



Achttiende jaargang nr. 1
 April 1994
 85

```

na      t
        ac      ;curlo
        adc      ;$30
        jmp      output
        ;
        ;*** TESTRAM: test 1 kilobyte of RAM starting at curkilo
        ;
stram    pha      ;save
        tya      ;pricipal
        pha      ;registers
        txa      ;on
        pha      ;stack
        lda      curkilo + 1
        sta      work
        lda      curkilo
        lsr      work
        rora      ;divide
        lsr      work
        rora      ;current
        sta      datram + 1
        lda      curkilo
        and      #%00000011
        asla      ;multiply by
        clc      ;four
        adc      #$10
        sta      rcurr + 1
        ldy      #0
        sty      rcurr
        ldx      ;prepare for add
        ;add frame address high
        ;set zero page address
        ;get low
        ;set that too
        ;get odd checkerboard
        ;test 1 kilobyte
        ;if error, exit
        ;checkerboard
    
```

In dit nummer o.a.:

Wat is DPMS?
Cursus C: alweer les 5
Norton Utilities 8.0 besproken
4DOS 5.0: Nog mooier dan vroeger
DOS65: RAMkaart testen en bestellen

Inhoudsopgave

De μ P Kenner

Nummer 85, april 1994
Verschijnt 5 maal per jaar
Oplage: 250 stuks
Druk: FEBO Offset, Enschede

De redactie:

Nico de Vries
Gert van Opbroek
Geert Stappers

Eindredactie:

Nico de Vries

Vormgeving:

Nico de Vries

Redactieadres:

p/a Nico de Vries
Van der Waalsstraat 46
2984 EP Ridderkerk

De μ P Kenner nummer 86 verschijnt op
20 augustus 1994.

Kopijsluitingsdatum voor nummer 86 is
vastgesteld op 6 augustus 1994.

Vereniging

Uitnodiging voor de clubbijeenkomst	5
Voortgang KGN68k (deel 15)	30
Van de voorzitter	41

Algemeen

Redactioneel	4
PostScript Verschoven	9
De Computerwetten van Murphy	25

Talen/Software

Select schakelaar voor Video card in MS-DOS hardware	6
Spuut DOS-geheugen nieuw leven in	7
Vijfde les 'C'	31
Afrekenen met DOS	37
Gearriveerd: 4DOS versie 5.0	39

Hardware/DOS65

DOS65 1 MByte RAMkaart: printen en prijzen	8
Bijtjes pesten of bytjes testen?	11

De μ P Kenner is het huisorgaan van de KIM gebruikersclub Nederland en wordt bij verschijnen gratis toegezonden aan alle leden van de club. De μ P Kenner verschijnt vijf maal per jaar, in principe op de derde zaterdag van de maanden februari, april, augustus, oktober en december.

Kopij voor het blad dient bij voorkeur van de leden afkomstig te zijn. Deze kopij kan op papier, maar liever in machine-leesbare vorm opgestuurd worden aan het redactieadres. Kopij kan ook op het Bulletin Board van de vereniging gepost worden in de redactie area. Nadere informatie kan bij het redactieadres of via het bulletin board opgevraagd worden.

De redactie houdt zich het recht voor kopij zonder voorafgaand bericht niet of slechts gedeeltelijk te plaatsen of te wijzigen. Geplaatste artikelen blijven het eigendom van de auteur en mogen niet zonder diens voorafgaande schriftelijke toestemming door derden gepubliceerd worden, in welke vorm dan ook.

De redactie noch het bestuur kan verantwoordelijk gesteld worden voor toepassing(en) van de geplaatste kopij.

Redactioneel

Het sprokkelen is weer gebeurd. Het nummer is weer redelijk gevuld. Beetje veel listing misschien. Maar daar is het dan ook de uP Kenner voor natuurlijk. Het blijft een probleem: kopij. Ik weet het: we zeuren er al jaren om. Maar toch blijven we zeuren. In het belang van de lezer. En dat bent uzelf.

Dus: als straks gedurende de zomer uw solar-cel gestuurde nikkel-cadmium ontlader/kool-zink bespaarder na 27 jaar van proefnemingen de eerste verschijnselen van functionaliteit begint te vertonen, aarzel dan niet om daar een verhaaltje over te tekstverwerken en dit op het BBS "The Ultimate" te deponeren.

Daarna krijgt de redactie de puzzle om erachter te komen in welk bestandsformaat het tekstje staat (we hebben nog steeds het liefst WP4.2, WP5.0, WP5.1 of plat ASCII, maar op alles wat de mensheid aan tekstverzieker-formaten heeft voortgebracht bijten we graag ons gebit nog een keer stuk). Vervolgens mag u ook proberen de layouter grijze haren te bezorgen door nieuwe uitdagingen in de tekst op te nemen. Een voorbeeld hiervan is de ogenschijnlijk ietwat onschuldige figuur bij de aardverschuivingen in PostScript, waar uw uitleg-slaaf ongeveer een hele avond door van de straat is geweest!

Zowel bij de KGN-68k als bij DOS65 komt er schot in de zaak. Bij DOS65 kunnen de soldeerbouten eenvoudig uit de kast: de printen kunnen worden besteld.

De prototypes van de 1 Mbyte RAMkaart voor DOS65 vertonen allemaal een ziekelijk gedrag: ze blijken direct de eerste keer betrouwbaar te werken. Inmiddels is er ook een programmaatje gebakken om al die 8 miljoen en nog en paar bitjes te pesten en te teisteren. Een beetje 6502 is daar toch wel even

bijna 7 minuten met druk. Eigenlijk best wel snel: 1 Mbyte testen in een minuut. De test is namelijk uitgebreider dan die op de gemiddelde PC. Kijk maar in dit nummer.

Bij de boys met de 32-bits in het gebit gaat het ook beter, lees het verhaaltje over hun voortgang maar eens. Bij hen begint de RAMP langzaam aan een RAM te worden, en kunnen ze met verfrist gemoed de refresh gaan testen. Hoeveel verfrissingen het systeem precies doet is nog niet duidelijk, maar ik hoop dat het ook met het warmere weer van de laatste dagen ruim voldoende zal zijn...

Ook op C-gebied maken we vorderingen: inmiddels zit deel 5 van de cursus in dit nummer geflanst. Lachen, gieren en brullen lukt ook deze keer. Gratis, dankzij de heer Murphy. U kent hem wel. Of nog niet soms?

Lachen, gieren en brullen lukt ook deze keer.

Even iets anders. De bijeenkomst is in Haarlem, dus niet in Almere of zo. Normaal zitten we na de zomer pas in Haarlem, maar deze keer is het anders. Erwin Visschedijk komt ons allemaal uitleggen hoe hij per 6502 ontsporingen en aanrijdingen van modeltreintjes denkt te kunnen voorkomen. De dienstregeling vindt u in de uitnodiging. En: als uw buurman toevallig treintjesgek is, moet u hem vooral meenemen. Het buurman heeft het namelijk op die buurman gemunt. Als lid weliswaar. De buurman mag als introduc  e zelfs gratis naar binnen. En de buurvrouw mag natuurlijk ook mee.

Rest mij u het weer te wensen dat bij het jaargetijde past, natuurlijk ook een prettige vakantie, en soldeer ze met DOS65!

Nico de Vries

Uitnodiging voor de clubbijeenkomst

Datum: 28 mei 1994
 Locatie: Wijkcentrum "De Ringvaart"
 Floris van Adrichemlaan 98
 2035 VD Haarlem
 Telefoon: 023-363856

Thema: Treinbesturing met microcomputers
 Entree: gratis, ook voor niet leden, dus breng uw
 buurman eens mee!

Programma:

9:30 Zaal open met koffie
 10:15 Opening door de voorzitter
 10:30 Voordracht door Erwin Visschedijk over
 de door hem ontworpen en gebouwde
 modeltreinbesturing met behulp van
 single chip microcontrollers en DOS65.
 12:00 Forum en markt
 12:15 Lunchpauze, consumpties tegen betaling

Routebeschrijving

Auto:

Komende uit de richting Utrecht, Amersfoort of
 Rotterdam: Afslag Haarlem-Zuid; tweede stoplicht
 links; bij de tweede kruising met stoplichten links;
 Floris van Adrichemlaan.

Komende uit de richting Alkmaar: afslag Haarlem-
 Zuid; verder zie boven.

Openbaar vervoer:

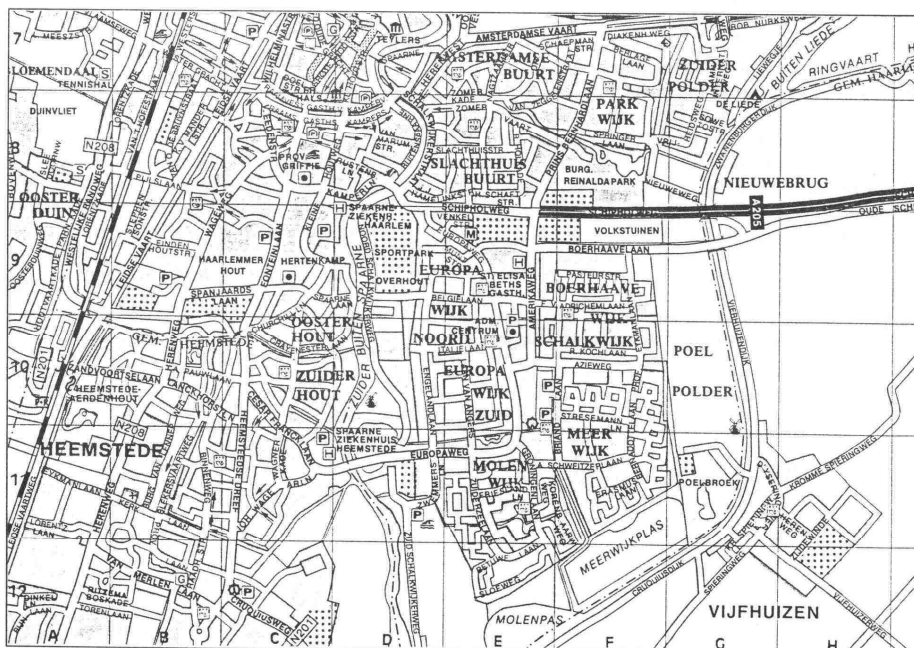
Vanaf het station Haarlem met buslijn 7, 71, 72 of
 77; halte Floris van Adrichemlaan.

Aansluitend het informele gedeelte met de moge-
 lijkheid om andermans systeem te bewonderen en
 niet-commerciële software uit te wisselen. U wordt
 uiteraard van harte uitgenodigd uw eigen systeem
 mee te nemen.

17:00 Sluiting

Let op

Het is ten strengste verboden illegale kopieën van
 software te verspreiden. Aan personen die deze
 regel overtreden zal de verdere toegang tot de bi-
 jeenkomst ontzegd worden. Breng verder alleen
 software mee die u legaal in uw bezit heeft. Het be-
 stuur aanvaardt geen enkele aansprakelijkheid voor
 de gevolgen van het in bezit hebben van illegale soft-
 ware.



Select schakelaar voor Video card in MS-DOS hardware

In het volgende stukje wil ik een probleem en een gerealiseerde oplossing voorleggen. Achteraf bleek het simpel, maar ik ben er toch lang mee bezig geweest. Op de vorige regels zitten (ook) geen auteursrechten en zijn dan ook vrij te gebruiken voor vele andere artikelen voor in de μ P Kenner.

Situatie omschrijving

MS-DOS machine, Hercules card, monochrome (hercules) monitor. Een VGA kaart en 1 VGA monitor die de meeste tijd aan een Macintosh zit.

Het probleem

Als de VGA kaart in de machine zit, gaat de informatie via die kaart naar buiten. De VGA kaart wordt bij Power On Self Test (POST) herkend en blijft dan de 'primary display'. Setup van het BIOS veranderen helpt niet. Als ik Linux draai heb ik aan het Hercules-scherm genoeg. Wil ik gaan print layouten moet ik eerst de VGA kaart er in schroeven. De rest van de week is het weer Linux en moet de VGA kaart er weer uit, want anders wordt het hercules-scherm niet gebruikt c.q. herkend.

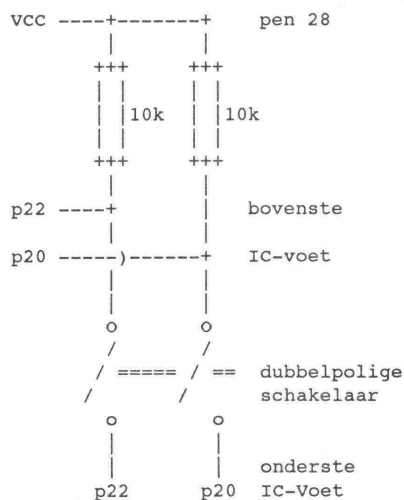
Verzamelde informatie

Het bleek dat VGA-kaart & Hercules-kaart er tegelijkertijd in mogen zitten. Wisselen hoeft dus niet. VGA erin of eruit is dus al genoeg. Onder MS-DOS zijn er zelfs programma's die het ondersteunen. Een vriend van mij ontwikkelt/debugt er software mee. Het VGA-scherm is dan voor de applicatie en het hercules-scherm bevat de debug-informatie. In "de IBM-PC en zijn klonen" beschreef Nico de Vries ook de POST. Een van de zelftests is het zoeken van andere BIOS-ROMs. De VGA kaart heeft een BIOS-ROM. Toen getest: BIOS ROM van VGA kaart verwijderd, zat toch in een voetje. VGA kaart zonder BIOS in machine gezet. Het beeld verscheen op de hercules-monitor!

Gerealiseerde Oplossing

De truc is te voorkomen dat het BIOS op de VGA kaart herkend wordt. Op het moment dat de chipse-

lect er niet meer komt, is het doel al bereikt. Om op zeker te spelen heb ik zowel het Chip Enable signaal als het Output Enable Signaal gewijzigd. De modificatie heb ik in een stapel (3) IC-voeten aangebracht. Uit de middelste is pen 20 & 22 verwijderd. Bij de bovenste voet zitten aan pen 20 & 22 ieder een weerstand van 10kohm, die aan de andere kant aan pen 28 (VCC) zitten. Van de onderste vertrekken vanuit pen 20 & 22 draden naar een dubbelpolige schakelaar. Van de schakelaar gaan weer draden naar 20 & 22 van de bovenste IC-voet.



