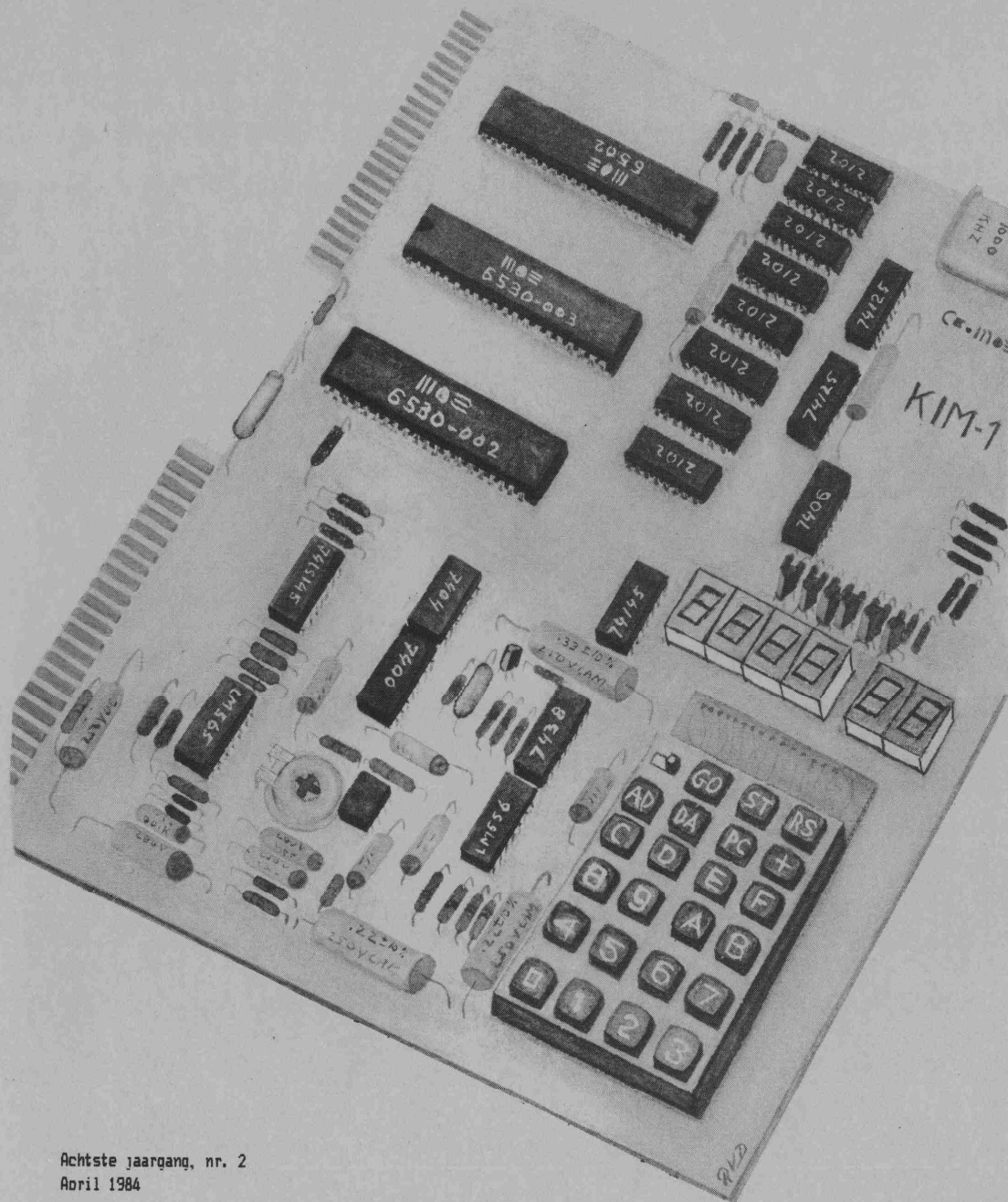




KIM GEBRUIKERS CLUB

NEDERLAND

een club van 6502
gebruikers

Het doel van de vereniging is: het bevorderen van de kennisuitwisseling tussen de gebruikers van 6502-computers, zoals KIM, SYM, JUNIOR, AIM-65, SYSTEM 65, ACORN, PET en CBM, APPLE, CHE-1, ATARI, de Proton-computers, ITT 2020, PEARCOM, PC-100, OSI, BBC, VIC-20 en COMMODORE 64, BASIS 108, PALLAS, SBC 65/68, FORMOSA, ORIC-1, ACE 1000, KEMPAC Systeem 4, NCS 6502, MINTA, enz. enz.

Deze kennisuitwisseling wordt o.a. gerealiseerd door zesmaal per jaar DE 6502 KENNER te publiceren, vijf maal per jaar clubbijeenkomsten te houden, een cassettebibliotheek in stand te houden en papierware-service te bieden. Daarnaast organiseren de leden zelf regionale bijeenkomsten.

Verschijningsdata DE 6502 KENNER =====

derde zaterdag
van de maanden:
februari
april, juni
augustus
oktober
december

Regio Den Haag/R'dam e.o. =====

Inl. Gerard van Roekel
v.d. Palmstraat 11-C
3135 LK Vlaardingen
Tel.: 010 - 351101

Bijeenkomsten van de club =====

derde zaterdag
van de maanden:
januari
maart
mei
september
november

Regio Arnhem e.o. =====

1e en 3e Ma-avond v.d. maand
in HET DORP
(zaal wordt aangegeven in de
hal van de personeelsflat)
Inl.: Henk Deutekom
Tel.: 08303 - 8987

De KIM GEBRUIKERS CLUB NEDERLAND is een volledig onafhankelijke vereniging met statuten en een bestuur. De club is ingeschreven bij de Kamer van Koophandel en Fabrieken voor Hollands Noorderkwartier te Alkmaar, onder nummer 634305.

Samenstelling van het bestuur is thans als volgt:

Voorzitter:	Dagelijks bestuur: Rinus Vleesch-Dubois Fl. Nightingalestraat 212 2037 NG Haarlem (023 - 330993)
Erevoorzitter:	Siep de Vries
Sekretaris:	Ruud Uhoff Voorburgpad 10 6843 EM Arnhem (085 - 816935) (19.00 - 20.00)
Penningmeester:	Riet Uhoff - Burghoorn voor adres: zie bij sekretaris
Ledenadministratie:	Leden: Ruud Uhoff (zie bij sekr.)
Redactie DE 6502 KENNER	Willem L. van Pelt Jacob Jordaensstraat 15 2923 CK Krimpen a.d. IJssel (01807 - 19881)
Hard-/Software:	Adri Hankel (05490 - 51151) John van Sorang (01807 - 20589) Erwin Visschedijk (05490 - 71416) Jan Tompot (070 - 931417)
Promotie:	Mw. H. de Vries - Van der Winden
Ereleden:	Anton Mueller
Adres van de vereniging:	zie adres sekretaris
Lidmaatschappen:	Fl. 45,- per kalenderjaar, postrekening 3757649 t.n.v. KIM Gebruikers Club te Arnhem
Advertentietarieven:	Op aanvraag bij de redactie

DE 6502 KENNER

DE 6502 KENNER is een uitgave van de KIM Gebruikers Club Nederland.

Adres voor het inzenden van en reacties op artikelen voor DE 6502 KENNER:

Willem L. van Pelt,
Jacob Jordaensstraat 15,
2923 CK KRIMPEN AAN DEN
IJSSSEL.
Tel.: 01807 - 19881

Redactie DE 6502 KENNER:
Vaste medewerkers:

Willem van Pelt
Gerard van Roekel
Frans Smeehuijzen
Jaap van Toledo

Freelance medewerkers:

Adri Hankel
Renee de Hoop
Fridus Jonkman
Herman Kuyvenhoven
Roger Langeveld
Anton Mueller
Koen van Nieuwenhove
Erwin Rosseeuw

Gehele of gedeeltelijke overname van de inhoud van DE 6502 KENNER zonder toestemming van het bestuur is verboden. Toepassing van gepubliceerde programma's, hardware etc. is alleen toegestaan voor persoonlijk gebruik.

DE 6502 KENNER verschijnt 6 x per jaar en heeft een oplage van 500 exemplaren.

Copyright (C) 1984 KIM Gebruikers Club Nederland.

De voorpagina is een aquarel van een KIM, geschilderd door:
Rinus Vleesch Dubois

INHOUDSOPGAVE DE 6502 KENNER NR. 31 APRIL 1984

1. Van het bestuur	2.
2. Agenda	3.
3. Van de redactie	3.
4. UITNODIGING Clubbijeenkomst Rijswijk	4.
5. Commodore 64 cassette-interface	2.
Boekeninformatie: Basic met de Commodore 64	10.
6. BUG in BOUW eens een MILJONOR! ... herziene schema's	2. 46.
7. BASIC ...De Proton Basic is er. Piet Fransen ...Uitbreidingen op de SYM-1 Basic. Frans Smeehuijzen Editor functies ...Berekening Autokosten. Will Cuijpers	2. 5. 47.
8. Brief van J. Kemfers PAORJK	10.
9. Obsessie: reactiespel voor hex-disolay ...Marcel van Doro	11.
10. Processor met ... PUSH X = PHX ??? ...Herman Burgers. Texas. USA	13.
11. Tips & Tricks ...Gerard van Roekel	16.
12. De VDU-kaart ...J.J.A. Janssen	17.
13. Program Cassette 2 (zie ook editie 30) ...Ad Brouwer	27.
14. FORTH. FORTH ...Reinier Kleibool Tafels van vermenigvuldiging tot max. 3200 ...Maarten Lamey	40. 45.
15. Marktinfo	48.
16. Vraag en Aanbod	10. 45.

VAN HET BESTUUR

Welkom Commodore-64.

Alhoewel de COMMODORE-64 al enige tijd geen onbekende in het land meer is, is nu het moment voor de vereniging aangebroken om meer aandacht aan dit lid uit het 6502 gezin te gaan besteden. Gewoon omdat de club steeds meer leden gaat tellen die in het bezit zijn van dit verkoopssucces van Commodore. Dat betreft dan zowel nieuwe leden, als mensen die zijn omgeschakeld op de C64, soms als tweede systeem naast KIM of JUNIOR.

Zoals we van Commodore gewend zijn, is het een machine die uitzonderlijke prestaties koppelt aan een aantal even uitzonderlijke klunzigheden. Naast het voortreffelijke "operating system", de KERNAL, vinden we een BASIC die al vrijwel klaar lag uit de PET/CBM-serie en dus geen enkele ondersteuning biedt aan de fantastische grafische, geluids- en kleurenmogelijkheden van de C64. Deze mogelijkheden komen overigens pas tot hun recht indien in machinetaal geprogrammeerd wordt, doch de machinetaalmonitor ontbreekt. Er zijn diverse monitors leverbaar op tape of in ROM-pack. De beste versie blijft "Supermon", dat U eenvoudig mag overnemen (bel mij even). Commodore zelf levert 64MON in ROM. Deze mist de uitbreidingen van "Supermon" maar functioneert. Trap er niet in om een ROM-pack 'TOOLKIT-64' te kopen voor de prijs van ca. F. 149,==. Een grof schandaal voor een toolkit die stommeg niet werkt en een monitor die routines uit de Basic-interpretator gebruikt zodat U hem kunt vergeten als de interpreter wordt uitgeschakeld om over 8K extra RAM te beschikken. Het ding heet ook wel "VIC AID" en is afkomstig van computercentrum CONDOR te Vlissingen.

We geven Commodore hier meteen even een dikke onvoldoende voor de documentatie die beneden elk pijl is. Het meegeleverde boekje is een leuk ding voor beginners. Wie echte documentatie wil hebben wordt verwezen naar de boekhandel waar iets beters te koop zou moeten zijn: "COMMODORE 64 PROGRAMMERS REFERENCE GUIDE". Hier staat wel zo ongeveer alles in maar een "reference guide" is het zeker niet. De gezochte informatie is zeer moeilijk te vinden. Niet kort, zakelijk en op een rijtje, maar veel gezwaam met voorbeelden. Een goed leerboek, dat wel!

Commodore levert ook een assembler die mij niet kon ontroeren. Er gaat toch maar niets boven ASSM/TED van C. Moser. Daarom hebben Anton Mueller en ik de PET-versie van deze assembler aangepast voor de C64. ASSM/TED wordt geleverd door ingenieursbureau Schroeder in Eindhoven. U zou dus de PET versie daar moeten kopen en onze aanpassingen er op los moeten laten. De club heeft echter toestemming om de kant en klare C64 versie namens de leverancier te mogen verkopen. Vanzelfsprekend tegen de normale prijs (ca. Fl. 152,==). U dient schriftelijk te bestellen op een software verklaring, welke verkrijgbaar is op het secretariaat. Ondertekening daarvan betekent dat U de onrechtmatigheid erkent van elke vorm van verder verspreiden. Doet U dat ook a.u.b. niet! Goede software wordt anders onbetaalbaar, omdat de auteurs net als U niet voor niets willen werken.

Tenslotte: Voor de COMMODORE-64 gaan we een documentatie-map aanleggen. Geïnteresseerden moeten even een briefkaart naar het secretariaat sturen. U ontvangt dan bericht terug hoe we een ander gaan organiseren. Ook voor andere systemen zullen we binnenkort iets dergelijks gaan starten, waarover later meer in DE 6502 KENNER.

RUUD UPHOFF.

BELANGRIJK PROTON NIEUWS

De Proton Basic is er.

Reeds lang was deze Basic in het vooruitzicht gesteld, maar is nu in mijn bezit. De Basic is een compiler-Basic die met labels kan werken. Het toepassen van labels maakt deze Basic zeer aangenaam tijdens gebruik. Als er bijv. een commando RENUM uitgevoerd wordt, blijft een programma in orde. De Basic beschikt over een aantal unieke instructies, zoals: REPEAT ... UNTIL, IF ... THEN ... ELSE, TRACE MODE, OPEN en CLOSE commando's voor directe printer en diskbesturing en nog een aantal andere instructies om o.a. hexadecimaal te werken. De Basic gaat vergezeld van een uitgebreide documentatie, die U op Uw floppy meegeleverd krijgt. Ook de FORTH-versie is vernieuwd en werkt nu bij mij uitstekend. De nieuwe FORTH bevat nu ook EDIT en ASSEMBLER functies. De Proton-bezitters die beide programma's in hun bezit willen krijgen, sturen mij een voor Proton geïnitieerde floppy en een postzegel voor het retourneren. Tot slot het volgende. De firma Proton heeft bij de utility-programma's ook een programma FORM* geleverd. Tijdens uittesten bleek dit programma bij mij niet te werken. De oorzaak was, dat ik een OKI-80 printer gebruik waarvoor dit programma niet ontworpen is. Ik heb inmiddels een nieuwe routine gemaakt die wel functioneert. Deze routine, *FORM*, herkent nu in een editor-file alle opties van de OKI-80 printer en voert deze korrekt uit. De files mogen in het geheugen staan, maar worden indien gewenst ook van van floppy gehaald. Verder zijn er een aantal commando's die tijdens het listen van de pagina's gebruikt kunnen worden. Diegene die geïnteresseerd is in de code van deze formlister, moet tevens even opgeven waar de routine geplaatst moet worden. Hij kan niet meer, gezien zijn lengte, op \$200 geassembleerd worden. Eventueel kan als startadres \$3000 gekozen worden omdat de assembler niet gelijktijdig geladen hoeft te zijn. Zonodig telefonisch overleg.

Piet Fransen
Voordewind 21
1034 KS Amsterdam
Tel.: 020-319271

BOUW eens een MILJONOR-bug.

In editie 30 publiceerde J.H. Vernimmer het artikel met de bovengenoemde titel. Helaas zijn in de schema's toch nog enkele fouten achtergebleven. Zodra de schema's zijn herzien, zullen wij deze opnieuw publiceren.

CASSETTE-INTERFACE voor COMMODORE-64.

De Firma Vogelzang adverteert al enige tijd over een interface waarmee het mogelijk is een normale cassette-recorder te gebruiken op de Commodore-64 en de VIC-20. Het ding kost volgens de advertentie Fl. 49,==. Alleen, als de firma er een gewoonte van maakt om ongefrankeerd af te zenden, dan moet U rekenen op ruim Fl. 7,== bijkomende kosten! Handiger lijkt het om meteen de schakeling na te bouwen uit het september 1983 nummer van MC of die uit CHIP van juli 1983.

Overigens vernemen we op de redactie keer op keer dat het nog veel handiger is om met een Snellaad/Toolkitmoduul te werken op de datasette-recorder. Dit moduul zorgt voor het laden en saven met een snelheid die hoger is dan mogelijk is met de C-1541 diskdrive. Om 16K te laden heeft de drive 58 sec. nodig en het moduul slechts 50 sec.! Retourneveloep en postzegel en U krijgt hierover informatie.

AGENDA

- DI 08-05-84 COMPEC EUROPE 84
 WO 09-05-84 Brussel, België.
- ZA 19-05-84 CLUBBIJEENKOMST TE RIJSWIJK (ZH)
 R.K. HTS "Rijswijk"
 Lange Kleiweg 4 te Rijswijk.
 Gastheer: Cees Bootsma.
 Tel.: 070 - 273351
 Ervaringen met OHIO-DOS op JUNIOR.
 Sprekende Commodore 64.
 Presentatie nieuwe DOS voor JUNIOR.
 Simulatie 65C02.
- DI 22-05-84 MICRO EXPO
 VR 25-05-84 Paris, France
 Info: Sybex, 4 Place Felix Eboue,
 75583 Paris Cedex 12, France.
- ZA 09-06-84 REGIONALE 65XX BIJEENKOMST
 CLUBHUIS DE BOSKANT, VILLEGOMEN, DEN HAAG
 Informatie: G. van Roekel, Tel. 010-351101
- ZA 15-09-84 CLUBBIJEENKOMST TE ALMELO
 Koppelen van computers.
 Erwin Visschedijk en Adri Hankel
 publiceren daarover in augustus a.s.
- ZA 17-11-84 CLUBBIJEENKOMST TE AMERSFOORT
 Algemene Ledenvergadering.
 Na de middag informeel gedeelte.
 Toegang gratis.
- Iedere 1e en 3e Ma-avond v.d. maand in HET DORP:
 REGIONALE 65XX BIJEENKOMST
 zaal wordt aangegeven in hal personeels-
 flat
 Informatie: Henk Deutekom, 08303-8987

Mentorencorps: Commodore-64 : Ruud Uthoff
 JUNIOR : Adri Hankel
 Erwin Visschedijk
 Apple : ?
 Kim-1 : ?
 SYM-1 : ?
 Aim-65 : ?
 Wie geeft zich op?

's Zondags vanaf 19.10 uur via Hilversum 2
 Hobbycoop, vaak met computerprogramma's, die
 m.b.v. een cassette recorder zijn op te nemen.
 Door het gebruik van 'Basicode' is elk uitge-
 zonden programma met Uw computer in te lezen.
 De Basicode voor de OSI en voor de APPLE II
 zijn gepubliceerd in de HCC-Nieuwsbrief nr.34
 van nov/dec 1981.
 De Basicode voor de JUNIOR werd gepubliceerd in de
 Elektuur van oktober 1983 en heeft betrekking op
 de Microsoft Basic KB-9.

Onze club heeft sedert de oprichting in 1977 veel nadruk
 gelegd op programmeren in machinetaal. De 'jongens van het
 eerste uur', w.o. onze ere-voorzitter en oprichter Siep de
 Vries, zwoeren erbij. Logisch ook, want alles dat boven de
 machinetaal is ontwikkeld, de assemblers en de hogere pro-
 grammaertalen, wordt eerst omgezet in machinetaal om de mi-
 croprocessor te laten werken. Het zijn 'slechts' hulpmiddel-
 en die het de programmeur gemakkelijk maken een programma te
 ontwikkelen. De programmeur moet dit gemak overigens betalen
 met verlies aan snelheid.

De club heeft ervaren dat de wereld van hobbyisten er een is
 waar 'meerdere talen' wordt gesproken of, althans, waar de
 behoefte bestaat niet alleen met machinetaal bezig te zijn.
 Na aarzende publikaties van Basic-programma's komen enige
 activiteiten los op het gebied van FORTH. Met de komst van
 de club-COMAL en de mogelijkheid om in FOCAL te programmeren
 mogen we verwachten dat het palet van programmeurs in onze
 club vele kleuren kent. In deze schakering vinden we
 ook de assemblers zoals de ASSM/TED van Carl Moser en de
 door Sebo Woldringh tot 8K uitgebreide Micro-ADE van Peter
 Jennings van Micro-Ware Ltd. Van de laatste heeft de club de
 copy-rights, zodat we de gehele, becommentarieerde source-
 listing kunnen aanbieden.

De oude vrees dat onze club een Basic-club zou worden als we
 de aandacht ook op hogere programmeertalen zouden richten is
 niet bewaarheid. We blijken zo nieuwsgierig, dat we niet al-
 leen genoeg nemen met het gebruik van een programmeertaal,
 maar we willen deze ook aanpassen, verbeteren, toesnijden op
 eigen behoeften. En dat kan alleen, zal ik het eens mooi zeg-
 gen: met een in-depth-look, de programmeertaal als een machi-
 taal te beschouwen. Dat hier een dilemma ontstaat door het
 ontbreken van documentatie houdt niet tegen dat in de club
 veel tijd wordt besteedt aan dit aspect. Waarmee maar gezegd
 wil zijn dat machinetaal het meest uitdaagt.

In het verlengde hiervan wil ik voor de nieuwkomers onder
 ons op het hart drukken dat kennisuitwisseling (onze pri-
 maire doelstelling) plaats vindt door het publiceren van
 artikelen in DE 6502 KENNER. Dat geldt voor alle leden. Er-
 van overtuigd dat wij ons hiertoe aangetrokken voelen stel-
 de ik eerder al eens: de club, dat bent U!! Daarmee wil
 ik tot uitdrukking brengen dat het lidmaatschap meer met
 zich brengt dan louter consumeren van hetgeen anderen publi-
 ceren. Je moet er zelf aan meedoen! Daarbij is het van be-
 tekenis te weten dat bestuursleden en redactieleden niet
 primair bezig moeten zijn met het schrijven van artikelen
 of met het ontwikkelen van programma's, hoe aantrekkelijk
 dat ook is. Mijn persoonlijke hartstocht heb ik onlangs nog
 eens geopenbaard bij een bezoek aan iemand met een C64. "De
 leden kennen nu de democratie achter hun clubblad: het is
 hun clubblad: zij moeten erin schrijven".

De in deze editie voorkomende auteurs hebben dat begrepen.
 Het is een greep uit de copy-buffer. Met een blik op de menu-
 kaart en gekoepeld aan actuele ontwikkelingen in de club. Al
 betekent het voor de inzender soms lang wachten, het levert
 een samenstelling op welke gezien, en gelezen mag worden. Zo
 konden we doorgaan met het verbeteren van de SYM-Basic, waar-
 we wellicht inspirerend werken om andere Basic's onder han-
 den te nemen. Er is weer wat voor hex-disolav, de nieuwe pro-
 cessor onder de loep. Tips en Tricks waartoe ik iedereen uit-
 nodig deze op te sturen, we gaan eens kijken of de VDU-kaart
 op een andere manier kan werken, en we hebben wat voor onze
 FORTH en BASIC-liefhebbers. Inzenders, bedankt!

