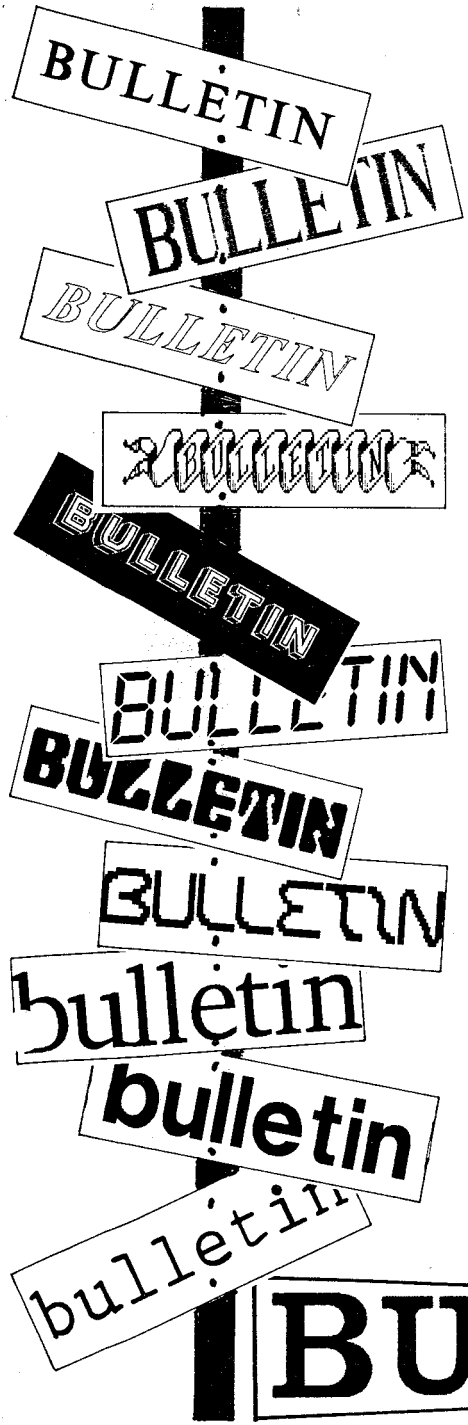


C-16 PLUS/4

3e jaargang nr. 2 – maart 1989

DRUKWERK

indien onbestelbaar retour aan:
Simon Groothuis
Slimpad 34, 1851 LB HEILOO

daaaag

BULLETIN

verscheen
verschijnt 6 maal per jaar,
halverwege de oneven maanden

Redactiesecretariaat:

alle kopij sturen aan
Simon Groothuis, Slimpad 34,
1851 LB Heiloo

Redactie:

Richard van Gelder *(hardware)*
Weiersweid 12, 1831 BW Alkmaar
tel. 072 - 61 57 78

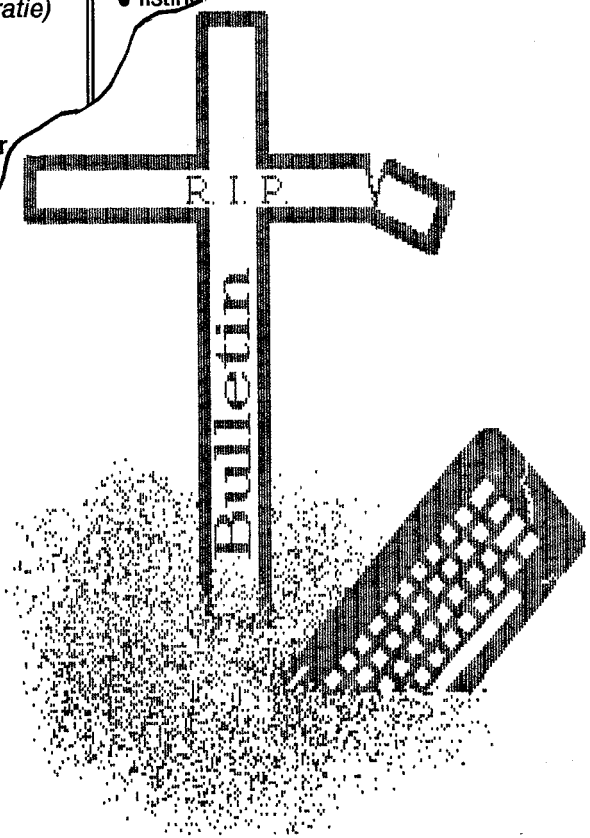
Simon Groothuis *(markt, redactiesecretariaat, administratie)*
Slimpad 34, 1851 LB Heiloo
tel. 072 - 33 62 65

Arthur Muller *(software)*
Bregwaard 6, 1824 EK Alkmaar
tel. 072 - 61 09 71

Ton Muller *(redactionele adv bureauredactie)*
Bregwaard 6, 1824 EK Alkm
tel. 072 - 61 09 71

Spelregels:

- de redactie gaat ervan uit, dat alle kopij oorspronkelijk is en zonder oogmerken die in strijd zijn met de wet, zoals het te koop aanbieden van software
- inzending van kopij betekent niet automatisch plaatsing, dit ter beoordeling aan de redactie die van een en ander de inzender op de hoogte zal stellen
- listing



MINIATUURTJE

```
10 GRAPHIC1,1:COLOR4,1
20 FORI=0TO319STEP2
30 DRAW1,I,0TO319-I,199
40 NEXT
50 FORI=198TO1STEP-2
60 DRAW1,0,ITO319,199-I
70 NEXT
```

van de redactie

Het allerlaatste redactionele voorwoord. Da's een gek gevoel!
Twee jaar lang was het Bulletin een vast punt in ons bestaan. Het inzamelen van de kopij, het ordenen, de redactievergaderingen, het indelen, vormgeven en drukklaar maken van het Bulletin, het achter de mensen die niet betaalden aanrennen, de verdere ledenadministratie bijhouden, postzegels kopen, stickers uitdraaien en plakken, de boekjes naar het postkantoor brengen, met tweemaandelijkse regelmaat kwamen de werkzaamheden terug. Toch wel een hoop werk, maar hartstikke leuk.
Soms waren we wanhopig omdat we veel te veel kopij hadden en we niet konden kiezen, later waren we wel eens wanhopig omdat we zo weinig kopij hadden... Soms waren we wanhopig omdat de aangeleverde kopij zo uit het grijze circuit leek te komen en nauwelijks te verwerken was. Maar dat was allemaal soms, meestentijds was het erg leuk en rolden er naar vorm en inhoud best aardige Bulletins van de pers.

Kwaliteit die zonder al te veel inspanning op je deurmat valt, maakt eenlijk een beetje gemakzuchtig. Een aantal pogingen om databanken op te zetten mislukte. Die voor de handleidingen lukte nog het best, al laat Piet Vliegers ons weten weinig of niets meer binnen te krijgen. Zoals hij ook schrijft een beetje teleurgesteld te zijn, dat op zijn verlanglijstje zo weinig reactie kwam. Van zijn idee om met de vaklieden op hard- en softwaregebied nieuwe dingen op te zetten, is dus niks terechtgekomen.

Daar zit 'm nou de kneep, beste Berry Celie: wat er is heeft allemaal in het Bulletin gestaan en voor wat er nog niet is lijken we niet zo erg warm te lopen. Geen kwestie van de centen, die problemen hebben het stoppen hooguit wat dichterbij gebracht. Maar probeer het. Graag! Hopelijk vegen nieuwe bezems ook in dit geval schoon.
Weet je wat: je hoeft niet eens een nieuwe club op te richten, je mag de oude voortzetten. Zelfs mag je naam en vorm van het Bulletin daarbij gebruiken. In de rubriek Vraag & aanbod staat Berry klaar om het stokje over te nemen. Succes!

Veel kopij begon ditmaal met "jammer". Dat streelt ons. Maar doet onze mening niet veranderen. De inhoud van dit nummer bevestigt die alleen maar.
Twee reacties plaatsen we, één in getekende, één in geschreven vorm. Beide erg origineel.

Een paar huishoudelijke mededelingen tot slot.

1. Wanneer we meer weten over een gebruikersdag krijg je een nieuwsbrief toegestuurd.
2. Oude Bulletins zijn voor *f* 3.– per stuk na te bestellen bij Simon Groothuis.

Onder het spelen van het gedigitaliseerde Wilhelmus nemen wij met weemoed in het hart afscheid van jullie allemaal.

De groetjes van
Arthur, Richard, Simon en Ton.

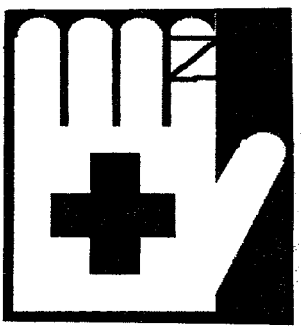
VIRUSSEN 0.2

BESTE REDAKTIE,

In het Bulletin stond dat onze computers geen virussen kennen. Helaas klopt dat niet. Er bestaat wel degelijk een virus. Een erg lastig virus zelfs, waar C-16 & Plus/4-gebruikers regelmatig het slachtoffer van worden. Het is door Commodore zelf ontwikkeld en geperfektioneerd. Eerst is het sluimerend aanwezig en merk je er niets van. Als een donderslag bij heldere hemel slaat het dan plotseling toe! (meestal geaktiveerd door een hoogtepunt).

Inderdaad: het beruchte "We-stoppen-er-mee-Virus" dat we van Commodore allemaal kennen. Kort na introduktie van de C-16 & Plus/4-lijn sloeg dit al toe. Voordat deze uitstekende machines voet aan de grond konden krijgen stopten ze de produktie.
De gevolgen daarvan kennen we maar al te goed. Software is haast niet te koop en in de 'algemene' Commodore-bladen worden we geboycot of (in het gunstigste geval) doodgedrukt tussen het C-64 en Amiga-geleuter.

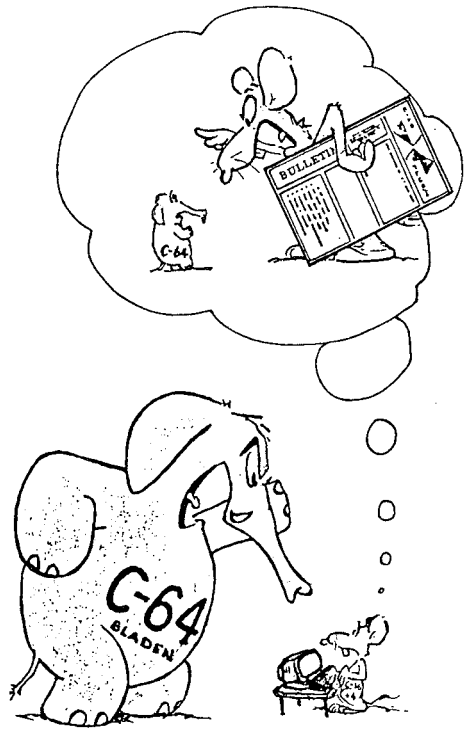
Gelukkig is er als alternatief het C-16 Plus/4 Bulletin. Ons eigen komputertijdschrift in onze eigen taal. Nog wel niet zo groot, maar toch al stevig in de groei.
Helaas, plotseling slaat het beruchte Commodore-virus weer toe. Dit keer nog wel bij onze eigen Bulletin-redaktie!
"We-stoppen-er-mee", zo meldt ons de januari-editie. "Op een hoogtepunt. Rond 1 APRIL verschijnt het laatste nummer."
Toegegeven, het is wel even schrikken als dat virus je te pakken krijgt. Maar, beste redaktie: 'Houdt moed!' Er is een prima middel tegen. Gewoon even RESETten en je bent er zo weer vanaf!
Van harte beterschap, dus maar. En verder: ook na 1 april gewoon doorgaan met het Bulletin. Zijn er problemen, dan kunnen we daar met z'n allen best wel een oplossing voor bedenken.



Met vriendelijke groeten,
Ronald & Gerard de Bruin.
Oud-Beyerland.

Ronald en Gerard de Bruin, Oud-Beijerland

STOPPEN DAT IS



TERUG NAAR AF!

vraag & aanbod

GEVRAAGD:

Wie heeft voor mij een lichtpen of wil er (wegens tijdgebrek) een voor mij in elkaar zetten? Voorts ben ik nog steeds vergeefs op zoek naar een Logo-module.
Siem van Westen, Zaandijk, tel. 075-216082.

GEVRAAGD:

Wie helpt mij mijn Protek-modem geschikt te maken voor Viditel en Girotel (1200/75 baud)? Wie schrijft voor mij een compleet programma om een stamboom bij te houden?
Piet Vliegers (die voor beide zaken *f* 25.– inzet), Hoogezand, tel. 05980-90026.

GEVRAAGD:

Wie heeft er voor mij de gebruiksaanwijzing van Speed-packer versie 4.0 of een korte instructie?
Wie wil Trivial Pursuit vertalen van het Duits naar het Nederlands? Het Duitse programma is bij mij aanwezig.
Wim Rijpstra, Drachten, tel. 05120-17126.

GEVRAAGD:

Schema's voor een eprombank en andere hardware-uitbreidingen, eproms, handleiding van tvw 1.1, Fasttext-cartridge.
Adres om Compute Mit te bestellen en graag informatie over kosten.
Martin Reumers, Noorbeek, tel. 04457-1710.

GEVRAAGD:

Wie helpt mij muziekprogramma's voor mijn C-16 op te sporen?
Ronald Beijma, Lemmer, tel. 05146-2794 (alleen in weekeinde).

GEVRAAGD:

Geheugenuitbreiding voor C-16, basicode-vertaalprogramma, educatieve software voor kinderen van 8 jaar en ouder.
Jos Roolvink, Veldkers 15, 7577 DB Oldenzaal.

GEVRAAGD:

Mede-oprichters voor een nieuwe C-16 Plus/4-club. Ik denk dat het mogelijk is om voor *f* 3,50 per stuk een nieuw Bulletin in elkaar te zetten.
Berry Celie, Amsterdam, tel. 02907-5483.

GEVRAAGD:

Een goed werkende gestabiliseerde voeding voor de C-16.
Marc Jans (die meedeelt zijn voeding te hebben opgeblazen door, zoals in Bulletin nr.1 beschreven, de brugcel te vervangen door 4 dioden van het type IN5406), Assen, tel. 05920-46050.

GEVRAAGD:

Hoe print ik met 3+1 van de Plus/4 op de C-16 (waarop het perfect werkt) op onze Citizen 120 CD-printer?
Wie heeft voor deze printer een Nederlandse of Duitse handleiding?
Wie helpt ons aan: leerprogramma's (denk aan grammatica voor Nederlands of buitenlandse talen) en rekenprogramma's voor de basisschool?
En wie heeft er zoiets als een agendaprogramma voor ons?
Wie heeft de disk van Sonderheft 3 van 64er of het programma Supergrafik uit dit tijdschrift?
Geert en Carla Hakze, Brunssum, tel. 045-253225.