Is de schakelaar open, dan winnen de pull-up weerstanden en komt er geen chipselect. Is de schakelaar gesloten, dan komt er wel een chipselect en herkent de POST wel het BIOS van de VGA-kaart.

Geert Stappers

Spuut DOS-geheugen nieuw leven in

Netwerk-gigant Novell introduceert een nieuw product dat de DOS-geheugengrens van de 640 KB weer wat verlegt. Dankzij DPMS kunnen aansturingsprogramma's en residente software in het 'extended geheugen' worden ondergebracht.

De ontwikkelaars bij Novell bieden ons een nieuwe techniek om de 640KB, de UMB's en de HMA nog meer te ontlasten. Zij hebben de zaken grondig aangepakt: in plaats van de TSR's en de drivers van randapparatuur onder te brengen in UMB's of in HMA, installeren zij deze direct in XMS. Daar zijn immers nog megabytes in overvloed beschikbaar. Deze techniek gaat door het leven als 'Dos Protected Mode Services' (DPMS).

Dankzij de DPMS kan men drivers en residente programma's ontwerpen die in de 'protected modus' werken op een 286- of 386-pc. Het principe is eenvoudig. DOS voelt zich onwennig in protected modus. Om er toch gebruik van te kunnen maken, moeten ontwikkelaars doorgaans gebruik maken van een zogenoemde 'DOS extender'. Deze legt een verbinding tussen DOS in de real modus en het toepassingsprogramma dat in de protected modus draait. Spijtig genoeg nemen deze kleinoden veelal verscheidene honderden KB's geheugen in beslag en zijn zij niet ontworpen voor het laden van drivers en TSR's.

Toen kwamen Novell-ingenieurs op het idee een sterk afgeslankte versie van een DOS extender te ontwerpen. Die werd eenvoudiger, kleiner en meer afgestemd op residente programma's en drivers voor randapparatuur.

In feite is DPMS een resident programma dat zichzelf in XMS nestelt en slechts een heel klein gedeelte toegankelijk houdt voor DOS, in het gewone geheugen of in een UMB. Het produkt dat Novell ons aanbiedt, is tweeledig: één aspect is voor de gebruiker, het andere richt zich op de ontwikkelaar. De gebruiker heeft wat aan het programma DPMS.EXE. Het betreft een uitbreiding van het besturingssysteem verantwoordelijk voor het hierboven vermelde fraais. Het presentje voor de ontwikkelaar bestaat slechts op papier. Het is de documentatie die ontwikkelaars moet toelaten programma's te schrijven die DPMS aanspreken. De SDK DPMS

(Software Developer's Kit voor DPMS) bestaat uit een handleiding van een dertigtal pagina's, een diskette met SPMS.EXE, een voorbeeldprogramma, een bestand voor verwerking in assembler-taal en een speciale versie van EMM386.EXE.

Na beëindiging van de installatie voegt DPMS een API (Application Program Interface) toe die toegankelijk is via interrupt 2Fh, voorzien voor dit soort toestanden. Via deze geïnstalleerde API kunnen programma's die gebruik willen maken van dit XMS-geheugen om er code of gegevens in onder te brengen, beschikken over alle noodzakelijke diensten.

Eenvoudig en tegelijk complex

Voor de gebruikers is DPMS gesneden koek. Het volstaat DPMS.EXE toe te voegen aan het Dos systeembestand CONFIG.SYS. De rest gebeurt vanzelf. Maar de ontwikkelaar komt er niet zo goedkoop vanaf. U hebt het reeds geraden: om van DPMS gebruik te kunnen maken, moet het programma daarop zijn ingesteld. Er is dus geen sprake van dat drivers die niet in UMB's kunnen worden ondergebracht voortaan dan maar in XMS terecht komen. Men zal zich nieuwe drivers dienen aan te schaffen, drivers die conform de DPMS-specificaties zijn

geschreven.

De ontwikkelaars zitten daar niet bepaald op te wachten. DPMS is inderdaad moeilijk te implementeren. De geboden diensten zijn zeer beperkt: alleen het strikt noodzakelijke, geen byte meer. Om gebruik te kunnen maken van DPMS, zal de ontwikkelaar ingewijd moeten zijn in de geheimen van assembler en van de protected modus. Tevens zal hij geen gebruik meer kunnen maken van een hele reeks programmeertechnieken die slechts in de real modus werken. Dat is de prijs die de ontwikkelaar betaalt voor toegang tot het gehele werkgeheugen.

De server en zijn klanten

DPMS werkt volgens het client/server principe. De server is DPMS, de klanten zijn de 16 bits .EXE of .COM programma's. Zij worden aangemaakt met de gewone programmeerhulpjes, in assembler of in een hogere taal.

Dankzij DPMS kunnen aansturingsprogramma's en residente software in het 'extended geheugen' worden ondergebracht.

Als men gebruik maakt van gewone hulpmiddelen, dan komt dit vooral doordat DPMS het gros van het werk overlaat aan de programmeur. Zo is een programma dat gebruik maakt van DPMS een gewoon programma dat geheel of gedeeltelijk ondergebracht wordt in XMS geheugen. De verplaatsing binnen het geheugen gebeurt in feite niet door DPMS, maar door de programmacode zelf. Vandaar de complexiteit van het probleem.

De programmeur zal moeten bepalen of DPMS wordt geïnstalleerd. Hij zal DPMS moeten oproepen om in de protected modus segment-descriptoren toe te kennen, XMS geheugen toe te wijzen, nog eens DPMS oproepen om de descriptoren in te stellen (basisadres, grenswaarden en types). Pas dan kan het programma de real modus-segmenten onderbrengen in het daartoe ingestelde geheugen. DPMS maakt wel een kopie, maar het is het programma zelf dat de vertaling van real modus naar protected modus en omgekeerd zal moeten voorzien. Wanneer alles klaar is in protected modus, hoeft de ontwikkelaar nog slechts het in real modus vrijgemaakte geheugen vrij te maken en de uitvoering van het programma verder te zetten.

Een ontwikkelaar die een programma met DPMS ondersteuning wil ontwerpen, staat er zo'n beetje alleen voor. Het momenteel enige beschikbare hulpmiddel is WDEB386, dat deel uitmaakt van de Windows-development kit. Deze debugger is beter dan niks, maar kan hoegenaamd niet tippen aan producten als CodeView of Turbo-Debugger.

De oplossing voor de toekomst?

DPMS is compatibel met MS-DOS, DR-DOS, Novell-DOS en Windows; tenminste met de huidige versies van die programma's. DPMS is een interessante techniek die nog maar eens het onvermijdelijke uitstelt, namelijk de overgang naar een 32 bits bestuurs-

systeem zoals OS/2, Windows NT of het langverwachte Windows 4 (codenaam Chicago).

De 32 bits modus is de enige afdoende manier om uit het keurslijf te breken en om schoonheid te maken met het verleden.

Voor u gelezen in CM Corporate: een artikel van Francois Piette: Novell DPMS beta doorgelicht

DOS65 1 MByte RAMkaart: printen en prijzen

Of: voortgang werkgroep DOS65

Elders in dit nummer staat een testprogramma voor de 1 Mbyte RAMkaart voor DOS65. Er is dus vooruitgang geboekt. Er is nog meer vooruitgang: de printen voor de kaart zijn in bestelling. Daarbij heeft de werkgroep in goed onderling overleg een veer gelaten.

De bedoeling is altijd geweest om de leden een dubbelzijdige, doorgemetalliseerde, van soldeermasker en opdruk voorziene print aan te bieden. Een tweede bedoeling was ook om de print niet veel duurder te maken dan ongeveer f 80,-. Deze twee wensen kunnen we helaas niet realiseren, zonder een onverantwoorde belasting te leggen op de gezonde financiële situatie binnen de club.

Daarom is besloten de printen aan te bieden zonder soldeermasker en componentenopdruk. De werkgroep was het hier unaniem over eens, en een aantal leden die we hierover hebben aangesproken vond het ook prima: de reeds gepubliceerde componentenopdruk is duidelijk genoeg en een beetje soldeer-kunst is ook voldoende voorhanden. Dus: dezelfde printen als de prototypen, want veranderingen zijn niet nodig gebleken.

De prijzen voor de diverse spullen bevatten ook een bijdrage aan de clubkas, om volgende projectjes op te kunnen zetten, en voorraad printen te kunnen financieren. Het lijstje is nu:

Print	f	80,00
Dallas DS1210	f	10,00
GAL22V10 (geprogr.)	f	15,00
8kx8 SRAM, 20 of 25 ns	f	8,00
128kx8 CMOS SRAM	f	30,00

De laatste prijs is een dagprijs. Afwijkingen groter dan 5 procent zullen worden doorberekend, zowel positief als negatief. Bij iedere print wordt een 5,25" diskette geleverd met daarop de source van VTEST en de JEDEC-file voor de GAL/PAL.

Bestellen

U kunt de spullen als volgt bestellen: maak een lijstje van wat u hebben wilt en tel de bedragen op. Vermeld op uw lijst ook uw giro- of bankrekening-nummer, zodat we een eventueel overschot (bij een prijsdaling van de SRAMs) kunnen terug storten. Stuur het lijstje naar Nico de Vries en maak het benodigde bedrag over op de girorekening van de club: 3757649. Binnen zes weken zullen de bestelde spullen worden geleverd.

Nico de Vries

PostScript Verschoven

Onlangs hadden we een print ontworpen van 32 bij 10 centimeter. Een afmeting die niet op een A4 past. Voor de printboer is dat geen bewaar, maar voor de dokumentatie wel. Als we bij het postprocessen het ontwerp 90 graden draaiden en dan afdrukten op 80% van de ware grootte paste het wel. De componentwaarden van de SMT-devices waren bij schaal 1:1 niet eens duidelijk te lezen en bij 0,8 keer vergroot helemaal niet meer. Bij een vergroting van 1,5 keer was maar een gedeelte van de print duidelijk te lezen. Er moest dus verschoven worden. Het post processing programma biedt wel die mogelijkheid maar dat was nogal een tijdrovende zaak. Eerst moest de post processor van het printontwerppakket er aan rekenen en dan de PostScript printer nog een keer. Voor het verschoven stuk moesten ze beiden weer aan het rekenen. Omdat de eerste verschuiving niet goed ging, begon het feest dan weer van vooraf aan.

Tijd om in de PostScript file de verschuiving uit te voeren. De post processor was al bezig geweest om de vergroting uit te rekenen. Die file hebben we een editor binnen gehaald en gezocht naar de text **translate**. Volgens het Engels woordenboek is *translate* onder andere *vertalen*. In PostScript is het verschuiven, zie ook in het kader op de volgende bladzijde.

Op die regel stond **28.8 28.8 translate**. De oorsprong die eigenlijk helemaal links onder in de hoek zit, wordt daar door uit de hoek geschoven. Maar wij wilden hem juist aan een kant omlaag schuiven. Met de editor is er het volgende van gemaakt: **28.8 28.8 -300 add translate**. **add** is voor de verschuiving uit te rekenen, **-300** de verschuiving. Ter herinnering, PostScript gebruikt de Reversed Polish Notation (RPN, HP rekenmachine). File bewaard

en naar de PostScript printer gestuurd. He, jammer, niet genoeg verschoven, maar wel al vast de juiste kant op. Gewijzigd in **28.8 28.8 -600 add translate**, opnieuw naar de printer. Nog net niet helemaal, maar wel sneller resultaat, omdat het postprocesprogramma er niet meer aan hoeft te rekenen.

Opnieuw in de editor. Toen bleek dat op de volgende regel **0.06 0.06 scale** stond (Zie ook het kader op de volgende bladzijde). Dan werd er dus nog waarschijnlijk in de default units gewerkt van 1/72 inch. Nog eens naar de vorig twee uitdraaien gekeken. Ja hoor, het was een verschil van iets meer dan 4 inch. Een A4 is wat langer dan 11 inch, we willen ook nog een stuk overlap hebben, laten we 10 inch opschuiven. **28.8 28.8 72 -10 mul add translate**. Save, naar de printer en jawel hoor, dat is wat we hebben wilden.

Als het printed circuit board nu breder geweest was hadden we bijvoorbeeld 5 inch naar links moeten schuiven, **28.8 72 -5 mul add 28.8 translate**. En met **28.8 72 -5 mul add 28.8 72 -10 mul add translate** de overgebleven hoek op papier gekregen. En dat met een en dezelfde PostScript file als uitgangspunt.

Geert Stappers

Literatuur:

1. PostScript Language Reference Manual, SECOND EDITION
ADOBE Systems Incorporated
Addison-Wesley Publishing company, Inc.
ISBN 0-201-18127-4

translate t_x t_y translate - translate builds a temporary matrix $T = \begin{Bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ t_x & t_y & 1 \end{Bmatrix}$ and concatenates this matrix with the current transformation matrix (CTM). Precisely, translate replaces the CTM by $T \times \text{CTM}$. The effect of this is to move the origin of the user coordinate system by t_x units in the x direction and t_y units in the y direction relative to the former user coordinate system. The sizes of the x and y units and the orientation of the axes are unchanged. add num_1 num_2 add sum returns the sum of num_1 and num_2. If both operands are integer and the result is within the integer range, the result is an integer; otherwise, the result is a real.	example 3 4 add $\Rightarrow$ 7 9.9 1.1 add $\Rightarrow$ 11.0 scale s_x s_y scale - scale builds a temporary matrix $S = \begin{Bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{Bmatrix}$ and concatenates this matrix with the current transformation matrix (CTM). Precisely, scale replaces the CTM by $S \times \text{CTM}$. The effect of this is to make the x and y units in the user coordinate system the size of s_x and s_y units in the former user coordinate system. The position of the user coordinate origin and the orientation of the axes are unchanged. mul num_1 num_2 mul $product$ returns the product of num_1 and num_2. If both operands are integer and the result is within the integer range, the result is an integer; otherwise, the result is a real.
--	--

Een stukje Postscript-uitleg

Ik heb interesse in de KGN en wil

☐ Lid worden van de KGN

☐ Meer informatie over de KGN

Naam : _____

Adres : _____

Postcode en Woonplaats : _____

Datum : _____ Handtekening : _____

Dit strookje kunt u ingevuld opsturen aan het secretariaat van: KIM Gebruikersclub Nederland
Postbus 1336
7500 BH Enschede

Bijtjes pesten of bytes testen?

