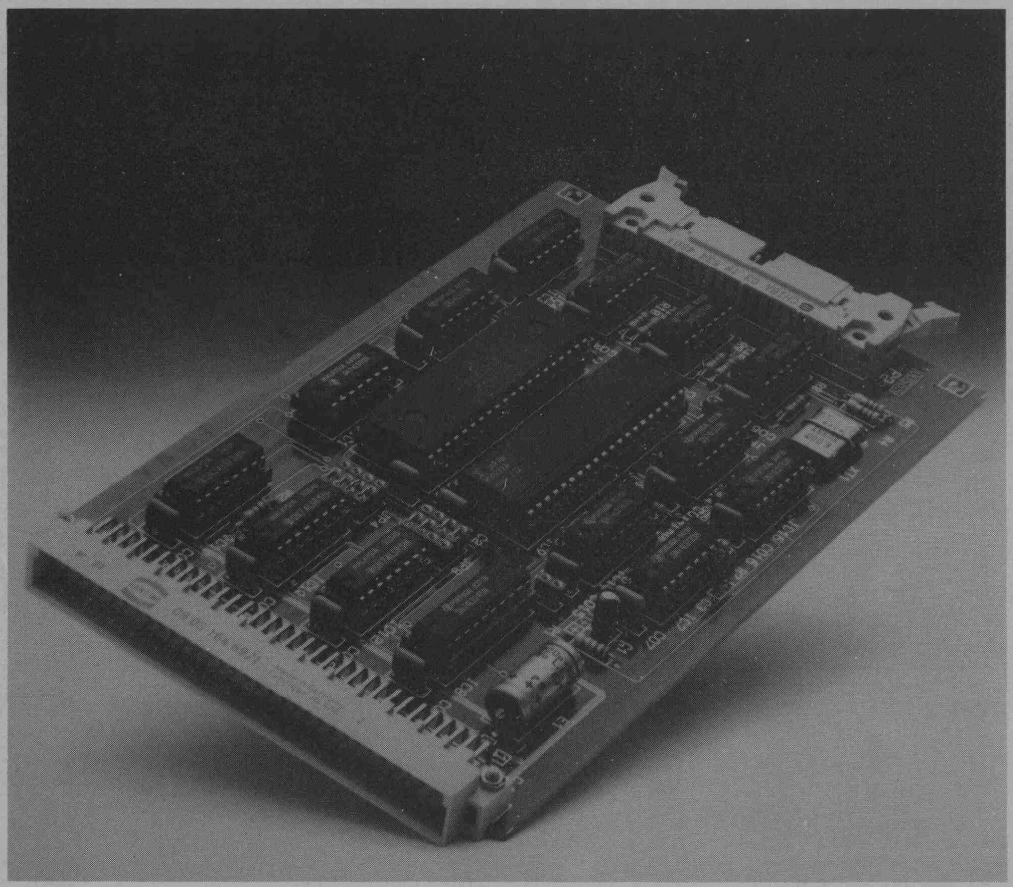
DE 6502 KENNER 48



Elfde Jaargang, Nr. 1
Februari 1987

DE 6502 KENNER

** DE 6502 KENNERS ** — EEN CLUB VOOR 6XXXX GEBRUIKERS

De vereniging heeft leden in Nederland, Belgie, Duitsland, Frankrijk, Spanje, Portugal, Amerika, Zambia, Denemarken. Het doel van de vereniging is: het bevorderen van de kennisuitwisseling tussen gebruikers van 6XXXX-computers, als COMMODORE-64, APPLE, CHE-1, PEARCOM, AIM-65, SYM, PET, BBC ATARI, VIC-20, BASIS 108, PROTON-computers, ITT-2020, OSI, ACC 8000, ACORN ELECTRON, SYSTEM 65, PC-100, PALLAS, MINTA FORMOSA, ORIC-1, STARLIGHT, CV-777, ESTATE III, SBC65/68, KIM, NCS, KEMPAC SYSTEM-4, Elektuur-computers (JUNIOR, en de OCTOPUS), LASER, maar ook 6800, 6809 en 68000-computers. De kennisuitwisseling wordt o.a. gerealiseerd door 6 maal per jaar DE 6502 KENNER te publiceren, door het houden van landelijke clubbijeenkomsten, door het instandhouden van een cassette-bibliotheek en door het verlenen van paperware-service. Regionale bijeenkomsten worden door de leden georganiseerd.

Verschijningsdata DE 6502 KENNER 1985	Bijeenkomsten van de club
derde zaterdag van februari, april, juni, augustus, oktober, december.	derde zaterdag van januari, maart, mei, september, november.
Inlichtingen over de regio-bijeenkomsten:	**Redaktie-adres en informaties over paperware etc.**
Gerard van Roekel Van der Palestraat 11 - C 3135 LK Vlaardingen Tel.: 010 - 351101	Willem L. van Pelt Jacob Jordaensstraat 15 2923 CK Krimpen/IJssel. Tel.: 01807 - 19881

De vereniging is volledig onafhankelijk, is statutair opgericht en ingeschreven bij de Kamer van Koophandel en Fabrieken voor Hollands Noorderkwartier te Alkmaar, onder nummer 634305.

Voorzitter: Rinus Vleesch-Dubois Fl. Nightingalestraat 212 2037 NG Haarlem Tel.: 023 - 330993	**Sekretaris:** Gert Klein Diedenweg 119 6706 CM Wageningen Tel.: 08370 - 23646
Penningmeester: John F. van Sprang Tulp 71 2925 EW Krimpen/IJssel. Tel.: 01807 - 20589	**Redaktie DE 6502 KENNER:** Willem L. van Pelt Jacob Jordaensstraat 15 2923 CK Krimpen/IJssel. Tel.: 01807 - 19881

Leden:

Adri Hankel	(05490 - 51151)	Hardware/software/DOS65
Erwin Visschedijk	(05490 - 71416)	Hardware/software/DOS65
Gert van Opbroek	(01729 - 8636)	
Nico de Vries	(010 - 302239)	Hardware/software/PET

Erevoorzitter: Siep de Vries
Ereleden : Mw. H. de Vries - Van der Winden
Anton Mueller
Lidmaatschap : Hfl. 45,= per kalenderjaar, postrekening 3757649 t.n.v. Penningmeester KIM Gebruikers Club Ned., Krimpen/IJssel.

Advertenties : Tarieven op aanvraag bij de redaktie.

** DE 6502 KENNER ** — EEN BLAD VOOR 6XXXX GEBRUIKERS

DE 6502 KENNER is een uitgave van de KIM Gebruikers Club Nederland. Het blad wordt verstrekt aan leden van de club. DE 6502 KENNER wordt van kopij voorzien door leden van de club, bij de opzaak van een publikatie bijgestaan door de redaktie. De inzendingen van programma's dienen voorzien te zijn van kommentaar in de listings en zo mogelijk door een inleiding voorafgegaan. Publikatie van een inzending betekent niet dat de redaktie of het bestuur enige aansprakelijkheid aanvaardt voor de toepassing ervan. De inzendingen kunnen geschieden in assembly-source-listings, in Basic, in Basicode, Forth, Focal, Comal, Pascal, Fortran, Cobol, Logo Elan, etc. etc.
De leden schrijven ook artikelen over de door hen ontwikkelde hardware en/of aanpassingen daarop. Zij schrijven tevens artikelen van algemene aard of reageren op publikaties van andere inzenders.