Praktijk is: het werkt, maar niemand weet waarom.
Theorie is: iedereen weet hoe het moet, maar het werkt niet.
[Zde stelling van Murphy]

PUBLIC DOMAIN SOFTWARE

PUBLIC DOMAIN software, ook wel FREEWARE genoemd, omvat programma's die door de makers vrij ter beschikking van de overige gebruikers gesteld worden. Er rust geen copy-right op. De programma's kunnen dan ook vrij gekopieerd en aan andere gebruikers doorgegeven worden. Er zijn gebruikers die zich op het verzamelen van Public Domain toeleggen en de verspreiding ervan bevorderen. Dat gebeurt dan tegen vergoeding voor gemaakte kosten, zoals aanschaf van diskettes en kassettes, porti, verpakking, publiciteit etc,
Voor zover ik weet is er in Nederland op dat gebied voor de C16-Plus/4 niet zo veel gaande. Ik ken eigenlijk alleen Courbois Software in Beuningen met tamelijk veel Public Domain op kassette.
In het Duitse 'Compute Mit' staan echter regelmatig flink wat kleine advertenties waarin Public Domain op diskette wordt aangeboden.
Nieuwsgierig geworden, heb ik een aantal adressen aangeschreven en daar de programmalijsten opgevraagd.
Er bleek een groot aantal diskettes met allerlei verschillende zaken leverbaar. Daar zaten erg veel utilities tussen en ook allerlei sound, hi-res en graphic programma's. Ook waren er vrij veel adventures en spellen bij.
De meeste spellen waren vaak vrij eenvoudig (basic), maar dat laatste kan niet gezegd worden van de Hongaarse Public Domain (o.a. Hungaroring, Godzilla en het 3-dimensionale schaakspel Colossus; allemaal 64Kb).
Ook zaten er prima Hongaarse sound- en muziek-demo's bij.
Al met al zou ik iedereen willen aanraden om eens een kijkje in de Public Domain-hoek te gaan nemen (vooral de Hongaarse).
Voor de prijs hoeft niemand het te laten; voor de dubbelzijdige diskettes die ik gekocht heb betaalde ik maar 5 DM per stuk (verzending inbegrepen).
Omdat niet iedereen de Compute Mit Sonderhefte bij de hand heeft wil ik hierbij een paar Public Domain-adressen doorgeven:

F. EITNER
Dr. Beberstrasse 17
D-2222 MARNE
Deutschland

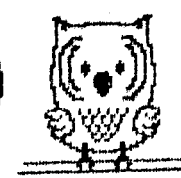
RAINER JACOBS
Zur Butzmuehle 1
D-3558 FRANKENBERG
Deutschland

VIKTOR JUERGENS
Prinzenstrasse 131
D-2330 ECKERNFOERDE
Deutschland

RONALD DE BRUIN

naschrift redactie
Er bestaat wel degelijk een verschil tussen public domain en freeware. Die laatste vorm van software wordt vaker shareware genoemd. Dat wil zeggen: de gebruiker beproeft het programma en als hij het echt gaat gebruiken, betaalt hij een bedragje aan de maker(s). De bedoeling is dat het programma circuleert, steeds weer gevolgd door nbijdrage. Met dat geld worden updates gemaakt en ook nieuwe programma's. Het is een uit Amerika afkomstig systeem.
En probleem: het betaalgedrag in Europa is niet al te best. Daarover ging dan ook het rechtsgeding van een paar softwarefirma's tegen de Hobby Computer Club in Nederland, die erg veel shareware verspreidt.




PRINTERBESTURING met 3+1 van de PLUS 4

Hieronder is een voorbeeld van het besturen van de printer HR-10C
LET VOORAL OP het gebruik van - tw - en - tf -
De - tw - stelt je instaat printeropdrachten mee te sturen bij het
uitprinten van de database (file manager)

```
ctrl9 asc27;69;:ctrl0
  ,,  asc27;87;:ctrl0  De BEATLES   LP'S  van Sander Sieswerda  1989
  ,,  asc27;82;: ,,                                  → spatie
  ,,  asc27;69;  ,,                          (2 x R)
  ,,  tf;:rc;    ,,
  ,,  tw;asc27;69;:asc27;87;:tf;:rc;:ttl1;: fld1;:tw;asc27;82;:fld2;:fld3;ctrl2
    ctrl9  tf;:rc; ctrl0
      ,,   tw;asc27;69;:tf;rc;:ctrl0     ctrl9 fld4; ctrl0
      ,,   tw;asc27;82;:tf;rc;: ,,         ,,  fld5;  ,,
           ctrl9 fld6; ctrl0
             ,,   fld7;   ,,
             ,,   fld8;   ,,
             ,,   fld9;   ,,
             ,,   fld10;  ,,
             ,,   fld11;  ,,
             ,,   fld12;  ,,
             ,,   fld13;  ,,
             ,,   fld14;  ,,
             ,,   fld15;  ,,
             ,,   fld16;  ,,
             ,,   fld17;  ,,
             ,,   eof?    ,,
```

Het resultaat van de besturing ;
****

THE BEATLES LP'S van Sander Sieswerda 1989

	fld2	fld3
LP: Abbey Road	26 sept	'69

ttl1 | fld1

side one	side two
Come together	Here comes the sun
Something	Because
Maxwell's silver hammer	You never give me your money
Oh! Darling	Sun King
Octopus's Garden	Mean Mr. Mustard
I want you (she's so heavy)	Polythene Pam
	She came in through the bathroomwindo
	Golden slumbers
	Carry that weight
	The end
	Her Mayesty

fld4
fld5
↓
enz.

Je ziet onderstrepen/vetafdrukken óók van de FILEMANAGER van de PLUS 4
****

P.J.Sieswerda.

V O O R B E E L D
printen op een bepaalde plaats(tab) van de FILEMANAGER / Plus 4

LP: Beatles for sale

uitgebracht op: 4 dec '64

nummers:

Side one	Side two
No reply	Eight days a week

R = Control 9 (reverse aan) Q = Control 0 (reverse uit) ← : Return

```
R center;:asc27;87;: Q V O O R B E E L D ←
R asc27;69;:center;: Q printen op een bepaalde plaats(tab) van de
FILEMANAGERR tf;:rc3; Q ←
R tw;:asc27;70;:tf;:rc;:ttl1;:tw;:asc27;9;5;:tf;:rc;:fld1; Q ←
R tw;:asc27;82;:tf;:rc;: Q ←
R tf;:rc;:ttl3;:tw;asc27;9;20;:tf;:rc;:fld3; Q ←
R ←
R tf;:rc;:ttl4;:tw;asc27;9;20;:tf;:rc;:fld4; Q ←
R tw;asc27;9;20;:tf;:rc;:fld5; Q ←
```

De opdracht -- tw;:asc27;9;20 -- schakelt de printersturing van de absolute tab-plaats aan.

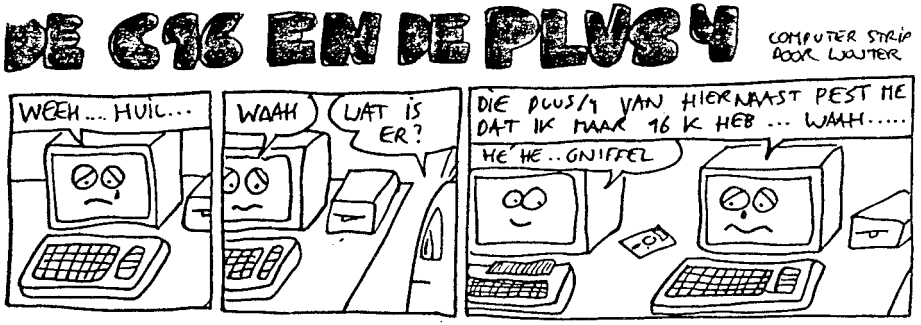
LET OP - Dit voorbeeld is van toepassing bij gebruik van een Brother HR-10C jouw/uw printer kan een andere ESC-sturing nodig hebben, KIJK in de handleiding van de printer.

Printersturing Brother HR-10C -----

P R I N T E R STURING (esc functies omgezet in asc funties)

CONTROL 9	asc27;69;:	CONTROL 0	- onderstrepen aan
,,	asc27;82;:	,,	- onderstrepen uit
,,	asc27;87;:	,,	- de regel die nu volgt wordt donkerder afgedrukt let op ; er mag geen RETURN tussen zitten.
,,	asc27;88;	,,	- schakelt donkerafdruk uit - RETURN schakelt dit ook uit
,,	asc27;30;6;	,,	- regelAFSTAND -instellen- - defal 6 kan verschillen van 0-20 - test met afstand - 7 -
,,	asc27;30;9;	,,	- afstandsgetal - 9 - is normaal
,,	asc27;70;:	,,	- dubbele aanslag aan
,,	asc27;38;:	,,	- dubbele aanslag weer uit
,,	asc27;35;:	,,	- automatisch regelopschuivenAAN
,,	asc27;34;:	,,	- ,, ,, ,, ,, UIT

Dit is een test met tussenin VETTE afdruk uitprinten
Dit is een test met R asc27;87;: Q tussenin VETTE afdruk R asc27;88;: Q uitprinten.
--- T I P ESC e wordt R asc27;69;: Q (ESC=asc27 en e=CHR$(69))



wist u dat?

- Your Commodore een speciale editie heeft uitgebracht bij de november-uitgave 1988: The Essential Guide. Daarin zaken als Money +, Text 80, gamefinder en adressen van software-huizen. Aldus meldt Wiebe ten Hoeve uit Bears.
- Speed/Plus een gigantische miskoop is geweest, naar het oordeel van Piet Vliegers. Dit programma doet het meer niet dan wel, roept hij vertwijfeld uit. Vraag van Piet: misschien kan onze vriend Frans Podbregar eens bij de maker van het programma informeren of er up-dates verschijnen?
- Dit een paar nuttige adressen zijn:
  Commodore Business Machines UK Ltd., Commodore House, The Switchback, Gardner Road, Maidenhead, Berks SL6 7XA, UK, tel. 0628-770088,
  voor onderdelen: HRS Electronics Ltd., Garrets Green Lane, Birmingham B33 OUE, UK, tel. 021-789-7575,
  of: DB Electronics, Great Baddon, Chelmstad, UK, tel. 0245-260874.
  Aldus meldt Harry van Grinsven, die op deze adressen kwam n.a.v. problemen met zijn 1551-drive. Doe er uw voordeel mee.

spellen

*** David's midnight magic +4 ***

Hoewel ik zelf een 64-guy ben en zodoende weinig met de C-16 doet heb ik toch nog een bijdrage voor ons LAATSTE bulletin!

Het spel 'David's midnight magic' is een bekend oud flipperspel op de C-64.
Op de C-16 (64k) / +4 is hij precies hetzelfde als op de C-64.
Het spel start met een getekende flipperkast met leuke effecten erbij.
De graphics zien er heel goed uit.
Het geluid is wat minder; je hoort alleen tijdens het spel wat geluidseffecten,die wel aardig klinken.
Er zit geen muziek in.
Je start het spel met 'shift` of 'C=` en je beweegt met 'shift - rechts`, 'C= - links`.
Het doel is met een balletje zoveel mogelijk punten te maken d.m.v. allerlei dingen te raken.
Je kan bonuspunten krijgen d.m.v. het balletje in bepaalde gaten te krijgen.
Deze bonus-punten kunnen ook verdubbeld. (tot 5X toe!)
Je kan ook het balletje in een bepaalde buis ketsen, zodat je 10 bonus-punten krijgt (X1000=aantal bonus-punten).
Het balletje blijft in deze buis zitten en je krijgt een nieuw balletje.
Gebeurt dit 3 keer,dan komen alle drie balletjes uit de buis vrij en je moet dan met drie balletjes te gelijk spelen, wat een geinig effect geeft!
Hoewel het balletje een beetje vierkant is, zit het spelletje goed inelkaar!!!

GRAPHICS : 9 (geinige graphics!)
SOUND : 6 (kon beter!)
SPEELBAARHEID : 8 (goed speelbaar!)
TOTALE CONCLUSIE : 8 (goed spel!!)

ANDREAS STREMLER..... (SPIDER INC.)

Saboteur

In dit spel ben je een saboteur in een pakhuis. Je moet een tijdbom en een disk vinden, en daarna ontsnappen voor de bom ontploft. Er zitten echter bewakers en honden op je hielen, die je kunt uitschakelen d.m.v. een aantal bewegingen. Ook kun je gooien met diverse voorwerpen die rondslingeren. Je moet ontsnappen per helicopter, die niet zo gemakkelijk te vinden is als je zou denken
Er zijn twee versies van dit spel: de plus/4 en de c-16 versie. Ik zal nu beide versies bespreken.

- de plus/4 versie:

GRAPHICS:
Zeer goede graphics en een werkelijk schitterende, vloeiende animatie van alle bewegingen.

GELUID:
Het geluid is goed. Terwijl je loopt hoor je een geluid, wat doet denken aan grint. Voor de rest is er in het spel eigenlijk weinig geluid, behalve de intro-tune aan het begin. Maar al met al is het geluid OK.
Saboteur +4 is een zeer goed spel dat zeker de moeite waard is te bemachtigen.

GRAPHICS:10
SOUND:7,5
PLAYABILITY:9
VALUE:10 (fl.10)
OVERALL:9,5

- de c-16 versie:

Deze versie is opmerkelijk slechter dan de plus/4 versie; je kunt hem alleen met het keyboard spelen, en je ziet alles op een zeer klein schermpje,dat hooguit 10 bij 10 groot is.

GRAPHICS:
In de c-16 versie wordt je in plaats van honden geconfronteerd met bananen op pootjes(dat is het enige waar ze op lijken). De bewakers (en jijzelf ook) zijn nogal vierkantig gebouwd.(ils ont des epaulles carrees). De bewegingen zijn slecht geanimeerd (meer het trekpop-idee dan NINJA-bewegingen). Tot zover de graphics.

GELUID:
Terwijl je loopt heb je ongeveer hetzelfde geluid als de plus/4 versie. Er zit alleen geen tune aan het begin.
Zelfs voor c-16 bezitters is deze versie af te raden. Je zult waarschijnlijk spijt hebben voor de moeite die je gedaan hebt om er aan te komen.

GRAPHICS:4
SOUND:6
PLAYABILITY:3 (alleen keyboard)
VALUE:4,5 (fl.10)
OVERALL:3,5

Conclusie: Plus/4 en c-16 + 64K bezitters kunnen voor weinig geld een schitterende aankoop doen. c-16 bezitters echter doen er beter aan dit spel te mijden.