W.L. van Pelt

UITNODIGING BIJeenkomst

Datum : zaterdag 19 mei 1984
Lokatie : R.K. HTS "Rijswijk"
Lange Kleiweg 4 te Rijswijk. Tel.: 070 - 907839

Reisroute :

- per auto - komende uit de richting Utrecht
Volg autoweg E8 Utrecht-Den Haag. Knooppunt Leidschendam via de hoge
rijbaan linksaf richting Delft/Rotterdam. Aan het eind daarvan rechts-
af richting Den Haag. Rechts aanhouden. Onder verkeersbord Ypenburg
door en rechtdoor met trambaan mee over Hoornbrug. Dan scherpe draai
naar rechts en onder brug door. Delft/Rotterdam en Wateringen aan-
houden. Bij stoplichten linksaf en Sir Winston Churchillaan uitrijden
tot laatste stoplichten voor spoorovergang. Links van U gebouw GAK en
ABN bank, hier linksaf Huis te Landelaan op, uitrijden tot eind en
rechtsaf. Utrijden tot schoolplein.
- komende uit de richting Amsterdam
Volg A4 A'dam-Rotterdam. Knooppunt Leidschendam passeren. E8/E10 aan-
houden tot eind, rechtsaf richting Den Haag. Rechts aanhouden. Onder
verkeersbord Ypenburg door met trambaan mee over Hoornbrug. Verder
als boven.
- komende uit de richting Rotterdam
Volg E10 R'dam-Den Haag tot vlak voor Hoornbrug. Onder verkeersbord
Ypenburg door en rechtdoor met trambaan mee over Hoornbrug. Verder
als boven.
- per trein - Station Rijswijk uitgang richting winkelcentrum De Boogaard. Bij uit-
gang linksaf. Spoorovergang Winston Churchillaan over tot stoplicht,
rechtsaf Huis te Landelaan. Verder als boven.

GEWIJZIGDE

TOEGANGSPRIJS: Fl. 10,=. Lunchpakket zelf meenemen.
Consumpties tegen betaling.
Deze regeling is een experiment op verzoek van
de ledenvergadering. (Toegangsprijs v/h Fl.25,=)

Programma:

- 9.30 Zaal open
10.15 Opening
10.30 Erwin Visschedijk en Adri Hankel
over hun ervaringen met OHIO-DOS;
voor- en nadelen van het gebruik.
11.45 Koffiepauze
12.00 Forum
Aansluitend Lunchpauze.
13.30 **INFORMEEL GEDEELTE**
Aan de marktkraam demo met BBC materiaal en software w.o. direkt op
diskette kijken (track voor track uitlezen). Programmeren van 2764
en 27128. Printer buffer 10K.
Presentatie nieuw DOS voor JUNIOR.
Een der leden/leerling der HTS is gevraagd zijn programma "Simuleren
65C02" te demonstreren.
Verder is er een sprekende Commodore 64.

BRENG OOK UW SYSTEEM MEE ALS DAT ENIGSZINS LUKT !!
MARKT. Op eigen tafel(s) te regelen.

17.00 Sluiting.

ID=

EDITOR 6502 ASSEMBLER V3.1 PAGE 01

0010: 0000

EDITOR ORG \$0000

0020:

0030:

0040:

UITBREIDING OP DE SYM-1 BASIC VOOR KIM-1 EN JUNIOR

0050:

0060:

E D I T O R - F U N K T I E S

0070:

0080:

0100:

AUTEUR: F. J. M. SMEEHUIJZEN

0110:

LIPPEDAL 19

0120:

2904 CL CAPELLE AAN DEN IJSSEL

0130:

TEL: 010-512507

0140:

0150:

WANNEER MET BEHULP VAN DE SYM-1 BASIC "MOEIZAAM" PROGRAMMA-
REGLS ZIJN INGETOETST EN ER BLIJKEN ACHTERAF NOG EEN AANTAL
(TYPE)FOUTEN IN TE ZITTEN, DAN ZIT ER NIETS ANDERS OP DAN
DE FOUTE REGELS OPNIEUW IN TE TOETSEN.
OF WANNEER IN EEN PROGRAMMA MEERDERE KEREN DEZELFDE OF BIJNA
DEZELFDE PROGRAMMAREGELS VOORKOMEN ZOU HET PRETTIG ZIJN DAT
NA EEN KEER INTOETSEN VERVOLGENS DE REGEL GEKOPIEERD KAN
WORDEN.

0240:

MET BEHULP VAN DE BASIC-EDITOR IS HET MOGELIJK OM EVENTUEEL
GEMAAKTE FOUTEN IN PROGRAMMA-REGLS EENVOUDIG EN SNEL TE
HERSTELLEN EN DOOR WIJZIGING VAN HET REGELNUMMER VAN EEN BE-
BESTAANDE REGEL IS DEZE REGEL TE KOPIEREN.

0250:

0260:

0270:

0280:

DE BASIC-EDITOR WORDT AANGEROEPEN M.B.V. HET KOMMANDO

0290:

0300:

#EDIT (REGELNUMMER) (ZIE BESCHRIJVING VAN HET TOEVOEGEN VAN
NIEUWE BASIC-KOMMANDO'S)

0310:

0320:

0330:

NA BOVENSTAANDE TE HEBBEN INGETOETST VERSCHIJNT OP HET BEELD-
SCHERM DE OPGEGEVEN REGEL, WAARBIJ DE CURSOR ACHTER DEZE REGEL
BLIJFT STAAN.
HET IS NU MOGELIJK OM MET DE CURSOR DOOR DE REGEL HEEN TE LOPEN
ALS VOLGT:

0340:

0350:

0360:

0370:

0380:

0390:

0400:

- DOOR HET INTOETSEN VAN EEN + OF CNTRL-H KAN DE CURSOR NAAR
LINKS GEPLAATST WORDEN EN VERVOLGENS EEN NIEUW TEKEN WORDEN
INGETOETST.

0410:

0420:

0430:

- DOOR HET INTOETSEN VAN EEN → OF EEN CNTRL-L KAN DE CURSOR
NAAR RECHTS GEPLAATST WORDEN EN VERVOLGENS EEN NIEUW TEKEN
WORDEN INGETOETST.

0440:

0450:

0460:

- DOOR HET INTOETSEN VAN EEN CNTRL-I WORDT ER EEN TUSSENRUIMTE
GEMAAKT IN DE REGEL WAARDOOR EEN TEKEN KAN WORDEN TUSSENGEVOEGD.

0470:

0480:

0490:

- DOOR HET INTOETSEN VAN EEN CNTRL-D WORDT OP DE PLAATS VAN DE
CURSOR HET TEKEN VERWIJDERD WAARBIJ GELIJKTIJDIG DE REGEL
WEER WORDT AANGESCHOVEN.

0500:

0510:

0520:

0530:

0540:

- DOOR HET INTOETSEN VAN EEN CNTRL-A KAN DE CURSOR VANAF EEN
WILLEKEURIGE POSITIE IN DE REGEL WEER ACHTER HET LAATSTE TEKEN
GEPLAATST WORDEN WAARNA DOOR HET INTOETSEN VAN DE RETURN DE REGEL
DOOR BASIC ALS INPUT WORDT GEZIEN EN VERWERKT.

0550:

0560:

0570:

0580:

0590:

OM EEN EN ANDER MOGELIJK TE MAKEN DIENEN DE VOLGENDE AANPASSINGEN
EN TOEVOEGINGEN PLAATS TE VINDEN:

0600:

0610:

0620:

0630:

0640:

- DE TOETS-HERKENNING VAN BASIC MOET WORDEN UITGEBREID.
- ER DIEN T EEN VLAG TE WORDEN GEZET OM NA HET INVOEREN VAN HET

EDITOR 6502 ASSEMBLER V3.1 PAGE 02

```

0650:      EDIT-KOMMANDO DE CURSOR ACHTER DE GE-DISPLAYDE REGEL TE LATEN
0660:      STAAN.
0670:      - VOORDEEMDE VLAG DIENST WEER GERESET TE WORDEN NADAT HET EDIT-
0680:      KOMMANDO IS UITGEVOERD.
0690:      - EEN ESCAPE MOGELIJKHEID OM EEN INGEVOERD EDIT-KOMMANDO ONGEDAAN TE
0700:      MAKEN.
0710:      - EEN ALARM-ROUTINE WELKE EEN BELL GENEREERT INDIEN ER MEER TEKENS
0720:      GE-DELETE WORDEN DAN ER ACHTER DE CURSOR STAAN.
0730:      - EEN ROUTINE WAARBIJ ZOWEL IN HET INPUTBUFFER ALS OOK OP HET BEELD-
0740:      SCHERM DE REGEL IN ELKAAR SCHUIFT BIJ EEN DELETE CHARACTER.
0750:      (OP HET DOOR MIJ GEBRUIKTE SCHERM GEBEURT DIT DOOR HET STUREN VAN
0760:      EEN ESCAPE-W)
0770:      - EEN ROUTINE WAARBIJ ZOWEL IN HET INPUTBUFFER ALS OOK OP HET BEELD-
0780:      SCHERM DE REGEL UIITEEN GESCHOVEN WORDT BIJ EEN INSERT CHARACTER.
0790:      (OP HET DOOR MIJ GEBRUIKTE SCHERM GEBEURT DIT DOOR HET STUREN VAN
0800:      EEN ESCAPE-Q)
0810:
0820:      NB. DE BASIC-EDITOR, ZOALS AL EERDER IS BESCHREVEN, WORDT GESTART
0830:      DOOR HET INTOETSEN VAN HET KOMMANDO #EDIT (REGENR), WAARNA
0840:      VERVOLGENS WORDT GESPRONGEN NAAR ADRES $9376.
0850:
0860:      DE NU VOLGENDE ROUTINES ZIJN IN VOLGORDE VAN ADRES OPGENOMEN.
0870:
0880: 9000      BASKEY ORG    $9000
0890:
0900:      *****
0910:
0920:      UITBREIDING OP DE BASIC GETKEY ROUTINE OP
0930:      ADRES $C38F.
0940:      HIER DIENST DE JSR $1ESA (KIM GETKEY-ROUTINE)
0950:      TE WORDEN GEWIJZIGD IN JSR $9000.
0960:
0970:      *****
0980:
0990: 9000      OUTCH *      $1E90
1000: 9000      GETKEY *     $1ESA
1010: 9000      DELETE *    $930F
1020: 9000      INSERT *    $933A
1030: 9000      COPYL *     $935C
1040: 9000      BUFIN *     $93D6
1050: 9000      WARM *      $C27E
1060: 9000      BASRES *    $946F
1070: 9000      ALARM *     $93EE
1080: 9000      TEMP *      $17E2
1090: 9000      CRLF *      $1E2F
1100: 9000      LFCR *      $C8FE
1110: 9000      BUFFER *    $001E
1120: 9000      CPOINT *    $00E8
1130: 9000      EPOINT *    $00ED
1140: 9000      PARAMS *    $0200
1150: 9000      EVLAG *     $00EE
1160: 9000      AVLAG *     $00EA
1170:
1180: 9000 20 5A 1E      JSR GETKEY ; SPRING NAAR KIM GETKEY-ROUTINE
1190: 9003 C9 08        CMPIM $08 ; IS HET EEN BACKSPACE ?
1200: 9005 D0 02        BNE NEXT
1210: 9007 A9 5F        LDAM $5F
1220: 9009 C9 01        NEXT CMPIM $01 ; IS HET EEN CNTRL-A ?
1230: 900B D0 03        BNE NEXT1
1240: 900D 4C 5C 93     JMP COPYL ; SPRING NAAR COPY LINE-ROUTINE
1250: 9010 C9 04        NEXT1 CMPIM $04 ; IS HET EEN CNTRL-D ?
1260: 9012 D0 03        BNE NEXT2
1270: 9014 4C 0F 93     JMP DELETE ; SPRING NAAR DE CHAR. DELETE ROUTINE
1280: 9017 C9 09        NEXT2 CMPIM $09 ; IS HET EEN CNTRL-I ?

```

BASKEY 6502 ASSEMBLER V3.1 PAGE 03

```

1290: 9019 D0 03      BNE NEXT3
1300: 901B 4C 3A 93    JMP INSERT ; SPRING NAAR DE CHAR. INSERT-ROUTINE
1310: 901E C9 7F      NEXT3 CMPIM $7F ; IS HET EEN RUB-OUT ?
1320: 9020 D0 03      BNE BACK
1330: 9022 20 6F 94     JSR BASRES ; SPRING NAAR DE BASIC RESET-ROUTINE
1340: 9025 60          BACK RTS
1350:
1360: 930F      BASIC2 ORG $930F
1370:
1380:
1390:
1400:
1410:
1420:
1430:
1440:
1450:
1460:
1470: 930F A9 1B      LDAIM $1B ; ESCAPE-W NAAR DE TERMINAL
1480: 9311 20 A0 1E     JSR DUTCH ; STUREN OM EEN 'DELETE CHAR-
1490: 9314 A9 57      LDAIM $57 ; RACTER' OP HET SCHERM TE
1500: 9316 20 A0 1E     JSR DUTCH ; BEWERKSTELLIGEN.
1510: 9319 86 E8      STXZ CPOINT ; CURSOR-POSITIE POINTER EVEN
1520: 931B E6 E8      INCZ CPOINT ; WEGZETTEN EN MET 1 VERHOGEN
1530: 931D A4 E8      LDYZ CPOINT ; POINTER OM DE CHARACTERS
1540: 931F C4 ED      CPVZ EPOINT ; VERGELIJKEN MET EINDE REGEL-POINTER
1550: 9321 10 13      BPL BELL ; NAAR BASIC ALARM-ROUTINE SPRINGEN
1560: 9323 89 1E 00    NCHAR LDAAY BUFFER ; NA DE CURSOR EEN POSITIE
1570: 9326 95 1E      STAZX BUFFER ; NAAR LINKS TE SCHUIVEN
1580: 9328 C8          INY
1590: 9329 E8          INX
1600: 932A C4 ED      CPVZ EPOINT ; POINTER VERGELIJKEN MET POSITIE
1610: 932C D0 F5      BNE NCHAR ; VAN LAATSTE CHARACTER IN DE REGEL
1620: 932E A6 E8      LDZX CPOINT ; X VULLEN MET DE OUDE CURSOR-POSITIE
1630: 9330 CA          DEX ; POINTER EN MET 1 VERLAGEN
1640: 9331 C6 ED      DECZ EPOINT ; $00ED GOED ZETTEN VOOR DE CNTRL-A
1650: 9333 4C D6 93    JMP BUFIN
1660: 9336 20 EE 93    BELL JSR ALARM ; BASIC ALARM-ROUTINE
1670:
1680: 933A      BASIC3 ORG $933A
1690:
1700:
1710:
1720:
1730:
1740:
1750:
1760:
1770:
1780:
1790: 933A A9 1B      LDAIM $1B ; EEN ESCAPE-Q NAAR DE
1800: 933C 20 A0 1E     JSR DUTCH ; TERMINAL STUREN OM EEN
1810: 933F A9 51      LDAIM $51 ; 'CHARACTER INSERT' TE
1820: 9341 20 A0 1E     JSR DUTCH ; BEWERKSTELLIGEN
1830: 9344 86 E8      STXZ CPOINT ; CURSOR-POSITIE POINTER SAVEN
1840: 9346 A4 ED      LDYZ EPOINT ; POINTER LAATSTE CHARACTER IN Y-REG
1850: 9348 E6 ED      INCZ EPOINT ; POINTER EEN OPHOGEN
1860: 934A A6 ED      LDZX EPOINT ; X-REG LADEN MET VERHOOGDE POINTER
1870: 934C 89 1E 00    NCHAR1 LDAAY BUFFER ; EN VANAF DE CURSOR-POSITIE
1880: 934F 95 1E      STAZX BUFFER ; TOT EN MET LAATSTE CHARACTER
1890: 9351 88          DEY ; EEN POSITIE NAAR
1900: 9352 CA          DEX ; RECHTS SCHUIVEN
1910: 9353 C4 E8      CPVZ CPOINT ; VERGELIJK MET CURSOR-POSITIE POINTER
1920: 9355 10 F5      BPL NCHAR1

```

```

BASIC3      6502 ASSEMBLER V3.1      PAGE 04

1930: 9357 A6 E8      LDZX CPOINT ; CURSOR-POSITIE POINTER NAAR X-REG
1940: 9359 4C D6 93    JMP  BUFIN ; TERUG NAAR INPUT-ROUTINE
1950:
1960: 935C      BASIC4 ORG $935C
1970:
1980: *****
1990:
2000:      DOOR HET INTOETSEN VAN EEN CONTROL-A IN DE BASIC
2010:      EDITOR IS HET MOGELIJK OM VANAF DE CURSOR-POSITIE
2020:      TOT AAN HET EINDE VAN DE GEDISPLAY-DE REGEL TE
2030:      KOPIEREN EN DERHALVE DE CURSOR ACHTER HET LAATSTE
2040:      CHARACTER TE PLAATSEN.
2050: *****
2060:
2070:
2080: 935C A4 EA      LDYZ $EA ; KONTROLEREN OF BASIC IN
2090: 935E C0 BB      CPYIM $BB ; EDITOR-MODE IS GEZET
2100: 9360 F0 03      BEQ  EDIT
2110: 9362 4C 7E C2    JMP  WARM
2120: 9365 E4 ED      EDIT CPXZ EPOINT ; KONTROLEREN OF CURSOR AL
2130: 9367 F0 0A      BEQ  BUF ; ACHTER LAATSTE CHARACTER STAAT
2140: 9369 B5 1E      EDIT1 LDAXZ BUFFER ; VANAF CURSOR-POSITIE TOT
2150: 936B 20 A0 1E    JSR  OUTCH ; EN MET LAATSTE CHARACTER KOPIEREN
2160: 936E E8          INX ; EN DISPLAY-EN
2170: 936F E4 ED      CPXZ EPOINT ; LAATSTE CHARACTER GEHAD ?
2180: 9371 D0 F6      BNE  EDIT1
2190: 9373 4C D6 93    BUF  JMP  BUFIN ; TERUG NAAR INPUT-ROUTINE
2200:
2210: 9376      BASIC5 ORG $9376
2220:
2230: *****
2240:
2250:      DEZE ROUTINE MAAKT DE OMGEVING VAN BASIC GEREED
2260:      VOOR HET GEBRUIK VAN DE BASIC-EDITOR.
2270: *****
2280:
2290:
2300: 9376 A9 99      LIST LDAM $99 ; BASIC LIST-TOKEN IN
2310: 9378 85 1E      STAZ BUFFER ; INPUT-BUFFER PLAATSEN
2320: 937A A2 00      LDXIM $00 ; PARAMETER OPHALEN
2330: 937C BD 00 02    PARAM LDAAZ PARAMS ; VAN ADRES $0200 EN IN
2340: 937F 95 1F      STAZ $1F ; INPUT-BUFFER PLAATSEN
2350: 9381 E8          INX
2360: 9382 C9 00      CMPIM $00
2370: 9384 D0 F6      BNE  PARAM
2380: 9386 A9 FF      LDAM $FF ; ZET DE PRINT-POSITIE
2390: 9388 85 ED      STAZ EPOINT ; POINTER OP -1
2400: 938A A9 D4      LDAM $D4 ; WIJZIG HET TERUGSPRING ADRES
2410: 938C 85 04      STAZ $04 ; OP $0004 EN $0005 VAN $C954
2420: 938E A9 93      LDAM $93 ; IN $93D4 (INPUT-ROUTINE)
2430: 9390 85 05      STAZ $05
2440: 9392 A9 BB      LDAM $BB ; ZET DE EDIT-MODE VLAG AAN
2450: 9394 85 EA      STAZ $EA
2460: 9396 68          PLA ; DE STACK-POINTER HERSTELLEN
2470: 9397 68          PLA
2480: 9398 68          PLA
2490: 9399 68          PLA
2500: 939A 68          PLA
2510: 939B 68          PLA
2520: 939C 68          PLA
2530: 939D 4C 92 C3    JMP  $C392 ; TERUG NAAR BASIC INPUT-ROUTINE
2540:
2550: 93A6      BASIC7 ORG $93A6
2560:

```

BASIC7 6502 ASSEMBLER V3.1 PAGE 05

```

2570: *****
2580:
2590: HIER WORDT GEKEKEN OF DE EDIT-MODE VLAG GEZET
2600: IS TER VOORBEREIDING VAN DE TE WIJZIGEN REGEL.
2610: *****
2620:
2630:
2640: 93A6 8C E2 17 STY TEMP ; ZET Y-REG EVEN WEG
2650: 93A9 A4 EA LDYZ $EA ; KONTROLEER OF DE
2660: 93AB C0 BB CPYIM $BB ; EDIT-MODE VLAG GE-
2670: 93AD F0 04 BEQ EDIT2 ; ZET IS
2680: 93AF AC E2 17 LDY TEMP ; EN HAAL Y-REG WEER OP
2690: 93B2 60 RTS ; EN TERUG NAAR BASIC
2700: 93B3 A4 1D EDIT2 LDYZ $1D ; TIJDELIJK $001D WEGZETTEN
2710: 93B5 84 EB STYZ $EB ; IVM MET OPBOUWEN VAN DE REGEL
2720: 93B7 A4 19 LDYZ $19 ; HAAL DE CURSOR-POSITIE POINTER OP EN
2730: 93B9 C9 0A CMPIM $0A ; VERGELIJK DE ACCU-INHOUD MET EEN LF
2740: 93BB F0 0C BEQ NEXTCH ; ZO JA, VOLGENDE CHARACTER OPHALEN
2750: 93BD C9 0D CMPIM $0D ; VERGELIJK DE ACCU-INHOUD MET EEN CR
2760: 93BF F0 08 BEQ NEXTCH ; ZO JA, VOLGENDE CHARACTER OPHALEN
2770: 93C1 99 1C 00 STAAY $001C ; PLAATS CHARACTER IN INPUT-BUFFER
2780: 93C4 20 A0 1E JSR DUTCH ; EN ZET HET OP HET BEELDSCHERM
2790: 93C7 E6 ED INCZ EPOINT ; VERHOOG DE CURSOR-POSITIE POINTER
2800: 93C9 A4 EB NEXTCH LDYZ $EB ; HAAL HET TIJDELIJK BEWAARDE BYTE
2810: 93CB 84 1D STYZ $1D ; WEER OP EN ZET HET OP $001E-1
2820: 93CD AC E2 17 LDY TEMP ; HAAL INHOUD Y-REG WEER OP
2830: 93D0 60 RTS ; EN GA WEER TERUG NAAR BASIC
2840:
2850: 93D4 BASIC8 ORG $93D4
2860:
2870: *****
2880:
2890: MET BEHULP VAN DEZE ROUTINE WORDT IN DE BASIC-
2900: EDIT-MODE EEN KOPPELING GELEGD TUSSEN DE BASIC
2910: GETKEY-ROUTINE EN DE EDIT FUNKTIES. VERVOLGENS
2920: WORDT ER NA GE-EDIT TE HEBBEN EEN CRLF VERZORGD
2930: EN DE EDIT-MODE VLAG AFGEZET.
2940: *****
2950:
2960:
2970: 93D4 A6 ED LDXZ EPOINT ; VUL X-REG MET CURSOR-POSITIE POINTER
2980: 93D6 20 61 C3 JSR $C361 ; EN SPRING NAAR BASIC GETKEY-ROUTINE
2990: 93D9 08 PHP ; SAVEN VAN DE ACCU EN DE
3000: 93DA 48 PHA ; REGISTERS I.V.M.
3010: 93DB 8A TXA ; HET VERWERKEN
3020: 93DC 48 PHA ; VAN DE CRLF
3030: 93DD 98 TYA
3040: 93DE 48 PHA
3050: 93DF 20 2F 1E JSR $1E2F ; KIM CRLF ROUTINE
3060: 93E2 20 6F 94 JSR BASRES ; RESET BASIC POINTER-ROUTINE
3070: 93E5 68 PLA ; DE ACCU EN DE
3080: 93E6 A8 TAY ; REGISTERS
3090: 93E7 68 PLA ; WEER VAN STACK
3100: 93E8 AA TAX ; HALEN
3110: 93E9 68 PLA
3120: 93EA 28 PLP
3130: 93EB 4C 8A C2 JMP $C28A ; BASIC INPUT-ROUTINE
3140:
3150: 93EE BASIC9 ORG $93EE
3160:
3170: *****
3180:
3190: ROUTINE OM BIJ EEN FOUTIEVE DELETE EEN ALARM
3200: NAAR DE TERMINAL TE STUREN.