DE 6502 KENNER IS EEN BLAD VAN EN DOOR DE LEDEN

Micro-ADE Assembler/Disassembler/Editor is een produkt van Micro Ware Ltd., geschreven door Peter Jennings en bestemd voor alle 6502-computers. De KIM Gebruikers Club Ned. heeft de copyrights verworven nadat ons lid Sebo Woldringh de 4 K KIM-1 versie uitbreidde tot 8 K KIM-1 versie. Adri Hankel paste deze aan voor de JUNIOR. Willem L. van Pelt stelde een nieuwe 8 K source-listing voor de JUNIOR samen.
De implementatie op andere systemen dan de KIM-1 en JUNIOR kan eenvoudig gebeuren door het aanpassen van de I/O-adressen, welke in de source-listing gemakkelijke te vinden zijn

FATE Format-lister/cond. Assembler/Tape-utilities/Editor is de door ons lid Rob Banen geschreven source-listing van een 12 K universeel systeem voor de JUNIOR-computer aan de hand van het universele disk operating system van de fa. Proton Electronics te Naarden, nu geschikt voor werken met tapes. FATE wordt beschikbaar gesteld met toestemming van Proton.

In de edities van DE 6502 KENNER worden regelmatig mededelingen gedaan over de door de club georganiseerde bijeenkomsten. Ook worden bestuurlijke mededelingen gedaan, naast informaties over hetgeen in de handel te koop is. Leden die iets te koop hebben of iets zoeken kunnen dit in de edities van DE 6502 KENNER bekend maken. Ook worden brieven aan de redaktie gepubliceerd, evenals specifieke vragen van leden. De edities worden samengesteld op basis van een groot aantal prioriteiten, welke door een redaktievergadering worden gehanteerd. Deze vergadering bestaat uit de vaste medewerkers zoals in de colofon vermeld. Het aantal inzendingen is groter dan in een enkele editie van minimaal 48 pagina's is te verwerken. Hierdoor kan het voorkomen dat een inzending eerst na enige tijd kan worden gepubliceerd.

DE CLUB HEEFT BEHOEFTE AAN MEER LEDEN. WIJ WILLEN MEER AAN KUNNEN BIEDEN DAN NU AL HET GEVAL IS. WERF DAAROM EEN LID!

WILT U EEN PRIJSLIJST? STUUR EEN GEFRANKEERDE ENVELOP AAN HET REDAKTIE-ADRES.

Een onafhankelijke jury kent jaarlijks een aantal aanmoedigingspremies toe aan auteurs van gepubliceerde artikelen in DE 6502 KENNER.

DE 6502 KENNER

De 6502 KENNER is published by the KIM Users Club The Netherlands.

Address all editorial, advertising and subscription inquiries DE 6502 KENNER:
c/o Willem L. van Pelt
Jacob Jordaensstraat 15
2923 CK Krimpen a/IJssel
The Netherlands

Editorial Staff:
Willem L. van Pelt
Gerard van Roekel
Frans Smeehuijzen
Coen Boltjes

Freelancers:
Fred Behringer, (Germany)
Andrew Gregory, (England)
Marc Lachaert, (Belgium)
Fernando Lopes, (Portugal)
Gert van Opbroek
Leif Rasmussen, (Denmark)
Ruud Uphoff
Frans Verberkt
Simon Voortman
Herman Zondag

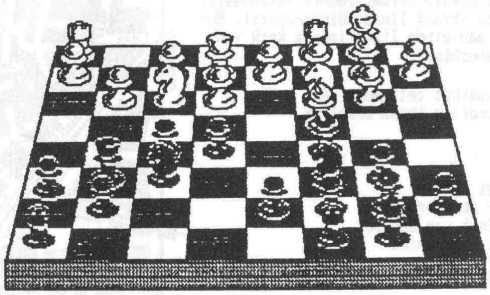
Translations:
Fred Behringer, (Germany)
Willem van Asperen
Frank Bens
Albert v.d. Beukel
Rene Hettfleisch
Coen Kleipool, (France)
Maarten van Lieshout
Antoine Megens
and many others.

DE 6502 KENNER appears in Febr, Apr, May, July, Sept, Oct, and Dec.

On frontpage is the DOS65 controllercard, developed by our member Ad Brouwer.
CAD/CAM: E. Visschedijk.
Coop. : A. Hankel
Photo : Fr. Visschedijk.

The redaktion is not resposible for the contents of the articles, programs in the issues. The articles to be published have to be written by the sender.

CONTENTS OF DE 6502 KENNER NO. 48, FEBRUARY 1987 VOL. 11/NO. 1



Print your articles, programs etc. with a new ribbon and use 8 lines/inch by max. 73 lines/page.
Write your articles, programs etc. in English unless you need help to do it.

Send the names and addresses of 6xxxx-users you know in your own country or you found in magazines to the editorial office. We will send information to those computerists and try to make them joining our club.

Send the names and addresses of the computer magazines in your country to the editorial office. We will send the magazines informations about our club to give attention to it and to get members in your own area.

PLEASE PAY YOUR 1987 SUBSCRIPTION (Hfl. 59,50 or Eurocheque = Hfl. 50,00)

UITNODIGING BIJEENKOMST

Datum : zaterdag 21 MAART 1987 ENTREEPRIJS: FL. 10,==
Lokatie : Gemeenschapshuis "De Zes Gehuchten"
Papenvoort 10 te Geldrop (bij Eindhoven)

Reisroute:

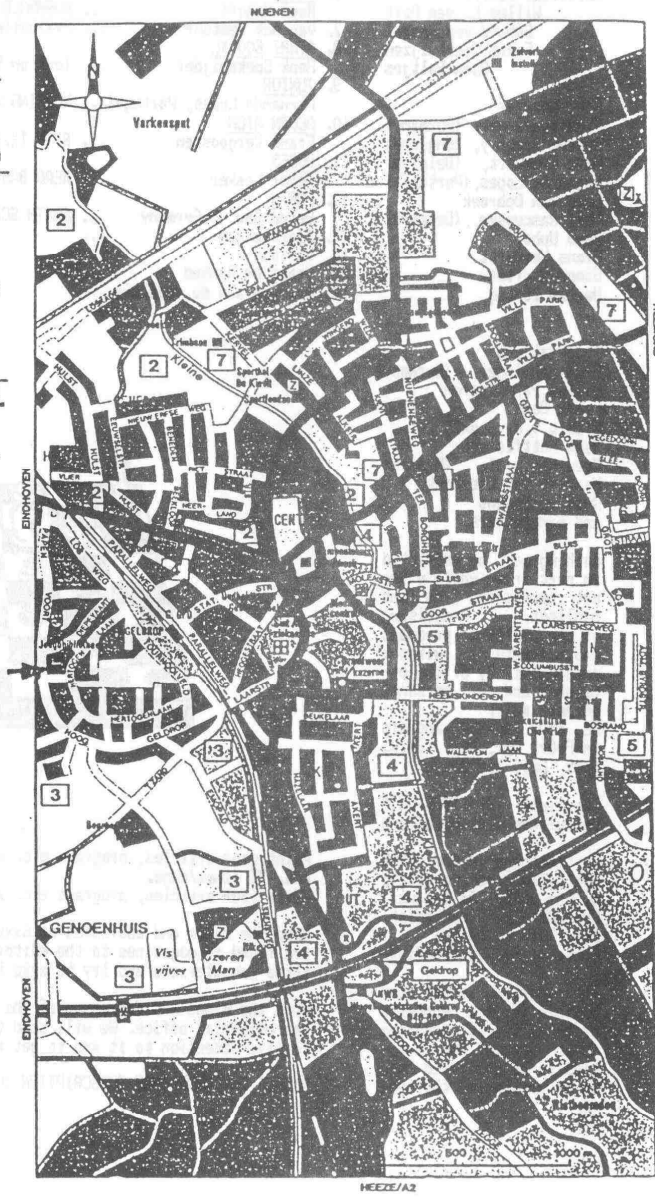
- per auto - vanaf snelweg E3 Eindhoven - Venlo
Afslag Geldrop bij Golden Tulip hotel. Bij eerste verkeerslicht richting centrum linksaf slaan (Emopad). Bij kruising na ca 800 m. linksaf slaan (Hoog Geldrop). Deze weg blijven volgen tot gebouw naast kerk aan rechterzijde.

