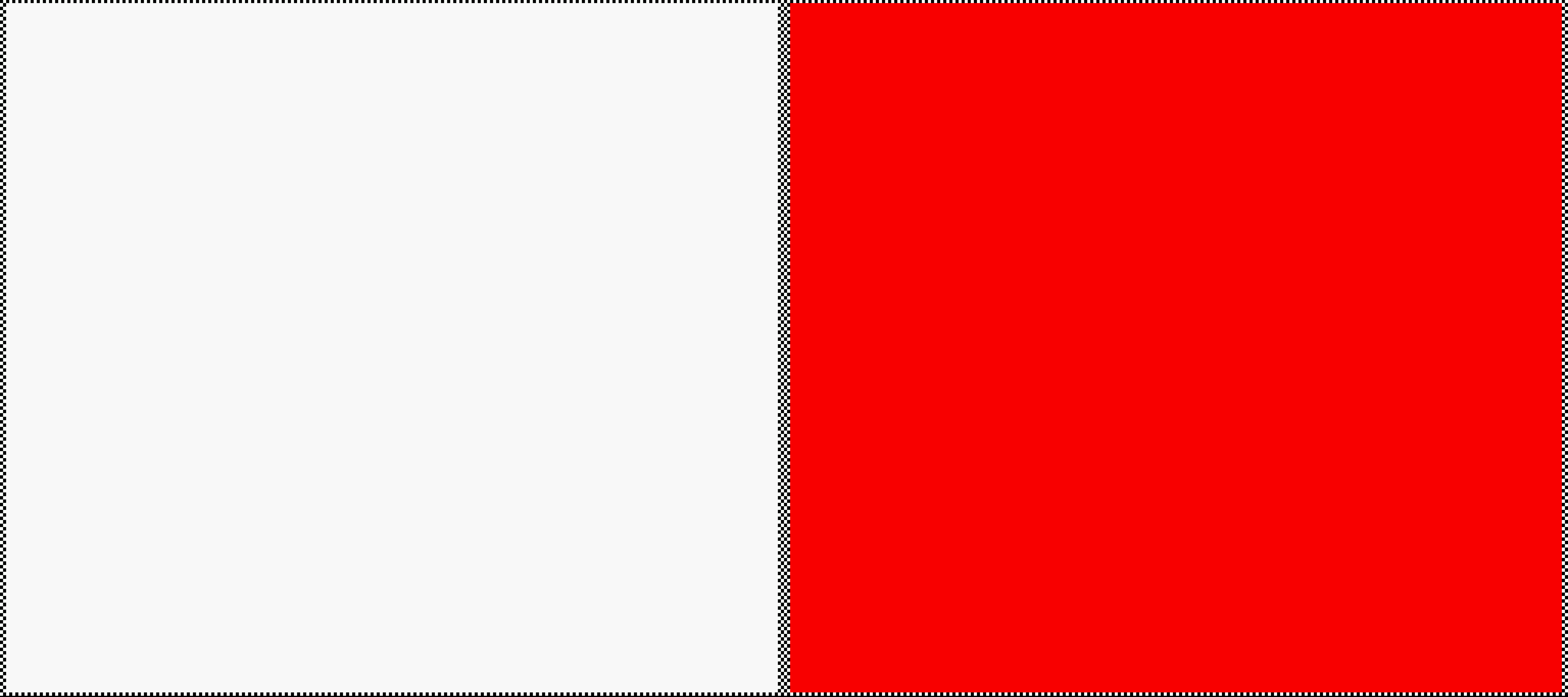
Existing QL Core based on the very limited MiST implementation.



My changes:

- Switched to cycle-exact fx68K core
- Emulation of QL speed including 8-bit bus and memory contention
- Fast speed up to 42MHz
- QL-SD support
- Support for SMSQ/E through a new “MiSTer Gold Card”
- Fixed keyboard problem through alternative co-processor firmware (“Hermes”)

## MISTER FPGA QL CORE



QUESTIONS? ‹blink›█‹/blink›