```

BASIC9 6502 ASSEMBLER V3.1 PAGE 06

```

3210:
3220: *****
3230:
3240: 93EE A9 07      LDAIM $07      ; LAAD ACCU MET EEN BELL-
3250: 93F0 20 A0 1E    JSR  OUTCH    ; CHARACTER EN "PRINT" HET
3260: 93F3 4C D6 93    JMP  BUFIN    ; TERUG NAAR BASIC INPUT-ROUTINE
3270:
3280: 946F             RESET ORG  $946F
3290:
3300: *****
3310:
3320: ROUTINE OM NA DE EDIT-MODE DE EDIT-MODE-
3330: VLAG AF TE ZETTEN EN HET BASIC TERUGSPRING-
3340: ADRES OP $0004 EN $0005 TE HERSTELLEN.
3350:
3360: *****
3370:
3380: 946F A9 54      LDAIM $54      ; HERSTEL OP DE ADRESSEN
3390: 9471 85 04      STAZ  $04      ; $0004 EN $0005 HET
3400: 9473 A9 C9      LDAIM $C9      ; TERUGSPRING-ADRES
3410: 9475 85 05      STAZ  $05      ; $C954 TBV BASIC
3420: 9477 A9 CC      LDAIM $CC      ; ZET DE EDIT-MODE
3430: 9479 85 EE      STAZ  EVLAG     ; VLAG EN DE
3440: 947B 85 EA      STAZ  AVLAG     ; AUTO-NUMBERVLAG AF EN GA
3450: 947D 60         RTS           ; TERUG NAAR BASIC
3460:

```

MICRO-ADE FOR THE 6502

Micro-ADE is a Text Editor, Assembler, Disassembler, originally a 4K version for the KIM, written by Peter Jennings. Our club became copyrights of that version. One of our members, S. Wol-drinhg made it a 8K version for the KIM. Afterwards Adri Hankel made it a 8K JUNIOR version. There is only a manual for the 4K KIM version. But no need for re-writing a manual especially for the the 8K JUNIOR version. Micro ADE has now commands like: Append, Blockmove, Clear Buffer, Delete Lines, Display address/number last line, Fix Line, Get Source Files, Append Source Files, Print Symbol/XREF Tables, Reset/Set Form Feed Flag, Reset/Set Page Mode, Insert Lines, Chose Line(s), List, Move Line(s), Number (renumber), Load ASCII-format files, Set Lines/Screen and Lines/Page, List Memory in use, Duplicate Files, Save Source Files, Print/Change string, Reset/Set Page-per-File/Eject Flag Search Line, List without Line Numbers, Save Chosen Lines, Assemble/Execute, Save ASCII-format Files, Disassemble.

JUNIOR/KIM Hypertape cassette with Manual and command summary: Fl. 47,50. Sourcelisting 8K version with comments: Fl. 65,00. Both in English language. To be payed on postal account 841433 of W.L. van Pelt at Kriboen a.d. IJssel, The Netherlands. Eurocheque allowed. Foreign countries Fl. 6,75 extra transfers.

BASIC MET DE COMMODORE 64

Dit boekje van ca. 200 bladzijden is een vertaling van Learn computer programming with the Commodore 64 door L.A.P. van den Wyngaert. Het stelt u in staat bruikbare programma's te schrijven in Basic. Er is geen voorkennis nodig. Tevens zijn een aantal praktische toepassingen verwerkt, w.o.:

- kleureffecten, animatie en grafieken met hoge resolutie
- geluidseffecten en morse-training

Uitg: Maarten Kluwer, Apeldoorn/Antwerpen
ISBN: 90 6215 092 6

HEEFT U EEN COMMODORE 64?
GEEF HET OP AAN DE SEKRETARIS!

VDU-software Paperware 4.
=====

Van J. Kemfers PAORJK
Herinckhave 30
7608 BH Almelo
Tel.: 05490 - 64110

ontvingen we de volgende brief:

Om met de VDU kaart van Elek-tuur te kunnen werken, moet men de software van Paperware 4 in TM en PM zetten. Dan moet alles werken volgens Elek-tuur. Dit klopt niet, omdat in PM 10 bytes nog veranderd moeten worden:

was	wordt
1000 D8	58
1004 8D 82 1A	4C DA 16
12DC AE 61 1A	4C F8 16
1334 BE 60 1A	4C 8A 16

Verder heb ik de VDU-kaart verzet van D000 naar B000. En de reg. van DB00 naar B800, dit i.v.m. SYM-Basic. De verandering op de kaart bestaat uit:

de 2 draden die op A14 zitten op A14 zetten;
de 2 draden die op A13 zitten op A13 zetten.

0E21 8E 00 D8	D8 wordt B8
0E24 8D 01 D8	Het is dus
0E2B 8E 00 D8	ook mogelijk
0E2E 8D 01 D8	de VDU op
0E37 8E 00 D8	nog andere
0E3A 8D 01 D8	plaats te
0E41 8E 00 D8	zetten door
0E44 8D 01 D8	deze 14 by-
0FDC 8E 00 D8	tes te ver-
0FE2 8D 01 D8	anderen.

Uiteraard dienen dan ook de adressen op de kaart te worden aangepast.

0E57 29 D7	D7 wordt B7
0E69 29 D7	
0E95 29 D7	
15AE A2 D0	D0 wordt B0

OBSESS

080284 PAGE 01

```

0010: ***** OBSESSIE *****
0020: *****
0030: *****
0040: *****
0050: *****
0060: *****
0070: *****
0080: *****
0090: *****
0100: *****
0110: *****
0120: *****
0130: *****
0140: *****
0150: *****
0160: *****
0170: *****
0180: *****
0190: *****
0200: *****
0210: *****
0220: *****
0230: *****
0240: *****
0250: *****
0260: *****
0270: *****
0280: *****
0290: *****
0300: *****
0310: *****
0320: *****
0330: *****
0340: *****
0350: *****
0360: *****
0370: *****
0380: *****
0390: *****
0400: *****
0410: *****
0420: *****
0430: *****
0440: *****
0450: *****
0460: *****
0470: *****
0480: *****
0490: *****
0500: *****
0510: *****
0520: *****
0530: *****
0540: *****
0550: *****
0560: *****
0570: *****
0580: *****
0590: *****
0600: *****
0610: *****
0620: *****
0630: *****
0640: *****

```

Benodigd : Standaard JUNIOR met 1K geheugen

EEN REAKTIESPELLETJE

Door : Marcel van Doro
De Ridderstraat 20
2313 JR Leiden

Soelregels: Na het programma gestart te hebben (startadres \$0200) zal in het display 00 00 00 verschijnen. Het beginnen geschiedt door op de + toets te drukken. Op de middelste twee displays zal een 'willekeurig' getal verschijnen (0 .. F). Dit getal zal daar maar een bepaalde tijd blijven staan. Wordt in die tijd op de toets met hetzelfde opschrift gedrukt, dan zal, evenals de beurtenteller, ook de scoreteller ogehoogd worden. Bij de volgende beurt zal de tijd waarin gereageerd moet worden, korter zijn. De linker twee displays geven het aantal beurten weer, de rechter twee de score. Na 25 beurten stopt het programma, de uitlezing blijft staan totdat er op de plustoeets gedrukt wordt.

SCANDS: De inhoud van de zero-page adressen F9, FA, FB worden op het display zichtbaar gemaakt. Op het display verschijnt dan van links naar rechts de inhoud van resp. \$00FB, \$00FA, \$00F9. (KIM SCANDS = \$1F1F)

GETKEY: Deze routine tast het toetsenbord eenmaal af. Als er een toets is ingedrukt, dan staat de toetswaarde na afloop in de accu. Is er geen toets ingedrukt, dan staat er 15 (hex) in de accu. (KIM GETKEY = \$1E5A)

Toetswaarden : 0 .. F voor de toetsen 0 .. F
10 voor de 'AD'-toets
11 voor de 'DA'-toets
12 voor de '+'-toets
13 voor de 'GD'-toets
14 voor de 'PC'-toets

```

0440: 0200 OBSESS ORG $0200
0450:
0460: 00 00 WAIT * $0000 : TELLER T.B.V. DE WACHTLUS
0470: 01 00 LIMCNT * $0001 : LIMET VAN DE WACHTLUS-TELLER
0480: 02 00 KEYVAL * $0002 : TOETSWAARDE LAATST INGEDRUKTE TOETS
0490: : VOOR EINDIGEN WACHTLUS
0500:
0510: F9 00 CNTSCO * $00F9 : SCORETELLER
0520: FA 00 FIELD * $00FA : 'SPEELVELD'
0530: FB 00 TURNS * $00FB : BEURTENTELLER
0540:
0550: 84 1A TIMER * $1A84 :
0560:
0570: 8E 1D SCANDS * $1D8E :
0580: F9 1D GETKEY * $1DF9 :
0590:
0600: 0200 A2 FF LDXIM $FF : WACHTLUS OP DE GROOTSTE TIJD
0610: 0202 86 01 STX LIMCNT :
0620: 0204 E8 INX :
0630: 0205 86 F9 STX CNTSCO : SCORETELLER OP NUL
0640: 0207 86 FA STX FIELD : DISPLAY WISSEN

```

OBSESS

080284 PAGE 02

```

0650: 0209 86 FB      STX  TURNS      : BEURTENTELLER OP NUL
0660: 020B 20 8E 1D  WAIT1 JSR  SCANDS   : ZIE INLEIDING
0670: 020E 20 F9 1D      JSR  GETKEY
0680: 0211 C9 12      CMPIM $12      : + TOETS INGEDRUKT ?
0690: 0213 D0 F6      BNE  WAIT1      : ZONEE, TERUG
0700: 0215 AD 84 1A  START LDA  TIMER    : IN ACCU 'N WAARDE VAN TIMER (KIM: $174C)
0710: 0218 29 0F      ANDIM $0F      : ALLEEN RECHTER NIBBLE
0720: 021A 85 FA      STA  FIELD    : ZICHTBAAR MAKEN
0730: 021C 20 40 02      JSR  INKEY
0740: 021F A5 02      LDA  KEYVAL    : HAAL TOETS OP
0750: 0221 C5 FA      CMP  FIELD    : BELIJK AAN GEVR. TOETS?
0760: 0223 D0 03      BNE  BEURT     : ZONEE, SCORE OVERSLAAN
0770: 0225 20 5D 02      JSR  SCORE
0780: 0228 F8      BEURT SED
0790: 0229 18      CLC
0800: 022A A5 FB      LDA  TURNS    : HAAL AANTAL BEURTEN OP
0810: 022C 69 01      ADCIM $01     : MET 1 VERHOGEN
0820: 022E 85 FB      STA  TURNS    : TERUGZETTEN
0830: 0230 D8      CLD
0840: 0231 C9 25      CMPIM $25     : 25 BEURTEN GEHAD?
0850: 0233 D0 E0      BNE  START     : ZONEE, TERUG NAAR START
0860: 0235 20 8E 1D  WAIT2 JSR  SCANDS
0870: 0238 20 F9 1D      JSR  GETKEY
0880: 023B C9 12      CMPIM $12     : + TOETS INGEDRUKT ?
0890: 023D D0 F6      BNE  WAIT2
0900: 023F 00      BRK      : STOP
0910:
0920:      : subroutine INKEY wacht een bepaalde tijd en haalt
0930:      : eventueel een ingedrukte toets op; komt in KEYVAL
0940:
0950: 0240 A9 00      INKEY LDAIM $00    : TELLER WORDT GERESET
0960: 0242 85 00      STA  WAIT
0970: 0244 A9 15      LDAIM $15     : WAARDE VAN 'GEEN TOETS'
0980: 0246 85 02      STA  KEYVAL    : IN TOETSBUFFER ZETTEN
0990: 0248 20 8E 1D  WAIT3 JSR  SCANDS
1000: 024B 20 F9 1D      JSR  GETKEY
1010: 024E C9 15      CMPIM $15     : GEEN TOETS INGEDRUKT?
1020: 0250 F0 02      BEQ  TELLER    : ZOJA, DAT DAN NIET WEGZETTEN
1030: 0252 85 02      STA  KEYVAL    : WEGZETTEN IN TOETSBUFFER
1040: 0254 E6 00      TELLER INC  WAIT  : TELLER VERHOGEN
1050: 0256 A5 00      LDA  WAIT
1060: 0258 C5 01      CMP  LIMCNT    : TELLER BELIJK AAN LIMIET?
1070: 025A D0 EC      BNE  WAIT3     : TERUG
1080: 025C 60      RTS      : TERUG NAAR HOOFDPROGRAMMA
1090:
1100:      : subroutine SCORE verhoogt de scoreteller en ver-
1110:      : kleint de wachtlus
1120:
1130: 025D F8      SCORE SED
1140: 025E 18      CLC
1150: 025F A5 F9      LDA  CNTSCO    : OPHALEN SCORETELLER
1160: 0261 69 01      ADCIM $01     : 1 ERBIJ OPTELLEN
1170: 0263 85 F9      STA  CNTSCO    : TERUGZETTEN
1180: 0265 D8      CLD
1190: 0266 A5 01      LDA  LIMCNT    : OPHALEN LIMIET
1200: 0268 E9 03      SBCIM $03     : LIMIET VERLAGEN
1210: 026A 85 01      STA  LIMCNT    : TERUGZETTEN
1220: 026C 60      RTS

```

6502 KENNER

Herman Burgers
3100 Canter Lane
AUSTIN, Texas 78759
USA.

Processor met... PUSH X = PHX ???

Zojuist heb ik mijn 6502 weggegooid. Want nu zit in dat voetje een betere processor. Hij is volledig compatible met de 6502, en hij draagt als code 65C02 op zijn 40 pins DIP pack. Deze is van Rockwell, en heeft alle nieuwe codes. De chips met dezelfde 65C02 code van GTE en NCR missen de RMBx/SMBx en de BBRx/BBSx instructies.

Er zijn een drietal redenen om een 65C02 te gebruiken.

- .1 Een serie nieuwe en zeer aantrekkelijke instructies erbij, totaal heb ik er nu 210 geteld.
- .2 "Bugs" van de 6502 zijn verdwenen.
- .3 Lager energie gebruik, wat meer vermogen voor extra Eproms beschikbaar bij mij.

De nieuwe instructies zijn o.a :

MNEMONICS	BYTES	UITLEG
PHX en PLX	1	push register X en pull X v stack.
PHY enPLY	1	push register Y en pull Y v stack.
BRA	2	branch always.
INC	1	increment register A.
DEC	1	decrement register A.
STZ	2/3	store zero (clear), zero pg/abs, zonder 0 in een register.
BBRx en BBSx	3	branch on bit x reset (R) en branch on bit x set (S), zero pg.
RMBx en SMBx	2	reset (R) bit en set (S) bit zpg.
TSB en TRB	2/3	test/reset (R) en test/set (S) bit, in zero page en abs.

De RMBx, SMBx, BBRx en BBSx instructies komen met 8 stuks tegelijk, voor ieder bit is er een aparte opcode. De BBRx en BBSx zijn een soort BIT test voor ieder bit en een BRA, in een instructie. De opcodes worden gevolgd door een zero page adres, en de BByx ook nog door een offset byte voor de branch. Ze zijn in de LSB 7 en F kolommen te vinden, die waren nog helemaal leeg.

Verder is er een zeer bruikbare aanvulling in de verschillende adressering modes gekomen, als eerste de niet-geïndexeerde indirecte (zero page) mode.

De ORA t/m SBC instructies zijn nu dus ook als ORA (zp) t/m SBC(zp) beschikbaar. Zie de LSB 1 kolom. De JMP is er nu weer als enige met een absolute indirect X indexed, JMP (abs),x ! Dat is wel de instructie geworden om een jumptable met max. 128 adressen te bewerken.

En BIT heeft er een stel weggelaten adressering modes bij, nu bestaat ook de BIT #nn bv.

Wat de weg gewerkte "bugs" betreft, is er ook nog wat te vertellen.

De bekendste is het indirecte adres op een page boundary (lage adres byte op een page en hoge adres byte op de volgende page. Dat ging fout, en bij de 65C02 gaat dat goed. Het kost wel een extra machine cycle.

De niet bestaande opcodes doen nu geen rare dingen, alles doet nu een NOP als het onbekend is. (De \$FF om een EPROM nog een keer plaatselijk te kunnen veranderen is niet meer bruikbaar die opcode is ingenomen door een BBS on bit 7.)

Ook verbeterd is de decimal mode. De D flag wordt bij reset op 0 (binary) gezet, in de stand D=1, decimal mode werken de Z,V en N flags als bij binary mode. Dit vereist wel een extra machine cycle in decimal mode.

De meest bruikbare vernieuwingen zij naar mijn idee de BRA, de non-indexed indirect instukties, de TSB/TRB, de STZ en de indexed indirecte JMP.

Voorbeelden:

```
80 42      BRA MYLABEL
9E 34 12    STZ MYTABLE,X
BF 02 42    BBS3 ZP2,MYLABEL

04 AA      LDA #MASK
           TSB ZPAA                ;Zet bits volgens mask, set Z
           BNE ONE_BIT_SET         ; flag volgens A or M->M
           BEQ NO_BITS_SET

           JSR GET_A_CMD           ;Command in .A
           ASL                     ;Multiply by 2
           TAX                     ;Use it as index
7C 08 EE    JMP (CMDTABL),X       ;Jump using adress table
```

6502 KENNER

Het geheel heb ik in tabelvorm gegoten. Een opcode kun je samenstellen door de Most Significant (MSD) en de Least Significant Digit (LSD) te combineren, bv. PHX = \$DA.

		Least Significant Digit -->															
MSD		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
V																	
0	BRK	ORA				TSB	ORA	ASL	RMB	PHP	ORA	ASL		TSB	ORA	ASL	BBR
		(nn,X)				nn	nn	nn	0,nn		#nn	A		nnnn	nnnn	nnnn	0,nn
1	BPL	ORA	ORA			TRB	ORA	ASL	RMB	CLC	ORA	INC		TRB	ORA	ASL	BBR
		(nn),Y	(nn)			nn	nn,X	nn,X	1,nn		nnnn,Y	A		nnnn	nnnn,X	nnnn,X	1,nn
2	JSR	AND				BIT	AND	ROL	RMB	PLP	AND	ROL		BIT	AND	ROL	BBR
		nnnn	(nn,X)			nn	nn	nn	2,nn		#nn	A		nnnn	nnnn	nnnn	2,nn
3	BNI	AND	AND			BIT	AND	ROL	RMB	SEC	AND	DEC		BIT	AND	ROL	BBR
		(nn),Y	(nn)			nn,X	nn,X	nn,X	3,nn		nnnn,Y	A		nnnn,X	nnnn,X	nnnn,X	3,nn
4	RTI	EOR				EOR	LSR	RMB	PHA	EOR	LSR			JMP	EOR	LSR	BBR
		(nn,X)				nn	nn	4,nn		#nn	A			nnnn	nnnn	nnnn	4,nn
5	BVC	EOR	EOR			EOR	LSR	RMB	CLI	EOR	PHY			EOR	LSR	BBR	
		(nn),Y	(nn)			nn,X	nn,X	5,nn		nnnn,Y				nnnn,X	nnnn,X	5,nn	
6	RTS	ADC				STZ	ADC	ROR	RMB	PLA	ADC	ROR		JMP	ADC	ROR	BBR
		(nn,X)				nn	nn	nn	6,nn		nnnn	A		(nnnn)	nnnn	nnnn	6,nn
7	BVS	ADC	ADC			STZ	ADC	ROR	RMB	SEI	ADC	PLY		JMP	ADC	ROR	BBR
		(nn),Y	(nn)			nn,X	nn,X	nn,X	7,nn		nnnn,Y			(nnnn),X	nnnn,X	nnnn,X	7,nn
8	BRA	STA				STY	STA	STX	SMB	DEY	BIT	TXA		STY	STA	STX	BBS
		(nn,X)				nn	nn	nn	0,nn		#nn			nnnn	nnnn	nnnn	0,nn
9	BCC	STA	STA			STY	STA	STX	SMB	TYA	STA	TXS		STZ	STA	STZ	BBS
		(nn),Y	(nn)			nn,X	nn,X	nn,Y	1,nn		nnnn,Y			nnnn	nnnn,X	nnnn,X	1,nn
A	LDY	LDA	LDA			LDY	LDA	LDA	SMB	TAY	LDA	TAX		LDY	LDA	LDA	BBS
		#nn	(nn,X)	#nn		nn	nn	nn	2,nn		#nn			nnnn	nnnn	nnnn	2,nn
B	BCS	LDA	LDA			LDY	LDA	LDA	SMB	CLV	LDA	TSX		LDY	LDA	LDA	BBS
		(nn),Y	(nn)			nn,X	nn,X	nn,Y	3,nn		nnnn,Y			nnnn,X	nnnn,X	nnnn,X	3,nn
C	CPY	CMP				CPY	CMP	DEC	SMB	INY	CMP	DEX		CPY	CMP	DEC	BBS
		#nn	(nn,X)			nn	nn	nn	4,nn		#nn			nnnn	nnnn	nnnn	4,nn
D	BNE	CMP	CMP			CMP	DEC	SMB	CLD	CMP	PHX			CMP	DEC	BBS	
		(nn),Y	(nn)			nn,X	nn,X	5,nn		nnnn,Y				nnnn,X	nnnn,X	5,nn	
E	CPX	SBC				CPX	SBC	INC	SMB	INX	SBC	NOP		CPX	SBC	INC	BBS
		#nn	(nn,X)			nn	nn	nn	6,nn		#nn			nnnn	nnnn	nnnn	6,nn
F	BEQ	SBC	SBC			SBC	INC	SMB	SED	SBC	PLX			SBC	INC	BBS	
		(nn),Y	(nn)			nn,X	nn,X	7,nn		nnnn,Y				nnnn,X	nnnn,X	7,nn	

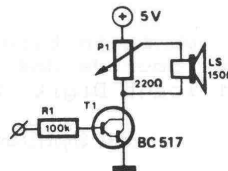
nn= zero page, (nn)= indirect zp, nnnn= absolute, (nnnn)= indirect absolute

Die 6502, hij doet het bij mij zonder mankeren. Bij oudere APPLE II en APPLE II+ modellen, die een clock voor phase 2 van phase 0 maken is deze chip (typen tot 4 MHz leverbaar) mogelijk niet geschikt.

T I P S & T R I C K S

Tips en tricks zijn die kleine handige trukjes die iedere hobbyist toepast op zijn computer. Welke echter nooit ingestuurd worden door hun schijnbare eenvoud. Toch ben ik er van overtuigd dat menige handigheid voor een ander de ideale oplossing kan zijn.

EEN L.S. INTERFACE MET VOLUME-REGLING



Om in basic de decimale toetscode op te vragen kan men het volgende simpele programma intoetsen.

```
10 INPUT A$:PRINT ASC(A$)
```

Na "RUN" kan men een willekeurige toets indrukken en een "RETURN". Op het beeldscherm verschijnt dan de decimale waarde van deze toets.