- vanuit rondweg Eindhoven
Afslag Geldrop bij DAF showroom (Eindhovense weg). Weg volgen tot tweede verkeerslicht voor spoorwegviadukt. Rechtsaf slaan en meest rechtse weg inslaan (Papenvoort). Deze weg blijven volgen tot gebouw voor kerk aan linkerzijde.

- vanuit Nuenen - Helmond
In Geldrop richting Eindhoven. Bij 1e verkeerslicht voorbij spoorviadukt linksaf (voorsorteren). Verder als bij route vanuit rondweg Eindhoven.

- per trein - Stoptrein Eindhoven Weert
In Geldrop direkt na uitgang station links voetgangerstunnel door (Tourmooiveld). Weg oversteken en rechtsaf gaan. Eerste laan links (Hertogenlaan). Na bocht eerste straat rechts (Gildestr.) Eerste straat links (Diepenvaart). Gebouw aan einde links (naast kerk van Zesgehuchten).

Lunchpakket zelf meenemen.
Consumpties tegen betaling.



Programma:

09.30 Zaal open met koffie
10.15 Opening door de voorzitter
10.30 Voordracht Gert van Opbroek
... Van 6502 naar 6xxxx
11.30 FORUM
12.00 Lunchpauze
13.00 MARKT
13.15 INFORMEEL GEDEELTE
In dit deel kunnen alle leden hun zelf ontwikkelde hardware en software demonstreren.
Het illegaal gebruik en het kopieeren van software waarop enige vorm van rechten rusten, ook die van de club, is ten strengste verboden!
BRENG UW EIGEN SYSTEEM MEE !!!
Het aanbieden van overbodige spullen is op eigen tafel(s) te regelen.
17.00 SLUITING.

De redaktie roept eenieder op via de radio ontvangen Basicode programma's in te sturen.

Geef U bij de redaktie op voor het maken van illustraties, vertalingen naar het engels, of illustratieve tekeningen.

KIM GEBRUIKERS CLUB NEDERLAND JAARSTUKKEN 1986

Balans per 31 december 1986.

AKTIVA	1986	1987
Inventaris	3410	5460
Voorraden	pm	pm
Te ontvangen posten	0	207
Geldmiddelen	28774	18540
	32184	24207
PASSIVA		
Vrije reserve	19833	14938
Reservering 6e uitgave DE 6502 KENNER	n.v.t.	n.v.t.
Vooruitontvangen contributie	12130	6795
Te betalen posten	221	2474
	32184	24207
BATEN		
Contributie	18353	17662
Opbrengst advertentie DE 6502 KENNER	900	0
Verkoop losse edities DE 6502 KENNER	1206	1204
Verkoop cassettes en manuals	365	1143
Opbrengst clubbijeenkomsten	1101	721
Rente bank en giro	534	501
DOS65	1226	765
Diversen, o.a. diskettes	3425	42
	27110	22038
LASTEN		
Drukkosten DE 6502 KENNER	13553	11250
Verzendkosten	1061	1114
Redaktiekosten	2867	1180
Bestuurskosten	1466	1970
HCC-dagen	88	2528
Afschrijving inventaris	3180	2490
	22215	20532

Opgemaakt door:
J.F. Van Sprang
Penningmeester.

EC65K - CPU BOARD

By : Bernd Dau, Germany

My 65K was not working correctly at 4MHz if the slow-down switch was connected (not switched but just connected). I have changed the 74F74 with a new 74LS74 ... that has no effect. I changed it with a 7474 and from now on the board is working alright. But if the Eprom is addressed, and the slow-down is putting on by the diode, the EC is going to Nirwana. Well, I have changed the 74F04 and the 74F00 with a 7404 and 7400 and from now on the board is running and running and ...

Het jaar 1986 is voor onze club het jaar van de aktiviteiten geweest. Dat is het beeld dat blijft hangen. De EC65, alias OCTOPUS, alias SAMSOM, zelfbouwprojekt van het tijdschrift Elektuur, was al eerder geboren, maar groeide vooral tot een voor de redaktie interessante computer door drie leden: Marc Lachaert uit België, Leif Rasmussen uit Denemarken en Coen Boltjes uit Delft. Er zijn er natuurlijk veel meer, en ook die zou ik willen noemen, maar dat wordt moeilijk met al zoveel plaatsgebrek. De club werd door OCTOPUS ook voor de vraag gesteld of we iets kunnen doen met Z80. In principe is dat technisch geen enkel probleem. De redaktie heeft de Z80 ook in de APPLE zitten. Voordat je je echter blind staart op toeters en bellen in plaats van op toepassing, moet je je afvragen of Z80 wel zo spectaculair is. Mij lijkt het dat de Z80 door kreten als CP/M, Wordstar en DBASE op een wereldwonder is gaan lijken, iets heel spectaculairs. Nu kan niet ontkent worden dat het handige programma's zijn als je er perse op gebrand bent. Met een assembler programmeren in Z80 is een ding, met een assembler goed programmeren is een tweede; de instruktieset van de Z80 is zo omvangrijk dat je het met de grootste moeite kunt overzien, het snel even iets programmeren, ik bedoel met kwaliteit, is weggelegd voor de doorgewinterde programmeurs. Wat moet een gemiddelde hobbyist nou eigenlijk met DBASE, vraag je je af. Je hebt er al gauw een harddisk bij nodig. Met 80 Kbyte geformatteerd op je floppy kom je dan niet zo heel erg ver. En als ik naar de programma's in z'n algemeenheid kijk, dan kan ik niet anders dan zeggen: alles kom ik in de wereld van de 6XXXX evenzo tegen.
Omdat in het buitenland DOS65 ook in gebruik is, hebben een aantal leden zich extra ingespannen om de manuals te vertalen. Een enorm karwei, een enorme solidariteit, dat ook terug te vinden is in de vertalingen van heel veel andere zaken. Tientallen leden zijn ermee bezig. Het ondersteunt onze club in de pogingen elders kennis te verwerven en uit te wisselen.
In 1986 verscheen onze naam in diverse binnen- en buitenlandse gedrukten. We hebben internationaal de aandacht getrokken. Elektor Electronics in Engeland, Elektor Portugal, RUN, Acorn Users, het zijn er slechts een paar. Ook lokale couranten, en Het Vrije Volk. We hebben nog een artikel in de Helmstedter Bild tegoed, door een lid zelf geschreven. Met betrekking tot de toeloop van buitenlandse leden in 1986 wil ik opmerken dat de blijvers daarvan zich beijveren in het vinden van nieuwe leden en er al in geslaagd zijn dat in nieuwe lidmaatschappen waar te maken. Het mooiste voorbeeld vond ik wel van Fred Behringer uit München. Hij plaatste zijn artikel C-16 naar 512 K in ons decembernummer. Dat kondigde hij tevoren al aan in diverse lezersrubrieken van verschillende computerbladen. Gevolg: een heleboel aanvragen om informatie over onze club. Gevolg daarvan: nieuwe leden. Een daarvan zorgt meteen al voor een nieuw lid, nog voor zijn lidmaatschap officieel is ingegaan.
Het idee van Karel Odon om modems van de leden in te schakelen heeft nu vaste vormen aangenomen en de medewerking breidt zich uit. Onze naam is nu in diverse databanken te vinden. Nieuwe ideëen zijn dus welkom.
Een oud, en eerder mislukt idee, krijgt in 1986 plotseling wel gestalte: kontaktpersonen in gemeenten in binnen- en buitenland. Spoedig zullen zij in de gemeentegidsen en de Gele Gids vermeld zijn. Ook hierdoor krijgen we er nieuwe leden bij, terwijl het op plaatselijk nivo tot onderlinge kontakten kan leiden.
In 1985 schreven 61 nieuwe leden in, in 1986 waren dat er 109. Het is het scala aan aktiviteiten en medewerking daarin dat het mogelijk maakt zulke successen te bereiken. Allen die dit rechtstreeks aangaat: bedankt, jullie hebben applaus verdient.
Een 1987 dat zo'n inzet en enthousiasme ontmoet kan nooit meer stuk.