Ronald de Bruin.

inhoud bulletins

a

aanvulling op austrospeed 2-3-21
aanvulling track 18 3-1-12
ace ii 2-5-22
ack-ack attack 2-4-22
adres-16 test 1-3-7
adventure ontleed 1 2-3-15
adventure ontleed 2 2-4-21
adventure ontleed 3 2-5-18
alphabeth zelf gemaakt 2-5-20
anti reset 1 1-4-11
anti reset 2 2-4-9
apfelmannchen 2-4-23
aproxi +4 2-4-23
arthur noid 3-1-20
assembler 1-4-8
auf wiedersehen monty 1-3-11
austrospeed aanvulling 2-3-21
austrospeed compiler 2-2-15
autostartprg diskdrive 2-1-10

b

basic uitbreiding bespreking 2-3-23
basic uitbreiding correcties 2-1-15
basic uitbreiding zelf maken 1-4-13
basic versneilen 2-4-5
basicode aanpassen prg tbv 1-1-26
basicode al verkrijgbaar !! 1-4-9
basicode oproep tot 1-1-9
basicode prg op cassette 2-4-7
basicode prg verkrijgbaar!! 1-3-3
basicode printroutine 2-4-7
basicode uitleg van de protocol 2-2-12
boekbespr 100 prg voor de c=16 2-4-19
boekbespr rom listing voor c=16 plus/4 2-4-19
boekbespr using the commodore c=16 1-1-9
boekbespr wegwijs in machine taal 3-1-5
boten v-e programma 1-4-8
bridgehead 2-1-30
briefhoofd 2-5-21
budgetplanner test 1-3-7

c

c=16 c=116 geheugenuitbreiding 1-1-4
c=16 en 3-plus-1 2-1-13
chips te koop 3-1-6
circle fantasy 2-1-15
citizen dp 560-cd printer 1-1-3
club bijeenkomst!! 2-2-4
commodore geschiedenis 3-1-9
compiler austrospeed 2-2-15
computer virussen 3-1-3
copy files bespreking 2-1-8
create page 2-6-18
create page handleiding 3-1-13

d

databank handleidingen 1 1-4-6

databank handleidingen 2 2-2-18
databank handleidingen 3 2-4-8
database allerlei gegevens 2-6-12
database allerlei gegevens? 2-4-15
database test 2-3-20
database voor peeks & pokes ? 2-2-23
datasette en 3-plus-1 1-4-12
datasette kopstand 1 1-2-21
datasette kopstand 2 1-3-18
datasette met basicode prg 2-4-7
datasette reparatie 1-3-15
datasette repareren 2-3-8
demolition 2-2-16
demolition vs. rebound 2-2-17
directory extensions 2-5-12
directory gebruik 2-6-14
diskdrive 1551 2-6-14
diskdrive 1581 op plus/4 c=16 test 2-5-6
diskdrive oc-118 test 2-1-22
diskette hoes printen 2-2-20
diskettes enkel - dubbel ? 1-2-10
drumkit van wizzard 2-5-13
dubbelhoge letters 3-1-12

e

ebs-1.0 operating system 2-2-14
eprom superbase 2-6-14
epromboard in cartridge 2-5-5
eprommer voor plus/4 en c=16 2-3-5
extra basic 1-2-28
extra hulptoetsen 2-6-5

f

fasttape-module 1-1-13
fasttext gebruikers verslag 2-6-6
fire ant 2-1-29
funktionenplotter 2-4-23

g

gebruikerdag + workshop 2-4-3
geheugen indeling c16 plus/4 2-6-17
geheugen uitbreiding voor plus/4 ? 2-2-10
geheugenuitbreiding 3-1-7
geheugenuitbreiding c16 1-1-4
geschiedenis van commodore 3-1-9
godzilla 3-1-19
grandmaster chess handleiding 1-3-27
graphic saver-loader 2-3-19
grote letters 3-1-21

h

handleiding chess 1-3-27
handleiding create page 3-1-13
handleiding databank 1 1-4-6
handleiding databank 2 2-2-18
handleiding databank 3 2-4-8

hardware uitbreidingen voor c=16 en plus/4 2-2-14
hires toolkit 1-3-4
hires toolkit 1-4-25
hires toolkit en u.d.g. 2-1-7
hires toolkit met lichtpen 2-4-10
home office test 2-3-20
hulptoetsen extra 2-6-5
hungaroring forma-1 car race 2-5-23
hustler 2-1-1

i

i-o pinouts 1-3-12
ikari warriors 2-4-22
ingebouwde software en c=16 2-1-13
ingebouwde software en datasette 1-4-12
ingebouwde tekstverw. op datasette 2-5-17
interfacing rs-232 1-4-19
interrupt mc taal 2-1-18
interrupt mc taal programma 2-4-17
italic op telrol 2-3-18

j

jailbreak 2-2-16
jet set willy 2-1-29
joystick besturing in mc 1-2-9
joystick tips 1-3-8
joystick verbouwen 2-3-4
joystickaansluiting 1-1-6

machinetaal op de c16/plus4

een inleiding in de wereld van monitor en assembler....

Iedereen heeft wel eens van machinetaal gehoord, maar nog niet iedereen kan er in programmeren. Daarom wou ik aan de hand van een mini-cursusje de mensen die nog niets van machinetaal afweten helpen om zelf eens wat te programmeren, want machinetaal is echt niet zo moeilijk. Dan nu het cursusje:

1. hoe werkt mijn computer ?

Zoals de meeste wel weten snapt de c16/plus4 en elke andere computer geen bal van basic, het snapt slechts machinetaal. Basic wordt door een machinetaalprogramma, de basicinterpreter, bij elke opdracht opnieuw naar machinetaal vertaald, wat langzaam, omslachtig en geheugenvretend is en weinig mogelijkheden geeft. (er zijn dan al gauw veel peeks en pokes bij nodig). Machinetaal werkt hoofdzakelijk met "peek","poke" en daarmee samenhangende opdrachten. Machinetaal bestaat eigenlijk uit allemaal getallen (op-codes), waar zo lastig mee te werken valt, dat men daarvoor een tussenstap heeft bedacht, assembler, welke bestaat uit woorden, die voor ons herkenbaar zijn zoals LDA en AND. Degene die de machinetaal uitvoert is de cpu 7501, ofwel de central process unit, zeg maar het brein van de c16/plus4. Deze heeft een aantal kenmerken, zoals dat hij werkt met getallen van 8 bits tegelijk (een byte),en dat hij 64k tegelijk kan adresseren. De cpu werkt met een stuk of 6 registers, welke je zou kunnen definieren als een soort "hoofdvariabelen"

2. De 7501 registers.

Ik zal ze alle 6 hier even kort beschrijven.

Het a register. Doorgaans de accumulator genoemd of kortweg de accu. Dit is de belangrijkste 'variabele', er wordt voor korte tijd voor b.v. het lezen en schrijven van geheugenplaatsen een byte bewaard, of een parameter voor een subroutine.

Het x en het y register. Deze zijn vrijwel gelijk aan de accumulator, maar je kunt er geen bewerkingen zoals or en and mee doen, wat met de accu wel kan. Vaak hebben ze andere adresseringsmethoden (kom ik later op terug), en worden ze als lusteller gebruikt.

De program-counter. Dit is het enige 16-bits register, welke de geheugenplaats aangeeft van de instruktie die op dat moment door de processor wordt uitgevoerd.

De stack pointer. Deze geeft aan waar de eerste lege plaats op de stack te vinden is. De stack is een soort 'stapel', waar door Kernal/Basic/de gebruiker tijdelijk dingen op geschoven kunnen worden (met oa de instrukties PHA en PLA). Wat het laatst op de stack geschoven wordt, komt er ook het eerst weer af, dus vandaar dat hier het sprong-adres opgeschoven wordt bij het aanroepen van een subroutine.

Het status register. Allemaal een-bits vlaggen (wel/niet gebeurd), die de status van de processor aangeven. O.a.: Carryvlag (als een bewerking groter dan 255 is uitgevallen), Zerovlag (als een vergelijking klopt), interuptdisableflag etc.

Al deze registers zijn vanuit machinetaalinstrukties beinvloedbaar.

3. Het hexadecimale en binaire talstelsel.

Dit zijn 2 nieuwe talstelsels waar je mee te maken krijgt. De computer werkt zoals je weet met nulletjes en eentjes, aan en uit, zwart en wit, wat je het ook maar noemen wilt. De gehele notatie gaat echter in hexadecimaal, kortweg hex. Je hoeft niet bang te zijn dat je eindeloos moet gaan omrekenen ofzo, als je een tijdje met hex gewerkt hebt weet je de belangrijkste geheugenplaatsen in hex, je rekent en programmeert zelf in hex, je denkt zelfs in hex. Voor het gemak nog even een herhaling:

Binair, een tweetallig stelsel, 0 en 1.

Decimaal, een tientallig stelsel, 0,1,2,3,4,5,6,7,8 en 9

Hexadecimaal, een zestien talligstelsel, 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E en F

De werking van alle 3 is het zelfde. Nog even: voor het omrekenen van hex naar binair (oa makkelijk bij registers), hoeft u slechts de eerste 16 binaire getallen te kennen. Een hexgetal als $ 9AC2 (het $ teken staat er ter aanduiding dat het hex is) splits je in vieren op, en je maakt er de volgende binaire getallen van:

9- 1001, A- 1010, C- 1100, 2- 0010, dus 9AC2 = 1001101011000010.

Voor het omrekenen bestaan er in basic de volgende funkties:

DEC ("hex-getal") levert een decimaal getal op, v.b: PRINTDEC ("FF") geeft 255.

HEX$ (decimaal-getal) levert een hex getal op, v.b.: PRINTHEX$ (100) geeft 64.

4. LDA en STA.

We beginnen met de eerste machinetaal-instruktie: LDA (load accumulator with memory). Met deze instruktie laad je de accu met een byte-waarde uit welke geheugenplaats dan ook. Achter deze instruktie typ je de desbetreffende geheugenplaats in hex, met een $ ervoor. V.b. LDA $2000. Hiermee wordt de accu gevuld met de inhoud uit adres $2000 (decimaal 8192). Je zou dit kunnen vergelijken met basic A= PEEK (8192). Dit zelfde kun je ook met de X en de Y registers doen, het heet dan LDX (load x with memory) en LDY. Voorbeeld: LDX $FF16 of LDY $03F7 (laad respectievelijk de x en de y met de geheugenplaatsen $FF16 en $03F7.) De tweede instruktie, STA, stopt die waarde weer in een geheugenplaats. V.b.: STA $065E (vult de geheugenplaats met de inhoud van de accu). Dit is weer te vergelijken met basic POKE 1630, A . Vanzelfsprekend kan dit ook met het X en het Y register. V.b.: STX $03F7 en STY $0494 . Je hebt nog een instruktie nodig om je eigen machinetaalprogramma te kunnen schrijven: BRK . Deze instruktie veroorzaakt een break in het programma, en je keert weer terug naar de monitor. Het wordt misschien allemaal wat duidelijker aan de hand van een voorbeeldje in de praktijk:

5. De monitor.

Het is handig goed met de monitor om te kunnen gaan om nut van machinetaal op je c16/plus4 te hebben. Zet de computer eens aan, want we gaan het vorige eens in praktijk brengen. Als de computer aanstaat, start je in basic. Je komt als volgt in de monitor: 1. Druk op RUNSTOP+RESET

2. Typ MONITOR (+return) of M shift+C.
3. Typ SYS 4 (of elke andere geheugenplaats waar 0 in staat, want 0 is de opcode voor BRK, waardoor je in de monitor komt.)

Al die getallen die nu voor je neus staan zijn de registers die we al gehad hebben Je kunt ze elke keer opnieuw opvragen dmv. het monitorcommando R (van registers). Een machinetaalprogramma intikken doe je met het commando A (van assemble). b.v.: A 03F7 LDA $FF15 . Het getal achter A is het adres waar het programma komt te staan. De monitor zet na RETURN zelf de opcodes ertussen (zoals het werkelijk in het geheugen komt te staan dus.) en print het volgende adres alvast voor. Tik nu in: LDX $FF19 return. STA $FF19 return. STX $FF15 return. BRK return. return. U ziet , alle instrukties herken je. Wat je moet weten is dat $FF15 en$FF19 respectievelijk de randkleur en de achtergrondkleur bevatten. De werking zal nu wel duidelijk zijn : De kleuren van rand en achtergrond worden bewaard in A en X en vervolgens verkeerdom teruggezet. Je kunt het programma runnen met het monitorcommando G (van go), gevolgd door het beginadres, dus in onsgeval G 3F7. Je ziet het effect. Niet opzienbarend, dit had u in basic ook wel kunnen bereiken (maar ja, we moeten toch ergens beginnen). We voegen nu een instruktie toe waardoor je een effect bereikt wat in basic niet mogelijk is, n.l. door de snelheid. Deze nieuwe instruktie heet JMP (van jump), en is vergelijkbaar met GOTO. we springen alleen niet naar een regelnummer, maar naar een nieuw adres. Achter JMP staat het nieuwe adres, b.v.: JMP $8000. In ons voorbeeld zouden we door een JMP naar $03F7 de computer kunnen laten blijven verwisselen. En heel snel. Typ in : A 0403 JMP $03F7 return return. Start hem weer met G 3F7. Dit geeft aan hoe snel machinetaal is, sneller als jouw oog en de kathodestraal van je TV/monitor. Je kunt de listing nog eens rustig bekijken als je het programma stopt met RUNSTOP+RESET. Geef het commando D (van disassemble, vgl. list) , dus D 3F7.

Tot slot nog een kleuriger versie van dit geheel. We gebruiken $FF1E om een RND kleur uit te krijgen, $FF1E geeft namelijk de momentele rasterregel aan. (en die verandert zo snel, dat we hem goed voor RND bytes kunnen gebruiken). Tik in :

```
A 03F7 LDA $FF1E    (haalt een willekeurig een kleurcode van 0 - 255)
A 03FA STA $FF15    (stopt deze in de rand)
A 03FD STA $FF19    (en in de achtergrond)
A 0400 JMP $03F7    (nog eens)
```

6. Het geheugen van de C16/plus4.

Om goed met machinetaal te kunnen werken, dien je de geheugenindeling een beetje te snappen. Uitgaande van een standaard c16 is die zo : Het geheugen loopt van $0000 tot $FFFF, de zeropage loopt officieel van $0000 tot $00FF, maar men noemt het gebied van $0000 tot $0800 zo. Dit is 2K waar o.a. het kernal en de basicinterpreter gegevens opslaan, een soort notitievelletje dus. vele plaatsen hiervan worden niet of amper gebruikt en zijn dus een nuttige plaats voor machinetaalprogramma's (o.a. van $03F7 tot $0436). De meeste mensen hebben daar wel eeen lijst van maar voor de genen die dat niet hebben citeer ik hier even een goede duitse zeropagemap.