OM MACHINETAAL PROGRAMMA'S AAN BASIC TOE TE VOEGEN M.B.V. DECIMALE DATA

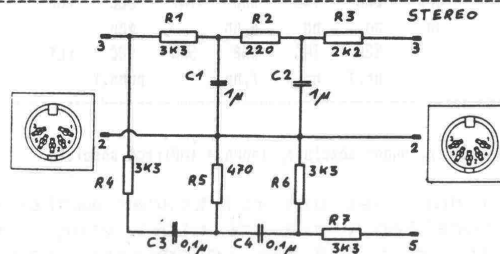
```
10 DATA 234,008,072,138,072,152,072,186
20 DATA 254,005,001,208,003,254,006,001
30 DATA ENZ.
1000 FOR I=8192 TO 8207
1010 READ A:POKE I,A
1020 NEXT I
1030 END
```

(I=DECIMALE HET BEGINADRES TOT HET EINDEADRES)

OM MACHINETAAL PROGRAMMA'S AAN BASIC TOE TE VOEGEN M.B.V. HEXADECIMALE DATA

```
10 AD=8192
20 READ QP$
30 GOSUB 900
40 POKE AD,QP
50 AD=AD+1
60 IF X<>135 THEN GOTO 20
70 END
100 DATA A9,7F,8D,83,1A,AD,82,1A
110 DATA 09,04,8D,80,1A,A9,18,8D
120 DATA ENZ.
900 QL$=LEFT$(QP$,1)
910 QR$=RIGHT$(QP$,1)
920 QL=VAL(QL$)
930 IF QL=0 AND QL$ ("0") THEN QL=ASC(QL$)-55
940 QR=VAL(QR$)
950 IF QR=0 AND QR$ ("0") THEN QR=ASC(QR$)-55
960 QP=QL*16+QR
970 RETURN
```

MET DEZE SIMPELE SCHAKELING BEREIKT MEN EEN IMITATIE VAN HET STEREO GELUID.



>PR

JJA Janssen
Gerardsweg 30
6525 RT NIJMEGEN
tel:080-562082

DE VDU - K A A R T

De VDU-kaart v/d ELEKTUUR heeft de 6845 als video-controller, om de VDU-kaart goed te laten werken moeten de registers v/d 6845 gevuld worden. De 6845 heeft de volgende registers:

register	R/W	aantal bits	data	data	data
address register	W	5	---	---	---
R0 horizontal total	W	8	\$7A	\$7A	\$7A
R1 horizontal displayed	W	8	\$50	\$40	\$5A
R2 hor sync position	W	8	\$58	\$58	\$5E
R3 hor sync width	W	4	\$08	\$05	\$08
R4 vertical total	W	7	\$1C	\$1C	\$1C
R5 vert total adjust	W	5	\$1F	\$1F	\$1F
R6 vertical displayed	W	7	\$19	\$10	\$16
R7 vert sync position	W	7	\$1B	\$16	\$19
R8 interlace mode	W	2	\$00	\$00	\$00
R9 max scan line address	W	5	\$0A	\$0A	\$0A
R10 cursor start	W	7	\$E0	\$E0	\$E0
R11 cursor end	W	5	\$07	\$07	\$07
R12 start address (h)	W	6	\$00	\$00	\$00
R13 start address (l)	W	8	\$00	\$00	\$00
R14 cursor (h)	R+W	6	\$00	\$00	\$00
R15 cursor (l)	R+W	8	\$00	\$00	\$00
R16 light pen (h)	R	6	---	---	---
R17 light pen (l)	R	8	---	---	---

In de tabel duidt R/W aan of het register gelezen (R=READ) dan wel geschreven (W=WRITE) kan worden.
Het aantal bits duidt aan hoe groot het desbetreffende register is bv. 5 geeft aan dat bit 0 t/m bit 4 v/h register gebruikt worden.
De data is de inhoud v/h desbetreffende register om een beeldformaat van 80 bij 25 te verkrijgen, voor de monitor van mij! (eerste data kolom)
De tweede data kolom is voor een beeldformaat van 64*16 en de laatste data kolom voor een beeldformaat van 90*22.
Waar de data niet van toepassing is staat ---.

Bespreking v/d registers.

address register:

In dit register moet men zetten welk registernummer men wilt lezen dan wel schrijven. Men moet dus dit register vullen met data van \$00 t/m \$11 (R0 t/m R17).
Dit register bevindt zich op adres \$D800.

De volgende registers (R0 t/m R17) zijn allemaal toegankelijk via adres \$D801, welk register wordt genomen hangt af van de data op adres \$D800.

Register R0:

In dit register komt de frequentie v/d horizontale synchronisatie en geeft dus aan de totale breedte v/h scherm.
Voor een mooi stilstaand beeld kan deze inhoud van monitor tot monitor verschillend zijn.

Register R1:

Geeft het totaal aantal karakters op een regel zichtbaar op het scherm wordt gezet. Voor 80 kar. op een regel is dit dus \$50.

Register R2:

Geeft de horizontale sync positie aan en bepaalt dus waar het beeld eindigt.

Register R3:

Geeft de breedte v/d horizontale sync puls aan, hiemee is het beeld dus horizontaal gezien een kar. te verschuiven.

DE 6502 KENNER

Registers R4 en R5:

Deze registers bepalen samen de verticale synchronisatie frequentie.
R4 in aantal regels, R5 als fijnafrekening nog evt. een aantal lijnen.

Register R6:

Geeft het aantal regels die op het scherm moeten komen aan.

Register R7:

Geeft aan waar de verticale sync moet komen in aantal regels.

Register R8:

Moet \$00 op staan zodat we niet interlinieeren.

Register R9:

Geeft aan uit hoeveel lijnen (minus een) een regel wordt opgebouwd.
minimaal dus 6 (7-1) en alles meer dan 6 geeft een spatie tussen de regels.

Registers R10 en R11:

Geven aan hoe de cursor van vorm is.

R10 bij welke lijn met de cursor begonnen wordt en R11 bij welke lijn de cursor eindigt.

Bit 5 en 6 van R10 hebben een speciale functie nl:

bit 5	bit 6	functie
0	0	stilstaande cursor
0	1	geen cursor
1	0	cursor snel knipperend
1	1	cursor langzaam knipperend

Register R12 en R13:

Deze twee registers geven aan bij welk adres (v/h interne RAM-gebied v/d VDU-kaart) de controller moet beginnen om kar. op het scherm te zetten.
Staat er bv. \$00 in R12 en R13 dan zal het kar. v/d eerste geheugenplaats links boven op het scherm komen (eerste geheugen plaats is hier dus \$D000 omdat RAM geheugen van \$D800 t/m D7FF loopt).

Door R12, R13 steeds met het max. aantal kar. op een regel te verhogen zal het beeld dus gaan scrollen.

Register R14 en R15:

Deze twee registers geven aan waar de cursor moet komen te staan indien hij natuurlijk zichtbaar is (afh. van bit 6 R10).

Voor de rest werken zij hetzelfde als R12 en R13.

Register R16 en R17:

Deze twee registers geven aan waar de lichttoen op het scherm wordt gehouden, omdat de lichttoen nog niet voorhande is wordt hier verder niet op ingegaan.

Het beste kan men volgende programmatje intypen waarmee men dan de registers kan vullen volgens de tabel en vervolgens, indien het beeld niet geheel goed is, enige registers iets veranderen.
Het programmatje werkt met het hexadecimale toetsenbord.

```
0200    20 6F 1D :vraag een byte
0203    10 FB
0205    8D 00 D8 :zet op address register en selecteer R0-R17
0208    20 6F 1D :vraag weer een byte
020B    10 FB
020D    8D 01 D8 :zet byte in het geselecteerde register
0210    4C 00 02 :herhaal dit alles
```

Als dit programmatje is opgestart moet men eerst het registernummer geven (0 t/m \$11) en daarna de data die op dat register moet komen.
Hiermee kan men eenvoudig bekijken hoe het een en ander werkt.

De software voor de VDU-kaart:

Al hoewel ELEKTUUR ook met soortgelijke software zal komen heb ik toch maar zelf de software voor de VDU-kaart geschreven.

Het programma heb ik in de TM-eprom gezet en loopt van \$0C80 t/m \$0FFF.
Om het goed te laten werken moet in de PM-eprom ook twee dingen veranderd worden, nl:

op \$12DC moet staan JMP INPUT (bij mij dus 4C 4A 0F)

op \$1334 moet staan JMP PRINT (bij mij dus 4C 8D 0E)

Dan heeft geheugenplaats \$1A58 (STATUS) een speciale functie nl:

bit 7	bit 1	bit 0	status
0	0	0	VDU met echo en normaal scrollen
0	0	1	VDU met echo en langzaam scrollen
0	1	0	VDU zonder echo en normaal scrollen
0	1	1	VDU zonder echo en langzaam scrollen
1	*	*	RS 232 bv. voor elekterminal of printer

(* = don't care)

Met deze drie bits kan men dus beslissen waar een karakter naar toe gaat, wel of geen echo v/e ingedrukte toets en hoe er gescrolled moet worden, over dit laatste nog een omerking:

De bedoeling was om een zgn. smooth scroll te maken dwz. dat er steeds met een lijn gescrolled wordt iov. met een hele regel, hierdoor kan er heel mooi gescrolled worden.

Dit smooth scrollen had ik zo gemaakt:

Door de vert. sync steeds te veranderen komt het beeld steeds met een lijn naar boven, dit (steeds met enige vertraging) hernalen totdat het scherm een gehele regel hoger staat. Dan de vert. sync terug zetten en normaal scrollen (bij dit laatste evt. het beeld even weghalen zodat dit niet te zien is).

Als ik dit echter bij mijn monitor doe dan vervormt het beeld dusdanig dat ik heb besloten smooth scroll te vervangen door langzaam scrollen.

De software omvat eigenlijk twee subroutines, nl. PRINT en INPUT.

Subroutine PRINT:

Deze subr. drukt een karakter af op het scherm indien bit 7 van STATUS is 0 is bit 7 1 dan wordt subr. PRCHA (\$1334 in PM) vervolgd.

Subroutine INPUT:

Deze subr. is eigenlijk het vervolg van subr. RECCHA (\$12AE in PM).

Er wordt gekeken of er een echo plaats moet vinden en of er een speciale toets is in gedrukt, deze zijn:

toets	functie
control a	initialiseert de VDU-kaart, zet alle registers goed en zet het beeld formaat weer op 80*25
control b	verwisselt de flag van normaal/langzaam scrollen
	tevens krijgt men een cursor die niet meer knippert
control d	verwisselt de flag van wel/geen echo
delete	geen echo naar scherm indien delete is ingedrukt

Indien control a, b of d is ingedrukt wordt er nog een keer naar RECCHA gesprongen voor een nieuwe toets, hierdoor kan men tijdens het intyoen van tekst deze drie toetsen gebruiken zonder dat uw programma foutmeldingen geeft.

Eventueel kunnen voor control a, b en d andere toetsen genomen worden, ik verwijs u hiervoor naar de listing.

Verder zitten er nog twee kleine subroutines in en wel:

FOR64 zet het beeld formaat op 64*16 (bij mij \$0FAB)

FOR90 zet het beeld formaat op 90*22 (bij mij \$0FD6)

Door een van deze subr. aan te roepen en beeld schoonmaken (dmv. \$0C naar het scherm te sturen) heeft u het gewenste formaat verkregen.

Dmv. control a kan men weer terug naar 80*25.

Met het eerste programma kan men ook nog andere formaten maken.

Met deze software blijft alle bestaande programma's werken, er wordt uiteindelijk niets van pag. 0 veranderd.

Het opstarten v/d computer gaat nu als volgt:

Zorg dat op STATUS \$00 staat, start PM op (1000 go), druk delete in, druk control a in (scherm initialiseren) en evt. control l (scherm schoonmaken). Het indrukken van control l heeft natuurlijk wel de melding WHAT? tot gevolg, omdat PM deze toets niet herkent.

In het begin heb ik in een tabel de data gegeven voor de formaten 80*25, 64*16 en 90*22 wilt u een ander formaat vast in eorum hebben dan kan dit u moet dan de tabel TAB, VMAX en HMAX zelf aanpassen. Ik verwijs u hiervoor weer naar de listing.

1 PASS 2

```

0005 :
0010 AREG .DE $D800 adres-register van de 6845
0015 DREG .DE $D801 data-register van de 6845
0020 HMAX .DE $1A52 max. aantal kar. horizontaal
0025 VMAX .DE $1A53 max. aantal regels op het scherm
0030 HCUR .DE $1A54 cursor positie horizontaal (0-HMAX)
0035 VCUR .DE $1A55 cursor positie verticaal (0-VMAX)
0040 TOPLS .DE $1A56 samen met TOPHS vormt dit het adres
0045 TOPHS .DE $1A57 dat links boven op het scherm staat
0050 STATUS .DE $1A58 flao die het volgende aangeeft
0055 :
0060 : STATUS = $00 echo en normaal scrollen
0065 : STATUS = $01 echo en langzaam scrollen
0070 : STATUS = $02 geen echo en normaal scrollen
0075 : STATUS = $03 geen echo en langzaam scrollen
0080 : STATUS = $FF kar. wordt serieel verzonden
0085 :
0090 CHA .DE $1A62 opslagplaats voor de ACCU
0095 :
0100 TOPL .DE $0 gebruikte adressen in pag 0
0105 TOPH .DE $1 deze worden allemaal gesaved
0110 TEMPL .DE $2 er gaat op pag 0 dus niks verloren
0115 TEMPH .DE $3
0120 :
0125 :dit programma heb ik achter TM gezet
0130 :
0135 .BA $C80
0140 .MC $200
0145 .OS
0150 :
0155 :de subr. INIT initialiseert de VDU-kaart
0160 :
0C80- A2 0F 0165 INIT LDX #$0F :vul register 0 t/m 15 met
0C82- 8E 00 D8 0170 LOOP1 STX AREG :data uit tabel TAB
0C85- BD B1 0C 0175 LDA TAB,X
0C88- 8D 01 D8 0180 STA DREG
0C8B- CA 0185 DEX
0C8C- 10 F4 0190 BPL LOOP1
0195 :
0200 : Max aantal kar. horizontaal
0205 :
0C8E- A9 50 0210 LDA #$50
0C90- 8D 52 1A 0215 STA HMAX
0220 :
0225 : Max aantal kar. verticaal
0230 :
0C93- A9 19 0235 LDA #$19
0C95- 8D 53 1A 0240 STA VMAX
0245 :
0C98- A9 00 0250 LDA #$00 :zet cursor links boven
0C9A- 8D 54 1A 0255 STA HCUR
0C9D- 8D 55 1A 0260 STA VCUR
0CA0- 8D 56 1A 0265 STA TOPLS
0CA3- A9 D0 0270 LDA #$D0 :TOP wijst naar $D000 dit is
0CA5- 8D 57 1A 0275 STA TOPHS :het begin v/h RAM-gebied
0CA8- AD 58 1A 0280 LDA STATUS :normaal scrollen
0CAB- 29 FE 0285 AND #$FE
0CAD- 8D 58 1A 0290 STA STATUS
0CB0- 60 0295 RTS
0CB1- 7A 50 58 0300 TAB .BY $7A $50 $58 :data
0CB4- 08 1C 1F 0305 .BY $08 $1C $1F : voor
0CB7- 19 1B 00 0310 .BY $19 $1B $00 : de
0CBA- 0A E0 07 0315 .BY $0A $E0 $07 : registers
0CBD- 00 00 00 0320 .BY $00 $00 $00 : v/d
0CC0- 00 0325 .BY $00 : 6845
0330 :
0335 :deze subr. maakt het scherm schoon
0340 :
0CC1- A9 D0 0345 CLS LDA #$D0 :zet TOP op $D000
0CC3- 85 01 0350 STA *TOPH :en vul $D000-$D800
0CC5- A0 00 0355 LDY #$00 :met soaties

```

```

0CC7- 84 00      0360      STY *TOPL
0CC9- A9 20      0365      LDA #$20
0CCB- 91 00      0370      STA (TOPL),Y
0CCD- C8         0375      INY
0CCE- D0 FB      0380      BNE LOOP3
0CD0- E6 01      0385      INC *TOPH
0CD2- A5 01      0390      LDA *TOPH
0CD4- C9 D8      0395      CMP #$D8      ;einde RAM-gebied?
0CD6- D0 F1      0400      BNE LOOP2
          0405      ;
          0410      ;deze subroutine geeft een FORM FEED
          0415      ;
0CD8- A9 D0      0420      FF      LDA #$D0
0CDA- 85 01      0425      STA *TOPH      ;TOP := $D000
0CDC- A9 00      0430      LDA #$00
0CDE- 8D 54 1A   0435      STA HCUR      ;HCUR := 00
0CEE- 8D 55 1A   0440      STA VCUR      ;VCUR := 00
0CE4- 85 00      0445      STA *TOPL
0CE6- A0 0C      0450      LDY #0C
0CE8- 8C 00 D8   0455      LOOP4     STY AREG      ;zet registers v/d 6845 goed
0CEB- 8D 01 D8   0460      STA DREG
0CEE- C8         0465      INY
0CEF- C0 10      0470      CPY #$10
0CF1- D0 F5      0475      BNE LOOP4
0CF3- 60         0480      RTS
          0485      ;
          0490      ;deze subroutine print een CR
          0495      ;
0CF4- AD 54 1A   0500      CR      LDA HCUR      ;voor aan regel?
0CF7- F0 44      0505      BEQ ENDCR      ;doe dan niks
0CF9- A0 0E      0510      LDY #$E      ;zet cursorpositie
0CFB- 8C 00 D8   0515      STY AREG      ;in TEMP
0CFE- AD 01 D8   0520      LDA DREG
0D01- 29 D7      0525      AND #$D7      ;moet tussen $D000
0D03- 09 D0      0530      ORA #$D0      ;en $D800 liggen
0D05- 85 03      0535      STA *TEMPH
0D07- C8         0540      INY
0D08- 8C 00 D8   0545      STY AREG
0D0B- AD 01 D8   0550      LDA DREG
0D0E- 85 02      0555      STA *TEMPL
0D10- AD 52 1A   0560      LDA HMAX
0D13- 38         0565      SEC
0D14- ED 54 1A   0570      SBC HCUR      ;HMAX-HCUR=aantal kar.
0D17- AA         0575      TAX      ;tot einde regel
0D18- 20 4B 0D   0580      JSR XSPATIES ;maak rest schoon
0D1B- A0 0F      0585      LDY #$F
0D1D- 8C 00 D8   0590      STY AREG
0D20- 38         0595      SEC
0D21- AD 01 D8   0600      LDA DREG      ;trek van register 14,15
0D24- ED 54 1A   0605      SBC HCUR      ;HCUR af, zodat de cursor
0D27- 8D 01 D8   0610      STA DREG      ;vooraan de regel komt
0D2A- 88         0615      DEY
0D2B- 8C 00 D8   0620      STY AREG
0D2E- AD 01 D8   0625      LDA DREG
0D31- E9 00      0630      SBC #0      ;de waarde van R14 moet
0D33- 29 0F      0635      AND #$0F      ;liggen tussen $0000-$1000
0D35- 8D 01 D8   0640      STA DREG
0D38- A9 00      0645      LDA #$00
0D3A- 8D 54 1A   0650      STA HCUR      ;HCUR := 00
0D3D- 60         0655      ENDCR      RTS
          0660      ;
          0665      ;deze subroutine verhoogt TEMP met 1 en
          0670      ;zorgt dat het resultaat tussen $D000
          0675      ;en $D800 blijft (video ram)
          0680      ;
0D3E- E6 02      0685      INCTEMP     INC *TEMPL
0D40- D0 08      0690      BNE ENDINC
0D42- E6 03      0695      INC *TEMPH
0D44- A5 03      0700      LDA *TEMPH
0D46- 29 D7      0705      AND #$D7
0D48- 85 03      0710      STA *TEMPH
0D4A- 60         0715      ENDINC     RTS
          0720      ;
          0725      ;deze subroutine zet vanaf het adres dat

```

```

0730 :in TEMP staat X spaties weg
0735 :
0D4B- A0 00 0740 XSPATIES LDY #$00
0D4D- CA 00 0745 DEX
0D4E- 30 0A 0750 BMI LOOP6 :klaar?
0D50- A9 20 0755 LDA #$20
0D52- 71 02 0760 STA (TEMP),Y
0D54- 20 3E 0D 0765 JSR INCTEMP
0D57- 4C 4D 0D 0770 JMP XSPATIES+2
0D5A- 60 0775 LOOP6 RTS
0780 :
0785 :deze subroutine print een LINE FEED
0790 :
0D5B- EE 55 1A 0795 LF INC VCUR :cursor 1 regel naar onder
0D5E- AD 55 1A 0800 LDA VCUR
0D61- CD 53 1A 0805 CMP VMAX :was het de laatste regel?
0D64- F0 19 0810 BEQ SCROLL :ja ga dan scrollen
0D66- A0 0F 0815 LDY #$0F
0D68- BC 00 D8 0820 STY AREG
0D6B- AD 01 D8 0825 LDA DREG
0D6E- 18 0830 CLC
0D6F- 6D 52 1A 0835 ADC HMAX :tel bij R14.R15 HMAX op
0D72- 8D 01 D8 0840 STA DREG :dus de cursor komt een
0D75- 88 0845 DEY :regel naar beneden
0D7E- 8C 00 D8 0850 STY AREG
0D79- AD 01 D8 0855 LDA DREG
0D7C- 4C EC 0D 0860 JMP AND :rond daar alles af
0865 :
0870 :SCROLL telt bij R12.R13 HMAX op, alles op scherm
0875 :gaat een regel naar boven en maakt onderste nieuwe
0880 :regel schoon
0885 :
0D7F- AD 58 1A 0890 SCROLL LDA STATUS :vertraagd scrollen?
0D82- 4A 0895 LSR A
0D83- 90 03 0900 BCC NSCROLL
0D85- 20 9D 0F 0905 SSCROLL JSR VERTRAG :ja vertraag dan
0D88- CE 55 1A 0910 NSCROLL DEC VCUR :zet VCUR weer op laatste regel
0D8B- A0 0D 0915 LDY #$D
0D8D- 8C 00 D8 0920 STY AREG
0D90- 18 0925 CLC
0D91- A5 00 0930 LDA *TOPL :TOP := R12.R13 + HMAX
0D93- 6D 52 1A 0935 ADC HMAX
0D96- 85 00 0940 STA *TOPL
0D98- 85 02 0945 STA *TEMPL :TEMP := TOP
0D9A- 8D 01 D8 0950 STA DREG
0D9D- 88 0955 DEY
0D9E- 8C 00 D8 0960 STY AREG
0DA1- A5 01 0965 LDA *TOPH
0DA3- 69 00 0970 ADC #$00
0DA5- 09 D0 0975 ORA #$D0 :resultaat moet tussen
0DA7- 29 D7 0980 AND #$D7 :$D000 en $D800 blijven
0DA9- 85 01 0985 STA *TOPH
0DAB- 85 03 0990 STA *TEMPH
0DAD- 29 07 0995 AND #$07 :resultaat moet tussen $000
0DAF- 8D 01 D8 1000 STA DREG :en $800 blijven (2K RAM)
1005 :
1010 :zet temp op eerste kar. laatste regel
1015 :
0DB2- AE 53 1A 1020 LDX VMAX :in TEMP staat TOP
0DB5- CA 1025 DEX
0DB6- F0 0E 1030 BEQ HMV
0DB8- 18 1035 CLC
0DB9- A5 02 1040 LDA *TEMPL :TEMP := HMAX * (VMAX-1) + TEMP
0DBB- 6D 52 1A 1045 ADC HMAX
0DBE- 85 02 1050 STA *TEMPL
0DC0- 90 F3 1055 BCC DEX
0DC2- E6 03 1060 INC *TEMPH
0DC4- D0 EF 1065 BNE DEX
0DC6- A5 02 1070 HMV LDA *TEMPL
0DC8- 48 1075 PHA
0DC9- A5 03 1080 LDA *TEMPH
0DCB- 48 1085 PHA
0DCC- 09 D0 1090 ORA #$D0 :zet resultaat weer
0DCE- 29 D7 1095 AND #$D7 :tussen $D000 en $D800
0DD0- 85 03 1100 STA *TEMPH