van Pelt.

FORTH SCREEN-COPY

```
System : Elektor's EC65
Author : Bern Dau, St. Barbara Str. 17,
         D-3333 Buddenstedt
         W-Germany.

This program is the first FORTH I've written :  it does only work
with Elektor's EC-65 FORTH Disk 19 ( 80 tracks full screen ). The
Word Copy-A)C must have 2 words on stack ) source screen destina-
tion screen (
It is very easy to extend the Word to copy a number of screens :

: COPY-ALL    57    11    DO  I  I   COPY-A)C   LOOP ;
(             end  begin   2 times the param for COPY-A)C )

: COPY-ALL-WITH-OFFSET  57 11 DO I I 99 + COPY-A)C LOOP ;
(             11 source = 100 destination
              57 source = 156 destionaton        )

SCR# 134
  0  cr ." COPY UTILITIES" CR ."     simply enter scr A scr C" cr
  1                        ."     and then the COPY.... Word" cr
  2 : COPY-A)C EMPTY-BUFFERS SWAP BLOCK DROP
  3            PREV @ !  UPDATE STRING " SE C" DOS$ !  !DOS FLUSH
  4   FORTH STRING " SE A" DOS$ ! !DOS
  5         " GO 2754" DOS$ ! !DOS " GO 2761" DOS$ ! !DOS FORTH ;
  6
  7 : COPY-A)B EMPTY-BUFFERS SWAP BLOCK DROP
  8            PREV @ !  UPDATE STRING " SE B" DOS$ !  !DOS FLUSH
  9   FORTH STRING " SE A" DOS$ ! !DOS
 10         " GO 2754" DOS$ ! !DOS " GO 2761" DOS$ ! !DOS FORTH ;
 11       ( the 2 times go statement is to unload head )
 12   ;S
 13
 14
 15
```

TE KOOP

OCTOPUS65 bestaande uit: CPU-kaart, 56K RAM-kaart, FDU-kaart, cassette-interfacekaart + software, keyboard, prof. voeding alle spanningen, het geheel ondergebracht in prof. bedrijfskast, terminal type op wielen, dus geen tafelmodel, werkend te zien, zonder drives en systeemsoftware. Prijs: Hfl. 650,--.

COMMODORE 801 PRINTER t.e.a.b.
Inlichtingen bij Peter van Klink
Tel. thuis: 070 - 679289
Tel. werk: 070 - 792805

BRIEF AAN DE REDAKTIE

Ik heb de oorspronkelijke systeemsoftware van mijn JUNIOR veranderd. Zo zit bijv. de basismo-nitor en videosoftware in Eprom vanaf adres $F2C0. Ik ben nu bezig de cassetteroutines van Ad Brouwer uit edities 30 en 31 van DE 6502 KENNER aan mijn systeem aan te passen, doch dit wil niet zo best lukken.
Vraag: Wil degene die deze cassetteroutines op z'n systeem heeft draaien zich met mij in verbinding stellen?

NOGMAALS: ELEKTUUR VDU-KAART

Deze kaart brengt kennelijk kopzorg voor vele clubleden. Ik had de volgende problemen bij het opbouwen:

1. De CLK-ingang van de 6845 bleek erg storingsgevoelig en de chip produceerde dan verkeerde adressen voor de karakters. Dit werd verholpen door een weerstand van 470 Ohm tussen deze lijn en plus te leggen, en gelijk aan de uitgang van IC21 (tussen pen 12 en 16);
2. De registers van de 6845 kregen blijkbaar niet de goede data binnen, waardoor er een verkeerd beeldformaat uitrolde. Met het verwijderen van de buffer-IC 16 werd het iets beter. Misschien moet de adresdekoder N38 van een snellere type zijn;
3. De 4 flipflops die de DEN en CUR-signalen met 2 karaktertijden moeten vertragen, brengen bij mij niet genoeg vertraging, waardoor het venster en de cursor een plaats te ver naar links komen. Dit heb ik niet opgelost;
4. Het beeld op mijn monitor is niet helemaal stilstaand te krijgen, het "golft" een beetje, alsof er een laagfrekwente storing in zit. Misschien ligt dit aan de massa-verbindingen. Als lezers soortgelijke ervaringen hebben gehad of oplossingen weten, graag contact opnemen met:

Hakon Austbo/, Waldecklaan 19, Bussum, 02159-35381.

BRIEF VAN ELEKTUUR

Ons lid Will Cuijpers ontving van Elektuur de mededeling dat, naar aanleiding van zijn vraag, de printen 86207 (65XXX-CPU-kaart) en 86208 (PSIO-RTC-kaart) weer in produktie zijn genomen, Elektuur verwachtte de printen in januari 1987 weer op voorraad te hebben.

From our member Bernd Dau, Germany :
For the German members the very best book to start with FORTH is the book from

DATA BECKER GmbH
Merowingerstr. 30
4000 Duesseldorf

title : ' DAS TRAININGSBUCH ZU FORTH '
author : ' MONADJEMI '

The book is also available in Computer-stores.

NIEUWE UITZENDTIJDEN HOBBYSCOOP

Woensdag Radio 1 en Radio 2 : 19.02 - 19.30 uur
Zondag Radio 5 : 22.40 - 23.00 uur

GRAPHICS IN APPLESOFT
By: Hans Bosch, Twente University of Technology, The Netherlands.
A machine language subroutine which carries out all the necessary functions to prepare a graph: scaling and plotting of the axes, and plotting of the functions to be represented. The text can be put anywhere. The subroutine may be called for either in direct mode or from the user's Applesoft program by using a statement which starts with the ampersand & command followed by a reserved word, the token of it specifying the particular action to be taken. The keywords allowable so far are &HLIN, &VLIN, &PLOT and &DRAW. The syntax and use of each of these commands will be described separately.
This article was published in DE 6502 KENNER 9(1985)6(DEC)
Send cheque of Hfl. 30,= (eurocheque 20,50) to W.L.v.Pelt.