Dezimal	Hexadezimal	Bemerkung
0	$0000	Datenrichtungsregister des 7501
1	$0001	Ein-/Ausgabe-Port des 7501
2	$0002	Flag für Schleifen
3- 4	$0003-0004	Neue Startadresse (Renumber)
5- 6	$0005-0006	Schrittweite (Renumber)
7	$0007	Gesuchtes Zeichen
8	$0008	Flag für Anführungszeichen-Modus
9	$0009	TAB-Spaltenzähler
10	$000A	Flag: 0 = Load, 1 = Verify
11	$000B	Zeiger für Eingabepuffer, Anzahl der Elemente
12	$000C	Flag für Standard-DIM
13	$000D	Datentyp: $FF=String, $00=Numerisch
14	$000E	Datentyp: $80=Integer, $00=Fließkomma
15	$000F	Flag für DATA/LIST
16	$0010	Flag: Element/FNx-Flag
17	$0011	Flag: $00=INPUT, $40=GET, $98=READ
18	$0012	Flag: Vorzeichen des ATN
19	$0013	Flag: Input-Prompt
20- 21	$0014-0015	Int. Adresse
22	$0016	Zeiger auf temporären Stringstapel
23- 24	$0017-0018	Letzter temporärer Stringvektor
25- 33	$0019-0021	Stapel für temporäre Strings
34- 35	$0022-0023	Bereich für Hilfszeiger 1
36- 37	$0024-0025	Bereich für Hilfszeiger 2
38	$0026	Bereich für Produkt bei Multiplikation
39	$0027	Bereich für Produkt bei Multiplikation
40	$0028	Bereich für Produkt bei Multiplikation
41	$0029	Bereich für Produkt bei Multiplikation
42	$002A	Bereich für Produkt bei Multiplikation
43- 44	$002B-002C	Zeiger auf Basic-Anfang
45- 46	$002D-002E	Zeiger auf Variablen-Anfang
47- 48	$002F-0030	Zeiger auf Beginn der Arrays
49- 50	$0031-0032	Zeiger auf Ende der Arrays (+1)
51- 52	$0033-0034	Zeiger auf Stringspeicher
53- 54	$0035-0036	Hilfszeiger für Strings
55- 56	$0037-0038	Zeiger auf Speichergrenze
57- 58	$0039-003A	Laufende Basiczeilennummer
59- 60	$003B-003C	Textpointer
61- 62	$003D-003E	Zeiger auf Basic-Statement für CONT
63- 64	$003F-0040	Nummer der aktuellen DATA-Zeile
65- 66	$0041-0042	Adresse des aktuellen DATA-Elementes
67- 68	$0043-0044	Sprungvektor für INPUT
69- 70	$0045-0046	Aktueller Variablenname
71- 72	$0047-0048	Adresse der aktuellen Variablen
73- 74	$0049-004A	Variablenzeiger für FOR...NEXT
75- 76	$004B-004C	Zwischenspeicher für Basic-Zeiger
77	$004D	Akkumulator für Vergleichssymbole
78- 79	$004E-004F	Arbeitsbereich (Zeiger etc.)
80- 81	$0050-0051	Arbeitsbereich (Zeiger etc.)
82	$0052	Arbeitsbereich (Zeiger etc.)
83	$0053	Grafik-Modus
84- 86	$0054-0056	Sprungvektor für Funktionen
87- 96	$0057-0060	Bereich für numerische Operationen
97	$0061	Fließkomma-Akkumulator 1 (FAC): Exponent
98-101	$0062-0065	Fließkomma-Akkumulator 1 (FAC): Mantisse
102	$0066	Fließkomma-Akkumulator 1 (FAC): Vorzeichen
103	$0067	Zeiger für Polynom-Auswertung
104	$0068	Fließkomma-Akkumulator 1 Überlauf
105-110	$0069-006E	Fließkomma-Akkumulator 2 Exponent etc.
111	$006F	Vorzeichenvergleich Akku 1 mit Akku 2
112	$0070	Fließkomma-Akkumulator 1 niederwertige Stel
113-114	$0071-0072	Kassettenpuffer Lange/Zeiger
115-116	$0073-0074	Automatisches Zeileninkrement 0=AUS
117	$0075	Grafik-Flag 0=Text 255=Grafik
118-120	$0076-0078	Arbeitsbereich
121-123	$0079-007B	Darstellung von DS$
124-125	$007C-007D	Basic-Pseudo-Stack-Pointer
126-143	$007E-008F	Arbeitsbereich (Sound)
144	$0090	Statuswort ST
145	$0091	Flag: Stoptaste RVS-Taste
146	$0092	Flag: Serieller Bus — Ausgabe Zeichenpuffer
147	$0093	Zeichenspeicher — Serieller Bus
148	$0094	Flag: Serieller Bus — Ausgabe Zeichenpuffer
149	$0095	Zeichenspeicher — Serieller Bus
150	$0096	Zwischenspeicher
151	$0097	Anzahl der geöffneten Files
152	$0098	Eingabegerät (Normalwert: 0)
153	$0099	Ausgabegerät (Normalwert: 3)
154	$009A	Flag: $80=Direkt- $00=Programm-Modus
155-158	$009D-009E	Zeiger auf Ende des Programms
159-160	$009F-00A0	Temporärer Datenspeicher
161-162	$00A1-00A2	Temporärer Datenspeicher
163-165	$00A3-00A5	Echtzeit-Uhr
166	$00A6	Register für seriellen Bus
167	$00A7	Register für Kassettenroutine
168	$00A8	Register für seriellen Bus
169	$00A9	Temporärer Farb-Vektor
170	$00AA	Register für Kassettenroutine
171	$00AB	Anzahl der Zeichen im Filenamen
172	$00AC	Aktuelle logische File-Nummer
173	$00AD	Aktuelle Sekundär-Adresse
174	$00AE	Aktuelle Gerate-Nummer
175-176	$00AF-00B0	Zeiger auf Filenamen
177	$00B1	Von Fehlerroutine benutzt
178-179	$00B2-00B3	I/O Startadresse
180-181	$00B4-00B5	Basis-Ladeadresse
182-183	$00B6-00B7	Zeiger: Anfang Kassettenpuffer
186-187	$00BA-00BB	Zeiger auf Zeichen im Kassettenpuffer
190-191	$00BE-00BF	Register für Long-Fetch-Routine
192-193	$00C0-00C1	Register für Scrolling
194	$00C2	RVS-Flag (Bildschirm)
195	$00C3	Zeiger: Ende der Zeile (Input)
196-197	$00C4-00C5	Cursorposition (Input), Reihe Spalte
198	$00C6	Tastaturabfrage (S40=keine Taste)
199	$00C7	Flag: Eingabe Bildschirm Tastatur
200-201	$00C8-00C9	Zeiger auf Bildschirmzeile
202	$00CA	Cursorposition in aktueller Bildschirmzeile
203	$00CB	Flag: Anführungszeichen-Modus ($00=nein)
204	$00CC	Länge der aktuellen Bildschirmzeile
205	$00CD	Zeile, in der sich der Cursor befindet
206	$00CE	Letztes Zeichen (I/O)
207	$00CF	Anzahl der Zeichen (Insert-Modus)
208-215	$00D0-00D7	Reserviert für Software
216-232	$00D8-00E8	Reserviert für Anwendungssoftware
233	$00E9	Arbeitsbereich
234-235	$00EA-00EB	Bildschirm-Editor (Farbe)
236-238	$00EC-00EE	Arbeitsbereich (Bildschirm)
239	$00EF	Anzahl der Zeichen im Tastaturpuffer
241-242	$00F1-00F2	Register für Monitor
250	$00FA	Register für X bei Stoptastentest
251	$00FB	Aktuelle Bank-Konfiguration
254	$00FE	Arbeitsregister (Editor)
256- 511	$0100-01FF	Prozessorstack
512- 600	$0200-0258	Eingabepuffer
601- 604	$0259-025C	Basic-Puffer
605- 684	$025D-02AC	Basic-/DOS-Arbeitsbereich
605	$025D	DOS-Schreiberzähler
606- 621	$025E-026D	Bereich für Filenamen
622	$026E	1. Filename (Länge)
623	$026F	DOS (Laufwerk 1)
624- 625	$0270-0271	1. Filename (Adresse)
626	$0272	2. Filename (Länge)
627	$0273	DOS (Laufwerk 2)
628- 629	$0274-0275	2. Filename (Adresse)
630	$0276	DOS logische Adresse
631	$0277	DOS (Geräteadresse)
632	$0278	DOS (Sekundaradresse)
633- 634	$0279-027A	DOS (Disketten-ID)
635	$027B	ID-Flag
636	$027C	DOS (Ausgabepuffer)
637- 684	$027D-02AC	DOS (Arbeitsbereich)
685- 688	$02AD-02B0	Grafik-Cursor
689- 692	$02B1-02B4	Grafik-Cursor (Register)
693- 715	$02B5-02CB	Grafik-Register
716- 739	$02CC-02E3	Print-Using, Grafik-Arbeitsbereich
740	$02E4	High-Byte-Adresse des Character-ROM
747	$02EB	Trace-Flag

752	$02F0	Anzahl der Grafik-Parameter
753	$02F1	Parameter: Relativ (1), Absolut (0)
754- 755	$02F2-02F3	Fließkomma-Vektor
756- 757	$02F4-02F5	Integer-Vektor
768- 769	$0300-0301	Sprungvektor: Fehlermeldung
770- 771	$0302-0303	Sprungvektor: Basic-Warmstart
772- 773	$0304-0305	Sprungvektor: Token-Generierung
774- 775	$0306-0307	Sprungvektor: Keywort erzeugen
776- 777	$0308-0309	Sprungvektor: Hauptschleife
778- 779	$030A-030B	Sprungvektor: Eval
780- 781	$030C-030D	Sprungvektor: Token-Generierung (User)
782- 783	$030E-030F	Sprungvektor: Keywort erzeugen
784- 785	$0310-0311	Sprungvektor: User-Token bearbeiten
786- 787	$0312-0313	Sprungvektor: Interrupt (Uhr)
788- 789	$0314-0315	Sprungvektor: Interrupt
790- 791	$0316-0317	Sprungvektor: Break Interrupt
792- 793	$0318-0319	Sprungvektor: Open
794- 795	$031A-031B	Sprungvektor: Close
796- 797	$031C-031D	Sprungvektor: Kanal für Eingabe öffnen
798- 799	$031E-031F	Sprungvektor: Kanal für Ausgabe öffnen
800- 801	$0320-0321	Sprungvektor: I/O zurücksetzen
802- 803	$0322-0323	Sprungvektor: Input
804- 805	$0324-0325	Sprungvektor: Ausgabe
806- 807	$0326-0327	Sprungvektor: Abfrage der Stoptaste
808- 809	$0328-0329	Sprungvektor: Getin-Routine
810- 811	$032A-032B	Sprungvektor: Schließen aller Files
812- 813	$032C-032D	Sprungvektor: Monitor Break
814- 815	$032E-032F	Sprungvektor: Lade-Routine
816- 817	$0330-0331	Sprungvektor: Save-Routine
818-1010	$0332-03F2	Kassettenpuffer
1011-1012	$03F3-03F4	Datenzähler (Write)
1013-1014	$03F5-03F6	Datenzähler (Read)
1015-1078	$03F7-0436	RS-232-Input-Puffer (64 Byte)
1079-1108	$0437-0454	System-Arbeitsbereich
1109-1138	$0455-0472	System-Arbeitsbereich
1139-1144	$0473-0478	Chrget-Routine
1145-1156	$0455-0484	Chrgot-Routine
1172-1244	$0494-04DC	Bankswitching-Routinen
1255-1258	$04E7-04EA	Print-Using-Parameter
1263-1270	$04EF-04F6	Trap- und Error-Flags
1280-1282	$0500-0502	USR-Sprungbefehl
1283-1287	$0503-0507	Startwert für RND
1288	$0508	Flag für Kalt- oder Warmstart
1289-1298	$0509-0512	Tabelle der logischen Filenummern
1299-1308	$0513-051C	Tabelle der Gerätenummern
1309-1318	$051D-0526	Tabelle der Sekundäradressen
1319-1328	$0527-0530	Tastaturpuffer
1329-1330	$0531-0532	Basic-Anfang
1331-1332	$0533-0534	Basic-Ende
1337	$0539	Zeiger: Kassettenpuffer
1338	$053A	Typ des Kassettenfiles
1344	$0540	Tasten-Widerholfunktion $80=alle, $40=keine, $00=nur DEL, CUR, SPC
1345-1346	$0541-0542	Zähler für Wiederholfunktion
1347	$0543	Shift-Flag
1348	$0544	Letztes Shift-Zeichen
1349-1350	$0545-0546	Sprungvektor: Keylog
1351	$0547	Text- Grafik-Flag
1352	$0548	Scroll-Flag
1355-1372	$054B-0551	Cpu-Arbeitsbereich
1362-1367	$0552-0557	Cpu-Register
1368-1372	$0558-055C	Cpu-Arbeitsbereich
1373	$055D	Zähler für Funktionstasten
1374	$055E	Zeiger: Text, Funktionstasten
1375-1510	$055F-05E6	Speicher für Funktionstasten
1516-1571	$05EC-06EB	1 Page, Bankingroutinen
1516-1519	$05EC-05EF	Adreßtabelle
1520-1521	$05F0-05F1	Long-Jump (Adresse)
1522	$05F2	Long-Jump (Akkumulator)
1523	$05F3	Long-Jump (X-Register)
1524	$05F4	Long-Jump (Status-Register)
1525-1629	$05F5-065D	RAM-Bereich für Bankswitching
1630-1771	$065E-06EB	RAM-Bereich für Benutzersoftware
1792-1967	$06EC-07AF	Basic-Pseudo-Stack
1968-1996	$07B0-07CC	Kassetten-Arbeitsbereich
1997-2000	$07CD-07D0	RS-232-Arbeitsbereich
2001	$07D1	RS-232: Pointer auf Anf. Eingabepuffer
2002	$07D2	RS-232: Pointer auf Ende Eingabepuffer
2003	$07D3	Anzahl der Zeichen im Eingabepuffer
2021	$07E5	Bildschirm: Unterer Rand
2022	$07E6	Bildschirm: Oberer Rand
2023	$07E7	Bildschirm: Linker Rand
2024	$07E8	Bildschirm: Rechter Rand
2026	$07EA	$FF=Automatisches Einfügen
2030-2033	$07EE-07F1	Bit-Tabelle
2034	$07F2	A-Register retten bei Sys-Kommando
2035	$07F3	X-Register retten bei Sys-Kommando
2036	$07F4	Y-Register retten bei Sys-Kommando
2037	$07F5	Status-Register retten bei Sys-Kommando
2038	$07F6	Register für Tastaturabfrage
2039	$07F7	Flag: CTRL-S: $00=offen, $06=gesperrt
2040	$07F8	RAM-ROM-Umschaltung für Monitor 0=ROM, $80=RAM
2044	$07FC	Motorsteuerung

De zeropage wordt vaak gebruikt voor allerlei parameters en pseudo registers in te bewaren, een handig gebied hiervoor is $00D0 tot $00E8.
Na deze 2K zeropage komt van $0800 tot $0C00 het kleurgeheugen. Hier staat van elke plaats op het scherm (in de volgorde zoals je ze leest) in de eerste 4 bytes de kleur, in de bits 4-6 de luminantie, in bit 7 flash on (bit=1). Van $0C00 tot $1000 bevindt zich het echte schermgeheugen, met de pokecode in bits 0-6, en in bit 7 reverse on/off. Van $1000 tot $4000 (of nog verder als je een uitgebreide C16 of een plus4 hebt) bevindt zich het basicgeheugen. dit is niet z'on handige ruimte voor machinetaalprogramma's, want zelfs bij kleine basicprogramma's zitten variabelen in alle uithoeken. Je kunt echter wel b.v. 1K afschermen ($3C00 - $4000) door de volgende pokes (zie memorymap): POKE 56,59 : POKE 52,59. met graphic on blijft er van deze 12K basic nog maar 2K over, n.l. van $1000 tot $1800. Van $1800 tot $2000 bevindt zich het hires-kleurgeheugen en van $2000 tot $4000 de werkelijke hires. Meer RAM is in de 16K versie te vinden van $FD00 tot $FF00 (o.a. TEDchip en I/O). Deze TEDchip stuurt videosignalen, die het beeld samenstellen op je TV/monitor, en brengt ook nog geluid voort. Deze chip kan daarom best belangrijk zijn bij het programmeren van allerlei grafische effecten. hier volgt een goede korte beschrijving (weer een duitse) van die registers :