```

```

1105 :
1110 : maak laatste regel schoon
1115 :
1120 LDX HMAX
1125 JSR XSPATIES
1130 :
1135 : zet de cursor goed
1140 :
1145 LDY #0F
1150 PLA
1155 TAX
1160 PLA
1165 CLC
1170 ADC HCUR :R14,R15 := R14,R15 + HCUR
1175 STY AREG
1180 STA DREG
1185 TXA
1190 DEY
1195 STY AREG
1200 AND ADC #00 :resultaat moet tussen
1205 AND #0F :$0000 en $1000 liggen
1210 STA DREG
1215 ENDSCROLL RTS
1220 :
1225 : deze subroutine print een VT, de cursor gaat een regel
1230 : naar boven mits deze nog niet bovenaan staat
1235 :
1240 VT LDA VCUR :staat cursor bovenaan?
1245 BEQ ENDVT :zo ja, doe dan niets
1250 DEC VCUR :VCUR := VCUR - 1
1255 LDY #0F
1260 STY AREG :zet cursor HMAX plaatsen terug
1265 LDA DREG :dus een regel naar boven
1270 SEC
1275 SBC HMAX :R14,R15 := R14,R15 - HMAX
1280 STA DREG
1285 BCS ENDVT
1290 DEY
1295 STY AREG
1300 DEC DREG
1305 LDA DREG
1310 AND #0F :resultaat moet tussen
1315 STA DREG :$0000 en $1000 liggen
1320 ENDVT RTS
1325 :
1330 : deze subroutine print een BACK SPACE
1335 :
1340 BS LDA HCUR
1345 BNE BS1 :kijk of HCUR = 0
1350 LDA VCUR :zo ja, kijk of VCUR = 0
1355 BEQ ENDBS :zo ja, doe dan niets
1360 DEC VCUR :VCUR := VCUR - 1
1365 LDA HMAX
1370 STA HCUR :zet cursor achteraan
1375 BS1 DEC HCUR :zet cursor een plaats terug
1380 LDY #0F
1385 STY AREG
1390 LDA DREG
1395 SEC
1400 SBC #01 :R14,R15 := R14,R15 - 1
1405 STA DREG
1410 BCS ENDBS
1415 DEY
1420 STY AREG
1425 LDA DREG
1430 SBC #00
1435 AND #0F :resultaat moet tussen
1440 STA DREG :$0000 en $1000 liggen
1445 ENDBS RTS
1455 :
1460 .LI VDU2V1.1

```

F02 11F9 4292-548B

```

1470 :
1475 :deze subroutine zet de cursor een plaats
1480 :verder mits deze niet rechts onderaan staat
1485 :
0E52- EE 54 1A 1490 HT INC HCUR ;zet cursor een plaats verder
0E55- AD 54 1A 1495 LDA HCUR
0E58- CD 52 1A 1500 CMP HMAX ;staat cursor geheel rechts?
0E5B- D0 10 1505 BNE HT1
0E5D- EE 55 1A 1510 INC VCUR ;ja dus naar nieuwe regel
0E60- AD 55 1A 1515 LDA VCUR
0E63- CD 53 1A 1520 CMP VMAX ;staat cursor op laatste regel?
0E66- F0 1F 1525 BEQ HT2 ;ja, ga scrollen
0E68- A9 00 1530 LDA #$00
0E6A- 8D 54 1A 1535 STA HCUR ;nee, zet cursor vooraan regel
0E6D- A0 0F 1540 HT1 LDY #$0F
0E6F- 8C 00 D8 1545 STY AREG
0E72- EE 01 D8 1550 INC DREG ;R14, R15 := R14, R15 + 1
0E75- D0 0F 1555 BNE HT3
0E77- 88 1560 DEY
0E78- 8C 00 D8 1565 STY AREG
0E7B- AD 01 D8 1570 LDA DREG
0E7E- 18 1575 CLC
0E7F- 69 01 1580 ADC #$01
0E81- 29 0F 1585 AND #$0F ;resultaat moet tussen
0E83- 8D 01 D8 1590 STA DREG ;$0000 en $1000 liggen
0E86- 60 1595 HT3 RTS
0E87- 20 F4 0C 1600 HT2 JSR CR
0E8A- 4C 7F 0D 1605 JMP SCROLL
1610 :
1615 :deze subr. print de kar. op scherm die in de ACCU staat
1620 :X en Y register worden gesaved de ACCU gaat verloren!
1625 :
1630 :op $1334 moet komen te staan JMP PRINT
1635 :
0E8D- 8D 62 1A 1640 PRINT STA CHA ;save ACCU
0E90- AD 58 1A 1645 LDA STATUS ;waar moet kar. heen ?
0E93- 10 06 1650 BPL PRINT1
0E95- 8E 60 1A 1655 STX $1A60 ;kar. gaat naar printer
0E98- 4C 3A 13 1660 JMP $133A ;ga verder in subr. PRCHA
0E9B- 8A 1665 PRINT1 TXA : ;kar. gaat naar scherm
0E9C- 48 1670 PHA : ;save X en Y-register
0E9D- 98 1675 TYA
0E9E- 48 1680 PHA
0E9F- A5 00 1685 LDA *TOPL ;save TOPL en TOPH
0EA1- 48 1690 PHA
0EA2- A5 01 1695 LDA *TOPH
0EA4- 48 1700 PHA
0EA5- A5 02 1705 LDA *TEMPL ;save TEMPL en TEMPH
0EA7- 48 1710 PHA
0EA8- A5 03 1715 LDA *TEMPH
0EAA- 48 1720 PHA
0EAB- AD 56 1A 1725 LDA TOPLS ;zet oude TOPL (=TOPLS)
0EAE- 85 00 1730 STA *TOPL ;oo TOPL
0EB0- AD 57 1A 1735 LDA TOPHS ;zet oude TOPH (=TOPHS)
0EB3- 85 01 1740 STA *TOPH ;oo TOPH
0EB5- AD 62 1A 1745 LDA CHA ;grafische symbolen?
0EB8- 30 04 1750 BMI GRAF ;ja, dan wel afdrukken
0EBA- C9 1F 1755 CMP #$1F ;ASCII waarde ( 20 ?
0EBC- 30 49 1760 BMI BS? ;ja, ga verder bij BS?
0EBE- A0 0F 1765 GRAF LDY #$F ;nee, dan gewoon afdrukken
0ECO- 8C 00 D8 1770 STY AREG
0EC3- AD 01 D8 1775 LDA DREG
0EC6- 85 02 1780 STA *TEMPL ;TEMP := R14, R15
0EC8- 88 1785 DEY
0EC9- 8C 00 D8 1790 STY AREG
0ECC- AD 01 D8 1795 LDA DREG
0ECF- 09 D0 1800 ORA #$D0 ;waarde van TEMP moet tussen
0ED1- 29 D7 1805 AND #$D7 ;$D000 en $D800 blijven
0ED3- 85 03 1810 STA *TEMPH
0ED5- A0 00 1815 LDY #$0
0ED7- AD 62 1A 1820 LDA CHA ;zet de af te printen kar.
0EDA- 91 02 1825 STA (TEMPL),Y ;oo de huidige cursorpositie
0EDC- 20 52 0E 1830 PHT JSR HT ;zet cursor een plaats verder
0EDF- A5 00 1835 XY LDA *TOPL ;save TOPL
0EE1- 8D 56 1A 1840 STA TOPLS

```

0EE4- A5 01	1845	LDA *TOPH	:save TOPH
0EE6- 8D 57 1A	1850	STA TOPHS	
0EE9- 68	1855	PLA	
0EEA- 85 03	1860	STA *TEMPH	:herstel TEMPH
0EEC- 68	1865	PLA	
0EED- 85 02	1870	STA *TEMPL	:herstel TEMPL
0EEF- 68	1875	PLA	
0EF0- 85 01	1880	STA *TOPH	:herstel TOPH
0EF2- 68	1885	PLA	
0EF3- 85 00	1890	STA *TOPL	:herstel TOPL
0EF5- 68	1895	PLA	
0EF6- A8	1900	TAY ;	herstel X en Y-register
0EF7- 68	1905	PLA	
0EF8- AA	1910	TAX	
	1915 ;		
	1920 ;indien hier LDA CHA staat wordt ACCU ook gesaved!		
	1925 ;test voor een break		
	1930 ;		
0EF9- 2C 80 1A	1935	BIT \$1A80	
0EFC- 10 01	1940	BPL BREAK	
0EFE- 60	1945	RTS	
0EFF- 2C 80 1A	1950 BREAK	BIT \$1A80	
0F02- 10 FB	1955	BPL BREAK	
0F04- 6C 7C 1A	1960	JMP (\$1A7C)	
0F07- C9 08	1965 BS?	CMP #8	:is het een BACK SPACE?
0F09- D0 06	1970	BNE HT?	
0F0B- 20 1D 0E	1975	JSR BS	:ja, zet cursor een plaats terug
0F0E- 4C DF 0E	1980	JMP XY	:herstel alles
0F11- C9 09	1985 HT?	CMP #9	:is het een HORIZONTAL TAB?
0F13- D0 03	1990	BNE LF?	
0F15- 4C DC 0E	1995	JMP PHT	:ja, zet cursor een plaats verder
0F18- C9 0A	2000 LF?	CMP #\$A	:is het een LINE FEED?
0F1A- D0 06	2005	BNE VT?	
0F1C- 20 5B 0D	2010	JSR LF	:ja, zet cursor een regel verder
0F1F- 4C DF 0E	2015	JMP XY	:herstel alles
0F22- C9 0B	2020 VT?	CMP #\$B	:is het een VERTICAL TAB?
0F24- D0 06	2025	BNE CLS?	
0F26- 20 F4 0D	2030	JSR VT	:ja, zet cursor een regel terug
0F29- 4C DF 0E	2035	JMP XY	:herstel alles
0F2C- C9 0C	2040 CLS?	CMP #\$C	:is het een FORM FEED?
0F2E- D0 06	2045	BNE CR?	
0F30- 20 C1 0C	2050	JSR CLS	:ja, maak scherm schoon
0F33- 4C DF 0E	2055	JMP XY	:herstel alles
0F36- C9 0D	2060 CR?	CMP #\$D	:is het een CARRIAGE RETURN?
0F38- D0 06	2065	BNE FF?	
0F3A- 20 F4 0C	2070	JSR CR	:ja, zet cursor vooraan regel
0F3D- 4C DF 0E	2075	JMP XY	:herstel alles
0F40- C9 1C	2080 FF?	CMP #\$1C	:is het een FILE SEPARATOR?
0F42- D0 9B	2085	BNE XY	:nee, druk dan niets af
0F44- 20 D8 0C	2090	JSR FF	:ja, cursor home
0F47- 4C DF 0E	2095	JMP XY	:herstel alles
	2100 ;		
	2105 ;subr. INPUT is het vervolg van subr. RECCHA in PM.		
	2110 ;op \$12DC moet staan JMP INPUT.		
	2115 ;		
0F4A- AE 61 1A	2120 INPUT	LDX \$1A61	:herstel X-register
0F4D- C9 01	2125	CMP #1	:is het control a?
0F4F- F0 1D	2130	BEQ INITREQ	
0F51- C9 02	2135	CMP #2	:is het control b?
0F53- F0 1F	2140	BEQ SSORNS	
0F55- C9 04	2145	CMP #4	:is het control d?
0F57- F0 39	2150	BEQ ECHO	
0F59- C9 7F	2155	CMP #\$7F	:is het DELETE?
0F5B- F0 10	2160	BEQ INEND	:ja, echo dan niet
0F5D- 48	2165	PHA	
0F5E- AD 58 1A	2170	LDA STATUS	:moet het serieel?
0F61- 30 09	2175	BMI NOECHO	:ja, echo dan niet
0F63- 4A	2180	LSR A	
0F64- 4A	2185	LSR A	:echo of geen echo?
0F65- B0 05	2190	BCS NOECHO	:geen echo
0F67- 68	2195	PLA ;	zet kar. weer in ACCU
0F68- 48	2200	PHA ;	save ACCU
0F69- 20 8D 0E	2205	JSR PRINT	:echo
0F6C- 68	2210 NOECHO	PLA ;	zet kar. weer in ACCU

```

OF6D- 60      2215 INEND      RTS
                2220 :
                2225 :CONTROL A,B en D hebben een speciale functie
                2230 :
OF6E- 20 80 0C 2235 INITREQ JSR INIT      :control a. init. de 6845
OF71- 4C 8C 0F 2240 JMP INPUT2    :vraag opnieuw een kar.
OF74- AD 58 1A 2245 SSDRNS LDA STATUS    :control b. verwissel
OF77- 49 01      2250 EOR #1          :langzaam en normaal scrollen
OF79- 8D 58 1A 2255 STA STATUS
OF7C- 4A      2260 LSR A
OF7D- AD BB 0C 2265 LDA TAB+<A
OF80- 90 02      2270 BCC NN
OF82- 29 00      2275 AND #$00
OF84- A2 0A      2280 LDX #$A          :verwissel ook
OF86- BE 00 DB 2285 STX AREG          :knipoerende en
OF89- 8D 01 DB 2290 STA DREG          :vaste cursor
OF8C- AE 61 1A 2295 INPUT2 LDX $1A61    :herstel X-register en
OF8F- 4C AE 12 2300 JMP $12AE      :haal volgende toets binnen
OF92- AD 58 1A 2305 ECHO LDA STATUS     :control d. verwissel
OF95- 49 02      2310 EOR #2          :dan van echo
OF97- 8D 58 1A 2315 STA STATUS
OF9A- 4C AE 12 2320 JMP $12AE      :haal volgende toets binnen
                2325 :
                2330 :subr. VERTRAAG vertraagt het scrollen
                2335 :
OF9D- A2 00      2340 VERTRAAG LDX #0
OF9F- A0 50      2345 LDY #$50
OFA1- CA      2350 VV DEX
OFA2- D0 FD      2355 BNE VV
OFA4- 88      2360 DEY
OFA5- D0 FA      2365 BNE VV
OFA7- 60      2370 RTS
                2375 :
                2380 :bij aanroep v/d subr. FOR64 wordt de beeldindeling 64*16
                2385 :
OFAB- A9 10      2390 FOR64 LDA #$10
OFAA- 8D 53 1A 2395 STA VMAX
OFAD- A0 06      2400 LDY #6
OFAF- 8C 00 DB 2405 STY AREG
OFB2- 8D 01 DB 2410 STA DREG
OFB5- C8      2415 INY
OFB6- A9 16      2420 LDA #$16
OFB8- 8C 00 DB 2425 STY AREG
OFBB- 8D 01 DB 2430 STA DREG
OFBE- A9 40      2435 LDA #$40
OFC0- 8D 52 1A 2440 STA HMAX
OFC3- A0 01      2445 LDY #1
OFC5- 8C 00 DB 2450 STY AREG
OFC8- 8D 01 DB 2455 STA DREG
OFCB- A0 03      2460 LDY #3
OFCD- A9 05      2465 LDA #5
OFCF- 8C 00 DB 2470 EFOR64 STY AREG
OFD2- 8D 01 DB 2475 STA DREG
OFD5- 60      2480 RTS
                2485 :
                2490 :bij aanroep v/d subr. FOR90 wordt de beeldindeling 90*22
                2495 :
OFD6- A9 5A      2500 FOR90 LDA #$5A
OFD8- 8D 52 1A 2505 STA HMAX
OFDB- A0 01      2510 LDY #1
OFDD- 8C 00 DB 2515 STY AREG
OFEO- 8D 01 DB 2520 STA DREG
OFE3- C8      2525 INY
OFE4- A9 5E      2530 LDA #$5E
OFE6- 8C 00 DB 2535 STY AREG
OFE9- 8D 01 DB 2540 STA DREG
OFEC- A9 16      2545 LDA #$16
OFEE- 8D 53 1A 2550 STA VMAX
OFF1- A0 06      2555 LDY #6
OFF3- 8C 00 DB 2560 STY AREG
OFF6- 8D 01 DB 2565 STA DREG
OFF9- C8      2570 INY
OFFA- A9 19      2575 LDA #$19
OFFC- 4C CF 0F 2580 JMP EFOR64
                2585 .EN

```

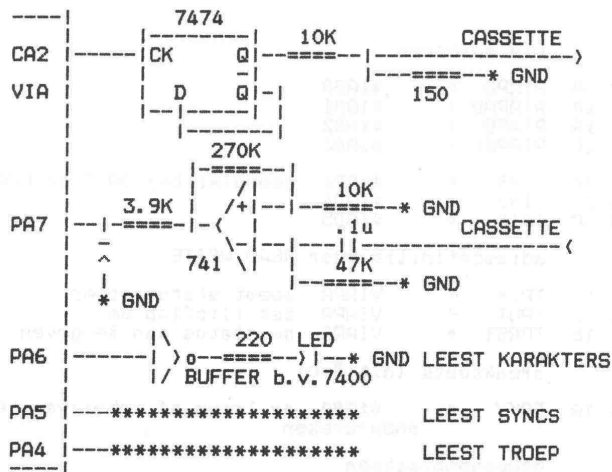
```

0010:      PROGRAM  CASSETTE 2
0020:
0030:      (C)      AD BROUWER
0040:
0050:
0060:
0070:      Dit programma doet de I/O die noodzakelijk is
0080:      om wat op cassette te schrijven of om er weer wat
0090:      vanaf te halen.
0100:      De subroutines worden aangeroepen vanuit het
0110:      programma CASSETTE 1.
0120:
0130:
0140:      Er zijn twee soorten formaten:
0150:
0160:      - JUNIOR  LOAD en SAVE
0170:      - APPLE   READ en WRITE
0180:
0190:
0200:      Het junior formaat is zodanig veranderd dat er ook
0210:      asciifiles geschreven en gelezen kunnen worden.
0220:
0230:
0240:      Het apple formaat vereist een hardware aanpassing
0250:      die op de VIA aangesloten kan worden.
0260:
0270:
0280:      Vlaggen na het lezen.
0290:
0300:      Na afloop is de zero (Z) gezet als het programma
0310:      foutloos gelezen is.
0320:      Als deze niet staat, staat de carry vlag (C) als het
0330:      programma niet past.
0340:      De overflow (V) is nul behalve bij LOAD waar V=1
0350:      als er een ander dan het gewenste label gevonden is.
0360:      CASSETTE 1 print dit label en begint opnieuw met lezen.
0370:
0380:      In het Y register staat een kopie van het
0390:      statusregister.
0400:
0410:
0420:      Er kan voortijdig met schrijven worden gestopt
0430:      door te breaken. SAVE wordt juist afgesloten maar
0440:      bij WRITE klopt de lengtespecificatie, die eerder
0450:      op cassette is geschreven, niet meer.
0460:
0470:      LOAD en READ kunnen ook onderbroken worden. Dit
0480:      wordt beschouwd als een data error en na CR wordt
0490:      er weer opnieuw begonnen (zie CASSETTE 1).
0500:      Het breaken tijdens lezen kan alleen als er data
0510:      binnenkomt.
0520:
0530:      Beschrijving APPLE formaat.
0540:
0550:      Het apple formaat ziet er zo uit:
0560:
0570:
0580:      header      start      1      0      0
0590:
0600:      |-----| |---| |-----| |---| |---| |---|
0610:      -|-----| |---| |-----| |---| |---| |---|
0620:      {-----} {---} {-----} {---}
0630:      1.4ms    0.5ms  1.0ms    0.5ms
0640:

```

0650:
0660: Na een header van ongeveer 5 sec komt een start bit.
0670: Direct hierachter komt de data. Er wordt begonnen met
0680: bit 7 uit te zenden. De bytes worden synchroon uit
0690: gezonden. Als het blok verzonden is volgt het crc byte.
0700:
0710:
0720: Als er een startadres en/of lengte op cassette wordt
0730: gezet gaat dit hetzelfde alleen zonder crc.
0740:
0750:
0760: Bij mode CO wordt eerst adres laag en daarna adres
0770: hoog op cassette gezet. Hierna volgt lengte laag
0780: en lengte hoog.
0790:
0800:
0810: Bij mode 80 wordt lengte laag gevolgt door lengte hoog
0820: op cassette gezet.
0830:
0840:
0850: Als de adressen op cassette staan wordt er opnieuw
0860: een header geschreven alleen deze duurt nu 2.5 sec.
0870: Na deze header wordt het blok weggeschreven.
0880:
0890:
0900: Bij het lezen kan de software de pointers van cassette
0910: halen. Voordat het blok wordt gelezen moet immers het
0920: adres van het laatste byte bekend zijn.
0930: Als er een puls wordt overgeslagen betekent dit dat de
0940: read routine niet tot een eind komt. Om hier toch uit
0950: te komen moet er gebreakt worden.
0960:
0970:
0980: De read routine begint met eerst syncs te zoeken.
0990: Hierna wordt er gelezen tot er een startbit wordt gezien.
1000: Na het startbit worden de adressen ingelezen en verwerkt
1010: of er wordt data binnen gehaald t/m het eindadres.
1020:
1030: In de apple routines kan de software worden aangepast
1040: als er maar rekening mee wordt gehouden dat het Y
1050: register als timer wordt gebruikt.
1060:
1070:
1080: Read en write sluit ook het relay dat de motor aanzet
1090: van de recorder.
1100:
1110:
1120: Hardware aanpassing voor READ en WRITE.
1130: -----
1140:
1150: Voor het schrijven is een flipflop en een
1160: spanningsdeler nodig.
1170: Voor het lezen moet het signaal van cassette
1180: tot blokvorm worden versterkt.
1190:
1200: Om geïnformeerd te worden over de leesstatus worden
1210: er drie ledjes gebruikt.
1220:
1230:
1240: Om een klok voor de flipflop te verkrijgen wordt
1250: port A van de VIA zo geschakeld dat als port A
1260: gelezen wordt, dat dan CA2 een puls afgeeft die
1270: als klok dient voor de flipflop.
1280:

1290:
1300:
1310:
1320:
1330:
1340:
1350:
1360:
1370:
1380:
1390:
1400:
1410:
1420:
1430:
1440:
1450:
1460:
1470:
1480:
1490:
1500:
1510:
1520:
1530:
1540:
1550:
1560:
1570:
1580:
1590:
1600:
1610:
1620:
1630:
1640:
1650:
1660:
1670:
1680:
1690:
1700:
1710:
1720:
1730:
1740:
1750:
1760:
1770:
1780:
1790:
1800:
1810:
1820:
1830:
1840:
1850:
1860: 4200
1870:
1880:
1890:
1900:
1910:
1920:



de 7474 flipflop heeft +5 en GND nodig
de 741 opamp heeft +5 en -5 nodig.
bij de opamp moet de voedingsspanning met
2 tantaals van 1u worden ontkoppeld.