UNI LISTER FOR THE COMMODORE 64
By: R. Fransen, The Netherlands.
Machine language program to translate the graphical signs in the print-commands into readable text.
This article was published in DE 6502 KENNER 9(1985)5(OKT)
Send cheque of Hfl. 14,50 (eurocheque 5,00) to W.L.v.Pelt.

Videotex-programme for the Junior/Octopus

by: Coen Boltjes tel: 015-136812
Nw. Plantage 9 vidibus: 400029830
2611 XH Delft
The Netherlands

Translated by: Elja van der Veer

After describing the hardware in issue 47 it is now time for the software. Although the intension is not to discuss the entire programme an introduction is presented with Prestel as standard.
A videotex page consists of 24 lines of 40 characters. As the uppermost and lowest lines are reserved for systeminformation, 880 characters remain for information. In order to unambigiously detect the start of a new page "NULL, Clear Screen, CR, CR, CR" is send as header. The videotex system works by means of the page mode, which means that the first line follows the last. This is in contrast to forinstance the fido network.
Each line is started in the so-called initial mode. This mode presents a character as follows: alfa-numerical white, black background, normal grafics, no flashing, normal height and not concealed. If a green picture must be drawn over a number of lines, each line must be started with a change to grafical green. As datatransport takes place with 7 bits (with even parity) the mode changes can not be coded with one character. Therefore this takes place as escape sequence. After receiving the escape character ($1B) the next sign is interpreted as display attribute which results in a mode change. In general the two characters result in printing one space.

Examining the videotex code tabel we can differentiate 4 groups of characters:
Columns 0 and 1 contain the control characterset C0;
Columns 2, 3, 4, 5, 6 and 7 contain the alfa-numerical set G0;
Columns 2a, 3a, 5a and 6a contain the mosaic character set L;
Columns 4b and 5b contain the display attributes control code set C1.
Sofar the description of the Prestel Standard. For furthes information we refer to the literature. (1)

In the programme a large number of commands have been implemented. These can be called with CTRL-keys. Normal keys can not be used as they have to be send to the connected videotex computer. The commands, which are described below, uses three information structures which are: Screen, Page and File.
A screen is defined as the information visible on the monitor.
A page is the information in the page buffer, which is in general all information received after the last Form Feed.
A file contains one or more pages seperated by one ore more FF's. The format of the file is the standard OHIO-format, and ends with a $00 as end of file mark.

The available commands are:

^A)ccess -sends your viditel access- and code number. Before assembling these codes must be inserted behind the label CODE.
^B)uffer -empties the page buffer. This command is needed only if the header is lacking.
^D)os -passes a DOS-Command.
^E)rase -erases the screen.
^G)et -reads the next page from the file and lists it. The readpointer is adjusted.
^H)elp -presents a list of the available commands.
^I)nit -sets the page readpointer to the beginning of the file.
^K)ey -reads the value of the key to be depressed next, and sends it to the connected computer, also when it it concerns a CTRL-key. This is used for databanks which are "almost standard".
^L)ist -lists the page on the screen.
^N)ew -creates a new file, replacing the old one.
^P)rint -prints a page on a printer. For an impleme-tation see issue 47.
^Q)uit -leaves the programme. The return address is pulled from the stack.
^R)eveal -list the page with the concealed information.
^S)ave -saves the page in the file, locating it in the end of the file.
^T)erminal -transfers to the terminal protocol, with only ^Q)uit and ^V)ideotex as commands. The user must take care of the size of the text. LF's are ignored, and CR's are send as #.
^V)idibus -saves and sends a character string. If ^V is followed by a characterstring, a string is defined. (ending with CR, 16 characters max.) If ^V is followed by CR the defined string is send to the computer. This command can be used for temporarily storing clientnumbers in message systems.
^Y) enables and disables the paritycheck on receiving charackters. A wrong parity will result in printing a blockcharacter.
^X) sends the cancel character.

A download programme is to be devellopped.

As videotex uses # for closing down a command, the CR key is changed into #. For the same reason it is practical to reserve a key for sending *. In this programme this is ^Z. Note that the videotex code for # deviates from the ASCII code. (see code tabel)

Although the programme with the adjusted hardware can very well be used as a videotex-terminal, it is impossible to support the entire standard. Below you will find a list of what is not possible.
-The programme works without any color. If the background color is other than black the character is inverted. Color information is, by the way, recorded.
-The hardware doesnot support double height. Characters unther a line with double height are represented as spaces, as is prescribed.
-The hardware doesnot support seperated grafics. These are represented as normal grafical characters.
-The programme writes a line only once. If after writing a line this line must be changed, it is possible that this change has not the right effect. In practice this is no serious drawback.

As the hardware used is not normally present, and consequently differt configurations may be used, software dependent from the computer is described below. In the listing two parts have been described twice. If the part which is not relevant is left out, the software can be used with all configurations. Take care that the I/O addresses agree with the addresses in your computer.

INIKB -initiates the keyboard I/O, in general this has already taken place.
INIMOD -initiates the modem I/O and the bufferparameters. A received character must cause an IRQ.
READMO -Is the IRQ routine. It reads the character of the modemACIA and inserts it into the variable input buffer (MODBUF). It also take care of the character flash.
GETMOD -reads a character from MODBUF. If the buffer is not empty, it ends with the character in AHOLD and C=0, else it ends with C=1.
GETKB -reads a character from the keyboard. If a key was depressed, it ends with the key in AHOLD and C=0, else it ends with C=1
PUTMOD -sends AHOLD trough the modemACIA.