Register	Adresse	HEX/DEZ	Funktion	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	$FF00 /	65280	Timer 1 (Low-Byte)	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	$FF01 /	65281	Timer 1 (High-Byte)	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
2	$FF02 /	65282	Timer 2 (Low-Byte)	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
3	$FF03 /	65283	Timer 2 (High-Byte)	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
4	$FF04 /	65284	Timer 3 (Low-Byte)	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
5	$FF05 /	65285	Timer 3 (High-Byte)	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
6	$FF06 /	65286	Vertikale Bildschirmposition	Test	ECM ein-aus	HiRes ein-aus	Bildschirm ein-aus	24 oder 25 Zeilen	Vertikale Bildschirmposition		
7	$FF07 /	65287	Horizontale Bildschirmposition	RVS mögl. / RVS unmögl.	PAL/NTSC	horizont. Position einfrieren	Multicolor ein/aus	38 oder 40 Spalten	Horizontale Bildschirmposition		
8	$FF08 /	65288	Tastatur-Latch	Tastatur-Matrix							
9	$FF09 /	65289	Interrupt-Request-Register	IQR	Timer 3	-	Time 2	Time 1	Lightpen-Interrupt	Raster-interrupt	-
10	$FF0A /	65290	Interrupt-Mask-Register/IRQ-Rasterzeile Bit 8	-	Time 3	-	Time 2	Time 1	Lightpen-Interrupt	Raster-interrupt	IRQ-Rasterzeile Bit 8
11	$FF0B /	65291	IRQ-Rasterzeile (Bit 0 bis 7)	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
12	$FF0C /	65292	Hardware-Cursor (Bit 8 bis 9)	- unbenutzt -						Bit 9	Bit 8
13	$FF0D /	65293	Hardware-Cursor (Bit 0 bis 7)	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
14	$FF0E /	65294	Frequenz Tongenerator 1 (Bit 0 bis 7)	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
15	$FF0F /	65295	Frequenz Tongenerator 2 (Bit 0 bis 7)	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
16	$FF10 /	65296	Frequenz Tongenerator 2 (Bit 8 bis 9)	- unbenutzt -						Bit 9	Bit 8
17	$FF11 /	65297	Soundregister	Ton ein/aus	Rauschen ein/aus	Stimme 2 ein/aus	Stimme 1 ein/aus	- Lautstärke -			
18	$FF12 /	65298	Frequenz Tongenerator 1 (Bit 8 bis 9)/Bit Map-Adr./RAM-ROM	-	-	Bit Map Basisadresse			Daten aus RAM/ROM	Tongenerator 1 Bit 9 + Bit 8	
19	$FF13 /	65299	Basisadresse Zeichengenerator/Status	Zeichensatz - Basisadresse						Single Clock	Status
20	$FF14 /	65300	Adresse Bild- und Farb-Speicher	- Bild- und Farbspeicher - Basisadresse -					- unbenutzt -		
21	$FF15 /	65301	Hintergrundfarbe	-	- Helligkeit -				- Farbe -		
22	$FF16 /	65302	Vordergrundfarbe	-	- Helligkeit -				- Farbe -		
23	$FF17	65303	Multicolor 1	-	- Helligkeit -				- Farbe -		
24	$FF18 /	65304	Multicolor 2	-	- Helligkeit -				- Farbe -		
25	$FF19 /	65305	Rahmenfarbe	-	- Helligkeit -				- Farbe -		
26	$FF1A /	65306	2. Cursorpositionierung	-	-	-	-	-	-	- Bit 8 und 9 von Register 28 -	
27	$FF1B /	65307	2. Cursorpositionierung	- Bildschirmpositionierung des 2. Cursors -							
28	$FF1C /	65308	Rasterzeile (Bit 8)	- unbenutzt -							Bit 8
29	$FF1D /	65309	Rasterzeile (aktuell) (Bit 0 bis 7)	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
30	$FF1E	65310	Aktuelle Rasterspalte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
31	$FF1F	65311	Cursorsteuerung	-	- Blinkgeschwindigkeit -				- Vertikale Subadresse -		
62	$FF3E /	65342	ROM-Bank	Schreibbefehl schaltet ROM-Bank ein							
63	$FF3F	65343	RAM-Bank	Schreibbefehl schaltet RAM-Bank ein							

Nu wat betreft het ROM (uitgaande van een standaard c16): Van $8000 tot $d800 bevindt zich de basicinterpreter (probeer maar eens G 8000). De karakterset staat van $D000 tot $D800, en de rest wordt gevuld door het kernal, waarvan het interessant te weten is dat er zich aan het eind van een sprongtabel bevindt ($FF81- $FFF3), waarvanuit je allerlei handige kernalroutines kunt oproepen, voor gebruik in je eigen progamma's. Hier even een (duits) overzichtje.

Adresse	Label	Funktion
$8818	LPUPDT	LinkPointer aktualisieren
$D83B	SETCR	Cursor setzen
$D849	GETCR	Cursor holen
$D88B	CLEAR	Bildschirm loschen
$D89A	HOME	Cursor in HOME-Position
$DC49	BSOUTSCREEN	Zeichen auf Bildschirm ausgeben
$FAFF	PUTWRD	2-Byte-Zahl hexadezimal ausgeben
$FB05	PUTHXS	Bytewert hexadezimal mit Space ausgeben
$FB10	PUTHEX	Bytewert hexadezimal zweistellig ausgeben
$FB20	MAKHEX	ASCII-Code der Hex-Darstellung berechnen
$FF81	VIRES	Video-Chip und Editor initialisieren
$FF84	IORES	I/O-Bausteine initialisieren
$FF90	STMSG	Flag für System-/Fehlermeldungen setzen
$FFBA	SETLFS	File-Parameter setzen
$FFBD	SETNAM	File-Namen setzen
$FFC0	OPEN	File öffnen
$FFC3	CLOSE	File schließen
$FFC6	CHKIN	Eingabegerat setzen
$FFC9	CKOUT	Ausgabegerat setzen
$FFCC	CLRCH	Tastatureingabe und Bildschirmausgabe setzen
$FFCF	BASIN	Zeichen von Eingabegerat holen
$FFD2	BSOUT	Zeichen auf Ausgabegerat ausgeben
$FFD5	LOAD	Programm laden
$FFD8	SAVE	Programm speichern
$FFE1	STOP	Stop-Taste abfragen
$FFE4	GETIN	Zeichen eingeben
$FFE7	CLALL	alle offenen Kanale schließen
$FFF0	PLOT	Cursor setzen (C=0) oder holen (C=1)
$FFF9	RESET	Software-Reset auslosen

Wat u verder nog dient te weten over het geheugen is dit: De computer slaat alle 16bits getallen in 2 bytes op, en wel als volgt: eerst de 8 laagste bits (van 0-7), en in de volgende byte de bits 8-15. je kunt dit o.a. zien door b.v in te tikken A 03F7 LDA $FF1E. na 2 maal return zie je dat $FF1E als $1E, $FF in het geheugen staat.

7. Adresserings methodes.

We hebben tot nu toe gewerkt met direkt daarachter het adres. Echter, er zijn nog veel meer methodes om geheugenplaatsen uit te lezen en te vullen. deze manier waarop noem je adresseringsmethodes. Veel instrukties hebben meerdere adresseringsmethodes, maar ik zal me in de voorbeelden hoofdzakelijk beperken tot LDA

1. Absoluut. Deze hebben we al eerder gehad. V.b: LDA $2000 laadt de accu met de inhoud van geheugenplaats $2000 (8192).

2. Immediate. Laadt de accu direkt met een waarde. V.b: LDA #$E5 vult de accu direkt met de waarde $E5. Het hekje staat er ten teken dat het GEEN adres is.

3. Zeropage. Het zelfde als absoluut, maar dan voor een geheugenplaats in de zeropage ($00 tot $FF), oftewel waarvan het adres maar een byte benodigd. V.b. LDA $3B, vult de accu met de waarde uit adres $003B

4. Absoluut X geïndexeerd. De inhoud van het X register wordt bij het adres opgeteld. Stel dat X de waarde $4C heeft, dan bewerkstelligd LDA $0C00,X dat de accu met de inhoud uit $0C4B gevuld wordt. (dit is o.a. handig bij lussen, zie volgend hoofdstuk).

5. Absoluut Y geïndexeerd. Zelfde als de vorige, maar nu met Y

6. Zeropage X geïndexeerd. Zelfde als de vorige twee, maar nu met een zeropage adres. V.b. met X= $12 vult LDA $56,X de accu met de waarde $68. Deze adresseringsmethode kan ook met Y, maar komt bijna nooit voor, zelfs niet bij LDA.

7. Indirekt geïndexeerd. Dit is een hele handige, vooral bij lussen. Bij de instruktie LDA ($D8),Y wordt uit de zeropage plaatsen $D8 (lowbyte) en $D9 (highbyte) een adres gehaald, en daar wordt Y bij opgeteld. V.b. Y=$10, adres $D8= $30, adres $D9= $20 : LDA ($D8),Y heeft tot gevolg dat de accu gevuld wordt met de inhoud uit adres $2040.

8. Geïmpliceerd. Dit is eigenlijk geen geen adresseringsmethode, dit omvat eigenlijk de instrukties waarop geen adres volgt, zoals TXA, TAX etc.

9. Absoluut indirekt. Deze komt voor bij JMP: hij springt naar de geheugenplaats, welke opgeslagen is in de aangegeven geheugenplaats, V.b: wanneer adres $03F7=$00 en $03F8=$20 springt JMP ($03F7) naar $2000.

De rest van de adresseringsmethodes (o.a. **Geïndexeerd indirekt**) zijn niet nuttig om hier te bespreken. Nu volgt eerst een praktijkvoorbeeld.

8. Toepassingen: lussen en vergelijkingen.

Je zou je kunnen voorstellen dat je snel een geheugengebied met een bepaalde waarde zou willen vullen, of een gebied wilt kopieren. Daarvoor hebt je een lus nodig. Je zou dan een waarde nemen en die bij elke keer eentje verhogen: een lusteller. Dat verhogen gaat met een instruktie: increase, en verlagen met decrease. Voor de accu en het geheugen zijn dat de instrukties INC en DEC, voor de X INX en DEX, voor de Y INY en DEY. Tik de volgende listing maar eens in:

```
(A 03F7) LDA #$A0      (waarde 160)
         LDX #$00      (lusteller)
         STA $0C00,X   ($0C00=scherm, vult schermpositie met reverse-spatie)
         INX           (volgende schermpositie)
         CPX #$C8      (al 200 stuks gehad?)
         BNE $03FB     (niet: nog eens)
         BRK           (end)
```

Verklaring: in A zetten we de waarde waarmee gevuld wordt, en X wordt de lusteller. Vervolgens maken we de eerste positie zwart en verhogen we de teller. Dan vergelijken we de teller met 200. Wat u moet weten voor deze vergelijkings instruktie (CMP voor accu en geheugen, CPX voor X, en CPY voor Y) dat de processor in gedachten (in gedachten?!?!) de twee waardes van elkaar aftrekt en dan kijkt of er $00 uitkomt. Is dat zo, dan wist hij de zerovlag. BNE (branch on not equal) springt naar $03FB als de zerovlag gezet is, dus als X nog geen 200 is. [Je zal je wel verbazen

waarom het adres na branch maar een byte is, want branchen gaat relatief. Die byte geeft aan hoeveel geheugenplaatsen hij naar voren of naar achteren moet springen (bit 7 geeft aan of de sprong positief (naar voren) of negatief (naar achteren) is. sprongen in de vorm van branchen kan dus nooit verder dan 128 byte verder zijn.]. Het resultaat van deze vul-aktie is duidelijk, en valt te bezichtigen met G 3F7. zorg wel dat de cursor in het midden staat.

9. Subroutines: Print,Char.

Leuker is het natuurlijk om met lussen komplete strings af te drukken, eventueel op een aangegeven plaats. We gaan nu een CHAR routine maken. We gieten deze in de vorm van een subroutine, zodat je hem elk moment kunt oproepen. Je kent dit wel uit basic. Je springt naar een subroutine met de instruktie JSR (jump subroutine), gevolgd door het adres. Deze subroutine dient te worden afgesloten door RTS (return from subroutine). Komt de computer echter een RTS tegen, zonder dat daar een JSR aan vooraf is gegaan, springt hij terug naar basic. Zo kun je vanuit basic met SYS subroutines aanroepen, en daarna weer doorgaan in basic (als ze afgesloten zijn met RTS). [Vervang b.v. in ons vorige voorbeeld BRK door RTS en typ in basic SCNCLR: SYS 1015 .] We gaan nu een aantal kernalsubroutines gebruiken. Tik eens in:

```
. 03F7  20 88 D8 JSR $D888
. 03FA  A2 0A    LDX #$0A
. 03FC  A0 00    LDY #$00
. 03FE  20 3B D8 JSR $D83B
. 0401  A0 00    LDY #$00
. 0403  A9 71    LDA #$71
. 0405  85 01    STA $01
. 0407  A9 84    LDA #$84
. 0409  85 02    STA $02
. 040B  A9 C8    LDA #$C8
. 040D  85 00    STA $00
. 040F  B1 01    LDA ($01),Y
. 0411  C8       INY
. 0412  20 49 DC JSR $DC49
. 0415  C4 00    CPY $00
. 0417  D0 F2    BNE $040B
. 0419  60       RTS
```

Uitleg: De eigenlijke routine begint op $040F. Daarvoor gebeurt het volgende: Het scherm wordt schoongemaakt. (Deze routine bevindt zich op $D88B). X en Y worden geladen met de x en y coordinaten, en vervolgens wordt de cursor gezet. (deze routine staat op $D83B). Vervolgens wordt de stringlengte en het beginadres van de string in $D0 tot $D2 gezet. Dan volgt de afdruklus gevolgt door een RTS. Wat u hier moet opmerken is dat de string niet direkt in het schermgeheugen gepoked wordt, maar dat gebruik gemaakt wordt van de kernalroutine BSOUTSCREEN ($DC49), welke de ascii code in A uitgeeft als een karakter. Je kunt nu zelf deze PRINT/CHAR routine aanroepen met JSR $040F, door te zorgen dat de stringlengte en het adres in $D0 tot $D2 staan. Je zou ook zoals in het voorbeeld eerst de cursor kunnen zetten en het scherm schoonmaken (in de tekst kunt u vanzelfsprekend ook stuur en kleurcodes gebruiken, zie daarvoor de ascii tabel in je handleiding.) Wat betreft de kernal routines, deze kun je vinden aan het eind van hoofdstuk 6.

10. IRQ programmeren en de rest van de instrukties.

Irq is een interrupt, een routine die 60 keer per seconde wordt uitgevoerd, ongeacht wat er verder gebeurd. De IRQ laat o.a. 60 maal per seconde de tijd bijstellen, het toetsenbord afvragen en af en toe de cursor knipperen. Het leuke is nu dat de vektor naar die routine in RAM staat, en wij die dus kunnen veranderen. We kunnen deze vektor naar onze eigen IRQ-routine laten wijzen en aan het eind van deze routine terugspringen naar de echte IRQ. Vraag: wat is nu leuk of handig om 60 keer per seconde uit te laten voeren ?!? Mogelijkheden te over. Je zou een extra toetsafvraag kunnen doen, zodat je elk moment een bepaalde routine kunt aanroepen door het indrukken van een toets . Je zou een bepaalde kleur of schermplaats elke keer kunnen veranderen, of, iets ingewikkelder, je zou m.b.v een joystick of muis een pijlcursor op het scherm kunnen laten bewegen en daar pulldown menus mee naarbenedenhalen. Ik pak even een simpel voorbeeldje om je te laten zien hoe je zoiets opstelt:

```
. 03F7  78       SEI              . 0402  58       CLI
. 03F8  A9 04    LDA #$04         . 0403  00       BRK
. 03FA  8D 12 03 STA $0312        . 0404  EE 19 FF INC $FF19
. 03FD  A9 04    LDA #$04         . 0407  4C 42 CE JMP $CE42
. 03FF  8D 13 03 STA $0313        . 040A  00       BRK
```

SEI en CLI dienen ervoor om tijdens het veranderen van de vektor geen IRQ toe te laten, anders zou de computer kunnen crashen. (SEI=set interrupt disable flag, CLI=clear interrupt disable flag). Op $0404 begint de werkelijke interruptroutine die de randkleur eentje ophoogd (probeer ook FF06, FF13, FF14, FF1E en andere). Dan wordt er gesprongen naar de echte routine op $CE42. Je kunt natuurlijk hier zelf je eigen routine invullen (bedenk wel dat hij 60 maal per seconde uitgevoert moet kunnen worden), je kunt kijken welke toets is ingedrukt in adres $C6, en vervolgens eventueel branchen naar je eigen routine.