Hieronder staat de source voor het programma VTEST. Het programma is oorspronkelijk gemaakt om het prototype van de nieuwe 1 Mbyte RAMkaart te testen. Het is niet "mooi" geschreven: quick and dirty is een betere benadering, want het programma doet wat het moet doen. Het kan dus vast wel mooier, slimmer en korter. Ook is er geen aandacht besteed aan snelheid en efficiëntie.

Dit doet het

Het programma doet het volgende. Als eerste wordt er gekeken of er wel een 1 Mbyte RAMkaart in het systeem zit. Is dat niet het geval dan wordt er gestopt met een foutmelding. Daarna wordt al het mogelijk aanwezige RAM gevuld met nul-bytes om een beginsituatie te maken. Vervolgens wordt gekeken hoeveel RAM er nu werkelijk op de kaart zit. Dit gebeurt relatief eenvoudig: op iedere kilobytegrens wordt het nummer van die kilobyte weggeschreven. Dit proces begint bij kilobyte nummer 1023 en telt naar beneden tot kilobyte nummer 57. 56 en lager is immers het werkgeheugen van het programma en

DOS65 zelf, en dat moet heel blijven. Als alle nummertjes zijn weggeschreven wordt er gezocht naar het hoogste kilobytenummer dat te vinden is: dat is de hoeveelheid RAM die werkelijk aanwezig is.

Pas dan begint het werkelijke testen. Dat gebeurt voor iedere kilobyte apart door \$55, \$AA, \$FF, \$01, \$02, \$04, \$08, \$10, \$20, \$40, \$80, \$FE, \$FD, \$FB, \$F7, \$EF, \$DF, \$BF, \$7F en tenslotte \$00 weg te schrijven en weer terug te lezen. Daarna volgt nog een test op kortgesloten adreslijnen, door slechts 1 bit weg te schrijven en te controleren of dit bit nergens anders opduikt.

Het programma sluit af met een test of het taaknummerregister werkt.

VTEST.MAC en VTEST.LIB staan op het BBS, en worden ook meegeleverd op diskette bij iedere print voor de 1 Mbyte RAMkaart.

Nico de Vries

```

ttl      'ftpFile: %sDate: %s Page %2d'
;*****
;*
;* VTEST
;*
;* A test program for the VDISK RAM card.
;*
;* The program will determine the amount of
;* RAM present on the VDISK card and thou-
;* roughly test all RAM on the card.
;*
;*****
;
; Name: VTEST.MAC
;
; Written by: Nico de Vries
; Mari Andriessenrade 49
; 2907 MA Capelle aan den IJssel
; The Netherlands
;
; History:
;
; 30-05-1993 Start of work
; 08-05-1994 V0.02: address of taskregister/datram changed to $FFE0
; 09-05-1994 V0.03: fixed bug in sizing, preventing finding size over 256k
; 09-05-1994 V0.04: added rudimentary stuck address line test

```

Fig. 1: VTEST.MAC, de RAM(p)tester

```

; 09-05-1994 V0.05: testing now starts at 56k, not at 64k
; 09-05-1994 V1.00: release version, equal to 0.05
; 09-05-1994 V1.01: bug fixed in card recognition part
;
; Description of DATRAM and TASKREG use:
;
; TASKREG:
; Initialized by IO65 at $FF
; All testing is done with tasknumber $FE loaded.
; The functioning of the taskregister is also tested.
;
; DATRAM:
; The 16 locations of DATRAM control the memory mapping as follows:
; DATRAM + 0: Addresslines A12 through A19 for $0000-$0FFF
; DATRAM + 1: Addresslines A12 through A19 for $1000-$1FFF
; DATRAM + 2: Addresslines A12 through A19 for $2000-$2FFF
; DATRAM + 3: Addresslines A12 through A19 for $3000-$3FFF
; DATRAM + 4: Addresslines A12 through A19 for $4000-$4FFF
; DATRAM + 5: Addresslines A12 through A19 for $5000-$5FFF
; DATRAM + 6: Addresslines A12 through A19 for $6000-$6FFF
; DATRAM + 7: Addresslines A12 through A19 for $7000-$7FFF
; DATRAM + 8: Addresslines A12 through A19 for $8000-$8FFF
; DATRAM + 9: Addresslines A12 through A19 for $9000-$9FFF
; DATRAM + A: Addresslines A12 through A19 for $A000-$AFFF
; DATRAM + B: Addresslines A12 through A19 for $B000-$BFFF
; DATRAM + C: Addresslines A12 through A19 for $C000-$CFFF
; DATRAM + D: Addresslines A12 through A19 for $D000-$DFFF
; DATRAM + E: Addresslines A12 through A19 for $E000-$EFFF: normally IO/video
; DATRAM + F: Addresslines A12 through A19 for $F000-$FFFF: normally IO65 ROM
;
; The data written into DATRAM + ? is as follows:
; D7 D6 D5 D4 D3 D2 D1 D0
;
; | | | | | | | |
; | | | | | | | |----- A12 for $?000-$?FFF
; | | | | | | | |----- A13 for $?000-$?FFF
; | | | | | | | |----- A14 for $?000-$?FFF
; | | | | | | | |----- A15 for $?000-$?FFF
; | | | | | | | |----- A16 for $?000-$?FFF
; | | | | | | | |----- A17 for $?000-$?FFF
; | | | | | | | |----- A18 for $?000-$?FFF
; | | | | | | | |----- A19 for $?000-$?FFF
;
; It is therefore possible to access the entire amount of RAM on the
; VDISK card through one window of 4 kbyte.
;
; VTEST uses the window at $1000 through $1FFF for testing.
;
; pag
;
; *** External Labels
;
; equ      "1V"          ;note: backwards!
; equ      "10"          ;note: backwards!
;
;
; lib      vtest.lib

```

```

;
; pag
;
; org      $a000                ;standard utility address
;
;*** Print sign-on message
;
vtest      jmp      vtest1        ;skip info part and variables
fcc        $c8,$c5,$cc,$d0        ;flag for help text
fcc        $4c,$4c,$4c            ;dummy bytes
fcc        'VTEST is a test program to verify that the'
fcc        ' VDISK RAM card is operating correctly.\r'
fcc        0                      ;end of help text
vtest1     jsr      prtext         ;print sign-on message
fcc        $0c,' VTEST '
fdb        rev
fcc        ','
fdb        release
fcc        $1b,'i\r by NdV ', $1b,'n\r'
fcc        0
jsr        remap                  ;set original mapping
lda        $1000                  ;get 1 address of testwindow
ldx        $1800                  ;get middle address
ldy        $1fff                  ;get last too
sta        deci1                  ;save
stx        deci2                  ;values
sty        deci3                  ;for later
ldx        #$11                   ;select 17th 4k block on card
stx        datram + 1             ;to map at $1000-$1fff
inc        $1000                  ;change
lda        $1800                  ;all
eor        #$ff                   ;three
sta        $1800                  ;memory
dec        $1fff                  ;locations
ldx        #$01                   ;select 2nd 4k block on card
stx        datram + 1             ;to map at $1000-$1fff
lda        $1000                  ;get data
cmp        deci1                  ;if different
bne        nocard                 ;no card is there
lda        $1800                  ;get data
cmp        deci2                  ;if different
bne        nocard                 ;no card is there
lda        $1fff                  ;if last data also different
cmp        deci3                  ;report error
beq        cardthr                ;else test card
nocard     dec        $1000         ;get
lda        $1800                  ;original
eor        #$ff                   ;data back
sta        $1800                  ;in memory
inc        $1fff                  ;locations
jsr        prtext                 ;report error
fcc        '\rThis system has no VDISK card!\r'
fcc        0
jmp        exit                   ;and exit
;
;*** Fill entire 1 Mbyte with zeroes to initialize the RAM

```



```

;*** The bottom 56 kbyte is skipped, because DOS65 and this
;*** program live there.
;
cardthr      jsr      prtext
fcc          '\rFilling 1 Mbyte of RAM with zeroes... '
fcc          0
ldy          #$FE          ;get tasknumber
sty          taskreg        ;set tasknumber
ldy          #$ff          ;get inner loop count
lda          #0             ;get data
1            sty      datram + 1 ;use $1000-$1fff
tax          ;start at beginning
2            sta      $1000,x    ;zero 1st kilobyte
            sta      $1100,x    ;zero 1st kilobyte
            sta      $1200,x    ;zero 1st kilobyte
            sta      $1300,x    ;zero 1st kilobyte
            sta      $1400,x    ;zero 2nd kilobyte
            sta      $1500,x    ;zero 2nd kilobyte
            sta      $1600,x    ;zero 2nd kilobyte
            sta      $1700,x    ;zero 2nd kilobyte
            sta      $1800,x    ;zero 3rd kilobyte
            sta      $1900,x    ;zero 3rd kilobyte
            sta      $1A00,x    ;zero 3rd kilobyte
            sta      $1B00,x    ;zero 3rd kilobyte
            sta      $1C00,x    ;zero 4th kilobyte
            sta      $1D00,x    ;zero 4th kilobyte
            sta      $1E00,x    ;zero 4th kilobyte
            sta      $1F00,x    ;zero 4th kilobyte
inx          ;count bytes
bne          2.b            ;and repeat
dey          ;count loops
cpy          #$0d           ;if not entering 1st 56k
bne          1.b            ;loop
jsr          prtext
fcc          'Done.\r'
fcc          0
jsr          remap
;
;*** Size RAM by writing size pattern on start of each
;*** kilobyte.
;*** Then search for the highest size number.
;
jsr          prtext
fcc          '\rSizing RAM.... '
fcc          0
ldx          #3             ;1 Mbyte is 1024 kbyte, is $3ff plus one
lda          #$ff           ;
tay
1            sta      datram + 1 ;use $1000-$1fff
            sty      $1c00        ;store count low
            stx      $1c01        ;store count high
            dey
            cpy      #$ff         ;
            bne      2.f          ;if Y went from 0 to FF
            dex          ;adapt X
2            sty      $1800        ;store count low

```

	stx	\$1801	;store count high
	dey		
	cpy	#\$ff	
	bne	3.f	;if Y went from 0 to FF
	dex		;adapt X
3	sty	\$1400	;store count low
	stx	\$1401	;store count high
	dey		
	cpy	#\$ff	
	bne	4.f	;if Y went from 0 to FF
	dex		;adapt X
4	sty	\$1000	;store count low
	stx	\$1001	;store count high
	dey		
	cpy	#\$ff	
	bne	5.f	;if Y went from 0 to FF
	dex		;adapt X
5	sec	;prepare for subtract	
	sbc	#1	;decrement A
	cmp	#\$0d	;if entering lowest 56k
	beq	6.f	;exit loop
	jmp	1.b	;and loop
6	lda	#\$0e	;start at 56k
	ldy	#\$56	;get expected size low
	ldx	#0	;get expected size high
7	sta	datram + 1	;set mapping
	cpy	\$1000	;compare size low
	bne	sizefnd	;if different, go print size
	cpx	\$1001	;compare size high
	bne	sizefnd	;if different, go print size
	iny		;adapt size
	bne	8.f	;if Y = 0
	inx		;adapt X
8	cpy	\$1400	;compare size low
	bne	sizefnd	;if different, go print size
	cpx	\$1401	;compare size high
	bne	sizefnd	;if different, go print size
	iny		;adapt size
	bne	9.f	;if Y = 0
	inx		;adapt X
9	cpy	\$1800	;compare size low
	bne	sizefnd	;if different, go print size
	cpx	\$1801	;compare size high
	bne	sizefnd	;if different, go print size
	iny		;adapt size
	bne	10.f	;if Y = 0
	inx		;adapt X
10	cpy	\$1c00	;compare size low
	bne	sizefnd	;if different, go print size
	cpx	\$1c01	;compare size high
	bne	sizefnd	;if different, go print size
	iny		;adapt size
	bne	11.f	;if Y = 0
	inx		;adapt X
11	clc		;prepare for add
	adc	#1	;adapt mapping