When the programme is started for the first time a DOS-error will follow. This is the result of loading the file VIDIWE which is not yet present. In this file telephone numbers or the welcome-page of viditel can be put.

```
                ;VIDITEL VERSION 2.6 86-09-27
                ;LAST REV.           86-12-03
                ;
                ;EXTERNAL DOS ROUTINES/VARIABLES
                ;
2A84=           XQDOSC=$2A84        ;DOS EXECUTIE ROUTINE
2363=           AHOLD =$2363
00E1=           DOSBUF=$00E1        ;COMMAND VECTOR
2300=           HIMEM =$2300        ;HIGHEST RAM ADDRESS
2311=           OUTV1 =$2311        ;OUTPUT VECTOR 1
                ;
                ;EXTERNAL COMPUTER DEPENDEND ROUTINES
                ;
F000=           VIDEO =$F000        ;PRINT AHOLD ON SCREEN
F32F=           RESET =$F32F        ;($F330) INIT CRTC
F0E7=           CRLDN =$F0E7        ;($F0B1) CURR LINE DOWN
F0A8=           CRLUP =$F0A8        ;($F0F0) CURR LINE UP
                ;
                ;EXTERNAL COMPUTER DEPENDEND VARIABELES
                ;
E7CB=           IRQVEC=$E7CB        ;($FA7E)
006C=           RAMPTR=$006C
E800=           VIDRAM=$E800        ;($D000)
E140=           CRTC  =$E140        ;($D800)
E7CD=           FLN   =$E7CD        ;($EFC0)
E7CF=           CLN   =FLN+2
E7D1=           LLN   =CLN+2
E7D3=           SCRPTR=LLN+2        ;SLAVE SCREEN POINTEER
E7D5=           CURSOR=SCRPTR+2     ;CURRENT CURSOR
E7D7=           INLINE=CURSOR+2     ;IN LINE OF WINDOW
E7D8=           COLUMN=INLINE+1     ;CURRENT COLOM
E7DA=           TEMCOL=COLUMN+2     ;SLAVE COLUMN
E7DB=           RAMBEG=TEMCOL+1     ;START REFRESH RAM
E7DD=           CHAPLN=RAMBEG+2     ;#CHARACTERS/LINE
E7DE=           LPSCR =CHAPLN+1     ;#LINES/SCREEN
E7E1=           TABLE =LPSCR+3      ;($EFDC) CRT FORMAT BUF.
                ;
                ;*** COMPUTER DEPENDEND I/O DEVICES ***
                ;
E100=           VABADR=$E100        ;(F800) VIA BASE ADDRESS
E101=           VAPAD =VABADR+1     ;PORT A
E104=           VAT1C =VABADR+4     ;TIMER 1 COUNTER
E106=           VAT1L =VABADR+6     ;TIMER 1 LATCH
E10B=           VAACR =VABADR+$0B   ;AUX CONTROL REG.
E10D=           VAIFR =VABADR+$0D   ;INT. FLAG REGISTER
E10E=           VAIER =VABADR+$0E   ;INT. ENABLE REGISTER
                ;
E400            *=$E400
                ;
                ;INTERNAL VIDEOTEX CONTROLE BYTES
E400=           GRAFLG=*            ;0=NORMAL <>0=GRAFIC
E401=           SEPFLG=*+1          ;0=NOT SEPERATED
E402=           GRAHOC=*+2          ;0=NORMAL
E403=           FLAFLG=*+3          ;0=NOT FLASHING
E404=           CONFLG=*+4          ;FF=NORMAL 00=CONCEALED
E405=           COLCHA=*+5          ;COLOR CHARACTER
E406=           COLBAC=*+6          ;COLOR BACKGROUND
E407=           HEIFLG=*+7          ;0=NORMAL HEIGHT
E408=           HEIGHC=*+8          ;>0 THIS LINE DUBBLE
                ;                   ;<0 PREVIOUS LINE DOUBLE
E409=           REVFLG=*+9          ;0=NORMAL PRINT
                ;                   ;<>0 DISABLE CONC. PRINT
E40A=           ESCFLG=*+10         ;
                ;
E40B=           SOP   =*+11         ;START OF PAGE POINTER
E40D=           EOP   =*+13         ;END OF PAGE POINTER
E40F=           XHOLD =*+15         ;SAVE FOR X
E410=           SHOLD =*+16         ;SAVE FOR SP
E411=           LASTCH=*+17         ;LAST PRINTED CHARACTER
E412=           PRTCNT=*+18         ;# CHARACTERS TO BE PRINT
E414=           MCNT  =*+20         ;MODEM INPUT COUNTER
E415=           MROFF =*+21         ;MODEM READ OFFSET
E416=           MWOFF =*+22         ;MODEM WRITE OFF
E417=           PARCHK=*+23         ;PARITY CHECK BYTE
E418=           VIBANK=*+24         ;TEMP FOR VIDEO PAGE
E419=           TIMCNT=*+25         ;SOFTWARE TIMER
E41A=           NUMBER=*+26         ;VIDIBUS BUFFER
E42A=           SAVEA =*+42         ;A TEMP
E42B=           SAVEX =*+43         ;X TEMP
E42C=           SAVEY =*+44         ;Y TEMP
E42D=           MODBUF=*+45         ;MODEM INPUT BUFFER
                ;
00FE=           PRTPNT=$00FE        ;PRINT POINTER
3A79=           SOF   =$3A79        ;START OF FILE POINTER
3A7B=           EOF   =$3A7B        ;END OF FILE POINTER
1800=           IB    =$1800        ;INPUT BUFFER AREA
007F=           DEL   =$7F          ;DELETE KEY
0027=           VDTCPL=39           ;#VIDIOTEX CHARACTERS/LINE
0017=           VDTLPS=23           ;#VIDIOTEX LINES/SCREEN
000A=           FLATIM=10           ;FLASH TIME
                ;
                ;   *** COMPUTER DEPENDEND DEFENITIONS ***
                ;
                ;   *** DEFENITIONS FOR 6850 MODEM ACIA ***
                ;
E1E0=           MOACTR=$E1E0        ;MODEM ACIA CONTR. REG
E1E0=           MOASR =MOACTR
E1E0=           MOACMR=MOACTR       ;DUMMY REGISTER
E1E1=           MOADR =MOACTR+1     ;MODEM ACIA DATA REG
E1C0=           MOBAUD=$E1C0        ;MODEM BAUDRATE SELECTOR
                ;
0003=           INIPA0=$03          ;BAUDRATE SELECTION
0003=           INIPA1=$03          ;DUMMY VALUE
0089=           INIPA2=%10001001    ;ACIA INITIALISATION
0002=           SENMAS=%00000010    ;SEND REGISTER FREE
0040=           PARMAS=%01000000    ;PARITY ERROR
                ;
                ;   *** DEFENITIONS FOR 6551 MODEM ACIA ***
                ;
E130=           MOADR =$E130        ;MODEM ACIA DATA REG
E131=           MOASR =MOADR+1      ;MODEM STATUS REG
E132=           MOACMR=MOADR+2      ;MODEM COMMAND REG
E133=           MOACTR=MOADR+3      ;MODEM ACIA CONTR. REG
E300=           MOBAUD=$E300        ;BAUDRATE SELECTOR
                ;
0000=           INIPA0=$00          ;BAUDRATE
0069=           INIPA1=%01101001    ;MODEM COMMAND REGISTER
0022=           INIPA2=%00100010    ;MODEM CONTROL REGISTER
0010=           SENMAS=%00010000    ;SEND REGISTER FREE
0001=           PARMAS=%00000001    ;PARITY ERROR
                ;
                ;   *** END OF DECLARATIONS ***
                ;
0200            *=$0200
                ;
0200 78         INITAL  SEI
0201 D8                 CLD
0202 AD1123             LDA OUTV1