Benodigde geheugenplaatsen:

In MODE (00F9) staat de mode.
In TPBG (00FA,00FB) staat het begin adres.
In TPEN (00FC,00FD) staat het eind adres.
Voor SAVE en LOAD staat in LABL (00FE) het label.
Bij READ en WRITE wordt LABL gebruikt als
hulpgeheugen.

SAVE en LOAD gebruiken CHSU en TEMP (4 PLAATSEN)
nadat deze plaatsen eerst op stack zijn gezet.

Bij alle routines wordt eerst de interrupt
geblokkeerd en na afloop weer teruggezet.

Het is mogelijk om een gedeelte te kiezen
van dit programma. In combinatie met CASSETTE 1
kan er gekozen worden om alleen de apple routines
over te houden.

Ik kies 4200 als startadres (willekeurig?).
Deze routines kunnen later in eeprom worden
gezet op de plaats van de huidige TM.

ORG \$4200

via adressen

01 18	VIAPA *	\$1801	gebruikt bij READ en WRITE
03 18	VIAPAD *	\$1803	
0C 18	VIAPCR *	\$180C	

```

1930:
1940:
1950:
1960:      pia adressen
1970:      80 1A PIAPA *    $1A80
1980:      81 1A PIAPAD *   $1A81
1990:      82 1A PIAPB *    $1A82
2000:      83 1A PIAPBD *   $1A83
2010:
2020:      F4 1A TIMR *     $1AF4 gebruikt bij SAVE en LOAD
2030:      D4 1A TIRD *     $1AD4
2040:      D5 1A INTR *     $1AD5
2050:
2060:      adresdefinitie voor READ WRITE
2070:      01 18 TPIN *      VIAPA leest status opamp
2080:      01 18 TPOT *      VIAPA zet flipflop om
2090:      01 18 TPRST *     VIAPA om status aan te geven
2100:
2110:      breaktoets (bit 7=0)
2120:
2130:      80 1A BRKY *      $1A80 om lezen of schrijven te
2140:      onderbreken
2150:
2160:      geheugenplaatsen
2170:
2180:      EC 00 CHSU *       $00EC checksum
2190:      EE 00 TEMP *       CHSU +02 tijdelijk gebruik
2200:
2210:      F9 00 MODE *       $00F9 mode
2220:      FA 00 TPBG *       $00FA begin
2230:      FC 00 TPEN *       $00FC eind
2240:      FE 00 LABL *       $00FE label/hulp
2250:
2260:      Om de programmas gemakkelijk aan elkaar te
2270:      kunnen passen, begint ook dit programma met
2280:      een aantal jumps.
2290:
2300: 4200 4C 13 44      JMP LOAD
2310: 4203 4C 95 43      JMP SAVE
2320: 4206 4C 7B 42      JMP READ
2330:
2340:      apple write i/o
2350:
2360: 4209 08      WRITE PHP      write apple formaat
2370: 420A 78      SEI           blokkeer interrupts
2380: 420B 20 3C 46 JSR STWRT     init pia via write
2390: 420E 24 F9   BITZ MODE
2400: 4210 10 39   BPL WRT      blokwrite
2410: 4212 38      SEC          bereken lengte
2420: 4213 A5 FC   LDAZ TPEN
2430: 4215 E5 FA   SBCZ TPBG
2440: 4217 A8      TAY
2450: 4218 A5 FD   LDAZ TPEN +01
2460: 421A E5 FB   SBCZ TPBG +01
2470: 421C C8      INY          tel er een bij op
2480: 421D D0 02   BNE WRB
2490: 421F 69 00   ADCIM $00
2500: 4221 48      WRB          PHA
2510: 4222 98      TYA
2520: 4223 48      PHA
2530: 4224 A9 20   LDAIM $20 5 sec header
2540: 4226 20 1B 43 JSR HEADR
2550: 4229 A0 2C   LDYIM $2C
2560: 422B 24 F9   BITZ MODE

```

2570:	422D	50	OE		BVC	WRC	
2580:	422F	A5	FA		LDAZ	TPBG	schrijf startadres
2590:	4231	20	12	43	JSR	WRBYTE	
2600:	4234	A0	2E		LDYIM	\$2E	
2610:	4236	A5	FB		LDAZ	TPBG	+01
2620:	4238	20	12	43	JSR	WRBYTE	
2630:	423B	A0	2E		LDYIM	\$2E	
2640:	423D	68			PLA		schrijf lengte
2650:	423E	20	12	43	JSR	WRBYTE	
2660:	4241	A0	2E		LDYIM	\$2E	
2670:	4243	68			PLA		
2680:	4244	20	12	43	JSR	WRBYTE	
2690:	4247	A0	10		LDYIM	\$10	2.5 sec header
2700:	4249	D0	02		BNE	WRT	
2710:	424B	A0	20		LDYIM	\$20	5 sec header
2720:	424D	A2	02		LDXIM	\$02	bewaar mode en begin
2730:	424F	B5	F9		LDAZX	MODE	
2740:	4251	48			PHA		
2750:	4252	CA			DEX		
2760:	4253	10	FA		BPL	WRTF	
2770:	4255	18			CLC		
2780:	4256	08			PHP		bewaar carry
2790:	4257	98			TYA		
2800:	4258	20	1B	43	JSR	HEADR	
2810:	425B	A0	27		LDYIM	\$27	
2820:	425D	A2	00		LDXIM	\$00	
2830:	425F	41	FA		EORIX	TPBG	bereken crc check
2840:	4261	48			PHA		
2850:	4262	A1	FA		LDAIX	TPBG	
2860:	4264	20	12	43	JSR	WRBYTE	schrijf byte
2870:	4267	A0	1B		LDYIM	\$1B	
2880:	4269	20	EA	45	JSR	NXTAA	volgend adres
2890:	426C	10	04		BPL	WRI	break
2900:	426E	68			PLA		
2910:	426F	90	EC		BCC	WRA	eind ?
2920:	4271	48			PHA		
2930:	4272	68			PLA		schrijf check
2940:	4273	A0	22		LDYIM	\$22	
2950:	4275	20	12	43	JSR	WRBYTE	
2960:	4278	4C	02	43	JMP	READRD	
2970:							
2980:							
2990:							
3000:	427B	08			READ	PHP	read apple formaat
3010:	427C	78			SEI		
3020:	427D	20	25	46	JSR	STRED	init via pia read
3030:	4280	24	F9		BITZ	MODE	
3040:	4282	10	4D		BPL	RDBLO	read blok
3050:	4284	20	62	43	JSR	SYNC	
3060:	4287	10	48		BPL	RDBLO	break
3070:	4289	A0	3B		LDYIM	\$3B	
3080:	428B	24	F9		BITZ	MODE	
3090:	428D	50	17		BVC	RDLA	
3100:	428F	20	3E	43	JSR	RDBYTE	read startadres
3110:	4292	48			PHA		
3120:	4293	A0	37		LDYIM	\$37	
3130:	4295	20	3E	43	JSR	RDBYTE	
3140:	4298	AA			TAX		
3150:	4299	A5	F9		LDAZ	MODE	negeer startadres?
3160:	429B	4A			LSRA		
3170:	429C	68			PLA		
3180:	429D	A0	34		LDYIM	\$34	
3190:	429F	B0	05		BCS	RDLA	ja
3200:	42A1	B5	FA		STAZ	TPBG	zet begin

3210:	42A3	86	FB		STXZ	TPBG	+01	
3220:	42A5	88			DEY			
3230:	42A6	20	3E	43	JSR	RDBYTE	read lengte	
3240:	42A9	48			PHA			
3250:	42AA	A0	37		LDYIM	#37		
3260:	42AC	20	3E	43	JSR	RDBYTE		
3270:	42AF	A8			TAY			
3280:	42B0	68			PLA			
3290:	42B1	AA			TAX		lengte in Y,X	
3300:	42B2	D0	01		BNE	RDBC		
3310:	42B4	88			DEY		verlaag	
3320:	42B5	CA			DEX			
3330:	42B6	18			CLC			
3340:	42B7	8A			TXA		bereken eind	
3350:	42B8	65	FA		ADCZ	TPBG		
3360:	42BA	AA			TAX			
3370:	42BB	98			TYA			
3380:	42BC	65	FB		ADCZ	TPBG	+01	
3390:	42BE	A8			TAY			
3400:	42BF	E4	FC		CPXZ	TPEN	vergelijk max eind	
3410:	42C1	D0	06		BNE	RDBK		
3420:	42C3	E5	FD		SBCZ	TPEN	+01 controleer gelijk	
3430:	42C5	F0	06		BEQ	RDBL	past dan ook	
3440:	42C7	D0	02		BNE	RDBM		
3450:	42C9	E5	FD		SBCZ	TPEN	+01	
3460:	42CB	B0	05		BCS	RDBD	past niet	
3470:	42CD	86	FC		STXZ	TPEN	past wel	
3480:	42CF	84	FD		STYZ	TPEN	+01	
3490:	42D1	18			CLC			
3500:	42D2	A2	02		LDXIM	#02	bewaar mode en begin	
3510:	42D4	B5	F9		LDAXZ	MODE		
3520:	42D6	48			PHA			
3530:	42D7	CA			DEX			
3540:	42D8	10	FA		BPL	REAA		
3550:	42DA	86	F9		STXZ	MODE	gebruikt voor crc check	
3560:	42DC	08			PHP			
3570:	42DD	20	62	43	JSR	SYNC		
3580:	42E0	30	07		BMI	RDCB		
3590:	42E2	28			PLP			
3600:	42E3	18			CLC			
3610:	42E4	A9	01		LDAIM	#01		
3620:	42E6	08			PHP			
3630:	42E7	90	19		BCC	READRD		
3640:	42E9	A0	3B		LDYIM	#3B		
3650:	42EB	20	3E	43	JSR	RDBYTE	read byte	
3660:	42EE	81	FA		STAIX	TPBG		
3670:	42F0	45	F9		EORZ	MODE	crc adjust	
3680:	42F2	85	F9		STAZ	MODE		
3690:	42F4	A0	33		LDYIM	#33		
3700:	42F6	20	EA	45	JSR	NXTAA	next	
3710:	42F9	10	E7		BPL	RDCB	break	
3720:	42FB	90	EE		BCC	RDC	eind ?	
3730:	42FD	20	3E	43	JSR	RDBYTE	read crc	
3740:	4300	C5	F9		CMPZ	MODE	crc ok?	
3750:	4302	18			CLC			
3760:	4303	B8			CLV			
3770:	4304	08			PHP		move status naar Y	
3780:	4305	68			PLA			
3790:	4306	A8			TAY			
3800:	4307	20	FC	45	JSR	INIVIA	reset via	
3810:	430A	28			PLP			
3820:	430B	B8			CLV			
3830:	430C	90	01		BCC	RDD	geen mem err	
3840:	430E	C8			INY		zet C	

```

3850: 430F 4C EF 44 RDD JMP LOADRD
3860:
3870: subroutines gebruikt door read en write i/o
3880:
3890: 4312 A2 10 WRBYTE LDXIM $10 schuift bits
3900: 4314 0A WRBYTB ASLA
3910: 4315 20 28 43 JSR WRBIT
3920: 4318 D0 FA BNE WRBYTB
3930: 431A 60 RTS
3940: 431B A0 4B HEADR LDYIM $4B schrijft header
3950: 431D 20 2D 43 JSR ZERDLY tijd in A
3960: 4320 D0 F9 BNE HEADR
3970: 4322 69 FE ADCIM $FE
3980: 4324 B0 F5 BCS HEADR
3990: 4326 A0 21 LDYIM $21
4000: 4328 20 2D 43 WRBIT JSR ZERDLY bittijd
4010: 432B C8 INY
4020: 432C C8 INY
4030: 432D 88 ZERDLY DEY
4040: 432E D0 FD BNE ZERDLY
4050: 4330 90 05 BCC WRTAPE
4060: 4332 A0 32 LDYIM $32
4070: 4334 88 ONEDLY DEY
4080: 4335 D0 FD BNE ONEDLY
4090: 4337 AC 01 18 WRTAPE LDY TPOT omzetten flipflop
4100: 433A A0 2C LDYIM $2C
4110: 433C CA DEX
4120: 433D 60 RTS
4130: 433E A2 08 RDBYTE LDXIM $08 read byte
4140: 4340 48 RDBYTB PHA
4150: 4341 20 4D 43 JSR RDBBIT
4160: 4344 68 PLA
4170: 4345 2A ROLA
4180: 4346 A0 3A LDYIM $3A
4190: 4348 CA DEX
4200: 4349 D0 F5 BNE RDBYTB
4210: 434B 60 RDBYRT RTS
4220:
4230: 434C 68 RDBBIS PLA
4240: 434D 20 50 43 RDBBIT JSR RDBIT stackaanoassing
4250: 4350 88 RDBIT DEY test 2 overgangen
4260: 4351 AD 01 18 LDA TPIN test 1 overgang
4270: 4354 45 FE EORZ LABL tijdsduur staat in Y
4280: 4356 10 FB BPL RDBIT geen overgang
4290: 4358 45 FE EORZ LABL
4300: 435A 85 FE STAZ LABL bewaar
4310: 435C C0 80 CPYIM $80 vergelijk Y
4320: 435E AD 80 1A LDA BRKY breaktest
4330: 4361 60 RTS
4340:
4350: 4362 20 4D 43 SYNC JSR RDBBIT synczoek routine
4360: 4365 A9 60 LDAIM $60 zet troep led
4370: 4367 8D 01 18 STA TPRST
4380: 436A A9 04 LDAIM $04
4390: 436C A0 20 SYB LDYIM $20
4400: 436E 48 PHA
4410: 436F 20 50 43 JSR RDBIT
4420: 4372 10 D8 BPL RDBBIS break
4430: 4374 68 PLA
4440: 4375 90 EB BCC SYNC start oornieuw
4450: 4377 CA DEX
4460: 4378 D0 F2 BNE SYB
4470: 437A 69 FE ADCIM $FE
4480: 437C B0 EE BCS SYB syncs genoeg

```

4490:	437E	A9	50		LDAIM \$50	zet sync led
4500:	4380	8D	01	18	STA TPRST	
4510:	4383	A0	1F		LDYIM \$1F	
4520:	4385	20	50	43	JSR RDBIT	
4530:	4388	10	C6		BPL RDBIT	break
4540:	438A	A0	21		LDYIM \$21	
4550:	438C	B0	F7		BCS RDBB	wacht tot korte
4560:	438E	A9	30		LDAIM \$30	start puls
4570:	4390	8D	01	18	STA TPRST	zet karakter led
4580:	4393	90	BB		BCC RDBIT	extra bit
4590:						
4600:					junior save i/o	
4610:						
4620:	4395	08			SAVE PHP	save junior formaat
4630:	4396	78			SEI	interrupts geblokkeerd
4640:	4397	20	1E	46	JSR STSAV	init pia save
4650:	439A	A2	02		LDXIM \$02	
4660:	439C	B5	F9		LDAZX MODE	bewaar mode en begin
4670:	439E	48			PHA	
4680:	439F	CA			DEX	
4690:	43A0	10	FA		BPL SVBB	
4700:	43A2	A2	04		LDXIM \$04	bewaar chksum en temp
4710:	43A4	B5	EB		LDAZX CHSU	-01
4720:	43A6	48			PHA	
4730:	43A7	CA			DEX	
4740:	43A8	D0	FA		BNE SVB	
4750:	43AA	86	EC		STXZ CHSU	init checksum
4760:	43AC	86	ED		STXZ CHSU	+01
4770:	43AE	A9	16		LDAIM \$16	255 syncs
4780:	43B0	20	31	45	JSR OUTCHT	
4790:	43B3	C6	EC		DECZ CHSU	
4800:	43B5	D0	F7		BNE SVBC	
4810:	43B7	A9	2A		LDAIM \$2A	hex startkarakter
4820:	43B9	24	F9		BITZ MODE	
4830:	43BB	10	02		BPL SAVB	
4840:	43BD	A9	02		LDAIM \$02	ascii startkarakter
4850:	43BF	20	31	45	JSR OUTCHT	
4860:	43C2	A5	FE		LDAZ LABL	save label
4870:	43C4	20	1E	45	JSR OUTBT	
4880:	43C7	24	F9		BITZ MODE	
4890:	43C9	30	0A		BMI SAVD	
4900:	43CB	A5	FA		LDAZ TPBG	save startadres
4910:	43CD	20	1B	45	JSR OUTBTC	
4920:	43D0	A5	FB		LDAZ TPBG	+01
4930:	43D2	20	1B	45	JSR OUTBTC	
4940:	43D5	A0	00		LDYIM \$00	
4950:	43D7	B1	FA		LDAIY TPBG	save bytes
4960:	43D9	24	F9		BITZ MODE	
4970:	43DB	30	05		BMI SAVC	
4980:	43DD	20	1B	45	JSR OUTBTC	hexformaat
4990:	43E0	F0	08		BEQ SAVG	
5000:	43E2	29	7F		ANDIM \$7F	ascii clear bit 7
5010:	43E4	20	6D	45	JSR CHKT	adjust checksum
5020:	43E7	20	31	45	JSR OUTCHT	
5030:	43EA	20	EA	45	JSR NXTAA	next
5040:	43ED	10	02		BPL SVGB	break
5050:	43EF	90	E4		BCC SAVD	eind?
5060:	43F1	A9	2F		LDAIM \$2F	hex eindkarakter
5070:	43F3	24	F9		BITZ MODE	
5080:	43F5	10	02		BPL SAVH	
5090:	43F7	A9	03		LDAIM \$03	ascii eindkarakter
5100:	43F9	20	31	45	JSR OUTCHT	
5110:	43FC	A5	EC		LDAZ CHSU	save checksum
5120:	43FE	20	1E	45	JSR OUTBT	

```

5130: 4401 A5 ED          LDAZ CHSU +01
5140: 4403 20 1E 45       JSR OUTBT
5150: 4406 A9 04          LDAIM $04 eindkarakter
5160: 4408 20 31 45       JSR OUTCHT
5170: 440B A9 04          LDAIM $04
5180: 440D 20 31 45       JSR OUTCHT
5190: 4410 4C E2 44       JMP LORD
5200:
5210:
5220:      junior load i/o
5230: 4413 08          LOAD PHP      load junior formaat
5240: 4414 78          SEI          interrupt off
5250: 4415 20 30 46       JSR STLOD  init pia load
5260: 4418 A2 02          LDXIM $02  bewaar mode en begin
5270: 441A B5 F9       LOB  LDAZX MODE
5280: 441C 48          PHA
5290: 441D CA          DEX
5300: 441E 10 FA       BPL LOB
5310: 4420 A2 04          LDXIM $04  bewaar chksum en tempo
5320: 4422 B5 EB       LDB  LDAZX CHSU -01
5330: 4424 48          PHA
5340: 4425 CA          DEX
5350: 4426 D0 FA       BNE LDB
5360: 4428 86 ED       STXZ CHSU +01
5370: 442A A9 FF       LOAB LDAIM $FF start sync
5380: 442C 20 A7 45       LOAC JSR INPULS
5390: 442F 20 D2 45       JSR DISER display error
5400: 4432 2C 80 1A       BIT BRKY
5410: 4435 10 1A       BPL LOANB break
5420: 4437 C9 16       CMPIM $16 sync
5430: 4439 D0 F1       BNE LOAC
5440: 443B 85 EC       STAZ CHSU
5450: 443D 20 A0 45       LOAEA JSR INASC haal karakter
5460: 4440 20 D2 45       JSR DISER display error
5470: 4443 C9 16       CMPIM $16
5480: 4445 D0 E3       BNE LOAB geen sync
5490: 4447 C6 EC       DECZ CHSU
5500: 4449 D0 F2       BNE LOAEA genoeg goede syncs
5510: 444B 20 A0 45       LOAE JSR INASC
5520: 444E 2C 80 1A       BIT BRKY
5530: 4451 10 70       LOANB BPL LOANA break
5540: 4453 20 D6 45       JSR DISSY display syncs
5550: 4456 C9 16       CMPIM $16
5560: 4458 F0 F1       BEQ LOAE syncs
5570: 445A 24 F9       BITZ MODE
5580: 445C 10 06       BPL LOAF
5590: 445E C9 02       CMPIM $02
5600: 4460 D0 C8       BNE LOAB
5610: 4462 F0 04       BEQ LOAG
5620: 4464 C9 2A       LOAF CMPIM $2A hex start
5630: 4466 D0 C2       BNE LOAB
5640: 4468 20 DA 45       LOAG JSR DISOK display ok
5650: 446B 20 79 45       JSR INBYT load label
5660: 446E C5 FE       CMPZ LABL
5670: 4470 F0 10       BEQ LOAH goede label
5680: 4472 A6 FE       LDXZ LABL label is 00
5690: 4474 F0 0C       BEQ LOAH
5700: 4476 E0 FF       CPXIM $FF label is FF
5710: 4478 F0 08       BEQ LOAH
5720: 447A 85 F9       STAZ MODE
5730: 447C 08          PHP
5740: 447D 68          PLA
5750: 447E 09 40       ORAIM $40 zet V vlag
5760: 4480 D0 64       BNE LODRR returnt met V vlag on

```

5770:	4482	48		LOAH	PHA		bewaar label
5780:	4483	24	F9		BITZ	MODE	
5790:	4485	30	21		BMI	LOAD	
5800:	4487	BA			TSX		
5810:	4488	20	6A	45	JSR	CHKIN	
5820:	448B	E6	FE		INCZ	LABL	
5830:	448D	F0	09		BEG	LOAHA	label FF
5840:	448F	24	F9		BITZ	MODE	
5850:	4491	70	05		BVS	LOAHA	negeer startadres
5860:	4493	85	FA		STAZ	TPBG	
5870:	4495	9D	07	01	STAAZ	\$0107	stack beginadres
5880:	4498	20	6A	45	LOAHA	JSR	CHKIN
5890:	449B	A4	FE		LDYZ	LABL	
5900:	449D	F0	09		BEG	LOAD	
5910:	449F	24	F9		BITZ	MODE	
5920:	44A1	70	05		BVS	LOAD	
5930:	44A3	85	FB		STAZ	TPBG	+01 start hoog
5940:	44A5	9D	08	01	STAAZ	\$0108	
5950:	44A8	68			LOAD	PLA	zet label in LABL
5960:	44A9	85	FE		STAZ	LABL	
5970:	44AB	24	F9		LOAK	BITZ	MODE
5980:	44AD	30	1E			BMI	LOAI
5990:	44AF	20	79	45		JSR	INBYT
6000:	44B2	F0	22			BEG	LOAP
6010:	44B4	B0	2C			BCS	LORD
6020:	44B6	20	DF	45	LOAM	JSR	DISPL
6030:	44B9	A0	00			LDYIM	\$00
6040:	44BB	91	FA			STAIY	TPBG
6050:	44BD	20	6D	45		JSR	CHKT
6060:	44C0	20	EA	45	LOANA	JSR	NXTAA
6070:	44C3	10	1D			LORD	
6080:	44C5	A5	F9			LDZ	MODE
6090:	44C7	F0	E2			BEG	LOAK
6100:	44C9	90	E0			BCC	LOAK
6110:	44CB	B0	16			BCS	LOADR
6120:	44CD	20	A0	45	LOAI	JSR	INASC
6130:	44D0	30	10			BMI	LORD
6140:	44D2	C9	03			CMPIM	\$03
6150:	44D4	D0	E0			BNE	LOAM
6160:	44D6	20	79	45	LOAP	JSR	INBYT
6170:	44D9	C5	EC			CMPZ	CHSU
6180:	44DB	D0	05			BNE	LORD
6190:	44DD	20	79	45		JSR	INBYT
6200:	44E0	C5	ED			CMPZ	CHSU
6210:	44E2	18			LORD	CLC	
6220:	44E3	B8			LOADR	CLV	status naar Y
6230:	44E4	08				PHP	
6240:	44E5	68				PLA	
6250:	44E6	A8			LODRR	TAY	
6260:	44E7	A2	FC			LDXIM	\$FC
6270:	44E9	68			LODRB	PLA	
6280:	44EA	95	F0			STAZX	CHSU
6290:	44EC	E8				INX	
6300:	44ED	D0	FA			BNE	LODRB
6310:							
6320:							
6330:							
6340:							
6350:	44EF	20	08	46	LOADRD	JSR	INIPIA
6360:	44F2	A5	F9			LDZ	MODE
6370:	44F4	70	0D			BVS	LODRC
6380:	44F6	A6	FA			LDXZ	TPBG
6390:	44F8	D0	02			BNE	LODRD
6400:	44FA	C6	FB			DECZ	TPBG

De rest van deze routine wordt ook gebruikt door read en write.