Label	Assembly Code	Comments
sizefnd	bne sty stx txa pha tya tax pla jsr jsr jsr prtext fcc ' kbyte found.\r\r' fcc 0 ; ;*** Test RAM. ; ldx #56 lda #0 stx curkilo sta curkilo + 1 lda curkilo + 1 cmp ramsize + 1 bne 2.f lda curkilo cmp ramsize beq testOK ldx #1 ldy #8 jsr crsadr jsr prtext fcc 'Testing RAM... ' fcc 0 ldx curkilo lda curkilo + 1 jsr testram php inc curkilo bne 3.f inc curkilo + 1 plp bcs endtest ldx curkilo lda curkilo + 1 jsr praxdec jsr prtext fcc ' kbyte OK.' fcc 0 jmp 1.b jsr crlf jsr prtext fcc \$1b,'iRAM error at ' fcc 0 ldx curkilo lda curkilo + 1 jsr praxdec jsr prtext	;if not all done, loop ;store size low ;and hi ;get high ;save high ;get low in A ;and in X ;get high in A ;print size in decimal ;get low ;get high ;set low ;set high ;get current kilobyte high ;if not same as end ;adapt and test ;else get low ;if same as last ;finish test ;column# ;line# ;position cursor ;get low ;get high ;test 1 kilobyte of RAM ;save the flags ;else adapt low ;if not zero, restore flags ;else adapt high ;restore flags ;if error, stop now ;get low ;get high ;print in decimal ;and test ;go to new line ;get low ;get high ;print in decimal
1	lda curkilo + 1 cmp ramsize + 1 bne 2.f lda curkilo cmp ramsize beq testOK	;get current kilobyte high ;if not same as end ;adapt and test ;else get low ;if same as last ;finish test
2	ldx #1 ldy #8 jsr crsadr jsr prtext fcc 'Testing RAM... ' fcc 0 ldx curkilo lda curkilo + 1 jsr testram	;column# ;line# ;position cursor ;get low ;get high ;test 1 kilobyte of RAM
3	php inc curkilo bne 3.f inc curkilo + 1 plp bcs endtest ldx curkilo lda curkilo + 1 jsr praxdec jsr prtext fcc ' kbyte OK.' fcc 0 jmp 1.b jsr crlf jsr prtext fcc \$1b,'iRAM error at ' fcc 0 ldx curkilo lda curkilo + 1 jsr praxdec jsr prtext	;save the flags ;else adapt low ;if not zero, restore flags ;else adapt high ;restore flags ;if error, stop now ;get low ;get high ;print in decimal ;and test ;go to new line ;get low ;get high ;print in decimal
endtest	jsr praxdec jsr prtext	;print in decimal

	fcc	' kbyte.', \$1b, 'n'	
	fcc	0	
testOK	jsr	crlf	;go to new line
	:		
	***	Test tasknumberregister by changing the mapping of	
	***	\$1000-\$1FFF 256 times and verifying that the window	
	***	can be changed by switching tasks only.	
	:		
	jsr	remap	
	lda	\$1000	;get original data
	sta	deci1	;save it
	lda	\$1fff	;get original data
	sta	deci2	;save it
	lda	ramsize + 1	;get ramsize high
	sta	work	;copy it
	lda	ramsize	;get size low
	lsr	work	;get maximum
	rora	;task	
	lsr	work	;count
	rora		;in A
	sta	work	;and in work
	jsr	prtext	
	fcc	'\rTesting taskregister....'	
	fcc	0	
	ldy	#0	
	ldx	#\$10	;start with 65k
1	cpx	work	;if RAM mapped
	beq	testtsk	;check storage
	sty	taskreg	;set task register
	stx	datram + 1	;map \$1000-\$1fff
	tya		;get data
	sta	\$1000	;move to begin
	sta	\$1fff	;and end of window
	iny		;else get next task
	inx		;and next map
	bne	1.b	;if not zero, loop
testtsk	ldy	#0	;else get task zero back
	ldx	#\$10	;start with 65k
2	cpx	work	;if entire RAM mapped
	beq	taskOK	;report no errors
	sty	taskreg	;set task register
	stx	datram + 1	;map \$1000-\$1fff
	tya		;get data
	cmp	\$1000	;check data
	bne	taskerr	;if error, report it
	cmp	\$1fff	;check data at end of window
	bne	taskerr	;if error, report it
	iny		;else get next task
	inx		;and next map
	bne	2.b	;then loop
taskerr	jsr	prtext	
	fcc	\$1b, 'iError.', \$1b, 'n\r'	
	fcc	0	
	jmp	exit	;and proceed
taskOK	jsr	prtext	
	fcc	'OK.\r'	

	fcc	0	
	;		
	*** Exit program.		
	;		
exit	lda	deci1	;get original data
	sta	\$1000	;save it
	lda	deci2	;in original
	sta	\$1fff	;location
	jsr	remap	;get original mapping for \$1000-\$1fff
	lda	#0	;get tasknumber
	sta	taskreg	;set task
	lda	#\$01	;get mapping for 2nd 4 kbyte
	sta	datram + 1	;set mapping RAM
	jmp	crlf	;start on next line and exit
	;		
	*** REMAP: set original mapping for \$1000-\$1FFF		
	;		
remap	pha		;save A
	lda	#\$FE	;get tasknumber
	sta	taskreg	;set task
	lda	#\$01	;get mapping for 2nd 4 kbyte
	sta	datram + 1	;set mapping RAM
	pla		;restore A
	rts		;and exit
	;		
	*** PRAXDEC: print AX in decimal		
	;NOTE: Stolen from IO65!		
praxdec	jsr	conver	;convert to decimals
	pha		;save A
	txa		;save X
	pha		;too
	jsr	pridec	;print result
	pla		;get old X
	tax		;in X
	pla		;get old A
	rts		;and exit
	;		
	*** Convert from 2 hex bytes to 3 decimal bytes ***		
	;NOTE: Stolen from IO65!		
conver	stx	deci1	;store low
	sta	deci2	;and high bytes
	pha		;save A
	txa		;save X
	pha		;too
	lda	#0	;zero result
	sta	deci3	;area
	tax		;get a zero byte
	sed		;set decimal mode
	beq	hdf	;and start (branch always)
hda	dec	deci2	;adapt hi
	txa		;get units
	adc	#\$56	;add 56
	tax		;to units
	pla		;get hundreds
	adc	#\$02	;add 200
	bcc	hdf	;if no carry, loop

	inc	dec13	;else adapt tenthousands
	bne	hdf	;and loop (branch always)
hdb	dec	dec1	;decrement low
	txa		;get units
	adc	#1	;add one
	tax		;save units
	pla		;restore carry
hdf	adc	#0	;add carry
	pha		;save carry byte
	clc		;prepare for add
	lda	dec1	;get low
	bne	hdb	;if not zero, add one to units
	lda	dec12	;if hi zero, end
	bne	hda	;else convert hi byte
	cld		;set hex mode
	pla		;restore hundreds
	sta	dec12	;store them as results
	stx	dec1	;store units too
	pla		;get old X
	tax		;in X
	pla		;get old A
	rts		;and exit
			;
			*** Print decimal numbers ***
			;NOTE: Stolen from IO65!
pridec	lda	dec13	;get tenthousands
	bne	conoe	;if any, print them
	lda	dec12	;get hundreds
	bne	conof	;if any, print them
	lda	dec1	;else get units
	jmp	nhxout	;and print least sign. byte
conoe	jsr	nhxout	;print tenthousands
	lda	dec12	;get hundreds
	jsr	prbyt	;print them
conog	lda	dec1	;get units
	jmp	prbyt	;print them
conof	jsr	nhxout	;print hundreds
	jmp	conog	;and print units
			;
			*** Print a byte (in decimal mode)
			;NOTE: Stolen from IO65!
nhxout	cmp	#\$10	;if smaller than 10 (decimal mode)
	bcc	prnibl	;print lo nibble only
			;
			*** Print a byte (in hex mode)
			;
prbyt	pha		;else save the byte
	lsra		;get high
	lsra		;nibble
	lsra		;into
	lsra		;low nibble
	jsr	prnibl	;print it
	pla		;restore the byte and:
			;
			*** Print low nibble as hex digit
			;

pnibl	and	##00001111	;get low nibble only
	cmp	#\$0a	;Number or A...F
	bcc	na	;if below A, go convert
	adc	#\$06	;else add offset for letter
na	adc	#\$30	;Calculate ASCII value
	jmp	output	;print digit
	;*** TESTRAM: test 1 kilobyte of RAM starting at curkilo		
testram	pha		;save
	tya		;pricipal
	pha		;registers
	txa		;on
	pha		;stack
	lda	curkilo + 1	;get curent kilobyte, high
	sta	work	;in work area
	lda	curkilo	;get current kilobyte, low
	lsr	work	;divide
	rora		;current
	lsr	work	;kilobyte
	rora		;by four
	sta	datram + 1	;map that into \$1000-\$1fff
	lda	curkilo	;get current kilobyte, low
	and	##00000011	;get kilobyte number within frame
	asla		;multiply by
	asla		;four
	clc		;prepare for add
	adc	#\$10	;add frame address high
	sta	rcurr + 1	;set zero page address
	ldy	#0	;get low
	sty	rcurr	;set that too
	lda	#\$55	;get odd checkerboard
	jsr	test1k	;test 1 kilobyte
	bcs	ramp2	;if error, exit
	lda	#\$AA	;get even checkerboard
	jsr	test1k	;test 1 kilobyte
	bcs	ramp2	;if error, exit
	lda	#\$FF	;get all ones
	jsr	test1k	;test 1 kilobyte
	bcs	ramp2	;if error, exit
	lda	##00000001	;single bit
bitloop	jsr	test1k	;test 1 kilobyte
	bcs	ramp2	;if error, exit
	asla		;shift single bit
	bcc	bitloop	;and repeat
	lda	##11111110	;single bit
bitlop2	jsr	test1k	;test 1 kilobyte
	bcs	ramp2	;if error, exit
	sec		;shift in a 1
	rola		;shift single bit
	bcs	bitlop2	;and repeat
	clc		;reset error flag
	jsr	testadr	;test for stuck address lines
	bcs	ramp2	;if error, set carry
	lda	#0	;
	jsr	test1k	;test 1 kilobyte and leave zeroes

ramp2	pla		;restore
	tax		;saved
	pla		;registers
	tay		;from
	pla		;stack
	rts		;and exit
	;		
	;*** TEST1K: test 1 kilobyte of RAM with pattern in A		
	;		
test1k	ldy	#0	;get start offset
1	sta	[rcurr],y	;store data
	iny		;do next byte
	bne	1.b	;if not whole page done, repeat
	inc	rcurr+1	;else adapt high byte of address
2	sta	[rcurr],y	;store data
	iny		;do next byte
	bne	2.b	;if not whole page done, repeat
	inc	rcurr+1	;else adapt high byte of address
3	sta	[rcurr],y	;store data
	iny		;do next byte
	bne	3.b	;if not whole page done, repeat
	inc	rcurr+1	;else adapt high byte of address
4	sta	[rcurr],y	;store data
	iny		;do next byte
	bne	4.b	;if not whole page done, repeat
5	cmp	[rcurr],y	;compare data
	bne	ramp	;if error, exit
	iny		;do next byte
	bne	5.b	;if not whole page done, repeat
	dec	rcurr+1	;else adapt high byte of address
6	cmp	[rcurr],y	;compare data
	bne	ramp	;if error, exit
	iny		;do next byte
	bne	6.b	;if not whole page done, repeat
	dec	rcurr+1	;else adapt high byte of address
7	cmp	[rcurr],y	;compare data
	bne	ramp	;if error, exit
	iny		;do next byte
	bne	7.b	;if not whole page done, repeat
	dec	rcurr+1	;else adapt high byte of address
8	cmp	[rcurr],y	;compare data
	bne	ramp	;if error, exit
	iny		;do next byte
	bne	8.b	;if not whole page done, repeat
	clc		
	fcc	\$24	;skip next instruction
ramp	sec		
	rts		
	;		
	;*** TESTADR: Test 1 kilobyte for stuck address lines		
	;		
testadr	ldy	#0	;get start offset
	tya		;A=0
1	sta	[rcurr],y	;store data
	iny		;do next byte
	bne	1.b	;if not whole page done, repeat

2	inc	rcurr + 1	;else adapt high byte of address
	sta	[rcurr],y	;store data
	iny		;do next byte
3	bne	2.b	;if not whole page done, repeat
	inc	rcurr + 1	;else adapt high byte of address
	sta	[rcurr],y	;store data
4	iny		;do next byte
	bne	3.b	;if not whole page done, repeat
	inc	rcurr + 1	;else adapt high byte of address
adrloop	sta	[rcurr],y	;store data
	iny		;do next byte
	bne	4.b	;if not whole page done, repeat
	lda	%%00000001	;get single bit
	sta	deci1	;save pattern
	lda	%%00000000	;get default data
	sta	deci2	;save too
	jsr	adrtest	;test for stuck address lines
	bcs	erradr	;if error, exit
	asl	deci1	;then shift bit
1	bcc	adrloop	;if not all bits tested, loop
	;repeat test with inverted data		
	ldy	#0	;get start offset
	lda	#\$ff	;
	sta	[rcurr],y	;store data
	iny		;do next byte
	bne	1.b	;if not whole page done, repeat
	inc	rcurr + 1	;else adapt high byte of address
	sta	[rcurr],y	;store data
	iny		;do next byte
2	bne	2.b	;if not whole page done, repeat
	inc	rcurr + 1	;else adapt high byte of address
	sta	[rcurr],y	;store data
3	iny		;do next byte
	bne	3.b	;if not whole page done, repeat
	inc	rcurr + 1	;else adapt high byte of address
4	sta	[rcurr],y	;store data
	iny		;do next byte
	bne	4.b	;if not whole page done, repeat
	lda	%%11111110	;get single bit
	sta	deci1	;save pattern
	lda	%%11111111	;get default data
	sta	deci2	;save too
	jsr	adrtest	;test for stuck address lines
	bcs	erradr	;if error, exit
	sec		;shift in a 1
adrlop2	rol	deci1	;shift bits
	bcs	adrlop2	;if not all bits tested, loop
	clc		
	rts		
	sec		
	rts		
	;*** ADRTTEST: store and verify single bit pattern		
	;		
	ldy	deci1	;get offset
	lda	deci1	;get data
erradr			
adrtest			

	sta	[rcurr],y	;store in RAM
	ldy	#0	;get start offset
	lda	dec12	;get default data
5	cmp	[rcurr],y	;if RAM location is zero
	beq	51.f	;do next byte
	lda	[rcurr],y	;else fetch data
	cmp	dec1	;if not test pattern
	bne	adrerr	;then address error
	cpy	dec1	;if test pattern, but not
	bne	adrerr	;on correct address, then address error
	lda	dec12	;else get default data back
51	iny		;else do next byte
	bne	5.b	;if not whole page done, repeat
	dec	rcurr + 1	;else adapt high byte of address
6	cmp	[rcurr],y	;if RAM location is not zero
	bne	adrerr	;then address error
	iny		;else do next byte
	bne	6.b	;if not whole page done, repeat
	dec	rcurr + 1	;else adapt high byte of address
7	cmp	[rcurr],y	;if RAM location is not zero
	bne	adrerr	;then address error
	iny		;else do next byte
	bne	7.b	;if not whole page done, repeat
	dec	rcurr + 1	;else adapt high byte of address
8	cmp	[rcurr],y	;if RAM location is not zero
	bne	adrerr	;then address error
	iny		;else do next byte
	bne	8.b	;if not whole page done, repeat
	inc	rcurr + 1	;else adapt
	inc	rcurr + 1	;high byte
	inc	rcurr + 1	;of address
	ldy	dec1	;get offset
	lda	dec12	;get default data
	sta	[rcurr],y	;overwrite test byte in RAM
	clc		
	rts		;and exit
adrerr	;pha		;save data
;	tya		;get offset
;	clc		;prepare for add
;	adc	rcurr	;add to address low
;	sta	rcurr	;store new low
;	bcc	nocary	;if a carry remained
;	inc	rcurr + 1	;adapt high byte
;nocary	jsr	prtext	;report error
;	fcc	'\rData: '	
;	fcc	0	
;	pla		;get data back
;	jsr	prbyte	;print data
;	jsr	prtext	;report error
;	fcc	' Physical address: '	
;	fcc	0	
;	lda	rcurr + 1	;get address high
;	jsr	prbyte	;print address high
;	lda	rcurr	;get address low
;	jsr	prbyte	;print address low

```

;      jsr      prtext      ;report error
;      fcc      '\r\r',0
;      sec
;      rts
;
;*** Various variables
;
;
ramsize    res      2      ;RAM size in kilobytes
curkilo    res      2      ;current kilobyte in test
work       res      1      ;work area
deci1      res      1      ;decimal byte: low
deci2      res      1      ;decimal byte: middle
deci3      res      1      ;decimal byte: high
;
end        vtest