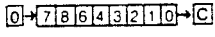
Dit was het wat betreft de introduktie in machinetaal. Je kunt nu zelf verder leren door er boeken over te lezen, bestaande programma's te bekijken, en vooral zelf te proberen. Als je van plan bent veel te programeren, kun je beter overschakelen op een assembler. Met zo'n programma kun je gebruikmaken van labels, gewone regelnummers, vaak ook decimale getallen en LIBS (subroutine libraries). Ik sluit af met een overzichtje van alle instrukties en adresseringsmethodes. Sommigen zal ik nog even kort beschrijven.

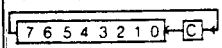
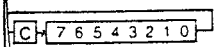
Nog vragen? (of commentaar) schrijf even en ik zal je beantwoorden als ik kan.
Wouter van oortmerssen, lyceumstraat 66, 1814BT Alkmaar.

Veel succes!

Name Description	Addressing Mode	Assembly Language Form	No Bytes	HEX OP Code	Status Register N V - B D I Z C
ADC Add memory to accumulator with carry	Immediate	ADC #Oper	2	69	• • • •
	Zero Page	ADC Oper	2	65	
	Zero Page X	ADC Oper X	2	75	
	Absolute	ADC Oper	3	6D	
	Absolute X	ADC Oper X	3	7D	
	Absolute Y	ADC Oper Y	3	79	
	(Indirect X)	AND (Oper X)	2	61	
	(Indirect) Y	ADC (Oper) Y	2	71	
AND "AND" memory with accumulator	Immediate	AND #Oper	2	29	• •
	Zero Page	AND Oper	2	25	
	Zero Page X	AND Oper X	2	35	
	Absolute	AND Oper	3	2D	
	Absolute X	AND Oper X	3	3D	
	Absolute Y	AND Oper Y	3	39	
	(Indirect X)	AND (Oper X)	2	31	
	(Indirect) Y	AND (Oper)Y	2	31	
ASL Shift left one bit (Memory or Accumulator) C ← 7 6 5 4 3 2 1 0 ← 0	Accumulator	ASL	1	0A	• • •
	Zero Page	ASL Oper	2	06	
	Zero Page X	ASL Oper X	2	16	
	Absolute	ASL Oper	3	0E	
	Absolute X	ASL Oper X	3	1E	
BCC Branch on carry clear	Relative	BCC Oper	2	90	

					N V - B D I Z C
BCS Branch on carry set	Relative	BCS Oper	2	B0	
BEQ Branch on result zero	Relative	BEQ Oper	2	F0	
BIT Test bits in memory with accumulator	Zero Page Absolute	BIT Oper BIT Oper	1 3	24 2C	M M • •
BMI Branch on result minus	Relative	BMI Oper	2	30	
BNE Branch on result not zero	Relative	BNE Oper	2	D0	
BPL Branch on result plus	Relative	BPL oper	2	10	
BRK Force Break	Implied	BRK	1	00	1 1
BVC Branch on overflow clear	Relative	BVC Oper	2	50	
BVS Branch on overflow set	Relative	BVS Oper	2	70	
CLC Clear carry flag	Implied	CLC	1	18	0
CLD Clear decimal mode	Implied	CLD	1	D8	0
CLI Clear interrupt flag	Implied	CLI	1	58	0
CLV Clear overflow flag	Implied	CLV	1	B8	0
CMP Compare memory and accumulator	Immediate Zero Page Zero Page X Absolute Absolute X Absolute Y (Indirect X) (Indirect) Y	CMP #Oper CMP Oper CMP Oper X CMP Oper CMP Oper X CMP Oper Y CMP (Oper X) CMP (Oper) Y	2 2 2 3 3 3 2 2	C9 C5 D5 CD DD D9 C1 D1	• • •
CPX Compare memory and index X	Immediate Zero Page Absolute	CPX #Oper CPX Oper CPX Oper	2 2 3	E0 E4 EC	• • •
CPY Compare memory and index Y	Immediate Zero Page Absolute	CPY #Oper CPY Oper CPY Oper	2 2 3	C0 C4 CC	• • •
DEC Decrement memory by one	Zero Page Zero Page X Absolute Absolute X	DEC Oper DEC Oper X DEC Oper DEC Oper X	2 2 3 3	C6 D6 CE DE	• •
DEX Decrement index X by one	Implied	DEX	1	DA	• •
DEY Decrement index Y by one	Implied	DEY	1	88	• •

					N V - B D I Z C
EOR "Exclusive Or" memory with accumulator	Immediate	EOR #Oper	2	49	• •
	Zero Page	EOR Oper	2	45	
	Zero Page X	EOR Oper,X	2	55	
	Absolute	EOR Oper	3	4D	
	Absolute X	EOR Oper X	3	5D	
	Absolute Y	EOR Oper Y	3	59	
	(Indirect,X)	EOR (Oper X)	2	41	
	(Indirect) Y	EOR (Oper) Y	2	51	
INC Increment memory by one	Zero Page	INC Oper	2	E6	N V - B D I Z C • •
	Zero Page X	INC Oper,X	2	F6	
	Absolute	INC Oper	3	EE	
	Absolute X	INC Oper X	3	FE	
INX Increment index X by one	Implied	INX	1	E8	N V - B D I Z C • •
INY Increment index Y by one	Implied	INY	1	C8	N V - B D I Z C • •
JMP Jump to new location	Absolute	JMP Oper	3	4C	N V - B D I Z C
	Indirect	JMP (Oper)	3	6C	
JSR Jump to new location saving return address	Absolute	JSR Oper	3	20	N V - B D I Z C
LDA Load accumulator with memory	Immediate	LDA #Oper	2	A9	N V - B D I Z C • •
	Zero Page	LDA Oper	2	A5	
	Zero Page X	LDA Oper X	2	B5	
	Absolute	LDA Oper	3	AD	
	Absolute X	LDA Oper X	3	BD	
	Absolute Y	LDA Oper Y	3	B9	
	(Indirect X)	LDA (Oper X)	2	A1	
	(Indirect) Y	LDA (Oper) Y	2	B1	
LDX Load index X with memory	Immediate	LDX #Oper	2	A2	N V - B D I Z C • •
	Zero Page	LDX Oper	2	A6	
	Zero Page Y	LDX Oper,Y	2	B6	
	Absolute	LDX Oper	3	AE	
	Absolute Y	LDX Oper Y	3	BE	
LDY Load index Y with memory	Immediate	LDY #Oper	2	A0	N V - B D I Z C • •
	Zero Page	LDY Oper	2	A4	
	Zero Page X	LDY Oper X	2	B4	
	Absolute	LDY Oper	3	AC	
	Absolute X	LDY Oper X	3	BC	
LSR Shift right one bit (memory or accumulator) 	Accumulator	LSR	1	4A	N V - B D I Z C 0 • •
	Zero Page	LSR Oper	2	46	
	Zero Page X	LSR Oper,X	2	56	
	Absolute	LSR Oper	3	4E	
	Absolute X	LSR Oper,X	3	5E	
NOP No operation	Implied	NOP	1	EA	N V - B D I Z C

Name Description	Addressing Mode	Assembly Language Form	No Bytes	HEX OP Code	Status Register
ORA "OR" memory with accumulator	Immediate	ORA #Oper	2	09	N V - B D I Z C • •
	Zero Page	ORA Oper	2	05	
	Zero Page,X	ORA Oper,X	2	15	
	Absolute	ORA Oper	3	0D	
	Absolute,X	ORA Oper,X	3	1D	
	Absolute,Y	ORA Oper,Y	3	19	
	(Indirect,X)	ORA (Oper,X)	2	01	
	(Indirect),Y	ORA (Oper),Y	2	11	
PHA Push accumulator on stack	Implied	PHA	1	48	N V - B D I Z C
PHP Push processor status on stack	Implied	PHP	1	08	N V - B D I Z C
PLA Pull accumulator from stack	Implied	PLA	1	68	N V - B D I Z C • •
PLP Pull processor status from stack	Implied	PLP	1	28	N V - B D I Z C • • • • • • • •
ROL Rotate one bit left (memory or accumulator) 	Accumulator	ROL	1	2A	N V - B D I Z C • • •
	Zero Page	ROL Oper	2	26	
	Zero Page,X	ROL Oper,X	2	36	
	Absolute	ROL Oper	3	2E	
	Absolute,X	ROL Oper,X	3	3E	
ROR Rotate one bit right (memory or accumulator) 	Accumulator	ROR	1	6A	N V - B D I Z C • • •
	Zero Page	ROR Oper	2	66	
	Zero Page,X	ROR Oper,X	2	76	
	Absolute	ROR Oper	3	6E	
	Absolute,X	ROR Oper,X	3	7E	
RTI Return from interrupt	Implied	RTI	1	40	N V - B D I Z C • • • • • • • •
RTS Return from subroutine	Implied	RTS	1	60	N V - B D I Z C
SBC Subtract memory from accumulator with borrow	Immediate	SBC #Oper	2	E9	N V - B D I Z C • • • •
	Zero Page	SBC Oper	2	E5	
	Zero Page,X	SBC Oper,X	2	F5	
	Absolute	SBC Oper	3	ED	
	Absolute,X	SBC Oper,X	3	FD	
	Absolute,Y	SBC Oper,Y	3	F9	
	(Indirect,X)	SBC (Oper,X)	2	E1	
	(Indirect),Y	SBC (Oper),Y	2	F1	
SEC Set carry flag	Implied	SEC	1	38	N V - B D I Z C 1
SED Set decimal mode	Implied	SED	1	F8	N V - B D I Z C 1
SEI Set interrupt disable status	Implied	SEI	1	78	N V - B D I Z C 1

					N V - B D I Z C
STA Store accumulator in memory	Zero Page Zero Page X Absolute Absolute X Absolute Y (Indirect X) (Indirect) Y	STA Oper STA Oper X STA Oper STA Oper X STA Oper Y STA (Oper X) STA (Oper) Y	2 2 3 3 3 2 2	85 95 8D 9D 99 81 91	
STX Store index X in memory	Zero Page Zero Page Y Absolute	STX Oper STX Oper Y STX Oper	2 2 3	86 96 8E	N V - B D I Z C
STY Store index Y in memory	Zero Page Zero Page X Absolute	STY Oper STY Oper X STY Oper	2 2 3	84 94 8C	N V - B D I Z C
TAX Transfer accumulator to index X	Implied	TAX	1	AA	N V - B D I Z C • •
TAY Transfer accumulator to index Y	Implied	TAY	1	A8	N V - B D I Z C • •
TSX Transfer stack pointer to index X	Implied	TSX	1	BA	N V - B D I Z C • •
TXA Transfer index X to accumulator	Implied	TXA	1	8A	N V - B D I Z C • •
TXS Transfer index X to stack pointer	Implied	TXS	1	9A	N V - B D I Z C
TYA Transfer index Y to accumulator	Implied	TYA	1	98	N V - B D I Z C • •

ADC (v.b. ADC #$4F) telt A bij $4F op, en plaatst de uikomst in A. is deze groter dan $FF wordt de carry geset. (9 bits = max. 511)
AND and met accu (als boven) v.b. AND $041F.
ASL alle bits worden 1 opgeschoven, bit 0 wordt $00 en bit 7 gaat naar de
BCC en BCS branchen resp. als de carry geset is danwel gewist. (hiermee kun je o.a. na een CMP controleren of de waarde groter is dan de accu. (carry gewist))
BPL en BMI branchen als bit 7 resp. geset danwel gewist is.
CLC en SEC wissen en setten de carry.
EOR en ORA (e)or bewerking (als AND).
LSR alle bits worden 1 opgeschoven, bit 7 wordt $00, bit 0 wordt carry.
NOP (dit is de leukste van allemaal. (No OPeration)
PHA en PLA zet A op de stack, en haalt hem er weer af. Handig om even de accu-waarde te bewaren. zorg wel dat je er niets afhaalt wat je er niet eerst opgezet hebt (de processor gebruikt de stack ook!)
PHP en PLP zelfde als boven, maar nu voor het statusregister.
ROR roteert de bits van de accu, bit 7 wordt carry, carry wordt bit 0.
ROL roteert de bits van de accu, bit 0 wordt carry, carry wordt bit 7.
SBC (substract with carry) als ADC, ook nu dient de carry als bit 9 voor A, waar de waarde van wordt afgetrokken. (V.b. SBC $4E).
TAX en TAY etc. verplaatst verschillende registers naar andere, waarbij A,X,Y,S staan voor A,X,Y,stackpointer. V.b. TAX copieert de inhoud van A naar X.

k
karakters in multicolor 2-6-9
kernal-rom versies 2-5-4
koppen stellen 1-2-23
kopstand 1 1-2-21
kopstand 2 1-3-18
kraken gekochte software 1-2-13
kraken winter events 1-4-17
kruiswoord puzzel 2-6-8

l
laad - save problemen 1-2-21
lichtpen 2-2-7
lichtpen 2-4-10
lichtpen maken 2-1-21
lijn '88 3-1-13
logo 1 1-3-10
logo 2 1-4-10

m
m&t basic 2-4-23
mastermind pgr 2-6-19
taal programma's aanpassen 2-1-4
domodule 2-2-23
mercenary plaatjespakker 3-1-20
merge prg 2-5-24
miniatuurtjes 2-1-16

miniatuurtjes 2-3-12

modem plus/4 2-5-4
moduulsoftware 2-3-14
monitor 1-1-5
monty in the run 2-1-30
multicolor demo pgr (karakters) 2-6-10
multicolor karakters 2-6-9

n
novaload copy system 2-4-22

o
oc-118 diskdrive test 2-1-22
old 1-3-6
operating system ebs-1.0 2-2-14
oude rom - script-plus 2-2-13

p
password 2-4-13
password programma 2-3-11
peeks & pokes 1-1-14
peeks & pokes 1-2-16
peeks & pokes 1-3-24
peeks & pokes 1-4-29
peeks & pokes 2-1-25
peeks & pokes 2-2-23
peeks & pokes database ? 2-3-17
persbericht 2-2-25
pgr anti reset 1 1-4-10
pgr anti reset 2 2-4-9
pgr basic uitbreiding 1-4-12
pgr basicode print routine 2-4-7
pgr circle fantasy 2-1-15
pgr dubbel hoge letters 3-1-12
pgr extra hulp toetsen 2-6-5
pgr graphic save/loader 2-3-19
pgr grote letters 3-1-21
pgr hardcopy 1-1-20
pgr interrupten 2-4-17
pgr lichtpen - hirestoolkit 2-4-10
pgr mastermind 2-6-19
pgr menumodule 2-2-23
pgr merge 2-5-24
pgr miniatuurtjes 2-1-17
pgr miniatuurtjes 2-3-12
pgr miniatuurtjes 3-1-22
pgr multicolor karakters 2-6-9
pgr password 1 2-3-11
pgr password 2 2-4-13
pgr pi in x-tig decimalen 2-5-21
pgr pilot 3.0 3-1-16
pgr screen print +4/c=16 3-1-24
pgr tips 2-5-11
pgr udg's 1-2-6
pgr vallende letters 3-1-21
pgr western 2-6-21
pgr write protect 1-4-15
phantom 1-4-5

phantom 2-1-11
pi met ?? decimalen 2-5-21
pilot 3.0 3-1-16
plaatjes saven 1-4-24
plaatjespakker mercenary 3-1-20
plus/4 en 3-plus-1 2-2-13
plus/4 en 3-plus-1 2-5-16
plus/4 geheugen uitbreiding ? 2-2-10
printer 803 met printkit 4 2-3-7
printer citizen dp 560-cd 1-1-3
printer italic op telrol 2-3-18
printkit 4 op de 803 2-3-7
printroutine van basicode 2-4-7
professional bouderdash 2-5-23
programs list bespreking 2-1-8
purple turles 2-5-24

q
quickshot verbouwen 2-3-4
quickshot voor c=16 plus/4 1-1-6

r
race-geluidje 1-1-13
ram uitbreiding voor plus/4 ? 2-2-10
rasterinterrupt 2-4-20
rebound 2-2-16
reparatie datasette 1-3-15
reparatie datasette 2-3-8
reset anti 1 1-4-11
reset anti 2 2-4-9
rom c=16 met fout 2-1-19
rom-kernal versies 2-5-4
romlisting voor c=16 plus/4 boekbespr 2-4-19
rs232 1-4-19

s
screen print c=16 3-1-24
screen print plus/4 3-1-24
script plus handleiding 2-6-18
script-plus en de oude rom 2-2-13
shape effect 2-3-13
sound op de plus/4 c=16 2-1-12
speech samples test 2-1-10
speed copy 2-6-18
spelcartridge als eprom bank 2-5-5
spellen bespreking 1-3-11
start programma voor diskdrive 2-1-10
statistikmacher 2-4-23
summer events 1-4-4
super cobra (+4) 2-5-23
superbase in eprom 2-6-14

t
tachtig kolommen 2-5-14
tekenen 1-4-16
tekstverwerkers 1-1-10
test getest 2-4-25

tijdschriften 1-1-21
tips en truuks 1-3-25
tips en truuks 2-5-11
track 18 2-6-14
track 18 aanvulling 3-1-12
trailblazer 2-1-30
trust 1-3-11
tuberunner 2-5-23