6410: 44FC CA	LODRD	DEX		
6420: 44FD 86 FC		STXZ	TPEN	verhuis naar TPEN
6430: 44FF A6 FB		LDXZ	TPBG	+01
6440: 4501 86 FD		STXZ	TPEN	+01
6450: 4503 AA	LODRD	TAX		
6460: 4504 68		PLA		haal mode en begin terug
6470: 4505 85 F9		STAZ	MODE	
6480: 4507 68		PLA		
6490: 4508 85 FA		STAZ	TPBG	
6500: 450A 68		PLA		
6510: 450B 85 FB		STAZ	TPBG	+01
6520: 450D 68		PLA		haal begin status
6530: 450E 29 04		ANDIM	\$04	interrupt test
6540: 4510 D0 04		BNE	LODINT	
6550: 4512 98		TYA		
6560: 4513 29 FB		ANDIM	\$FB	reset I vlag
6570: 4515 A8		TAY		
6580: 4516 98	LODINT	TYA		Y naar status
6590: 4517 48		PHA		
6600: 4518 8A		TXA		
6610: 4519 28		PLP		
6620: 451A 60		RTS		einde alle routines
6630:				
6640:				
6650:				
6660: 451B 20 6D 45	OUTBTC	JSR	CHKT	eerst chksum
6670: 451E 48	OUTBT	PHA		outhex
6680: 451F 4A		LSRA		
6690: 4520 4A		LSRA		
6700: 4521 4A		LSRA		
6710: 4522 4A		LSRA		
6720: 4523 20 27 45		JSR	HEXT	
6730: 4526 68		PLA		
6740: 4527 29 0F	HEXT	ANDIM	\$0F	
6750: 4529 C9 0A		CMPIM	\$0A	
6760: 452B 90 02		BCC	HEXAT	
6770: 452D 69 06		ADCIM	\$06	
6780: 452F 69 30	HEXAT	ADCIM	\$30	
6790: 4531 A0 08	OUTCHT	LDYIM	\$08	karakter out
6800: 4533 84 EE		STYZ	TEMP	
6810: 4535 A0 02	TRY	LDYIM	\$02	
6820: 4537 84 EF		STYZ	TEMP	+01
6830: 4539 BE 66 45	ZON	LDXAY	NPUL	
6840: 453C 48		PHA		
6850: 453D 2C D5 1A	ZONA	BIT	INTR	tijd ?
6860: 4540 10 FB		BPL	ZONA	
6870: 4542 B9 67 45		LDAAY	TIMG	
6880: 4545 8D F4 1A		STA	TIMR	
6890: 4548 AD 82 1A		LDA	PIAPB	zet pia B om
6900: 454B 49 80		EDRIM	\$80	
6910: 454D 8D 82 1A		STA	PIAPB	
6920: 4550 CA		DEX		
6930: 4551 D0 EA		BNE	ZONA	
6940: 4553 68		PLA		
6950: 4554 C6 EF		DECZ	TEMP	+01
6960: 4556 F0 05		BEQ	SETZ	
6970: 4558 30 07		BMI	ROUT	
6980: 455A 4A		LSRA		
6990: 455B 90 DC		BCC	ZON	
7000: 455D A0 00	SETZ	LDYIM	\$00	
7010: 455F F0 D8		BEQ	ZON	
7020: 4561 C6 EE	ROUT	DECZ	TEMP	
7030: 4563 D0 D0		BNE	TRY	
7040: 4565 60		RTS		

subroutines gebruikt door load en save i/o

7050:	4566	02		NPUL	=	\$02	aantal laag
7060:	4567	C3		TIMG	=	\$C3	tijd laag
7070:	4568	03			=	\$03	
7080:	4569	7E			=	\$7E	
7090:							
7100:	456A	20	79 45	CHKIN	JSR	INBYT	
7110:	456D	48		CHKT	PHA		checksum adjust
7120:	456E	18			CLC		
7130:	456F	65	EC		ADCZ	CHSU	
7140:	4571	85	EC		STAZ	CHSU	
7150:	4573	90	02		BCC	CHKTA	
7160:	4575	E6	ED		INCZ	CHSU	+01
7170:	4577	68		CHKTA	PLA		
7180:	4578	60			RTS		
7190:							
7200:	4579	20	7E 45	INBYT	JSR	TPASC	haalt hex binnen
7210:	457C	B0	20		BCS	TPASJ	geen hex
7220:	457E	0A		TPASC	ASLA		
7230:	457F	0A			ASLA		
7240:	4580	0A			ASLA		
7250:	4581	0A			ASLA		
7260:	4582	85	EE		STAZ	TEMP	
7270:	4584	20	A0 45		JSR	INASC	
7280:	4587	C9	2F		CMPIM	\$2F	
7290:	4589	F0	13		BEQ	TPASJ	eind
7300:	458B	90	10		BCC	TPASI	geen hex
7310:	458D	C9	47		CMPIM	'G	
7320:	458F	B0	0C		BCS	TPASI	geen hex
7330:	4591	E9	3F		SBCIM	\$3F	
7340:	4593	90	02		BCC	TPASA	
7350:	4595	69	08		ADCIM	\$08	
7360:	4597	29	0F	TPASA	ANDIM	\$0F	
7370:	4599	05	EE		ORAZ	TEMP	
7380:	459B	C8			INY		
7390:	459C	60			RTS		C=0
7400:	459D	C8		TPASI	INY		
7410:	459E	38		TPASJ	SEC		
7420:	459F	60			RTS		C=1
7430:							
7440:	45A0	A9	40	INASC	LDAIM	\$40	haal 8 bits
7450:	45A2	20	A7 45	INCHA	JSR	INPULS	
7460:	45A5	90	FB		BCC	INCHA	
7470:	45A7	48		INPULS	PHA		
7480:	45A8	20	AE 45		JSR	INPUS	
7490:	45AB	68			PLA		
7500:	45AC	6A			RORA		
7510:	45AD	60			RTS		
7520:	45AE	2C	82 1A	INPUS	BIT	PIAPB	wacht op puls
7530:	45B1	10	FB		BPL	INPUS	
7540:	45B3	AD	D4 1A		LDA	TIRD	
7550:	45B6	A0	FF		LDYIM	\$FF	
7560:	45B8	8C	F6 1A		STY	TIMR	+02
7570:	45BB	A0	13		LDYIM	\$13	
7580:	45BD	88		INPA	DEY		
7590:	45BE	10	FD		BPL	INPA	
7600:	45C0	2C	82 1A	INPB	BIT	PIAPB	
7610:	45C3	30	FB		BMI	INPB	
7620:	45C5	38			SEC		
7630:	45C6	ED	D4 1A		SBC	TIRD	tijd berekening
7640:	45C9	8C	F6 1A		STY	TIMR	+02
7650:	45CC	A0	06		LDYIM	\$06	
7660:	45CE	88		INPC	DEY		
7670:	45CF	D0	FD		BNE	INPC	
7680:	45D1	60			RTS		

```

7690:
7700: 45D2 A0 36      DISER LDYIM $36      error
7710: 45D4 D0 06      BNE  DISPA
7720: 45D6 A0 69      DISSY LDYIM $69      sync
7730: 45D8 D0 02      BNE  DISPA
7740: 45DA A0 55      DISOK LDYIM $55      ok
7750: 45DC 8C 80 1A   DISPA STY  PIAPA
7760: 45DF A8          DISPL TAY
7770: 45E0 AD 82 1A   LDA  PIAPB      andere display
7780: 45E3 49 02      EORIM $02
7790: 45E5 8D 82 1A   STA  PIAPB
7800: 45E8 98          TYA
7810: 45E9 60          RTS
7820:
7830:
7840:      de subroutines hierna worden door alle
7850:      vier de i/o routines gebruikt.
7860: 45EA A5 FA      NXTAA LDAZ  TPBG      vergelijk TPBG met TPEN
7870: 45EC C5 FC      CMPZ  TPEN
7880: 45EE A5 FB      LDAZ  TPBG      +01
7890: 45F0 E5 FD      SBCZ  TPEN      +01
7900: 45F2 E6 FA      NXTA  INCZ  TPBG      verhoog TPBG
7910: 45F4 D0 02      BNE  NXTB
7920: 45F6 E6 FB      INCZ  TPBG      +01
7930: 45F8 AD 80 1A   NXTB  LDA  BRKY      break test
7940: 45FB 60          RTS
7950:
7960: 45FC A9 00      INIVIA LDAIM $00      reset via
7970: 45FE 8D 0C 18   STA  VIAPCR
7980: 4601 8D 03 18   STA  VIAPAD
7990: 4604 8D 01 18   STA  VIAPA
8000: 4607 60          RTS
8010: 4608 AD 82 1A   INIPIA LDA  PIAPB      reset pia
8020: 460B 29 7F      ANDIM $7F
8030: 460D 09 7E      ORAIM $7E
8040: 460F 8D 82 1A   STA  PIAPB
8050: 4612 A9 7F      LDAIM $7F
8060: 4614 8D 80 1A   STA  PIAPA
8070: 4617 8D 81 1A   STA  PIAPAD
8080: 461A 8D 83 1A   INPIA STA  PIAPBD
8090: 461D 60          RTS
8100:
8110: 461E 20 44 46   STSAV JSR  STSA      set pia save
8120: 4621 A9 FF      LDAIM $FF
8130: 4623 D0 F5      BNE  INPIA      bit 7 pia B outout
8140: 4625 20 FC 45   STRED JSR  INIVIA      set via read
8150: 4628 A9 70      LDAIM $70
8160: 462A 8D 01 18   STA  VIAPA
8170: 462D 8D 03 18   STA  VIAPAD
8180: 4630 20 08 46   STLOD JSR  INIPIA      set pia load
8190: 4633 A9 33      LDAIM $33      groene led
8200: 4635 8D 82 1A   STA  PIAPB
8210: 4638 A9 20      LDAIM $20
8220: 463A D0 12      BNE  WAIT
8230: 463C 20 FC 45   STWRT JSR  INIVIA      set via write
8240: 463F A9 0A      LDAIM $0A
8250: 4641 8D 0C 18   STA  VIAPCR
8260: 4644 20 08 46   STSA  JSR  INIPIA
8270: 4647 A9 5E      LDAIM $5E      rode led
8280: 4649 8D 82 1A   STA  PIAPB
8290: 464C A9 80      LDAIM $80
8300: 464E 48          WAIT  PHA          wacht routine
8310: 464F A9 FF      LDAIM $FF
8320: 4651 20 5D 46   WAA  JSR  WAJ
8330: 4654 E9 01      SBCIM $01
8340: 4656 B0 F9      BCS  WAA
8350: 4658 68          PLA
8360: 4659 E9 00      SBCIM $00
8370: 465B B0 F1      BCS  WAIT
8380: 465D 60          WAJ  RTS
8390:

```

FORTH

=====

Reinier Kleipool
Rotterdamseijweg 122
3042 AS Rotterdam
Tel.: 010 - 154025 (week-ends)

FORTH wordt over het algemeen gezien als het buitenbeentje van de computer-talen. Dit komt, omdat FORTH in zo veel dingen afwijkt van "conventionele" talen. Omdat FORTH nu ook draait op onze 6502-systemen, is het goed er wat aandacht aan te besteden.

Een typisch FORTH-systeem heeft disc's. Dit, omdat FORTH niet gebonden is aan het 64K bereik van een micro. Voor FORTH ziet het geheugen er uit als een grote verzameling "blocks". Een block kan een stuk tekst bevatten, of data. Elk FORTH programma wordt in zo'n block geschreven.

U zult waarschijnlijk wel weten dat je FORTH kan uitbreiden met eigen instructies. Eigenlijk doe je dat de hele tijd, want elk programma is op zich een nieuwe instructie in FORTH.

Meestal bestaat het machinetaalprogramma van FORTH uit drie sub-programma's, namelijk: FORTH, de editor en eventueel een assembler. FORTH is het gedeelte dat de programma's uitvoert. De editor helpt bij het maken van nieuwe programma's. De assembler kan handig zijn voor snelle toepassingen.

Omdat iedereen zijn eigen woorden (het FORTH-jargon voor instructie) kan maken (en die een eigen naam kan geven) zou niemand op andermans systeem kunnen werken. Daarom zijn er standaarden gemaakt, die aangeven welke woorden er minimaal in FORTH moeten zitten, en welke namen ze moeten hebben (bijvoorbeeld: de FIG-FORTH en FORTH-79 standaarden). Verder is er ook een standaard afspraak over de vorm van het commentaar van nieuwe woorden.

Bijvoorbeeld:

SWAP (N1 N2 -- N2 N1)
Verwisselt de bovenste twee stack-waarden.

{SWAP} is de naam van het woord. Hetgeen tussen haakjes staat geeft aan wat er met de stack gebeurt (wat de stack precies is, komt later nog aan de orde). Met deze speciale notatie kan iedereen het nieuwe woord begrijpen en gebruiken. De lijst waarop alle woorden met commentaar staan heet een "glossary".

Na deze algemene kennismaking moeten we maar eens aan het werk gaan! Het woord "stack" is al gevallen. Dit is geen toeval, want bij FORTH draait alles om die stack. De stack in FORTH wordt niet gebruikt om return-adressen te bewaren, maar om te rekenen (net als bij de reken-

DE 6502 KENNER

machines van Hewlet Packard). Om twee getallen op te tellen moeten ze eerst op de stack staan. Bijvoorbeeld: {5 6 + .} {+} is het FORTH-woord voor optellen (ja, logisch!). {.} is het FORTH-woord voor druk op beeldscherm af. De spaties scheiden getallen en FORTH-woorden. Pas dus goed op! We kunnen nu het FORTH-woord {+} eens documenteren.

```
+ (N1 N2 -- SOM)
```

Tel de bovenste twee stack-waarden op en laat het antwoord op de stack achter.

(Na "--" staat wat op de stack achterblijft na het woord).

Een gevolg van deze methode is wel dat alles omgekeerd moet worden: 2 + 3 wordt dus 2 3 + . Dit lijkt een domme methode, maar als U dit eenmaal door hebt, weet U al meer dan de helft van FORTH! Want alle woorden werken zo.

Goed, dan nu een programma in FORTH: U toetst twee getallen in, en Uw super-snelle hyperintelligente computer berekent de som! We noemen het nieuwe woord "SOM".

```
SOM (N1 N2 --)
```

Bereken de som en druk deze af.

(Na "--" staat niets, er wordt dus niets op de stack achter gelaten) Om het te laten werken moet U eerst Uw FORTH laden en starten. Na het laden start U op \$2000. FORTH antwoordt met FIG-FORTH 1.1. Dan toetst U in {BREAK <RETURN>}. Dit, om de NMI- en IRQ-vectoren te zetten. Dan {EMPTY-BUFFERS <RETURN>} om de editor te initialiseren. Nu is FORTH voor gebruik gereed.

Nu maken we {SOM}. Toets in: {: SOM + . ; <RETURN>} (alleen wat tussen haakjes staat). Let op de spaties tussen de woorden! FORTH antwoordt nu met "OK". Het woord bestaat nu.

Als U dit wilt controleren kunt U {VLIST <RETURN>} intoetsen. FORTH zal dan een list geven van alle woorden die het op dat moment kent.

```
VLIST (--)
```

Geef de lijst van FORTH-woorden die nu bestaan.

Om het programma nu te gebruiken, toetsen we: {5 6 SOM <RETURN>}

FORTH antwoordt dan met: 11 OK.

GEFELICITEERD EUREKA !!

Om een woord dus te "run-nen" (om eens een Basic-term te gebruiken) is het genoeg om de naam in te toetsen met <RETURN>, de CR-toets dus.

Vanaf het begin heeft U dus al vier woorden gebruikt:

{BREAK}, {EMPTY-BUFFERS}, {VLIST} en {SOM}.

In het programma kwamen de woorden {:} en {;} voor. Deze woorden zorgen dat wat na {:} komt als nieuw woord wordt toegevoegd, totdat {;} wordt tegen gekomen. Het is op deze wijze dus mogelijk FORTH-program-

DE 6502 KENNER

ma's te schrijven. Maar dit is niet ideaal. Want, als het programma niet mocht werken, dan kan het niet worden verbeterd. Immers, het is dan al gecompileerd (door {;} en {;}). Hier biedt de editor uitkomst. Maar eerst lijkt het me verstandig om wat meer woorden te leren kennen.

(STACKMANIPULATIE)

DUP (N -- N N)
Kopieer het bovenste getal op de stack.

OVER (N1 N2 -- N1 N2 N1)
Kopieer het eennabovenste getal op de stack.

ROT (N1 N2 N3 -- N2 N3 N1)
Roteer de drie bovenste stackwaarden.

DROP (N --)
Vernietig de bovenste stackwaarde.

-DUP (0 -- 0)
(N1 -- N1 N1)
Kopieer alleen als <>0.

(REKENEN)

+ (N1 N2 -- SOM)
Bereken de som van twee getallen.

- (N1 N2 -- VERSCHIL)
Bereken het verschil van twee getallen.

* N1 N2 -- PRODUKT)
Bereken het produkt van twee getallen. Spreek * uit als star.

/ (N1 N2 -- QUOTIENT)
Bereken het integer quotient van N1 en N2. Spreek / uit als "slash".

FORTH kan alleen fixed-point rekenen. Dit is echter geen bezwaar, want er is ruimschoots voorzien in hulomiddelen om de getallen als breuken af te drukken.

(IN/UITVOER)

. N1 --)
Druk N1 af in getalstelsel "Base" als integer met teken.
Spreek . uit als "dot".

EMIT (N1 --)
Druk N1 af als ASCII-karakter (vergelijk: CHR\$(N1)).

." (--)

Druk de string volgend aan ." letterlijk af, totdat een " tegengekomen wordt (vergelijk: PRINT"PIET"). Soreek {."} uit als "dot-quote" Let op, {."} is een woord, dus een soatie erna. " is geen woord, het dient slechts om het einde aan te geven. Soatie dus niet verolicht.

CR (--)

Druk een carriage return af op de terminal.

SPACE (--)

Druk een soatie af op de terminal.

SPACES (N --)

Druk N spaties af op de terminal.

(DICTIONARY MANAGEMENT)

FORGET (--)

Verwijder het woord volgend op FORGET uit de dictionary.

COLD (--)

Verwijder alle eigen woorden uit de dictionary.
Het standaard-FORTH blijft dus over.

De Editor.

De editor is er om FORTH-programma's makkelijker te kunnen ontwikkelen. In de editor maak je eerst een tekst, die dan door FORTH gecompileerd zal worden, net zoals bij de Micro-ADE assembler. Allereerst moeten de editor-woorden aan FORTH gekoppeld worden. Dit klinkt ingewikkeld, maar je hoeft alleen {EDITOR (RETURN)} te toetsen en klaar is kees. Als je dan nu VLIST geeft, zul je zien dat er een hoop woorden bij zijn gekomen.

De editor gebruikt de blocks om de tekst op te slaan. Om die reden moeten we eerst een block kiezen om in te werken. Laten we veronderstellen dat we nog geen programma's in de blocks hebben staan. Als we nu bijvoorbeeld {5 CLEAR (RETURN)} intoetsen, dan wordt block 5 geselecteerd, en meteen automatisch schoongemaakt. Met L (RETURN) komt de inhoud op het scherm. Dit moet er zo uitzien:

SCR # 5

0

1

2

enzovoort, tot regel 15. Als het scherm niet leeg is, maar allemaal troep vertoont, heeft U of iets niet goed gedaan, of is er geen geheugen aangesloten op \$6000. Vergeet ook niet om {EMPTY-BUFFERS} te geven want dat kan ook leuke resultaten leveren!

Goed, we hebben nu een vrij scherm (block). Laten we er dan nu iets in zetten. Met {1 P HALLO KOM ER EENS KIJKEN. (RETURN)} hebben we het op regel 1 gezet. Bewijs het maar met {L (RETURN)}!

Met {2 P WAT IK IN MIJN BLOCKJE VIND. (RETURN)} zetten we de rest op de regel 2, enzovoort.

Maar wat is nu een editor zonder dat er fouten kunnen worden verbeterd. Als U nu zorgt voor de fouten, dan zal de editor ze wel oplossen. Met {2 D (RETURN)} bijvoorbeeld wordt regel 2 verwijderd uit het geselecteerde scherm. De andere regels schuiven 1 omhoog. Regel 15 wordt blank. Met {2 T} wordt regel 2 in de regelbuffer (PAD genaamd) gezet. Dit is te zien doordat de editor de regel afdruckt met een pijltje erin: 2 ^WAT IK IN MIJN BLOCKJE VIND . Het pijltje is de regel-cursor. Met {F IN (RETURN)} wordt de cursor achter "IN" gezet. Met {N} vindt de editor de volgende verschijning van "IN", dus: 2 WAT IK IN MIJN BLOCKJE VIN^D .

{TOP} doet de cursor weer terug bovenaan (HOME).
{X PIET} zoekt PIET op in PAD (oftewel de current regel) en verwijdert het.
{C KLAAS} zet vanaf de cursorpositie het woord KLAAS neer.
{3 I} voegt de inhoud van PAD in voor regel 3: de andere regels zakken een regel.
{3 S} voegt een lege regel in voor regel 3.
{4 E} vult regel 4 met spaties, verwijdert de regel niet.
{4 R} vervangt regel 4 met inhoud van PAD.
{B} zet cursor voor het zojuist gevonden woord.
{TILL PIET} verwijdert op current regel vanaf cursor positie tot PIET.

Dit zijn de meest gebruikte editor-commando's. Probeer ze eens rustig met een stukje tekst. Pas op: het is beter om na elke editor-sessie het woord {FLUSH (RETURN)} te geven om te zorgen dat de verbeteringen in PAD ook echt in het geheugen staan (dit is ongeveer hetzelfde als met {EMPTY-BUFFERS}).

We kunnen nu de editor eens gaan gebruiken voor een programma. We selecteren een vrij block. Om gewoon eens te kijken, kun je ook {1 LIST} intoetsen.

Op regel 0 moet commentaar komen over de inhoud van het block (tussen haakjes). Bijvoorbeeld: {0 P (PIET-MET-DE-KORTE-ACHTERNAAM TEST)}.
Op regel 1 komt oer afspraak de definitie van het getalstelsel.
{1 P DECIMAL} (DECIMAL zet de variabele BASE op 10 (HEX op 16)).
{3 P : TEST 10 1 DO I . LOOP :S} (print de getallen 1 tot 10).
Dit is genoeg voor een kort programma. In Basic staat hier eigenlijk FOR X = 1 TO 10:PRINT X:NEXT.