```

```

;*****
;*
;* VTEST.LIB
;* External labels for VTEST.MAC
;*
;*****
; Name: VTEST.LIB
; Written by: Nico de Vries
;            Mari Andriessenrade 49
;            2907 MA Capelle aan den IJssel
;            The Netherlands
;
;
; datram      equ      $ffe0      ;mapping RAM base address
; taskreg     equ      datram + $10 ;tasknumber register
;
; *** DOS labels
;
; output      equ      $c023      ;print the character in A
; crlf        equ      $c02f      ;output CRLF
; hnout       equ      $c035      ;print A as one or two hex nibbles
; prbyte      equ      $c038      ;print A as two hex nibbles
; prtext      equ      $c03b      ;print string until zero byte
;
; *** IO65 labels
;
; crsadr      equ      $f024      ;move cursor to X,Y
;
; *** Page zero variables
;
; rcurr       org      $80
;             res      2      ;current RAM address

```

Fig. 2: VTEST.LIB, externe labels en variabelen voor VTEST

De Computerwetten van Murphy

De wet van Murphy

Wanneer er iets mis kan gaan, gaat het ook mis. Met de uitvinding van de computer probeerde de mens voor het eerst levenloze materie een zekere mate van intelligentie mee te geven. Een fatale beslissing. Immers, tot op de dag van vandaag zijn computers weliswaar intelligent noch creatief, maar achterbaktheid, geniepigheid en sluwheid zijn al optimaal ontwikkeld.

Eerste digitale afleiding

De wet van Murphy wordt door de computer geoptimaliseerd. Echter, aangezien moderne computers verschillende dingen tegelijk kunnen volgt de:

Tweede digitale afleiding

Alles gaat op hetzelfde moment mis. Met de uitvinding van checksums, correctie- en backup-programma's en fouttolerante systemen krijgt de stomverbaasde, tot object gedegradeerde mens toegang tot de veelzijdige wereld van de informatica via de:

Het gaat ook mis wanneer het eigenlijk niet mis kan gaan.

Derde digitale afleiding

Het gaat ook mis wanneer het eigenlijk niet mis kan gaan. Puttend uit de rijke ervaring van gebruikers, programmeurs, ontwikkelaars en andere arme donders kunnen we nu de wet van Murphy en haar digitale afleidingen toepassen op de elektronische wereld van alledag.

Eerste elektronische toepassing van de wet van Murphy

Bij de computers is niets ondenkbaar, laat staan onmogelijk. Behalve het wenselijke.

Tweede elektronische toepassing van de wet van Murphy

In de wereld van de informatica gaan storingen niet over, maar gaan, elkaar overlappend, in elkaar over.

Derde elektronische toepassing van de wet van Murphy

Computerstoringen wachten geduldig op het ongelukkigste moment en slaan dan meedogenloos toe.

Vierde elektronische toepassing van de wet van Murphy

Je kunt bij computers nergens van op aan. Je kunt er niet eens niet van op aan dat je nergens van op aan kunt.

Vijfde elektronische toepassing van de wet van Murphy

1. Je kunt nooit een ernstige storing voorkomen door een kleine te veroorzaken
2. In het gunstigste geval zal een kleine storing zich voegen bij een grote, om die te versterken.

Zesde elektronische toepassing van de wet van Murphy

Niemand kan zoveel storingen veroorzaken als er zich in een computer voordoen.

Digitale kwartetreglement

1. Wanneer je gebruiker bent zul je het afleggen tegen computers, hardware-fabrikanten en programmeurs.
2. Wanneer je een hardware fabrikant bent, zul je het moeten afleggen tegen computers, gebruikers en programmeurs.
3. Wanneer je een programmeur bent, zul je het afleggen tegen computers, hardware-fabrikanten en gebruikers.

Gevolgtrekkingen uit het digitale kwartetreglement

1. Een menselijk winnaar is onmogelijk.
2. De Computer wint altijd.

Dubbelwet van de complexe hardware

1. Complexe systemen neigen naar complexe fouten.
2. Eenvoudige systemen daarentegen neigen naar complexe fouten.

Udo's knutseltheorieën

1. Een elektronisch apparaat demonteren is doodeenvoudig.
2. Het weer zo in elkaar zetten dat het weer werkt, is onmogelijk.
3. Ertegen schoppen helpt alleen bij anderen.

De ultieme prijswet

Hoe duur je een computersysteem schat, het wordt uiteindelijk altijd duurder dan verwacht.

Platts berekening van de ultieme prijswet(ook bekend onder de naam "gewone uitbreidingskoorts")

$$K > (J * (1000 + A/15)) + (1,5 * B) + A/20$$

Hierin zijn K de totale kosten na J jaren, wanneer de gebruiker een jaarinkomen heeft van A en denkt dat zijn systeem B zou kosten.

Het dimensiemirakel

Elke computer is te klein

Nadere uitwerkingen

1. Beschikt hij over een voldoende grote harde schijf, dan is het werkgeheugen te klein.
2. Heeft hij genoeg werkgeheugen, dan is de harde schijf te klein.

De MS-DOS-uitbreiding van het dimensiemirakel

Wanneer harde schijf en werkgeheugen groot genoeg zijn dan heeft de computer een besturingssysteem dat :

- a) een van beide of beide niet ondersteunt;
- b) een geheugendriver nodig heeft die het beschikbare toepassingsprogramma niet begrijpt.

De fysische uitbreiding van het dimensiemirakel

1. In elk geval beschikt je computer over één uitbreidingsslot te weinig.
2. Dat merk je pas wanneer je een nieuwe insteekkaart hebt gekocht.

De batterijbanaliteit

De accu van een shootcomputer is een minuut voor het opslaan van gegevens uitgeput.

Jaruks servicewet

1. Als het goedkoper is om een nieuwe computer te kopen, staat de firma erop de oude te repareren.

2. Als het goedkoper is om je oude systeem te repareren, staat de firma erop het nieuwste model te leveren.

De fundamentele toetsenbordtheorieën

1. Je toetsenbord heeft altijd één toets minder dan je favoriete programma ondersteunt.
2. Je toetsenbord heeft altijd een toets te veel, die defect kan en zal raken.

De fundamentele muistheorie over de compatibiliteit

Wanneer je een Mouse-System-compatibele drieknopsmuis koopt, zul je je hele leven niet een programma vinden dat die derde knop ondersteunt. Vanaf het moment dat je overstapt op een Microsoft-compatibele tweeknopsmuis, zul je overwegend moeten werken met een programma dat die derde knop zinvol gebruikt.

Wet van de "Einde?"-"Nee"-dubbelslag

Wanneer je per abuis de toetsenbordcombinatie indrukt waarmee je programma wordt beëindigd, zul je ook op de toets drukken die de vraag of de aangebrachte veranderingen moeten bewaard worden ontkent.

De backup-premissen

1. Voor een backup is altijd 1 diskette meer nodig dan je in voorraad hebt.
2. Een backup programma zal het laten afweten zodra je het nodig hebt.

Eerste afleiding

Het backup-programma zal hierbij het beschadigde bestand over je enige veiligheidskopie heen schrijven.

Tweede afleiding

Wanneer je de backup wilt terugzetten, zul je merken dat de enige versie van RESTORE op de schijf (en alleen daar) stond voordat je die formatteerde.

GÜtz' eerste theorie van de alomtegenwoordige onzekerheid

Pas wanneer je, bijvoorbeeld bij het formatteren, de vraag van het programma "Weet u het zeker?" met J hebt beantwoord, schiet je te binnen dat je er alles-behalve zeker van bent.

GÜtz' aangescherpte theorie van de alomtegenwoordige onzekerheid

Wanneer je daarna de diskette controleert, weet je het zeker : je hebt net je belangrijkste bestand gewist.

De diskettebakken-wetten

1. Je krijgt een diskette gemakkelijker in het bakje dan er weer uit.
2. De plastic tussenschotjes in het diskettebakje dienen er alleen voor om het uitzicht op de gezochte diskette te belemmeren.
3. In alle andere gevallen klappen ze de gezochte diskette naar voren.
4. Een diskette zit nooit in het vak waar je hem zoekt.
5. Je zult de sleutel nooit kwijt zijn, alleen wanneer je de bak per ongeluk een keer op slot hebt gedaan.
6. Diskettebakken kunnen niet op elkaar worden gestapeld.
7. Ze zullen je van het tegendeel overtuigen tot je er een aanraakt. Dan zullen ze allemaal tegelijk omvallen en hun inhoud over de vloer verspreiden.

Vuistregel voor een juist begrip van de software-industrie

Alle grote software-ontwikkelingen worden verwezenlijkt op grond van ernstige programmafouten.

Eerste conclusie uit de software vuistregel
Elk programma bevat fouten.

Tweede conclusie uit de software-vuistregel
Elk programma heeft altijd één fout meer.

Derde conclusie uit de software-vuistregel
Het herstellen van een fout veroorzaakt minstens twee nieuwe.

Persoonlijke afleiding uit de software-vuistregel
Wanneer fouten aan het daglicht treden, is dat bij jou.

Heini's theorieën over computerspellen

1. Je hebt altijd één punt te weinig voor de hoogste score
2. Als je een spel zo vaak hebt gespeeld, dat je onverslaanbaar bent geworden, zal een vriend het voor de eerste keer spelen en moeiteloos de eerste plaats bereiken.
3. Als je de beste bent, zal dat geen mens interesseren.

Postulaat van de multifunctionaliteit

Hoe minder functies een programma heeft, hoe beter het die uitvoert.

De fundamentele virustheorie

Computervirussen worden in principe verspreid via "gegarandeerd virusvrije" programma- en systeem-diskettes.

De algemene virustheorie

Je loopt een computervirus juist dan op wanneer je denkt dat je er geen hebt.

De termijngebonden virustheorie

Je loopt een computervirus juist dan op wanneer je het minst kunt gebruiken.

De partnerschaps-virustheorie

Gegarandeerd onschone virustheorie

Je loopt een computervirus juist dan op wanneer je denkt dat je er geen hebt.

De termijngebonden virustheorie

Je loopt een computervirus juist dan op wanneer je het minst kunt gebruiken.

De partnerschaps-virustheorie

Gegarandeerd onschuvoor geniepigheid en destructiviteit.

Onwrikbare wetten van de tekstverwerking

1. Wanneer je een woord wil wissen, verdwijnt er gegarandeerd een hele regel.

Algemeen?

2. Wanneer je een hele regel wil wissen, verdwijnt een hele alinea.
3. Wanneer je een alinea wilt wissen, verdwijnt de hele tekst.
4. Wanneer je de hele tekst wil wissen, gebeurt er helemaal niets.

Het baby-op-schoot-axioma

Een kind dat het toetsenbord in handen krijgt, vindt bij de eerste aanraking de enige toetsencombinatie waarmee iets te vernietigen valt. Wanneer er meer mogelijkheden zijn, zoekt hij de meest onheil aanrichtende uit.

Beperkt Baby-op-schoot-axioma

Wanneer je het kind belet een fatale toetsencombinatie aan te slaan, dan leidt dat minstens to een jebbbbbb kjllfdk hjjkhk ?.;fyo ijhv;.;?kikj oif

Preciseren van het ander-systeem-fenomeen

- Als je een Amiga hebt, wordt je uitgelachen om je spelletjescomputer
- Als je een Atari hebt, wordt je uitgelachen om je would-be grafische computer.
- Als je een commodore 64 hebt, wordt je uitgelachen om je peuterspeelzaalcomputer.
- Als je een Macintosh hebt, wordt je uitgelachen om je peperdure computer.
- Als je een MS DOS computer hebt, wordt je uitgelachen om je achterlijkheid.
- Als je een werkstation hebt, word je uitgelachen om je onderontwikkelde besturingssysteem
- Als je een ander systeem hebt, wordt je uitgelachen omdat je achterlijke exoot gevonden wordt.

Axels theorie van het debuggen

Wanneer debuggen een methode is om fouten uit een programma te verwijderen, is programmeren een methode om fouten in te bouwen.

Wetten van de werkkamer (1)

1. Alle horizontale vlakken zijn in een mum van tijd bezaaid met troep.
2. De diskettes liggen daaronder.
3. Het dringend noodzakelijke handboek is nergens te vinden.
4. Tussen dat alles bevinden zich sigae om fouten in te bouwen.

Wetten van de werkkamer (2)

1. Alle horizontale vlakken zijn in een mum van tijd bezaaid met troep.
2. De diskettes liggen daaronder.
3. Het dringend noodzakelijke handboek is nergens te vinden.
4. Tussen dat alles bevinden zich sigan door mensen die op het punt staan het hoogste niveau van onbekwaamheid te bereiken.

Frankes bloementheorie

Waarmee je de bloemen ook water geeft, de helft stroomt steevast over de listings.

Leerstelling van het nut van een programma

- Wat je met een programma wil doen,
- staat niet in het handboek;
 - wordt pas in de update van het handboek uitgelegd.
 - wordt pas in de volgende versie van het programma opgenomen.

De vier programma-basisregels

1. Elk programma dat feilloos werkt, is verouderd.
2. Elk nuttig programma wordt veranderd.
3. Elke onzinnige mogelijkheid wordt ogenblikkelijk gedocumenteerd.
4. Elke fout wordt onmiddellijk als nieuwe functie opgenomen.