0205 48                 PHA
0206 AD1223             LDA OUTV1+1
0209 48                 PHA             ;SAVE OUTPUT VECTOR
020A BA                 TSX
020B 8E10E4             STX SHOLD       ;SAVE STACK FOR DOS ERROR
020E A953               LDA #DOSPRT-1
0210 8D1123             STA OUTV1
0213 A904               LDA #DOSPRT-1/256
0215 8D1223             STA OUTV1+1 ;SET NEW OUTVECTOR
0218 20AC03 INITAM      JSR MOVCRT
021B 202FF3             JSR RESET       ;RESET THE VDU
021E 202003             JSR INIKB       ;INIT THE KEYBOARD
0221 208702             JSR INIMOD      ;INIT THE MODEM ACIA
0224 20FD05             JSR RESMOD      ;RESET MODES
0227 202206             JSR RESDH       ;NORMAL HEIGHT
022A 202E06             JSR RESIB       ;RESET INPUTBUFFER
022D 202806             JSR RESRVF      ;RESET REVEAL FLAG
0230 207D0A             JSR INISOF      ;RESET SOF
0233 20880A             JSR INIEOF      ;RESET EOF
0236 200D03             JSR INITIM      ;START TIMER
0239 ADCEE7             LDA FLN+1
023C 8D18E4             STA VIBANK      ;SELECT FIRST BANK
023F A90A               LDA #FLATIM
0241 8D19E4             STA TIMCNT
0244 A95F               LDA #$5F
0246 8D1AE4             STA NUMBER      ;RESET VIDIBUS NR
0249 A901               LDA #PARMAS ;
024B 8D17E4             STA PARCHK      ;ENABLE PARITY CHECK
024E A966               LDA #DOSERR
0250 8D4F2A             STA $2A4F *
0253 A902               LDA #DOSERR/256
0255 8D502A             STA $2A50       ;SET DOSERROR VECTOR
0258 A27E               LDX #INICOM
025A 86E1               STX DOSBUF
025C A202               LDX #INICOM/256
025E 86E2               STX DOSBUF+1
0260 20842A             JSR XQDOSC      ;EXECUTE COMMAND
0263 203509             JSR GETSCR      ;GET SCREEN AND PRINT IT
0266 58         DOSERR  CLI
0267 AE10E4             LDX SHOLD
026A 9A                 TXS             ;SET STACK BACK
026B 202103 VDTEL       JSR GETKB       ;LOAD FROM KEYBOARD
026E 9003               BCC VDTEL1      ;=>NO CHARACTER
0270 20B907             JSR XQUSER      ;EXECUTE USER COMMAND
0273 205503 VDTEL1      JSR GETMOD      ;LOAD FROM MODEM
0276 90F3               BCC VDTEL       ;=>NO CHARACTER
```

```
1750 0278 20D103          JSR XQDATA  ;EXECUTE COMMAND
1760 027B 4C6B02          JMP VDTEL
1770              ;
1780 027E 4C      INICOM  .BYTE 'LO VIDIWE'
1790              ;
1800              ;INIMOD INIT. THE MODEM I/O
1810              ;IT ALSO INIT THE SYCLIC BUFFER PARAMETERS
1820              ;RECEIVING A CHARACTER WILL CAUSE AN IRQ.
1830              ;
1840 0287 A903    INIMOD  LDA #$03     ;INIT THE MODEM IO
1850 0289 8D31E1          STA MOASR    ;RESET MODEM AXCIA
1860 028C A900            LDA #INIPA0  ;
1870 028E 8D00E3          STA MOBAUD   ;SET SPEED
1880 0291 A969            LDA #INIPA1  ;PARAMETER 1
1890 0293 8D32E1          STA MOACMR   ;SELECT
1900 0296 A922            LDA #INIPA2  ;PARAMETER 2
1910 0298 8D33E1          STA MOACTR   ;INIT MODEM ACIA
1920 029B A9B1            LDA #MOREAD
1930 029D 8DCBE7          STA IRQVEC
1940 02A0 A902            LDA #MOREAD/256
1950 02A2 8DCCE7          STA IRQVEC+1 ;INIT IRQVEC
1960 02A5 A900            LDA #$00
1970 02A7 8D14E4          STA MCNT
1980 02AA 8D15E4          STA MROFF
1990 02AD 8D16E4          STA MWOFF    ;RESET INPUT PARAMETERS
2000 02B0 60              RTS
2010              ;
2020              ;MOREAD IS IRQ ROUTINE FOR READING THE
2030              ;MODEM ACIA AND THE FLASH
2040              ;
2050 02B1 48      MOREAD  PHA          ;
2060 02B2 8A              TXA
2070 02B3 48              PHA          ;SAVE A & X
2080 02B4 AD31E1          LDA MOASR    ;RESET IRQ
2090 02B7 1026            BPL TIMER    ;=>NO ACIA IRQ
2100 02B9 2C17E4          BIT PARCHK   ;
2110 02BC F007            BEQ MOREA1   ;=>NO PAR ERROR
2120 02BE AD30E1          LDA MOADR    ;CLEAR IRQ
2130 02C1 A97F            LDA #$7F     ;LOAD BLOCK
2140 02C3 D003            BNE MOREA2   ;=>ALWAYS
2150 02C5 AD30E1 MOREA1   LDA MOADR    ;LOAD CHARACTER
2160 02C8 AE14E4 MOREA2   LDX MCNT
2170 02CB E0FF            CPX #$FF
2180 02CD F03A            BEQ TIMER1
2190 02CF 297F            AND #$7F     ;STRIP BIT 7
2200 02D1 AE16E4          LDX MWOFF
2210 02D4 9D2DE4          STA MODBUF,X
2220 02D7 EE16E4          INC MWOFF
2230 02DA EE14E4          INC MCNT     ;ADJUST BUF PAR.
2240 02DD D02A            BNE TIMER1   ;=>ALWAYS
2250 02DF AD04E1 TIMER    LDA VAT1C    ;RESET IRQ
2260 02E2 CE19E4          DEC TIMCNT   ;
2270 02E5 D022            BNE TIMER1
2280 02E7 A90A            LDA #FLATIM  ;GET FLASH TIME
2290 02E9 8D19E4          STA TIMCNT   ;RESET FLASH TIME
2300 02EC AD18E4          LDA VIBANK   ;LOAD OLD BANK
2310 02EF 4904            EOR #%00000100
2320 02F1 8D18E4          STA VIBANK   ;SET NEW BANK
2330 02F4 A20C            LDX #12
2340 02F6 8E40E1          STX CRTC
2350 02F9 8D41E1          STA CRTC+1   ;SWITCH VDU BANK
2360 02FC A20E            LDX #14
2370 02FE A904            LDA #%00000100
2380 0300 8E40E1          STX CRTC
2390 0303 4D41E1          EOR CRTC+1
2400 0306 8D41E1          STA CRTC+1   ;SWITCH CURSOR
2410 0309 68      TIMER1  PLA
2420 030A AA              TAX
2430 030B 68              PLA
2440 030C 40              RTI
2450              ;
2460 030D A940    INITIM  LDA #%01000000
2470 030F 8D0BE1          STA VAACR    ;SELECT CONTINOUS
2480 0312 A9FF            LDA #$FF
2490 0314 8D04E1          STA VAT1C
2500 0317 8D05E1          STA VAT1C+1  ;START TIMER
2510 031A A9C0            LDA #%11000000 ;ENABLE IRQ
2520 031C 8D0EE1          STA VAIER
2530 031F 60              RTS
2540              ;
2550              ;GETKB READS AN KEY. IT ENDS WITCH C=0 IF
2560              ;NO KEY, ELSE C=0 AND KEY IN AHOLD
2570              ;
2580              ;*** KEYBOARD  ROUTINES FOR OCTOPUS ***
2590              ;
2600 0320 60      INIKB   RTS          ;INIT THE KEYBOARD IO
2610              ;
2620 0321 18      GETKB   CLC          ;C=1 KEY C=0 NO KEY
2630 0322 AD0DE1          LDA VAIFR    ;VIA INTERRUPT FLAG REG
2640 0325 2902            AND #%00000010