Het woord I haalt de waarde van de loopteller op de stack.
Als het programma staat kunnen we FORTH intoetsen om de editor weer uit te schakelen (loskoppelen). Nu moet FORTH de woorden van het programma nog toevoegen aan de dictionary. Dit gaat door {N LOAD} te doen. Hierbij is N het nummer van het scherm waar de tekst staat. Met een {VLIST} moet U kunnen zien of {TEST} er nu staat.
Dit was een (te??) korte inleiding in het gebruik van deze mooie taal. Mochten er nog vragen zijn (ik hoop het), bel dan eens.
In het week-end als bovengenoemd.

DE 6502 KENNER

Van Maarten Lameij
Prins Clauslaan 81
7242 GN Lochem
Tel.: 05730 - 4427

ontvingen we een FORTH-programma'tie voor mensen met kinderen die vaak starfwerk krijgen. naar een idee uit het Engelse tijdschrift Computing Today.

```
1 LIST
SCR # 1
0 ( TAFELS VAN VERMENIGVULDIGING TOT MAX. 3200 . 1 VAN 2 )
1 DECIMAL
2 VARIABLE VANAF VARIABLE TOT ( VOOR VASTLEGGEN BEREIK TAFELS )
3 : TEKST CR ." DIT PROGRAMMA BEREKENT DE TAFELS VAN " CR
4      ." VERMENIGVULDIGING VANAF EN TOT EEN DOOR " CR
5      ." DE GEBRUIKER OP TE GEVEN WAARDE. " CR CR
6      ." DE GETALLEN MOETEN STEEDS IN EEN " CR
7      ." VIER-CIJFER COMBINATIE WORDEN OPGEGEVEN. " CR
8      ." BV. : 10 OPGEVEN ALS 0010. " CR CR :
9 : TAFEL 1 TOT +! ( EINDWAARDE PLUS 1 VOOR DE DO-LOOP )
10      TOT @ VANAF @ DO 11 1 DO CR I ." X "
11      J . ." = " I J * . LOOP LOOP :
12 --)
13
14
15
OK
2 LIST
SCR # 2
0 ( TAFELS VAN VERMENIGVULDIGING TOT MAX. 3200 . 2 VAN 2 )
1 : INVOER-1 CR ." DE TAFELS MOETEN BEGINNEN BIJ: "
2      KEY 48 - DUP . 1000 * VANAF !
3      KEY 48 - DUP . 100 * VANAF +!
4      KEY 48 - DUP . 10 * VANAF +!
5      KEY 48 - DUP . VANAF +! :
6      ( INVOER EN OMZETTEN BEGINWAARDE )
7 : INVOER-2 CR ." EN EINDIGEN BIJ: "
8      KEY 48 - DUP . 1000 * TOT !
9      KEY 48 - DUP . 100 * TOT +!
10     KEY 48 - DUP . 10 * TOT +!
11     KEY 48 - DUP . TOT +! :
12     ( INVOER EN OMZETTEN EINDWAARDE )
13 : TAFELS TEKST INVOER-1 INVOER-2 TAFEL :
14
15 :S
OK
```

SCHUIVEN EN SPRINGEN

Soeloprogramma voor 6502 comp.
met 7-segment display.

In editie 29 en 30, d.i. dec.
'83 en feb.'84, publiceerde
Roland de Preetere zijn pro-
gramma Schuiven en Springen.
Het programma, dat in princi-
pe geschikt is om op elke
6502-computer te draaien, is
bestemd voor gebruik met een
terminal.

Roland heeft dit programma
aangepast voor gebruik met 7-
segment display. Door alleen
de adressen voor 'scands' en
'getkey' aan te passen is de
routine geschikt voor elke
6502-computer met 7-segm.dis-
play. Het programma neemt de
adressen \$200 t/m \$7F9 in be-
slag, naast de zero-page en
de stackruimte. De source-
listing telt 17 pagina's, in-

clusief inleiding en label-
file. De source-listing is
te verkrijgen door Fl.12.50
over te schrijven op postre-
kening 841433 t.n.v. W.L. v.
Pelt te Krimpen a.d. IJssel.
Of storten op postgiro 3050
van de AMRO-bank aldaar ten
gunste van W.L. van Pelt, nr.
44.11.06.471.
Eurocheque is ook toegestaan.
mits volledig ingevuld.

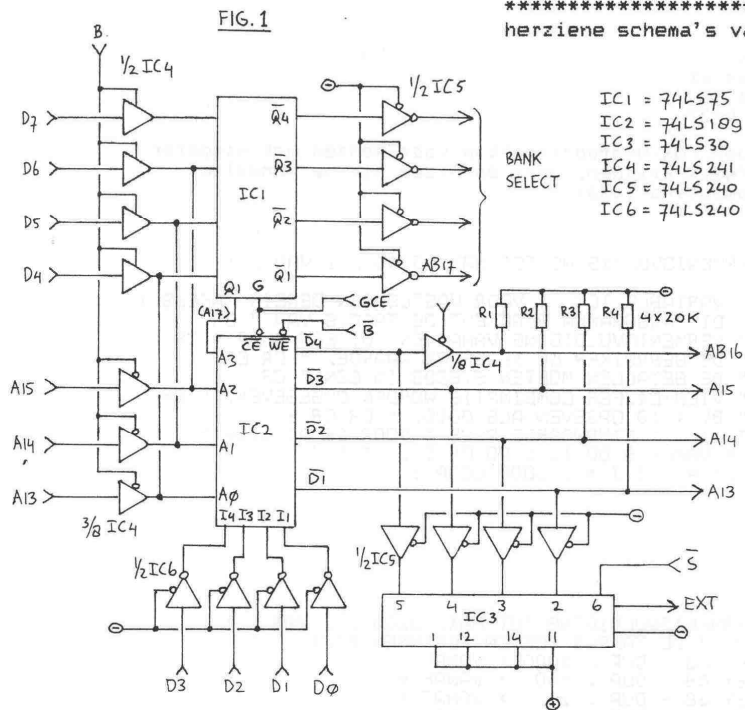
GEVRAAGD:

Wie kan mij helpen aan aan-
sluitgegevens om een elektro-
nische schrijfmachine Olivetti
Praxis 30 om te bouwen tot
printer/terminal?
D. Uittenbosch, Distelvink 29,
3738 SH Maartensdijk, Telefoon
03461 - 3319

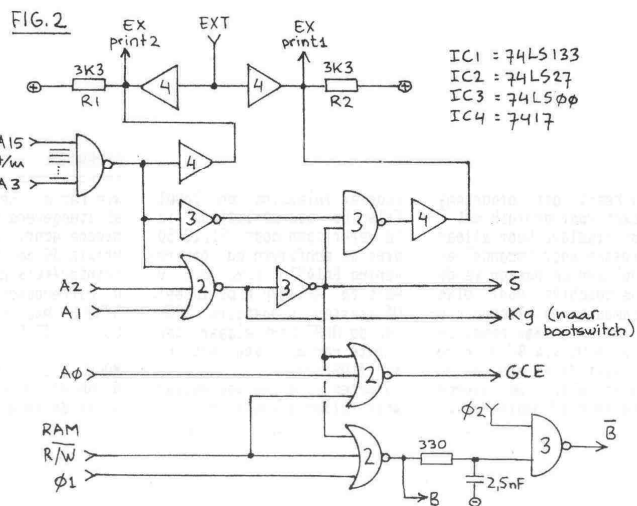
MAAKT U GEBRUIK VAN DE PROCES-
SORKAART VAN ELEKTUUR? Wilt
U dit de redactie dan opgeven?

BOUW EENS EEN MILJONOR !

herziene schema's van editie 30. pag.15



Aan de tekentafel:
Fridus Jonkman



```

1 REM BEREKENING AUTOKOSTEN
2 REM DOOR
3 REM WILL CUIJPERS
4 REM MOSSELLAAN 63
5 REM 4691 KE THOLEN
6 REM 01660-3566
7 REM VOLGENDE VARIABELEN ZIJN GEBRUIKT:
8 REM G=GELDWAARDE K=KILOMETERS
9 REM L=LITERS BRANDSTOF V=VASTE KOSTEN
10 REM R=REPARATIES S=BELASTINGEN
11 REM V=VERZEKERINGEN M=MAANDWAARDE
12 REM J=JAARWAARDE A=BEGINWAARDE
13 REM E=INGEGEVEN WAARDE
15 REM NAAR EEN IDEE UIT CHIP SPECIAL NO 13
20 DIMR(12),V(12),S(12)
30 GOTO 90
40 REM OP 0D45 STAAT BIJ MIJ DE CLS ROUTINE
50 CLS=USR("&"0D45",0):RETURN
60 REM OP 0400 STAAT BIJ MIJ DE ROUTINE OM BASICTABELLEN TE SAVEN
70 REM OP 045E STAAT BIJ MIJ DE ROUTINE OM BASICTABELLEN TE LAZEN
90 GOSUB 50:PRINTCHR$(10)
100 INPUT"BEGIN OP EERSTE MAAND VAN HET JAAR ?":X$
110 IF X$("&"JA")THEN 140
120 INPUT"KILOMETERSTAND 31 DECEMBER ?":KA
130 GOTO 200
140 INPUT"ONDER WELK ID-NUMMER STAAN DE OUDE GEGEVENS OP TAPE?":D
150 POKE 6777,D
160 PRINT"START RECORDER OP PLAY.":PRINT
170 PRINT:PRINT:PRINT"ID-NUMMER ":D;" WORDT GELEZEN"
180 Q=USR("&"045E",0)
185 PRINT"GELADEN !!!"
190 PRINT:PRINT
200 GOSUB 50:PRINT CHR$(10)
210 INPUT"BEREKENING VOOR DE MAAND (GEEF NO VAN DE MAAND JAN=1)":M
220 INPUT"KILOMETERSTAND ":KE
230 INPUT"HOEVEEL BRANDSTOF GETANKT ":LE
240 INPUT"BRANDSTOFKOSTEN PER LITER ":BE
250 BM=BM+BE:LM=LM+LE
260 INPUT"NOG EEN KEER GETANKT ":X$
270 IF X$="JA" THEN 230
280 KM=KE-KA:VD=KM/LM
290 IF KM<0 THEN GOSUB 1000:GOTO 200
300 INPUT"VASTE MAANDELIJKSE KOSTEN (STALLING) ":FM:PRINT
400 PRINT"TIJDSAFHANKELIJKE KOSTEN"
410 INPUT"ZOALS REPARATIES EN SERVICEKOSTEN ":RE
420 RM=RM+RE
430 INPUT"NOG MEER REPARATIES ":X$
440 IF X$="JA" THEN 400
450 INPUT"OVER HOEVEEL MAANDEN VERDELEN ":VT
460 RM=INT(RM/(VT+0.0001)*100+.5)/100
470 INPUT"WEGENBELASTING ?":SE
471 INPUT"IS DIT PER KWARTAAL OF JAAR ?":X$
473 IF LEFT$(X$,2)="KW"THEN SM=INT(SE/3*100+.5)/100:GOTO 480
474 IF LEFT$(X$,2)="JA"THEN SM=INT(SE/12*100+.5)/100:GOTO 480
475 GOSUB 1000:GOTO 470
480 INPUT"VERZEKERING ?":VE
481 INPUT"IS DIT PER MAAND, KWARTAAL, HALFJAAR OF JAAR ?":X$
482 IF LEFT$(X$,2)="MA"THEN VM=VE:GOTO 500
483 IF LEFT$(X$,2)="KW"THEN VM=INT(VE/3*100+.5)/100:GOTO 500
484 IF LEFT$(X$,2)="HA"THEN VM=INT(VE/6*100+.5)/100:GOTO 500
485 IF LEFT$(X$,2)="JA"THEN VM=INT(VE/12*100+.5)/100:GOTO 500
490 GOSUB 1000:GOTO 480
500 K=M
510 FOR J=1 TO 12
520 V(K)=V(K)+VM:S(K)=S(K)+SM
530 K=K+1
540 IF K>12 THEN K=1
550 NEXT J
560 K=M:FOR J=1 TO VT
570 R(K)=R(K)+RM:K=K+1
580 IF K>12 THEN K=1
590 NEXT J
700 GOSUB 50:PRINTCHR$(10)TAB(10)"AFREKENING VOOR DE MAAND ":M
710 PRINTTAB(10)"*****"
720 PRINT"GEREDEN KILOMETERS ":KM

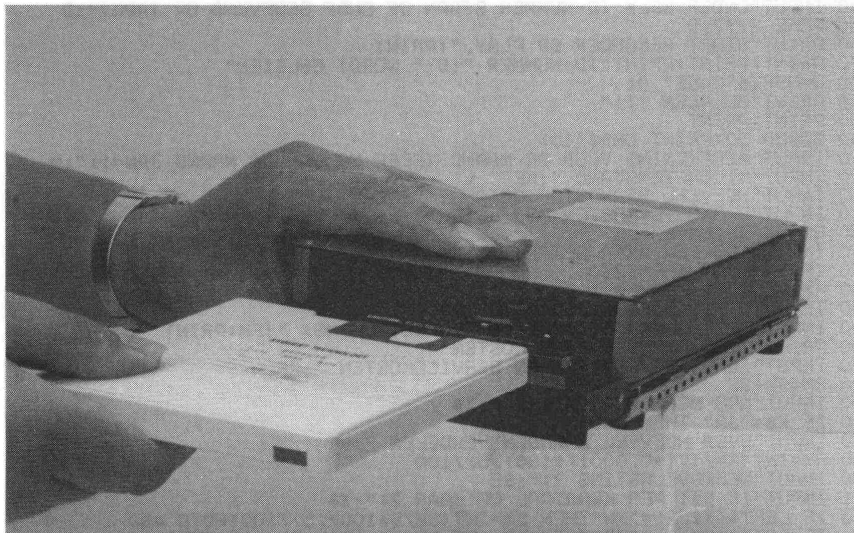
```

```

730 PRINT"GEMIDDELD VEBRUIK      :1 OP ":VD
740 GM=GM+BM+FM+R(M)+S(M)+V(M):KJ=KJ+KM
745 R(M)=O:V(M)=O:S(M)=O
750 PRINT"KOSTEN IN DE MAAND    :":GM:" DFL":PRINT
760 GJ=GJ+GM:GD=GJ/M:LJ=LJ++LM:LD=KJ/LJ
780 PRINT"GEMIDDELTE MAANDELIJKSE KOSTEN  :":GD:"DFL"
790 PRINT"TOTALE BRANDSTOFHOEVEELHEID    :":LJ:" LITERS"
800 PRINT"JAARGEMIDDELTE          :1 OP ":LD
810 PRINT"KOSTEN PER KILOMETER        :":GJ/KJ:" DFL"
900 PRINT:PRINT:PRINT:INPUT"ONDER WELK ID-NUMMER WILT U DEZE GEGEVENS OPSLAAN "
:D
910 POKE 6777,D
920 PRINT:PRINT"CASSETTE OP 'RECORD' EN RETURN"
930 Q=USR(&"0400",0)
940 PRINT"GESAVED"
950 END
1000 GOSUB 50:PRINT"FOUTIVE INVDER.NOGMAALS":RETURN
OK

```

MARKTINFO



Micro Plan introduceert een winchester drive met removable cartridge van het fabrikaat Syquest. De drive heeft een ongeformateerde opslagcapaciteit van 6.380.000 tekens en heeft ten aanzien van snelheid en betrouwbaarheid specificaties exact gelijk aan het type met vaste schijf, een uitvoering die ook leverbaar is. De drive's zijn leverbaar voor o.a. APPLE, hebben afmetingen kleiner dan een standaard slim-line flop en kunnen, als tenminste de standaard behuizing dit toelaat, ingebouwd worden in de centrale processor behuizing. De drive's worden ook compleet ingebouwd in een omkasting geleverd, met per omkasting een of twee drive's met vaste of removalble cartridge. De uitvoering met dubbele drive's biedt een vol-

ledige back-up mogelijkheid van drive naar drive, het removable zijn van het opslagmedium betekent dat data en programmatuur buiten de machine bewaard worden. De systemen worden geheel compleet met bijbehorende software geleverd. De drive's zijn schokbestendig zowel tijdens transport als bedrijf en temperatuurongevoelig. Als globale prijsindicatie ingebouwd per drive in computer of omkasting geldt ca. Fl. 5.000,00. De prijs van een losse cartridge bedraagt ca. Fl. 300,00 en dat dan voor een opslagcapaciteit van 5.000.000 tekens.

Voor nadere informatie kan men zich wenden tot:
Micro Plan,
Zwaanhoefstraat 12,
4702 LC Roosendaal
Tel.: 01650 - 35616.

JUNIOR AAN 8" OF 5" FLOPPY

In editie 27 van augustus 83 maakte Koen van Nieuwenhove uit Leuven (Belgie) middels een artikel bekend dat zijn disk operating system met de controller FD 1771 klaar was voor zijn 8" drives en ook voor 5". Het systeem, dat in single density draait, evenals dat van DHIO en PROTON, maakt gebruik van de geheugenplaatsen \$1900-\$1904. Deze adressen kunnen worden vrijgemaakt zoals hij al bekend maakte in editie 25 van februari 1983. De hard- en software is ontleend aan artikelen uit BYTE van mei en juni 1981 en het boek 'A Floppy disk operating system for the 8080' van Kenneth B. Welles, uitgegeven door BYTE Books. Cassettebesturing zal mogelijk blijven, terwijl de DOS al draait met de assembler van Moser, de KB9-Basic en Pascal. Koen hoopt in samenwerking met clubleden ook Micro-Ade, Forth etc. aan te passen te krijgen. Het 8" DOS maakt gebruik van de configuratie 77 tracks, 26 sectoren per track en 128 bytes per sector. Het 5" DOS heeft 40 tracks. IBM-compatibel dus.

Om een indruk te geven van de bevelen, hier een opsomming: Save, Run, Limits, Go, Quit, Execute, Copy, Delete, Directory, Exit, Free, List, Read, Rename, Trace, Store, Format, Random, Undelete, Memtest, Validate, Backup, Key (sleutel toekennen aan disk), Lock (beveiligen van een file). Het gepubliceerde artikel is verkrijgbaar door storting van Fl. 10,- op postrekening 841433 t.n.v. W.L. van Pelt te Krimpen a.d. IJssel. Een Eurocheque, mits volledig in gevuld, mag ook. Of overmaken op de AMRO-bank te Krimpen a.d. IJssel t.g.v. bankrekening 44.11.06.471 t.n.v. W.L. van Pelt.

FORTH voor JUNIOR is op band verkrijgbaar door Fl. 12.50 over te schrijven.

VERKRIJGBAARHEID BASIC

Microsoft Basic KB9 of KB9 P

Johnson Computer
P.O. Box 523
Medina, OHIO 44256

Itty Bitty Computers
P.O. Box 23189
San Jose, CA 95153

MSB-Verlag
Moangoldstrasse 10
D 7778 Markdorf
West Duitsland

Basic Compiler CMB 8032 en
Commodore 64

Gesellschaft für Math. Beratung & Software Dr. August C. Quint m.b.h.
Kaiser-Friedrich Ring 55
D 6200 Wiesbaden
Tel.: (06121) 842686

5.5K zelf-ontwikkelde Basic

J.J. Janssen
Gerardsweg 30
6525 RT Nijmegen

ASSEMBLER MOSER VERKRIJGBAAR

KIM-1 versie op band

Ing.büro Schröder
Echternachlaan 161
5625 KC Eindhoven
Tel.: 040 - 421821

VERKRIJGBAARHEID PASCAL

Pascal op floppy voor C 64

Micro-Delta Holland BV
Kaarder 100
1625 TN Hoorn
Tel.: 02290 - 41555

VERKRIJGBAARHEID FOCAL

Fast Focal, een Basic-achtige interpreter met veel wiskundige functies, afkomstig van The 6502 Program Exchange, 2920 West Moana, Reno, NV 89509, America, bij:

J.J. Janssen
Gerardsweg 30
6525 RT Nijmegen.
VERMELD DE NAAM VAN UW CLUB!

CASSETTES VOOR JUNIOR

NR.1 FL.12,50

Elektuur Eorum Programmer + Aanvulling John van Sorang. 4-Kolom printer, Conversie Dag naar week, Zenuwslag, Music Box, Dokatimer, Automaregister Uitlezing, Lotto.

NR.2 FL.12,50

Frequentieteller met 6522. List Mnemonics (7-segm. disp) Shift, Basiccode met de SC/MP interface, RTTY, Easy Editing Supporter (PME), Single Step Debug, Wallbreaker

NR.3 FL.1250

Renumber SYM-Basic voor de Junior. Append SYM-Basic v. Junior, Basic Compressor v. Junior, NARCHA (nieuwe print routine op \$1634). Adress (Micro-Ade verplaatsen). Mas termind R. Vissers, Outclk (Klok met 8 schakelporten).

Programma's in de cassettebibliotheek zijn afkomstig van onze clubleden of bewerkingen uit het First Book of KIM, of anderszins. Alle rechten van de programma's blijven aan de auteurs voorbehouden. De kosten van de cassette zijn alleen maar kostendekkend en niet bedoeld om de programma's te kopen. DE CLUB IS EN WORDT GEEN HANDEL!

SYM-BASIC CASSETTES v JUNIOR

ook in te lezen met KB9

NR.1 FL.12,50

A. Bruto-netto/Netto-bruto loonberekening B. Hypotheekberekening (% van Hintum) C. Kalender D. Eeuwkalender E. Magische vierkanten F. Hoofdrekenen G. Bunny, H. Fruitautomaat (G.v. Roekel) I. Olifant J. Zeegevecht K. Poker L. Maanlander M. Afstandsspel N. Bioritmes

NR.5 FL. 12,50

Service Rapporten Bestand.

SYM-BASIC CASSETTES v JUNIOR

NR.2 FL.12,50

A. Maanlanding B. Olifant C. Betal Raden D. Zeegevecht E. Claire (adresbestand) F. Poker G. Vier op een rij H. Zoek het woord I. Woordspel J. Faktuur (Fr. Smeehuisen: zie hiervoor ook de publikatie van Ruud Ushoff: (editie 28).

NR.3 FL.17,50

A. Superstartrek (nederlands) met handleiding

NR.4 FL.1250

A. Othello (cursoreestuurd) B. Afstandsspel C. Safari D. Letters schrijven (M. van Hintum) E. Letters/cijferspel F. Getal verwisselen G. Autodoolhof H. "Dammen" J. X-spel K. Vier op een rij L. Puzzel M. Uitgebreide dis assembler (Gerard v. Roekel)

COMAL*, KGN

Do de landelijke bijeenkomst introduceerde Ruud Ushoff de zelf ontwikkelde COMAL*KGN, welke vrijwel gelijk is aan COMAL-80. Er is sprake van gestructureerd programmeren als in PASCAL, maar in de op tekenstijl van BASIC. Zie de pag.2 in editie 30. Voor de JUNIOR op band verkrijgbaar met handleiding voor Fl.22,50 op giro 841433 t.n.v. W.L. van Pelt te Krimpen a.d. IJssel.

CFAST5

Tape Management for digital recording. The program provides JUNIOR to tape management of a digital cassetterecorder over the VIA 6522. Fastness: 6000 baud. Base functions: GET, SAVE, REWIND, INITIALISE and ABORT (jump to PM). The command-set is capable to be extended. 14 pag. incl. label-file. Price: Fl.12,50 nett incl. costs of transm./admin. To be payed on postal account 841433 of Mr.W.L. van Pelt at Krimpen a.d. IJssel.