Wet van de computeraanschaf

1. Beweringen van de fabrikant over de prestaties moeten worden vermenigvuldigd met de factor 0,5
2. Beweringen van de gebruiker over de prestaties moeten vermenigvuldigd worden met de factor 0,25
3. Bijgeleverde handboeken en systeemdiskettes worden op het postkantoor onmiddellijk afgelegd en blijven daar onvindbaar.
4. Als je na lang zoeken eindelijk een computer hebt gekocht, wordt die de week daarop de helft goedkoper. Als alternatief verschijnt er een model dat voor dezelfde prijs dubbele prestaties biedt.

Theorie van de architecten en programmeurs

Als architecten op dezelfde manier zouden bouwen als programmeurs programma's schrijven, zou één enkele specht hele steden kunnen verwoesten.

Gods tegenwerping

Als God de mens had geschapen om computers te gebruiken, had hij hem zestien vingers gegeven.

De archiefmythe

Het gebruik van computers op kantoor zal papier overbodig maken.

De boekmythe

Er bestaat een begrijpelijk computerboek, waarmee ik greep krijg op mijn problemen.

De hardware-mythe

Liever nog een paar jaartjes wachten, tot computers echt uitgerijpt zijn.

De prijsmythe

Liever nog een paar jaartjes wachten, tot computers nog goedkoper zijn geworden.

De software-mythe

Liever nog een paar jaartjes wachten, tot de software echt uitgerijpt is.

De afdeling-specifieke afleiding van de dubbelwet van de complexe hardware

Als je op je afdeling een permanente uitvlucht voor je eigen fouten wil hebben, rust haar dan uit met computers.

De algemene computersmoes(1)

"Daar is mijn PC niet compatibel genoeg voor."

De algemene computersmoes(2)

"Daar is mijn PC te compatibel voor"

De bestandensmoes(1)

"Mijn harde schijf is vol"

De bestandensmoes(2)

"Daar is mijn harde schijf te langzaam voor."

De bestandensmoes(3)

"Mijn harde schijf heeft ineens beschadigde sectoren"

De ontzettend slappe smoes

"Ze hebben me niet verteld wat bij dit toetsenbord de bovenkant is."

Slotsom

In de strijd tussen jou en de digitale wereld kun je maar beter de kant van de digitale wereld kiezen

B. Happy

KGN promotie team

Julie als lid van de KGN mogen deel uitmaken van het KGN promotie team.

Het is gelukkig geen full-time job, want we hebben het waarschijnlijk toch wel druk genoeg. We zoeken een groep mensen die bereid zijn eventueel een stand te bemannen tijdens een beurs of ander evenement. Als je nu ja zegt betekent dat niet dat je straks nog aan die verplichting vastzit. Pas als de KGN naar een namelijk in het aansluiten van de te demonstreren computer(toepassing). Tijdens de beurs zelf promotiemateriaal uitdelen, eventueel wat vertellen. In overleg met andere KGN promotie team leden wie er bij de stand blijft en wie er even over de beurs wandelt. Aan het einde van de dag de zaken weer netjes afbreken.

Hoe wordt je KGN promotie team lid?

Een briefje, bericht of telefoontje naar Geert Stappers!

Voortgang KGN68k (deel 15)

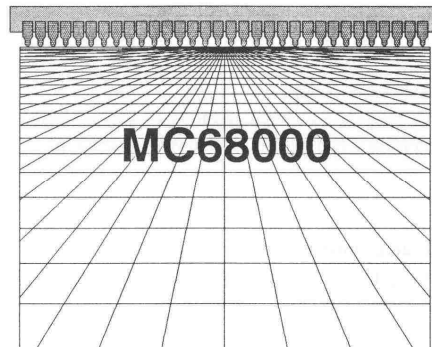
Opnieuw verslag over de voortgang van het KGN68k project.

Hardware

Het principe van een bi-directional latch als databuffer op de I/O-kaarten is ons goed bevallen. De MC68030 processor is namelijk heel erg snel met het weg halen van de signalen. Met die tussengeheugens wordt het geheel een stuk robuuster. Op de processorkaart hebben we nu hetzelfde type chips toegepast. Hier moesten echter 4 devices (32-bits) in plaats van 1 vervangen worden. Dat draaide naar behoren. De volgende stap tijdens die testsessie was de DRAM kaart. Ad Brouwer, de ontwerper, vertelde ons dat zonder DRAM ook al een groot gedeelte van de schakeling te controleren is. Zag er goed uit. De eerste Single In line Memory Module, SIMM, er in. Meteen vierhonderd gulden DRAM er op vonden we wat voorbarig. Om de vier geheugen lokaties konden we toen schrijven en correct teruglezen. De overige drie SIMMs er in. Toen konden we dus longs (32bit breed) 'met de hand' lezen & schrijven. Grote glimlachen op de gezichten. Complimenten voor de ontwerper. Toen verder getest. De vreugde verdween. Het blijkt dat het niet helemaal goed werkt. 17 april hebben we het niet meer in orde gekregen. Wel een afspraak gemaakt voor 1 mei. Stom hè, als het spannend wordt moet er aan de μ P Kenner gewerkt worden. Misschien kan het in een aanvulling verteld worden hoe de testsessie van 1 mei afgelopen is.

Software

De software is ook goed op gang. De KGN68k assembler en Linux omgeving zijn beter op elkaar ingespeeld. Een anekdote: Naar aanleiding van een upgrade in de assembler vertelt Pieter dat de oude foutmeldingen van de linker weg zijn en dat er nieuwe foutmeldingen voor in de plaats zijn gekomen. Gert vertelt dat fouten wel weg zijn uit de assembler GNU output, maar dat Pieter nu verder is gegaan met testen en dan nieuwe c.q. andere fouten tegen komt. Ze zijn er samen toch uitgekomen. Nu wordt er zelfs over features gesproken. Afgelopen woensdag hebben we een EPROM geprogrammeerd met behulp van een file die uit de Linux omgeving kwam. Dit was de eerste echte test van de combinatie KGN68k assembler en Linux ld68 linker. Met de melding 'Hello world' en 'Bus error at 0000000R' waren die avond wel tevreden. Een



mogelijk oorzaak en een remedie hiervoor hebben we ook al, maar een herkansing is er nog niet geweest. De disassembler is in een zodanig stadium dat hij geïntegreerd kan worden. Wim kan ondertussen verder met de uitbreiding van herkenning van 68030 instructies. Uitwisseling van versies verloopt prima via The Ultimate, waar de KGN68k werkgroep een eigen area heeft.

Printed Circuit Board

Het printontwerp heeft de laatste tijd op zijn gat gelegen. Door de verschuivingen binnen de werkgroep is het bij mij terecht gekomen. Niet echt de ideale oplossing, maar wel een oplossing. Als de KGN68k dadelijk bij de software mensen is, zal ik er meer tijd voor hebben.

Marketing

Ondertussen is het zo ver, dat we na moeten denken hoe we de markt opgaan. Een papieren tijger is de KGN68k allang niet meer. Nu kunnen we zelfs de basisset compleet tonen. Is er iemand binnen de KGN die zijn marketing plannen wil uitproberen? Uitdagen wil ik jullie niet, maar wel de kans geven om het te proberen. Het technische werk is gebeurd, nu kunnen we ons dus opmaken voor de volgende hindernis.

Geert Stappers (04781-)41279

Is er iemand binnen de KGN die zijn marketing plannen wil uitproberen?

Vijfde les 'C'

Wij gaan ons bezighouden met lezen en schrijven van files in 'C'. Er zullen twee manieren worden behandeld.

Re-direction

De eerste manier is gebaseerd op een truc. Hierbij maken wij gebruik van Re-direction. Dit is een bekend DOS command en - helaas - alleen werkend onder DOS (Ik gebruik vrijwel uitsluitend OS/2 als operating systeem).

Bij DOS wordt re-direction aangegeven met het teken: > om gegevens naar een file te schrijven en met het teken: < om gegevens van een file te halen. Van deze mogelijkheid gaan wij nu gebruik maken:

Je begint met het schrijven van een simpel programma: (stel, het programma heet: INVOER)

```
main()
{
    while ( getche() != 'X' );
}
```

Voordat je dit programma uitvoert, eerst achter de prompt van DOS intikken:

```
C> INVOER > file.txt
C>
```

Wanneer je nu intikt: 'Dit is een test. X', gebeurt er helemaal niets!

Tik nu in:

```
C> type file.txt
Dit is een test. X
```

Er is dus data in de file opgeslagen...

Helaas is ook de letter 'X' meegegaan. Deze letter was nodig om de while() loop af te breken. Dit is niet te veranderen maar het is wel mogelijk een teken te kiezen dat minder opvallend in beeld verschijnt.

Dit is het gemodificeerde programma:

```
main()
{
    while ( getche() != '\x1A' );
}
```

Dit teken is het 'Ctrl - z' teken en kan ik niet printen.

Zo, nu heb je iets in een file geplaatst en zou het leuk zijn als het ook weer gelezen zou kunnen worden door een programma. Ik demonstreer dat aan de hand van twee nieuwe programma's. Het eerste programma noem ik 'codeer' en leest de records uit programma 'invoer', bouwt dit om naar onleesbaar schrift en plaatst het daarna in een nieuwe file.

Het tweede programma leest de file met daarin gekodeerde records, bouwt dit weer terug in normale tekst en plaatst deze in een file:

Het eerste programma ziet er als volgt uit en noem ik 'codeer':

```
#define CTL_Z '\x1A'
main()
{
    char ch;
    while ( (ch = getch()) != CTL_Z )
        putchar ( ch + 1 );
    putchar ( CTL_Z );
}
```

Schrijf in een nieuwe file m.b.v. programma 'invoer' (het tweede, gemodificeerde voorbeeld hierboven) de volgende zin:

```
C> type file1.txt
Dit is te gek! (afsluiten met Ctrl-z)
```

Enter nu:

```
C> codeer < file1.txt > file2.txt (file1.txt bevat zin)
```

en lees file 'file2.txt' uit d.m.v. 'type...'

Wat je ziet op het scherm is onleesbaar!

- De inhoud van file1.txt is door het programma ingelezen, omgezet en geplaatst in file2.txt.
- De plaats in de ASCII tabel van elke letter is met 1 verhoogd (ch + 1)
- Voor main() staat een #define. Hierin wordt een variabele assigned die in het programma meerdere keren voorkomt. Dit behoort ook tot de compiler aanwijzingen en deze 'weet' nu dat CTL_Z in werkelijkheid een '\x1A' betekent.

Nu het beloofde programma dat van de inhoud van file2.txt weer leesbaar schrift maakt: (programma heet: terug)

```
#define CTL_Z '\x1A'
main()
{
    char ch;
    while ( (ch = getch()) != CTL_Z )
        putchar ( ch - 1 );
    putchar ( CTL_Z );
}
```

Enter nu:

```
C> terug < file2.txt > file3.txt
C> type file3.txt
Dit is te gek!
```

Duidelijk is te zien dat dit programma sterk lijkt op 'codeer' maar nu de plaats in de ASCII tabel van elk karakter met 1 vermindert waardoor weer een leesbare zin ontstaat.

Character I/O (schrijven)

Nu pakken we het probleem wat serieuzer aan en gaan eens kijken op welke wijze wij dit met 'C' zouden kunnen oplossen:

```
#include "stdio.h"
main()
{
    FILE *fptr;
    char ch;
    fptr = fopen ("test.txt", "w");
    while ( (ch = getch()) != '\r')
        putc ( ch, fptr );
    fclose ( fptr );
}
```

- het #include statement is noodzakelijk
- #include is nu gebruikt met " i.p.v. < > . Waarom dat is valt buiten het bestek van deze cursus.
- FILE is een speciaal statement dat dient voor de filedefinitie (qua doel vergelijkbaar met #define uit het vorige voorbeeld)
- FILE moet in hoofdletters!
- fopen() opent nieuwe file en zorgt voor bewerkingen
- letter "w" in functie fopen() bepaalt wat gebeuren moet
Deze letters worden hierna behandeld; deze letter altijd tussen " . "
- fclose() sluit de file
- putc() schrijft data naar file
- Let op! putc() lijkt op putchar()
- i.p.v. getch() mag ook getche() worden gebruikt (wat was het verschil?)
- het teken: '\r' in het while() statement is behandeld in les 1. Het betekent: 'Carriage Return' en ontstaat op het moment dat de ENTER of de Carriage Return toets wordt gebruikt. Je kunt dus met dit programma slechts 1 record, bestaande uit een of meerdere woorden, intikken.