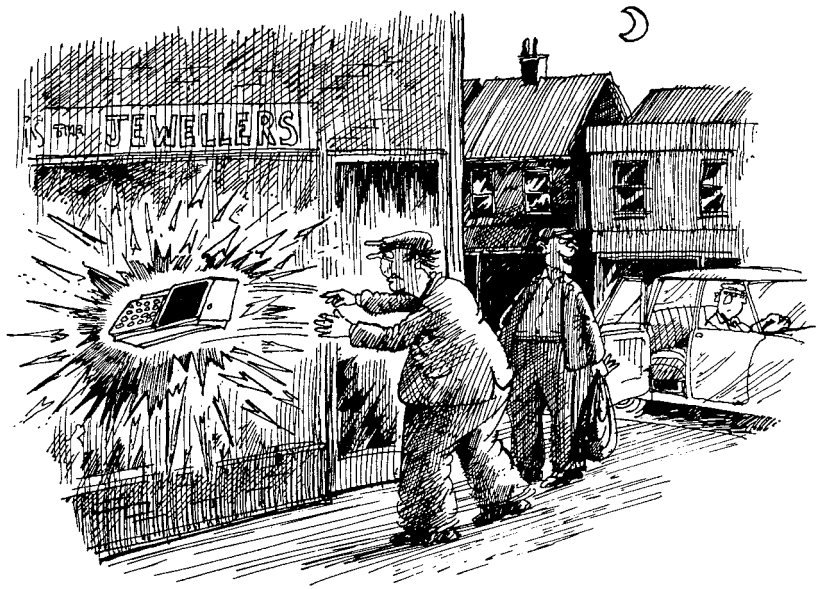
turbo-plus uitbreiding 2-4-4
turbo-plus-module 1-1-27
turbobase 1-3-7
tweede hands computer kopen ? 2-1-19
typewriter 1-4-22

u
u.d.g. in hires toolkit 2-1-7
u.d.g. voor c=16 & plus/4 1-2-6
unex-16 card 1-4-23
using the commodore 16 boekbespr. 1-1-9

v
vallende letters 3-1-21
verem 2-4-22
vic 2-4-23
vinput 2-3-23
virussen 3-1-3
voeding plus/4 3-1-8
voedingsproblemen 1 1-1-7
voedingsproblemen 2 1-2-26
voedingsproblemen 3 1-3-15

w
western c16 pgr 2-6-21
wisp aankondiging 2-1-7
wizzard's drumkit 2-5-13
word pro test 1-3-7
write protect 1-4-15

samengesteld door s.j. francke ermelo



'Fred's very advanced, he uses a computer on all our robberies now.'

listings

MINIATUURTJE

```
10 COLOR4,2:COLOR0,2:COLOR1,1:GRAPHIC1,1
20 DEF FN FX(I)=160+130*SIN(I*121*π/180)
30 DEF FN FV(I)=100+100*SIN(I*61*π/180)
40 X0=FN FX(0):Y0=FN FV(0)
50 FORI=1TO360
60 X1=FN FX(I):Y1=FN FV(I)
70 DRAW1,X0,Y0TOX1,Y1
80 X0=X1:Y0=Y1
90 NEXTI
```

MINIATUURTJE

```
10 GRAPHIC1,1
15 FORI=1TO360STEP12
20 XM=160+65*COS(I*π/180)
30 YM=100+50*SIN(I*π/180)
40 CIRCLE1,XM,YM,65,50
50 NEXTI
```

```
5 REM "GEHEUGENTEST"
10 SCNCLR
20 INPUT "HOEVEEL GETALLEN(MAX. 9)";N
30 SCNCLR
40 PRINT "DIT ZIJN DE ";N;"GETALLEN"
50 PRINT:PRINT:PRINT
60 FOR G=1 TO N
70 A(G)=INT(100*RND(0)+1)
80 PRINT A(G);
85 LET C(G)=A(G)
90 NEXT G
100 FOR Z=1 TO 2000:NEXT Z
110 SCNCLR
120 PRINT "TIK NU DE";N;"GETALLEN IN"
130 FOR T=1 TO N
140 INPUT B(T)
145 LET S(T)=B(T)
150 NEXT T
160 P=0
170 FOR I=1 TO N
180 FOR T=1 TO N
190 IF B(I)=A(T) THEN P=P+1:GOTO 205
200 NEXT T
205 B(I)=0
206 A(T)=0
210 NEXT I
220 PRINT "MIJN GETALLEN ZIJN";
230 FOR G=1 TO N
240 PRINT C(G);
250 NEXT G
260 PRINT:PRINT:PRINT
270 PRINT "UW GETALLEN ZIJN";
280 FOR T=1 TO N
290 PRINT S(T);
300 NEXT T
310 PRINT:PRINT:PRINT
320 PRINT "U HAD DUS";P;"GETALLEN GOED"
330 PRINT:PRINT
340 INPUT "NOG EENS (J/N)";Q$
350 IF Q$="J"THEN 10
360 PRINT "TOT ZIENS"
370 END

READY.
```

Louis van Hunt
Spijkenisse

```
2 REM "GETAL RADEN"
5 COLOR 4,6,2
7 COLOR 0,3,5
10 SCNCLR
20 G=INT(100*RND(0)+1)
22 A=A+1
25 PRINT
28 T=0
29 PRINT
30 T=T+1
40 INPUT"RAAD MIJN GETAL";R
50 IF R=G THEN PRINT "GERADEN IN";T;"BEURTEN":GOTO 81
55 PRINT
60 IF R>G THEN PRINT   "        TE HOOG":GOTO 29
70 PRINT "TE LAAG"
75 PRINT
80 GOTO 30
81 IF T<=5 THEN PRINT "GOED ZO !!!!!!"
84 FOR Y=1 TO 5000:NEXT Y
88 IF A<30 THEN GOTO 10:ELSE END

READY.
```

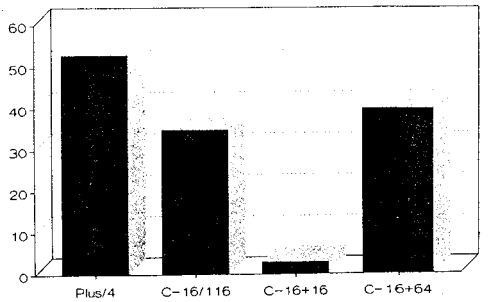
c16/plus4 in beeld.

toen ik laatst weer eens lekker aan het programmeren was, en mijn hoofd brak over een goede printer-routine, vroeg ik mij af of er eigenlijk een soort van standaard was bij c16/plus4 gebruikers waar ik rekening mee te houden had. ik werd nieuwsgierig en heb aan de hand van die handige adreslijst daar eens een enquêtje van gemaakt. als je hem bekijkt valt al meteen op: standaard? niet dus. de nummertjes aan de zijkant zijn de aantallen, (dus geen procenten) gebaseerd op 115 leden. bij de computers zijn duidelijk de 64K machines in de meerderheid. diskdrives zijn er bijna evenveel als datasettes, hoewel er toch nog veel mensen zonder zitten. onder de printers is de verdeeldheid het grootst. opvallend is de standaard van de citizen dr1560, de 'billigdrucker' (met umlaut!), zoals ze hem in duitsland noemen. bij de rest van de randapparatuur is evenwel opvallend het grote aantal cartridges, als betreurenswaardig het aantal muizen. het wordt eens tijd dat er software komt voor dat leuke besturingsapparaatje.

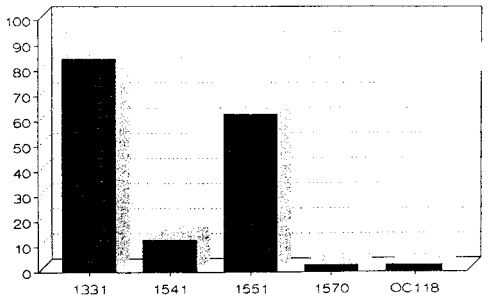
WOUTER

grafieken Tom Muller
op SlideWrite Plus 3.0

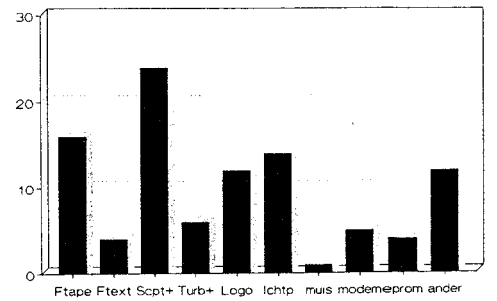
welke computer?



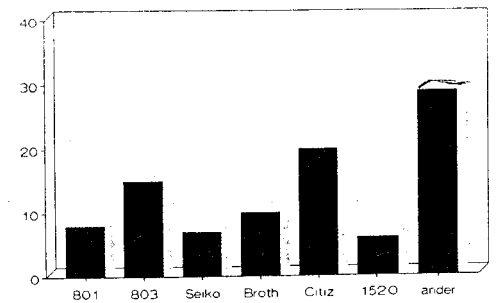
welke dataopslag?



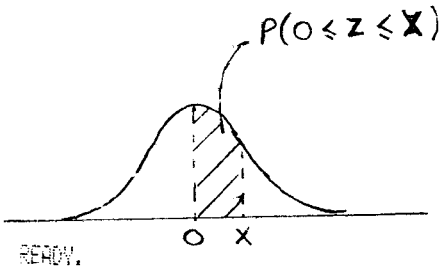
welke randapparatuur?



welke printer?



Voor de statistici onder ons heb ik een programma geschreven, dat de Z-waarden berekent die je gebruikt bij de normale verdeling.
De benadering die ik gebruikt heb stemt redelijk overeen met de werkelijke waardes.
Voor de duidelijkheid heb ik op de tekening
nog even aangegeven welk oppervlak nu berekend wordt.
Mochten er nog vragen zijn dan kun je mij bellen (weekends) of schrijven.



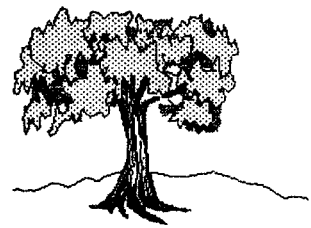
READY.

READY.

```
10 REM *** RONALD BEIJMA ***
20 REM *** TABEL NORMALE VERDELING ***
30 DEFFNA(X)=EXP(-X*X/2)/(SQR(2*π))
40 INPUT"GEEF WAARDE 0<Z<";X
50 Y=FNA(X)
60 X=X+0.05
70 Z=FNA(X)
80 IF Z<0.00001 THEN GOTO 110
90 Y=Y+2*Z
100 GOTO 60
110 Y=Y*0.5*0.05
120 Y=0.5-Y
130 PRINT"DE GEVONDEN Z-WAARDE IS ";Y
140 END
```

READY.

Ronald Beijma



TOP 10 van 1988
door de Rijpstra's

1.-SUMMER EVENTS
2.-WINTER EVENTS
3.-A.C.E +4
4.-SPY VS SPY
5.-HUNGAROING +4
6.-DAVIDS MIDNIGHT +4
7.-KIKSTART
8.-ICICLE WORKS +4
9.-STRIPPOKER II +4
10.-KARATE +4

PILOT INSTRUCTIETAAL

Hier dan het 2E DEEL van het PILOT programma.
Na de listing de cursus over hoe je kan programmeren en wat het programma zelf allemaal kan.
Veel succes bij het intikken van het tweede deel.

```
63550 PC=PC+1:GOTO63510
63560 PC=PC+1:C=PEEK(PC)-65
63570 IFC)25THENPOKE194,1:PRINT"LABEL "
;:GOTO63200
63580 L(C)=PC-5:GOTO63540
63590 GOSUB63230
63600 IFC=82THEN63090
63610 IFC=0THEN63190
63620 PC=PC+1:C=PEEK(PC)
63630 IFC()172THENPC=K:GOTO63190
63640 C=PEEK(PC+1)-65
63650 IFL(C)=0THENPOKE194,1:PRINT"LABEL "
BEL ";:PC=K:GOTO63200
63660 PC=L(C):GOTO63080
63670 RA=RA+1:IFRA=25THENPOKE194,1:PRINT
"STACK OVERFLOW ";:PC=K:RA=24:GOTO63200
63680 R(RA)=PC-1
63690 GOSUB63230
63700 IFC=82THENRA=RA-1:GOTO63090
63710 IFC=0THENRA=RA-1:GOTO63190
63720 PC=PC+1:C=PEEK(PC)
63730 IFC()172THENPC=K:GOTO63190
63740 C=PEEK(PC+1)-65
63750 IFL(C)=0THENPOKE194,1:PRINT"LABEL
BEL ";:PC=K:GOTO63200
63760 PC=L(C):GOTO63080
63770 GOSUB63230
63780 IFC=82THEN63090
63790 IFC=0THEN63190
63800 RA=RA-1:IFRA=-1THENTPOKE194,1:PRIN
T"STACK UNDERFLOW ";:PC=K:RA=0:GOTO6320
0
63810 PC=R(RA+1):C=82:GOTO63090
63820 GOSUB63230
63830 IFC=82THEN63090
63840 IFC=0THEN63190
63850 MA=0
63860 A$=""
63870 PC=PC+1:C=PEEK(PC)
63880 IFC=0THENPC=K:MA=0:GOTO63190
63890 IFC()44THENA$=A$+CHR$(C):GOTO63870
63900 IFM$=A$THENMA=1:C=82:GOTO63090
63910 IFPEEK(PC+1)=0THENMA=0:PC=PC+2:GOT
O63080
63911 GOTO63920
63912 PRINT"■■ OEET JE ZEKER DAT JE HET
 ■■■ PROGR.
63913 PRINT"■ WIL WISSEN ? Y/ !!!!!!
63914 GETA$:IFA$=""THEN63914
63915 IFA$="J"THENPRINT"■■ DRUK DAN OP -
7":PRINT"■DEL1-60000"
63916 IFA$()"J"THENPRINT"■■ HET ■■■ PR
OGRAMMA BLIJFT."
63917 END
63920 GOTO63860
63930 C=8:POKE65286,PEEK(65286)AND239:PO
KE1351,128
63931 COLOR0,2:COLOR4,3,3:PRINTCHR$(14)"
■■■■■■■ -1= STARTEN, -2= LISTEN, -3= WIS
SEN   "
63932 KEY6,""
63940 PRINT"■ -4= MENU 1 (H), -5= MENU 2
, ■■■= HELP ■"CHR$(27)CHR$(84)
63941 KEY1,"■R/:■START PILOT PROGRAMMA"+
CHR$(13)
```

```
63942 KEY2,"■L 1-62000:REM■■LISTING :"+C
HR$(13)
63943 KEY3,"■R/63912■"+CHR$(13):KEY7,"■T
TT"+CHR$(13)
63944 KEY4,"■RUN63951"+CHR$(13)
63945 KEY8,"?HE"+CHR$(13)+"TTTT     ■■■■
■"+"■■■■■LIST"+CHR$(13)
63946 KEY5,"■RUN62500"+CHR$(13)
63950 GOTO63040
63951 POKE806,103
63952 POKE65286,PEEK(65286)AND239:PRINT"
▶▄▟▛                    ▛T-▔ \▽/
  "
63953 PRINT"■   ( PILOT )  ──────
 ( PILOT )   "CHR$(27)CHR$(84)
63954 PRINT"■■■ / KUNT KIEZEN UIT DE VOL
GENDE            MOGELIJKHEDEN :"
63955 PRINT"■    1. ▛L▛T ▛▛T▛▄■\\■ O▛■
-▙ \▽/.
63956 PRINT"■    2. ▛L▛T ▛▛T▛▄■\\■ L■▔
/.
63957 PRINT"■    3. ▛▔▛/▛▔/."
63958 PRINT"■    4. /■■_ ▛L▛T ▛▛T▛▄■\\▛
▔_ \▽▔.
63959 PRINT"■    5. /■■_ \▽/ 2 (HULPJES
 BIJ HET          PROGRAMMEREN)
63970 PRINT"■■■  OAT IS UW KEUZE ?:
63971 POKE65286,PEEK(65286)OR16
63972 GETKEY$
63973 IFA$="3"THENPRINT"■I■■ / HEEFT VAN
 DE";:GOTO63983
63974 IFA$="4"THENPOKE806,101:RUN
63975 IFA$="1"THEN62800
63976 IFA$="2"THEN62600
63977 IFA$="5"THEN62500
63992 GOTO63951
63983 PRINTPEEK(56)*256+PEEK(55)-(PEEK(4
4)*256+PEEK(43));"BYTES DIE U
63984 PRINT" GEWOONLIJK HEEFT NOG";FRE(0
)"BYTES OVER■■"
63985 PRINT"TI VOOR UW ▛L▛T PROGRAMMA.■
■":PRINTTAB(10)"■■■DRUK OP EEN TOETS !!"
63986 GETA$:IFA$=""THEN63986
63987 GOTO63951
```

PILOT cursus 1 :De Fuctietoetsen:

Nadat je het programma hebt geladen en gestart krijg je een schoon scherm met alleen bovenaan 2 regels met wat informatie over de functietoetsen :

F1 : Spreekt voor zich, als een PILOT programma heeft gemaakt drukt u op F1 en het programma wordt gestart.
Denk nu niet dat de computer op hol is want het programma wordt eerst nagekeken en dergelijke kleine foutjes worden er uitgehaald.

F2 : Geeft u de listing van het PILOT programma.

F3 : Met dit kunt u uw PILOT programma wissen (GEBRUIK NOOIT HET COMMANDO - NEW -)
Nadat u op deze knop heeft gedrukt vraagd de computer of je het zeker weet. Drukt u dan op :N: dan zegt hij dat het PILOT programma blijft en drukt u op :J: dan moet u F7 indrukken.

F4 : Met dit krijgt u het HOOFDMENU. Hier kunt u het PILOT pros. wegschrijven en/of laden, op DISK of TAPE.
Hoe u dat moet doen staat in het prosr. zelf.

Stel u weet de naam van het programma niet meer dan drukt u gewoon op RETURN. Ook kunt u kijken hoeveel geheugen je nog vrij hebt om nog te programmeren. Met toets 4 gaat u weer terug naar de programmeermode maar start eerst nog het programma voordat u verder kan.
Pas als het programma loopt kunt u hem nog breaken (stoppen) als het een lang programma is.

F5 : Met F5 gaat u naar MENU 2. U kunt ook via het hoofdmenu daar komen.
In menu 2 zijn wat programmeer hulpjes aanwezig.
Als u een hulp gestart heeft en u wilt dat hij daarmee ophoudt dan moet u dat ook eerst even melden want de computer gaat er gewoon mee door.

HELP ; werkt hetzelfde als gewoonlijk Alleen knippert de regel niet.
Gewoon help helpt niet, vandaar dat ik zelf dat ook in het programma heb bijgevoegd.

PILOT cursus 2 :Wat moet je zeker weten:

1. PROGRAMMEER tussen 0 en 62500
2. Geef GEEN waarde aan regelnummer 0 , 62500 en boven 62500 of veeg ze weg.
3. Zet altijd END: aan het eind van uw PILOT programma
4. Geef uw PILOT programma nooit een langere naam dan 16 (a 17) karakters.
5. en vergeet hem niet eerst weg te schrijven voordat u het programma weg-veegd als u hem wilt bewaren.
6. Geef NOOIT het commando NEW, maar gebruik F3 om het programma weg te vegen.
7. Gewoon het commando HELP helpt niet.
8. Wilt u FUCTIE toets 6 (F6) gebruiken dan niet meer dan 12 KARAKTER.
9. Als het beeld uit is druk dan niet op RUN/STOP, gebeurd dat toch druk dan nogmaals op F1 en wacht rustig af.

PILOT cursus 3 :De taal:

(COMMANDO T: (PRINT))

```
VB :  10 A:$N
   t  20 T:''Hallo $N , ik heet PILOT
      30 END:
```

Het gaat dus nu over REGEL 20.
Daar ziet u na de regelnummer T: staan.
Deze staat voor PRINT.
Na T: krijg je '' oftewel AANHALINGS-TEKENS.
Na de aanhalingstekens kunt u de tekst plaatsen.
Daar kunt u zoals in basic ook gewoon CURSOR tekentjes kwijt.
Variabelen of STRINGS (in basic zo : PRINT''Hallo'';n$;''.....'') doet u in PILOT gewoon binnen de aanhalings-tekens.
U geeft dan eerst het STRING teken en dan een letter.
Na de letter kunt u weer gewoon teksten schrijven.
U heeft dus maar 26 VARIABELEN.

(COMMANDO A: (INPUT))

```
VB :  10 T:''Hallo, hoe heet jij ?
   †  15 A:$V
      20 T:''En waar woon je ?
   †  25 A:$X
      30 T:''Hallo $V, Waar list $X ?
   †  35 A:$U
      40 T:''list $X helemaal in $u ?
```

.let bij de regels 15, 25 en 35 A:$ en dan een letter.
A: staat voor input en $ en dan een willekeurige letter staat voor een variabele.
Als u uw computer een vraag laat stellen in een programma en het antwoordt moet een cijfer zijn dan moet u toch een $ gebruiken omdat wanneer u het in de T: opdracht zou willen verwerken dat niet kan, want hij print dan alleen die letter uit.

```
VB :  10 T:''Hoe heet jij ?
      15 A:G  (inplaats van A:$G)
      20 T:''Hallo G ik heet PILOT
```

. reeel 20 geeft hij nu gewoon :

```
Hallo G ik heet PILOT
```

Wanneer u er bij bijde G's er een $ voor zet dan geeft hij wel :

```
Hallo /en dan hier wat u heeft
insetikt bij REGEL 15/ ik heet
PILOT
```

U kunt zoveel variabelen achter elkaar zetten als u wilt, en u zult ook zien wanneer u gaat printen ziet u de tekst in in de reeel stond met de T: opdracht die print hij per letter af en de variabelen zit u in een keer op het beeld komen.

(COMMANDO J: (GOTO))

```
VB.† 10 *H
     20 T:''C16 & PLUS 4
   † 30 J:*H
```

De opdracht GOTO is in PILOT J: maar achter J: zet u zoals met GOTO geen reeelnummer maar een (*) sterretje en dan een letter.
Ergens anders in het programma staat het zelfde alleen zonder J:.
Op de reeel waar dan *(en dan een letter) staat gaat hij naar toe.
U kunt er zoveel in het programma zetten als u wilt, bijvoorbeeld :

```
10 *A            90 J:*E
20 T:''a         100 *Z
25 J:*Z          110 T:''en nu de
30 *B            120 J:*G
40 T:''b         130 *Y
50 *C            140 T:''en de
60 T:''c         150 J:*D
65 J:*Y          155 *E
70 *D            160 T:''dit was het
80 T:''d         165 end:
```

Maar zo iets is alleen met adventures nodig.

(COMMANDO R: (REM))

```
UB.† 10 R:***** het programma *****
     20 *N
     30 T:''peter meijer, TEL. 03240 -
        20822
   † 40 R:***** deze regels zijn niet
        voor tekst maar om het programma
        even te
   † 50 R:laten wachten !!!!!!!!!!!!!!!!
        !!!!!!!!!!!!!
   † 60 R:!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
     70 J:*N
```

R: staat dus voor REM oftewel commentaar maar omdat R: zoveel tijd in beslag neemt kunt u het ook als FOR.NEXT gebruiken omdat PILOT geen wacht lussen heeft.

(COMMANDO END: (END))

END: is gewoon END maar in PILOT moet u dit woord ALTIJD gebruiken aan het einde van uw programma anders loopt het vast. Achter END: kunt u trouwens van alles zetten zonder een ERROR melding.
UB. 30 end:** het einde van het progr.**

(COMMANDO U: & E: (GOSUB/RETURN))

U: (GOSUB) werkt hetzelfde als J: (GOTO) Achter U: zet u weer *en dan een letter en ergens anders in het programma staat dan *en dan een letter.
Met E:en dan * en een letter gaat hij weer terug waar hij werd geGOSUBt.
UB. 5 J:*B

```
10 *A
20 T:''even geduld
30 R:******************************
35 R:******************************
40 E:*A
50 *B
60 T:''hall
70 U:*A
80 J:*B
```

ding te melden : er is nog 1 commanc
waar ik nog niet achter ben hoe je het kan gebruiken. (het commando M:)
Mocht u er achter zijn gekomen dan zou ik dat graag even willen weten en als

zo dit was het dan, er rest mij nog 1 ding te melden : er is nog 1 commando waar ik nog niet achter ben hoe je het kan gebruiken. (het commando M:)
Mocht u er achter zijn gekomen dan zou ik dat graag even willen weten en als ik er achter ben dan kan u mij altijd nog bellen of schrijven hoe het zit.
U kunt trouwens nog steeds het programma bij mij op bandje krijgen. zie vorig BULLETIN.
Ik kreeg al dergelijke positieve reacties van mensen, DANK hiervoor.
Heeft u nog vragen BEL dan naar PETER MEIJER : 03240 - 20822.

JOYSTICK WAARDES !!!

Tot nu toe heb ik in geen enkel boek alle juiste waardes (om precies te zijn 17) van de Joystick zien staan.
Het boek "VAN START MET DE C16 IN BASIC 3.5" van Luc Sala stonden er wel een paar waardes maar nog niet alles en er zat nog een fout in ook : RECHTS is 3 en geen 6 zoals zij vermeldden.
M terzake, hier volgen alle 17 waardes van de Joystick :

OMHOOG = 1 RECHTS OMHOOG = 2
RECHTS = 3 RECHTS OMLAAG = 4

Verder raadt u het waarschijnlijk al :

OMLAAG = 5 LINKS OMLAAG = 6

Enz..... De FIRE BUTTON of de VUUR KNOP kent de waarde 128, als u nou een programma wilt maken waar je de VUURKNOP in moet drukken en tegelijkertijd de JOYSTICK omhoog moet doen dan telt u die twee waardes bij elkaar op : 128 + 1 dus : FIRE + LINKS = (128 + 7) = 135
POORT 1 heeft dezelfde wardes als POORT 2, alleen bij P2 heeft de VUURKNOP zelf geen waarde. Maar wel VUURKNOP + RECHTS.

Om een programma te maken met een JOYSTICK besturing is vrij makkelijk :
Inplaats van (voor toetsenbord) :
IF A$="-"THEN SOUND 1,100,10 /doet u:/
IF JOY(P)= 128 THEN SOUND 1,100,10
Bij P vult u in 1 of 2, hangt er van af welke J-POORT u wilt gebruiken.

PETER MEIJER, ALMERE, 03240 - 20822

```
10 PRINT"□     L  T  O  6  T  L
 E "
20 PRINT"       O  T       E  A  L
  N":FORL=1TO1000:NEXT
30 GOSUB 1000
40 Z=INT(A)
50 GOSUB 1000
60 U=INT(A)
70 GOSUB 1000
80 R=INT(A)
90 GOSUB 1000
100 F=INT(A)
110 GOSUB 1000
120 T=INT(A)
130 GOSUB 1000
140 H=INT(A)
150 GOSUB 1000
160 O=INT(A)
170 PRINT"□TOETS VOOR VERDER >>>":GETKEY
WH$
180 PRINT"□DE LOTTOGETALLEN ZIJN :"
190 PRINT"■■■Z"-"U"-"R"-"E"-"T"-"H"-"O;
GOTO 1'00
1000 SCNCLR
1010 A=RND(INT(X))*10
1020 FORL=1TO 9
1030 PRINT L" LOTTO-GETALLEN"
1040 NEXT L
1050 PRINT"□——"
1060 PRINT"] "INT(A)
1070 PRINT"——
1080 PRINT"■■■TOETS >>>":GETKEYWH$
1090 RETURN
1100 PRINT"■■■NOG EENS (J=JA  N=NEE)"
1110 GETKEYWH$:IFWH$="J"THEN RUN
1120 IF WH$="N"THENPRINT"■■JAMMER HOOR !
!":END
1130 GOTO 1110
```

Ben de Groot, Capelle a/d Yss.

WOUTERS PROGRAMMING LANGUAGE MOUSE OPERATING SYSTEM.

It's new, it's great and it's programmed by Woutersoft for C16/plus4 computers. This program provides not only a non-keyboard operating system (everything is done by mouse or joystick), it also provides real IBM/Amiga windows, pulldownmenues, IRQ control, swapfiles up to 0,75 megabytes, unique hires system (multi hiresscreens on any place of textscreen), programming by selecting instruktion out of pulldownmenue, keyboard on screen for typing actions, autostart, programprotection, programming language faster than basic and almost as fast as machine language by unique idiotproof entering system, use of ram on disk, files of any size, append files, read and write at same time, direct acces to any byte of file, colourfull drawing options, sample tape at any moment, small programs mergable while running, directory windows, programmable characters, double size characters, sprite-like objects and all features direct accessable from program istruktions.

and offcourse, there's more . . . . .

WPL MOUSE OPERATING SYSTEM; BUY IT OR YOUR COMPUTER WILL FALL ASLEEP FOR NEVER TO WAKE UP AGAIN . . .

Wouters programming language mouse operating system by woutersoft (c) 1989 call the netherlands 072-125515, lyceumstraat 66 , Alkmaar.