De functie fopen() opent een file met de naam zoals benoemd in FILE. Er kan gekozen worden voor een van de volgende modes:

"r"	Open voor lezen. File moet al bestaan
"w"	Open voor schrijven. Oude data wordt overschreven. Indien file niet bestaat, wordt nieuwe file gemaakt
"a"	Open voor data toevoegen. Nieuwe data wordt toegevoegd aan eind van bestaande data. Als file niet bestaat, wordt nieuwe gemaakt
"r +"	Open file voor lezen en schrijven. File moet al bestaan
"w +"	Open file voor lezen en schrijven. Nieuwe file wordt gemaakt indien deze nog niet bestaat, doch alleen bij schrijven. Oude gegevens worden overschreven
"a +"	Open file voor lezen en toevoegen. Nieuwe file wordt gemaakt indien nodig

Nu een voorbeeld voor het lezen van een hele file waarbij getest wordt op EOF konditie. Let ook op de nieuwe functie `getc()`.

```
#include <stdio.h>
main()
{
    FILE *fptr;
    int ch;
    fptr = fopen ( "test.txt", "r" );
    while ( (ch = getc( fptr )) != EOF )
        printf( "%c", ch );
    fclose ( fptr );
}
```

- EOF is een systeemvariabele en heeft een waarde van -1. Daarom altijd declareren als `int` !!
(Als je het niet declareert als een `'int'`, is de compiler zo verstandig aan te nemen dat je toch een `'int'` bedoelt! Helaas niet bij elke compiler)
(EOF wordt automatisch gevonden bij het bereiken van het eind van het laatste record. In ons geval is dit 1 record)
- `#include <stdio.h>` moet
- `getc()` is een nieuwe functie. Het haalt een karakter op uit de file overeenkomstig de definitie van `fptr`

In bovenstaand voorbeeld wordt karakter voor karakter gelezen. Natuurlijk is er ook een mogelijkheid meerdere records naar een file te schrijven:

```
#include <stdio.h>
main()
{
    FILE *fptr;
    char string[81];
    fptr = fopen ( "test.txt", "w" );
    while ( strlen ( gets( string ) ) > 0 )
    {
        fputs ( string, fptr );
        fputs ( "\n", fptr );
    }
    fclose ( fptr );
}
```

- `strlen()` is een nieuwe functie. Uitgaande van de naam is het duidelijk wat deze functie doet - het bevat de lengte van de string (= record) en deze inhoud wordt gebruikt in de `while()`
Zodra je niets intikt en alleen ENTER of Carriage Return geeft, is de lengte van de string 0 en wordt de `while` loop afgebroken. En daarmee ook het programma.
- een andere nieuwe functie is `gets()`. Deze functie haalt een string op
- `fputs()` is ook nieuw en plaatst de inhoud van de string op het scherm nadat eerst gesprongen is naar de volgende regel.
- let op de optie in `fopen()` nodig voor het schrijven
- **BELANGRIJK!**
De nieuwe functies werken nauw samen met de functie: `fopen()`. `gets()` plaatst de inhoud van `fptr` in `string`

waarna `strlen()` de lengte bepaalt. `fputs()` plaatst de inhoud van `string` in een veld van `fptr`, waarna het wordt weggeschreven naar het scherm.

Nu een voorbeeld van het lezen van strings:

```
#include <stdio.h>
main()
{
    FILE *fptr;
    char string[81];
    fptr = fopen ( "test.txt", "r" );
    while ( fgets ( string, 80, fptr ) != NULL )
        printf ( "%s", string );
    fclose ( fptr );
}
```

Ook nu leer je weer een nieuwe functie: `fgets()`. Deze functie hakt de file in mootjes van 80 posities lang en plaatst deze in 'string'.

Dat gaat net zo lang door totdat 'string' leeg is (NULL). `printf()` zorgt voor de weergave op het scherm van de inhoud van 'string'.

De NULL zijn wij al in een eerdere les tegengekomen. Het is een z.g. systeem variabele.

De oplettende leerling zal het opvallen dat 81 posities zijn ge-declareerd voor 'string'. De 81e positie van elk record bevat het LF/CR karakter welke het eind aangeeft van een record. Dat is nodig om de records uit elkaar te kunnen houden.

Nesting van statements

Ik kom nog even terug op een van de voorbeelden. Hierbij is het `while()` statement opgebouwd uit enkele andere functies. Ook dit wordt 'NESTING' genoemd.

Als we deze constructie willen begrijpen gaat dat het beste als je van binnen uit naar buiten werkt. Dus tel je de openingshaken van links naar rechts af totdat je een sluihaak aantreft. Tussen de meest rechtse openingshaak en de eerste sluihaak zit het begin (Kunnen wij eindelijk onze vergaarde kennis van de basisschool gebruiken!).

Daar staat: 'string', welke niet op zichzelf kan bestaan in 'C'. Dus verbreden wij onze blik totdat een functie is gevonden: `gets(string)`.

Dat is duidelijk. Vervolgens kijken wij nog wat verder en vinden: `strlen ( gets(string))`. Let op dat je altijd een gelijk aantal openings- en sluihaken ziet. Het resultaat van de `gets(string)` functie wordt nu gebruikt als input voor de `strlen()` functie.

Zo kun je verder gaan in het uiteenrafelen van een genest statement.

Uiteraard is het ook mogelijk hiervoor aparte regels te schrijven. Dat maakt het programma overzichtelijker maar heeft als nadeel dat er vaak extra variabelen moeten worden benoemd.

Wil je zelf een genest statement bouwen, werk dan altijd van binnen uit naar buiten toe. Op die manier begrijp je wat je aan het doen bent en vergeet je ook niet zo gemakkelijk een haak.

```
#include <stdio.h>
main()
{
    FILE *fptr;
    char string[81];
    fptr = fopen ("test.txt", "w");
    while ( strlen ( gets( string) ) > 0 )
    {
        fputs ( string, fptr);
        fputs ( "\n", fptr);
    }
    fclose ( fptr );
}
```

Ik doe nu een poging het voorbeeld te herschrijven zonder nesting:

```
#include <stdio.h>
main()
{
    FILE *fptr;
    char string[81];
    int lengte;
    lengte = 1
    fptr = fopen ("test.txt", "w");
    while lengte > 0 )
    {
        gets( string)
        lengte = strlen (string )
        fputs ( string, fptr);
        fputs ( "\n", fptr);
    }
    fclose ( fptr );
}
```

Ik weet niet of dit werkt (probeer het zelf maar). Je ziet dat wij nu de lengte eerst moeten instellen op een waarde groter dan 0 om te bereiken dat de eerste keer dat de while() functie wordt getest, het programma door-gaat.

Daarna wordt een string gelezen en geplaatst in variabele: 'string'. Vervolgens wordt de lengte van de string bepaald en geplaatst in: 'lengte'.

De string wordt als record in een file geplaatst en het hele circus begint opnieuw.

Bij het intikken van een lege string gebeurt er iets vervelends. Deze lege string wordt weggeschreven naar de file. Pas bij het opnieuw ingaan van de loop wordt de while() functie getest en de loop afgebroken. Dat zou ook wel kunnen worden ondervangen door een extra statement te plaatsen onder 'lengte = strlen (string)' waarmee de lengte wordt getest en indien '0', de loop wordt afgebroken (misschien met een 'break?').

Ik denk dat nu wel duidelijk is dat nesting voordelen biedt.

Oefeningen

De voorbeelden uit deze les kun je intikken en uitvoeren. Daarna is het misschien interessant een combinatie te schrijven van schrijven en daarna lezen van een file....

Hans van Boheemen

Afrekenen met DOS

Peter Norton blijft zijn gelijknamige DOS-hulpmid- delen verder ontwikkelen. Het besturingssysteem DOS dreigt het stiefkindje van de Norton-familie te worden.

Norton Disk Doctor is en blijft de hoofdmoot van Norton Utilities. In de jongste uitvoering van NU, versie 8.0, blijkt hij zelfs in twee versies te worden aangeboden: één voor DOS en een andere voor Windows. De DOS-versie heeft niet zoveel wijzigingen ondergaan en is vooral toegespitst op het opfrissen van gegevens, de optimalisatie en het repareren van schijven. NDD met het Windows-gezicht deed zijn intrede met Norton Desktop voor Windows 3.0.

Het is een licht verbeterde uitvoering van de schijvendokter. Norton Disk Doctor Windows doet zijn werk voortaan volledig op de achtergrond. De grafische animatie en de muzikale begeleiding zijn nog steeds van de partij, een kwestie van het aangenamen met het nuttige verenigen.

De huidige versie ondersteunt ook harde schijven gecompri- meerd met Stacker 2.0, 3.0 en 3.1, SuperStor 2.0x, Pro 1.0x, SuperStor DR-DOS 6, IBM DOS 6.1 SuperStor/DS en DoubleSpace van MS-DOS 6.x. Het is ongetwijfeld één van de meest complete programma's op dit gebied. De bèta-versie betekent nog niet de 32-bit bestandstoegang van Windows 3.11. Het is nog niet ingesteld op beheer van de File Allocation Tables (FAT) in de protected modus, dit in tegenstelling tot Windows.

Masochisme

SpeedDisk, de module waarmee harde schijven gedefragmenteerd kunnen worden, bestaat zowel voor DOS als voor Windows. Ook dit programma kan zijn werk in de achtergrond verrichten. Het is echter nog maar de vraag of het nu wel zo verstandig is een dergelijk pakketje voor Windows te gebruiken. De inhoud van schijfsectoren verplaatsen in een multi-tasking omgeving ruikt naar masochisme.

Zodra er sprake is van verminkte informatie op de harde schijf, ligt het niet zo voor de hand dat de gebruiker Windows opnieuw installeert om een Master Boot Record of een beschadigde FAT te repareren. Nou goed dan, Symantec ontsnapt niet aan de Windows-manie en de nieuwe gebruikers zullen minder

snel worden afgeschrikt door een gebruikersvriendelijke grafische interface.

Jacht op overtollige INI's

Andere nieuwigheden onder Windows kunnen ons meer overtuigen. Zij hebben te maken met een complexer wordend probleem, namelijk het onderhoud van de configuratiebestanden. Windows bevat nog steeds geen enkel standaard installatieprogramma. Een willekeurig Setup.exe of Install.exe van om het even welke softwareproducent strooit maar kwistig DLL- en INI-bestanden rond en wijzigt naar hartelust vitale bestanden zoals System.ini of Config.sys.

Maar met Norton Utilities 8.0 doet 'Tracker' zijn intrede. Het programma bewaart op regelmatige tijdstippen een kopie van de systeembestanden (onder andere Config.sys, System.ini, Win.ini en Autoexec.bat) zodat een overhaaste wijziging van een configuratiebestand wordt opgemerkt. De kopie wordt genomen telkens wanneer Tracker wordt gestart of beëindigd. Een andere mogelijkheid is het zelf instellen van tussenpozen (via een config-

uratiemenu) waarop Tracker zijn werk moet doen. De gebruiker kan een indrukwekkend aantal versies van systeembestanden bewaren en op verzoek terughalen. Tracker is bijna de tegenhanger van het Version Control System, een bekend gegeven voor programmeurs onder onze lezers. Het idee is uitstekend. De gebruiker wordt gevraagd twee versies van hetzelfde bestand te vergelijken. Daarvoor beschikt hij over het Windows-programma Fcompare dat de varianten of afleidingen van de bewuste bestand naast elkaar afbeeldt.

De verschillen worden in een kleur aangegeven, de verplaatste segmenten in een andere kleur. De gebruiker kan onmiddellijk wijzigingen aanbrengen aan een versie naar keuze, ondermeer door middel van de 'drag and drop'-techniek.

Gelikt controlepaneel

IniEdit is, zoals de naam doet vermoeden, een editor voor INI bestanden. Er bestaan een tiental vergelijkbare public domain toepassingen. Het werkvenster wordt in tweeën verdeeld. Het bovenste gedeelte toont de bestanddelen van een INI bestand. Terwijl het onderste gedeelte een gedetailleerde beschrijving geeft van elk item. Bij elke parameter be-

De inhoud van schijfsectoren verplaatsen in een multi-tasking omgeving ruikt naar masochisme.

schikt de gebruiker over een contextgevoelige en uitgebreide helpfunctie.

Een ander NU 8.0 onderdeel, IniTuner. Het programma vervangt het Windows controlepaneel dat tegelijk wordt uitgebreid en vereenvoudigd. Het veranderen van systeminstellingen gebeurt naar keuze via de traditionele Windows-methode (grafisch) of op het niveau van de INI-bestanden.

Voor elke parameter van de INI-bestanden, wordt de gebruiker ingelicht over de huidige waarde, de standaardwaarde en de syntax. Teacher, een soort helpbestand, zet een grove nalatigheid recht van Microsoft: het biedt namelijk een complete documentatie over de Windows configuratiebestanden. Teacher is even compleet als de Windows Resource Kit en veel gebruikervriendelijker.

Norton pakt de resources aan

System Watch is een praktisch hulpmiddel dat verschillende aspecten van de Windows Resources 'bewaakt'. In geen tijd ziet men het gebruik van de processor, het beschikbare fysieke en virtuele geheugen, het aantal bestanden of handles in gebruik, de resterende harde schijf-ruimte of de onbezette User Resources en GDI's. Als de gebruiker het wenst, kunnen de tijdgebonden verschillen van de

verschillende waarden in grafiekvorm worden weergegeven. Hierna kan de gebruiker een alarmdrempel instellen waaronder een foutboodschap en/of een bepaald geluidsbestand wordt geactiveerd.

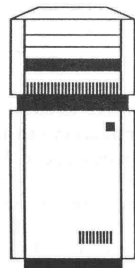
De andere wijzigingen zullen amper worden opgemerkt: zo kan het bestandsdoktertje FileFix voortaan Excel 5 en Wordperfect 6 bestanden aan. Norton Diagnostics geeft nu een meer gedetailleerd rapport over interrupts of joystick-poorten. Een ander hulpje, WipeInfo, ondersteunt in zijn nieuwste reïncarnatie ook gecomprimeerde harde schijven. De traditionele DOS-gerichte Norton Utilities gooien het steeds meer over een Windows-boeg. Symantec beconcurrert zowaar zijn eigen Norton Desktop voor Windows.

De nieuwigheden in Norton Utilities 8.0 zullen slechts door de Windows-freaks enthousiast worden ontvangen. Andere gebruikers hebben er niet zoveel boodschap aan. Het is niet makkelijk nog iets toe te voegen aan de zoveelste versie van een programma, maar oompje Norton is er toch opnieuw in geslaagd ons in positieve zin te verrassen.

Voor u gelezen in CM Corporate: een artikel van Eric Lapaille: Norton Utilities 8.0 BETA beproefd.

De andere wijzigingen zullen amper worden opgemerkt.

The Ultimate The BBS for all systems



Telefoon:
053-303902, 053-328506 of
053-327457

- 053-303902 (2 lijnen!) -
V22, V22bis, V23, V32bis, HST/14k4, V42bis, MNP5
- 053-328506 -
V22, V22bis, V23, V32bis, V.Fast (28k8), V42bis, MNP5
- 053-327457 -
V21, V22, V22bis, V32bis, V42bis, MNP5

Gearriveerd: 4DOS versie 5.0

Tijdens de laatste bestuursvergadering bleek dat op ons aller bulletin board al weer enkele maanden de laatste versie van de inmiddels bekende vervanger van COMMAND.COM te vinden was: 4DOS in de versie 5.0. Het is derhalve dus hoog tijd voor een verhaaltje over wat er nu weer bijbedacht is. Want ondergetekende was al zeer tevreden met de voorgaande versies (4.01 en 4.02, de laatste speciaal aangepast aan de nukken van MS-DOS 6.0). Hier komt het: dit is er veranderd.

Positieve discriminatie

Zij die 4DOS al langer genieten zijn al gewend aan de zeer krachtige manier waarop je in 4DOS uit een hele stapel files precies die kunt specificeren die je wilt hebben. Daarvoor heeft 4DOS onder andere krachtiger wildcards en kun je deze ook omdraaien met het woord EXCEPT. In 5.0 is dit feest nog verder uitgebreid: je kunt files nu ook aanwijzen op grond van hun grootte, hun datum of hun tijd. Alsof dat allemaal nog niet genoeg is, kun je ook nog een bereik opgeven waarin de datum, tijd of grootte moet liggen. En tot overmaat van ramp mag je tijd, datum en grootte ook nog combineren. Voorbeeldje:

```
DIR [/d-2,-1] /s[2048,65535]
```

Dit geeft een directory van alle files waarvan de datum ligt tussen eergisteren (vandaag min 2) en gisteren (vandaag min 1), en de grootte tussen 2 kbyte en 64 kbyte.

De tijd/datum/grootte-bereiken werken voor de commando's ATTRIB, COPY, DEL, DESCRIBE, DIR, FOR, LIST, MOVE, RD, REN, SELECT en TYPE.

Erfgenamen

Al eens ontdekt? Een nieuwe 4DOS shell erft de aliases en history van de vorige shell die hem heeft aangemaakt niet! Dus dat ingewikkelde commando van 2 shells en 7 commando's geleden staat mooi niet in je history!. Met versie 5.0 is dat dus niet langer nodig. In het stuurbestandje 4DOS.INI kun je keurig opgeven dat de history respectievelijk de aliases moet worden doorgegeven aan de volgende shells met LocalAliases=Yes/No en LocalHistory=Yes/No. Wil je de history en de aliases niet in lower memory maar in een UMB hebben? Kan ook. Zet even UMBAlias=Yes en UMBHistory=Yes in 4DOS.INI en reboot. Klaar!

Geheimpjes bewaren

4DOS had al snellere batch-files: gewoon .BAT vervangen door .BTM en 4DOS voert de file niet vanaf disk, maar vanuit het geheugen uit, zodat het doodleuk veel harder loopt. In versie 5.0 kun je met het hulpprogrammaatje BATCOMP .BTM-files comprimeren, waardoor ze niet alleen kleiner, maar ook nog onleesbaar voor anderen worden. Schitterend voor netwerkbeheerders die niet willen dat domme users aan hun batch-files knoeien met tekstverwerkers in plaats van ASCII-editors en zo! Voor 4DOS 5.0 maakt het niet uit of een .BTM-file gecomprimeerd is of niet: in beide gevallen gebeurt er het zelfde: snellere uitvoering van een batch-file.

ANSI.SYS

4DOS gebruikte voor een aantal zaken ANSI.SYS als je tenminste voortdurend aan het feit herinnerd wilde worden dat je een kleurscherm had door middel van gekleurde directories, en batch-files die complete regenbogen op je scherm kalkten. De meeste van die kleurrijke mogelijkheden werken nu ook zonder dat ANSI.SYS geladen is. De manier waarop 4DOS een ANSI-driver detecteert is trouwens betrouwbaarder en ook netter geworden.

MS-DOS 6.0 en 6.2

Deze versie van 4DOS is geheel aangepast aan MS-DOS 6.0 en 6.2/6.21. Zoals bekend kunnen deze

DOS-versies AUTOEXEC.BAT en CONFIG.SYS stap voor stap uitvoeren. Om dit ook met 4DOS voor de bestanden 4START.BAT en AUTOEXEC.BAT mogelijk te maken, ondersteunt de startregel in het SHELL-commando van CONFIG.SYS een nieuwe switch: /Y. Normaal wordt deze switch niet gebruikt, maar zal MS-DOS hem automatisch toevoegen als na het starten op F8 gedrukt wordt.

De tweede wijziging is de ondersteuning van de /C switch bij het DIR-commando, dat de mate van compressie van een gecomprimeerde schijf toont. Dat werkt dus niet meer in MS-DOS 6.21... Overigens ondersteunt 4DOS 5.0 schijven die gecomprimeerd werden met Stacker, SuperStor en DblSpace.

Windows zonder windows!

Er zijn in 4DOS 5.0 twee nieuwe vensters gedefinieerd. Naast het commandohistory-venster (PgUp) is er nu ook een directory-history-venster met alle recent gebruikte directories erin (via Ctrl-PgUp,

compleet met de voor 4DOS nodige backslash aan het einde) en een filenaam completeer-venster. Dat laatste werkt als volgt: type de eerste paar letters van een filenaam. Druk dat op F7 of Ctrl-Tab. Er verschijnt nu een venster met alle beschikbare filenamen die beginnen met de getypte letters. Wijs de gewenste filenaam aan en druk op Enter. Het venster verdwijnt weer en de filenaam verschijnt op de commandoregel op de plaats waar eerst de eerste letters stonden.

Verbeteringen aan bestaande commando's

COPY, DEL, MOVE en REN hebben een nieuwe switch: /T van Total. Vroeger kon je alleen kiezen tussen alle filesnamen tonen (wel duidelijk, maar vertraagt soms nogal) of met /Q(uiet) helemaal niets laten zien. De tussenweg die COMMAND.COM bewandelt, namelijk geen filenamen tonen maar wel de totalen was er nog niet. Nu dus wel. DEL en MOVE tonen nu bovendien de schijfruimte die gewonnen werd als gevolg van de verwijderde bestanden.

Ook DIR heeft naast de al genoemde /C een aantal nieuwe switches. Het alfabet is nu bijna helemaal gebruikt... /H geeft een gewone directory, maar toont . en .. niet. /Itekst toont alleen die files waarvan de description overeenkomt met de tekst achter de /I.

DRAWBOX kan de getekende box nu laten opzoomen.

INKEY accepteert nu /C (maak eerst de keyboardbuffer leeg voordat er input gevraagd wordt) en /P (echo de ingetoetste tekens niet op het scherm). INPUT accepteert de deze switches ook, met daarnaast /E (geef alvast een waarde laat toe

deze te veranderen) en /L (beperk de input tot een maximum op te geven lengte).

KEYBD is een nieuw commando, dat toestaat de status van CapsLock, NumLock en ScrollLock te veranderen in bijvoorbeeld batch-files.

LIST is geheel opnieuw geschreven. Het is allereerst veel sneller geworden. Verder ondersteunt het nu (eindelijk) hexdump mode, inclusief zoeken en printen. Daarnaast kan er een venstertje geopend worden waarin de complete filenaam met pad, de grootte en de tijd/datum getoond worden. Tenslotte geeft LIST nu ook regel- en kolomnummers.

Variabelen en functies

Dit was nog niet alles. 4DOS had ook al een groot aantal variabelen en functies waarmee als handige batch-schrijver de schitterendste zaken kan opvragen en uitvoeren. Onnodig te zeggen dat deze verzameling in 4DOS 5.0 ook weer groter is geworden. Het gaat een beetje ver om hier de hele waslijst op te voeren.

Hoe kom ik eraan?

Eenvoudig: op het BBS staan in de 4DOS-area 3 ZIP-files:

4DOS5A.ZIP, 4DOS5B.ZIP en 4DOS5C.ZIP. Samen vormen zij 4DOS 5.0 en de bijbehorende documentatie. Voor bezitters van een geregistreerde 4DOS 4.0X is er een patcher voor je bestaande BRAND.EXE, zodat je de gedownloade 4DOS.COM van je eigen naam en serienummer kunt voorzien. Eigenlijk is er maar 1 nadeel aan dit alles: 4DOS en zijn documentatie is weer behoorlijk groter geworden....

Nico de Vries

LIST ondersteunt nu (eindelijk) hexdump mode.

Van de voorzitter

Dat men volop belangstelling toonde in de technische kant van de computers op die beurs, had ik jullie al verteld. Het doet ons als bestuur goed, als blijkt dat er interesse is voor het beleid dat er uitgespeld.

Beleid is inderdaad vooral een bestuurstaak, maar suggesties zijn natuurlijk welkom. Een paar jaar terug was er een idee dat door een lid ingebracht werd. Toen zag het bestuur niet het nut er van in. Nu zijn we er alsnog mee begonnen om het alsnog te realiseren. Omdat we nu als bestuur wel het nut er van inzien. Als je zelf ook een idee hebt laat het dan weten. En het hoeft niet twee jaar zijn tijd vooruit te zijn.

Om contact met de KGN te vergemakkelijken is nu ook de entreeprijs voor niet-leden vervallen. Tijdens de discussie op de bestuursvergadering was er het standpunt dat er een verschil moet zijn tussen leden en niet-leden. Een lid heeft wel degelijk zijn voordelen, (blad, BBS, werkgroep, gezamenlijke componenten inkoop, promotie team, data sheet service, etc.) was hier het antwoord op. Nu heb je als lid een reden te meer om je collega, klasgenoot e.d. mee te vragen naar een bijeenkomst.

Nu Henk Speksnijder in het bestuur zit en Nico weer terug is, is er ook meer aandacht voor DOS65. Prachtig wat een enthousiasme Henk heeft. Als dat een afspiegeling van de werkgroep DOS65 is, komt dat wel goed, ook al gaan we nu de zomer in.

De eerste vrijwilligers voor het KGN promotie team hebben zich al gemeld. Hun voorwaarden waren heel reëel. Bij niet al te hoge eisen mag ook jij, als KGN-lid, deel nemen aan het KGN promotie team.

Er is ook meer aandacht voor DOS65.

Niet alles is rozengeur & maneschijn. Kopij komen we nog steeds te kort. Naar een mogelijke oorzaak kunnen we alleen maar gissen. Bescheiden zijn is op zich een goede deugd. Bij het schrijven voor de μ P Kenner mag je royaal zijn met informatie. Delen met anderen vind ik nog beter dan bescheiden zijn.

Jullie Voorzitter,

Geert Stappers

KGN promotie team

Jullie als lid van de KGN mogen deel uitmaken van het KGN promotie team.

Het is gelukkig geen full-time job, want we hebben het waarschijnlijk toch wel druk genoeg. We zoeken een groep mensen die bereid zijn eventueel een stand te bemannen tijdens een beurs of ander evenement. Als je nu ja zegt betekent dat niet dat je straks nog aan die verplichting vastzit. Pas als de KGN naar een namelijk in het aansluiten van de te demonstreren computer(toepassing). Tijdens de beurs zelf promotiemateriaal uitdelen, eventueel wat vertellen. In overleg met andere KGN promotie team leden wie er bij de stand blijft en wie er even over de beurs wandelt. Aan het einde van de dag de zaken weer netjes afbreken.

Hoe wordt je KGN promotie team lid?

Een briefje, bericht of telefoontje naar Geert Stappers!

Informatie

De μ P Kenner (De microprocessor Kenner) is een uitgave van de KIM gebruikersclub Nederland. Deze vereniging is volledig onafhankelijk, is statutair opgericht op 22 juni 1978 en ingeschreven bij de Kamer van Koophandel en Fabrieken voor Hollands Noorderkwartier te Alkmaar, onder nummer 634305. Het gironummer van de vereniging is 3757649.

De doelstellingen van de vereniging zijn sinds 1 januari 1989 als volgt geformuleerd:

- Het vergaren en verspreiden van kennis over componenten van microcomputers, de microcomputers zelf en de bijbehorende systeemsoftware.
- Het stimuleren en ondersteunen van het gebruik van micro-computers in de meer technische toepassingen.

Om deze doelstellingen zo goed mogelijk in te vullen, wordt onder andere 5 maal per jaar de μ P Kenner uitgegeven. Verder worden er door het bestuur per jaar 5 landelijke bijeenkomsten georganiseerd, beheert het bestuur een Bulletin Board en wordt er een softwarebibliotheek en een technisch forum voor de diverse systemen in stand gehouden.

Landelijke bijeenkomsten:

Deze worden gehouden op bij voorkeur de derde zaterdag van de maanden januari, maart, mei, september en november. De exacte plaats en datum worden steeds in de μ P Kenner bekend gemaakt in de rubriek Uitnodiging.

Bulletin Board:

Voor het uitwisselen van mededelingen, het stellen en beantwoorden van vragen en de verspreiding van software wordt er door de vereniging een Bulletin Board (BBS) beschikbaar gesteld.

De telefoonnummers zijn: 053-328506, 053-303902 of 053-327457.

Software Bibliotheek en Technisch Forum:

Voor het beheer van de Software Bibliotheek en technische ondersteuning streeft het bestuur ernaar zgn. systeemcoördinatoren te benoemen. Van tijd tot tijd zal in de μ P Kenner een overzicht gepubliceerd worden. Dit overzicht staat ook op het BBS.

Correspondentie adres

Alle correspondentie betreffende verenigingszaken kan gestuurd worden aan:

KIM Gebruikersclub Nederland
Postbus 1336
7500 BH Enschede

Het Bestuur

Het bestuur van de vereniging wordt gevormd door een dagelijks bestuur bestaande uit een voorzitter, een secretaris en een penningmeester en een viertal gewone leden.

Geert Stappers (Voorzitter, KGN/68k coördinator)
Engelseweg 7
5825 BT Overloon
Telefoon 04781-41279

Jacques H.G.M. Banser (penningmeester)
Haaksbergerstraat 199
7513 EM Enschede
Telefoon 053-324137

Gert van Opbroek (secretaris)
Del Del 16
5071 TT Udenhout
Telefoon 04241-3795

Jan Veninga
Klimopstraat 51
7601 SJ Almelo
Telefoon 05490-27910

Henk Speksnijder
Albert Cuijpsstraat 43
2902 GA Capelle aan den IJssel
Telefoon 010-4586879

Nico de Vries (redactie μ P Kenner)
Van der Waalsstraat 46
2984 EP Ridderkerk
Telefoon 01804-29207

Ereleden:

Naast het bestuur zijn er een aantal ereleden, die zich in het verleden bijzonder verdienstelijk voor de club hebben gemaakt:

Erevoorzitter:
Siep de Vries

Ereleden:
Mevr. H. de Vries-van der Winden
Anton Müller
Rinus Vleesch Dubois