2650 0327 F009            BEQ GETKB1   ;=>NO KEY
2660 0329 AD01E1          LDA VAPAD    ;GET CHARACTER
2670 032C 297F            AND #$7F     ;STRIP PARITY
2680 032E 8D6323          STA AHOLD
2690 0331 38              SEC
2700 0332 60      GETKB1  RTS
2710              ;
2720              ;*** KEYBOARD ROUTINES FOR JUNIOR ***
2730              ;
2740 E1D0=        KBACR =$E1D0         ;KEYB.ACIA CONTROL REG
2750 E1D1=        KBADR =KBACR+1       ;ACIA DATA REGISTER
2760 E1F0=        KBBAUD=$E1F0         ;KEYBOARD BAUDR. SELECT
2770              ;
2780 0333 A903    INIKB   LDA #$03
2790 0335 8DD0E1          STA KBACR    ;RESET ACIA
2800 0338 A911            LDA #$11
2810 033A 8DD0E1          STA KBACR    ;INIT ACIA
2820 033D A933            LDA #$33
2830 033F 8DF0E1          STA KBBAUD   ;1200 BD
2840 0342 60              RTS          ;
2850              ;
2860 0343 18      GETKB   CLC          ;C=1 KEY C=0 NO KEY
2870 0344 ADD0E1          LDA KBACR    ;KEYBOARD ACIA CONTR.REG
2880 0347 2901            AND #%00000001
2890 0349 F0E7            BEQ GETKB1   ;=>NO KEY
2900 034B ADD1E1          LDA KBADR    ;GET CHARACTER
2910 034E 297F            AND #$7F     ;STRIP PARITY
2920 0350 8D6323          STA AHOLD
2930 0353 38              SEC
2940 0354 60      GETKB1  RTS
2950              ;
2960              ;GETMOD READS AN CHARACTER OF THE MODBUF
2970              ;IT ENDS WITH C=1 IF NO CHARACTER, ELSE
2980              ;WITH C=0 AND CHARACTER IN AHOLD
2990              ;IF CHARACTER IS $0C THEN THE INPUTBUFFER
3000              ;IS RESET (JSR RESIB)
3010              ;
3020 0355 18      GETMOD  CLC
3030 0356 AD14E4          LDA MCNT
3040 0359 F042            BEQ GETMO3   ;=>NO MODEM INPUT
3050 035B AE15E4          LDX MROFF    ;MODEM READ OFFSET
3060 035E BD2DE4          LDA MODBUF,X
3070 0361 CE14E4          DEC MCNT     ;DEC MODEM COUNTER
3080 0364 EE15E4          INC MROFF
3090 0367 8D6323          STA AHOLD    ;SAVE CHARACTER
3100 036A C90C            CMP #$0C
3110 036C D006            BNE GETMO1   ;=>NO CLEAR SCREEN
3120 036E 202E06          JSR RESIB    ;RESET INPUTBUFFER
3130 0371 AD6323          LDA AHOLD
3140 0374 AE0DE4 GETMO1   LDX EOP
3150 0377 86FE            STX PRTPNT
3160 0379 AE0EE4          LDX EOP+1
3170 037C 86FF            STX PRTPNT+1
3180 037E A200            LDX #$00
3190 0380 81FE            STA (PRTPNT,X) ;CHARACTER IN PAGE
3200 0382 EE0DE4          INC EOP
3210 0385 D015            BNE GETMO2
3220 0387 EE0EE4          INC EOP+1
3230 038A AE0EE4          LDX EOP+1    ;GET MSB BUFFER
3240 038D E022            CPX #$22
3250 038F D00B            BNE GETMO2   ;=>BUFFER NOT FULL
3260 0391 CE0EE4          DEC EOP+1
3270 0394 48              PHA          ;SAVE CHARACTER
3280 0395 202205          JSR BELL
3290 0398 68              PLA          ;RESTORE
3300 0399 8D6323          STA AHOLD
3310 039C 38      GETMO2  SEC
3320 039D 60      GETMO3  RTS
3330              ;
3340              ;PUTMOD SENDS AHOLD TO THE MODEM ACIA
3350              ;
3360 039E A910    PUTMOD  LDA #SENMAS  ;LOAD SEND MASK
3370 03A0 2D31E1          AND MOASR
3380 03A3 F0F9            BEQ PUTMOD   ;=>TRANSM. REG. FULL
3390 03A5 AD6323          LDA AHOLD
3400 03A8 8D30E1          STA MOADR    ;TRANSMIT CHARACTER
3410 03AB 60              RTS
3420              ;
3430              ;MOVCRT PLACES THE CRTTAB TO THE TABLE
3440              ;
3450 03AC A900    MOVCRT  LDA #VIDRAM
3460 03AE 8DDBE7          STA RAMBEG
3470 03B1 A9E8            LDA #VIDRAM/256
3480 03B3 8DDCE7          STA RAMBEG+1 ;SET START OF RAM
```

```
3490 03B6 A211           LDX #$11
3500 03B8 BD9F07 MOVTAB  LDA CRTTAB,X
3510 03BB 9DE1E7         STA TABLE,X
3520 03BE CA             DEX
3530 03BF 10F7           BPL MOVTAB
3540 03C1 A211           LDX #$11
3550 03C3 BD9F07         LDA CRTTAB,X
3560 03C6 8DDEE7         STA LPSCR        ;#LINES/SCREEN
3570 03C9 CA             DEX
3580 03CA BD9F07         LDA CRTTAB,X
3590 03CD 8DDDE7         STA CHAPLN       ;#CHARACTERS/LINE
3600 03D0 60             RTS
3610              ;
3620              ;MODEM-RECEIVE EXECUTION ROUTINE
3630              ;
3640 03D1 AD6323 XQDATA  LDA AHOLD    ;LOAD CHARACTER
3650 03D4 297F           AND #$7F     ;JUST FOR SURE
3660 03D6 8D6323         STA AHOLD
3670 03D9 C920           CMP #$20
3680 03DB 100E           BPL XQDATB   ;=>NO COMM. CHAR. (C0)
3690 03DD 200B06         JSR RESESC   ;NO ESCFLG
3700 03E0 0A             ASL A        ;ACCU*2
3710 03E1 AA             TAX
3720 03E2 BD4007         LDA C0+1,X
3730 03E5 48             PHA
3740 03E6 BD3F07         LDA C0,X
3750 03E9 48             PHA          ;PUSH ROUTINE ADRES
3760 03EA 60             RTS          ;AND EXECUTE THE ROUTINE
3770 03EB AE0AE4 XQDATB  LDX ESCFLG   ;
3780 03EE F032           BEQ XQDATE   ;=>NO ESC. COMMAND (C1)
3790 03F0 200B06         JSR RESESC   ;RESET ESCAPE FLAG
3800 03F3 20F704         JSR C1PRT    ;PRINT/NOPRINT
3810 03F6 AD6323         LDA AHOLD    ;LOAD THE COMMAND
3820 03F9 C93E           CMP #'>
3830 03FB F055           BEQ START
3840 03FD C93F           CMP #'?
3850 03FF F052           BEQ STOP
3860 0401 291F           AND #%00011111
3870 0403 4A             LSR A
3880 0404 4A             LSR A
3890 0405 4A             LSR A
3900 0406 4A             LSR A
3910 0407 B004           BCS XQDATC   ;=>NO COLOR COMMAND
3920 0409 20DD06         JSR COLOR    ;SET THE COLORS
3930 040C 60             RTS
3940 040D 4A     XQDATC  LSR A
3950 040E AD6323         LDA AHOLD    ;RESTOR COMMAND
3960 0411 B002           BCS XQDATD   ;=>COLUMN 5B
3970 0413 2907           AND #%00000111
3980 0415 290F   XQDATD  AND #%00001111
3990 0417 0A             ASL A        ;ACCU*2
4000 0418 AA             TAX
4010 0419 BD8007         LDA C1+1,X
4020 041C 48             PHA
4030 041D BD7F07         LDA C1,X
4040 0420 48             PHA          ;PUSH ROUTINE ADRES
4050 0421 60             RTS          ;EXECUTE THE ROUTINE
4060 0422 AE00E4 XQDATE  LDX GRAFLG   ;
4070 0425 F015           BEQ XQDATG   ;=>NORMAL
4080 0427 AD6323         LDA AHOLD    ;LOAD GRAFIC CODE
4090 042A C940           CMP #$40
4100 042C 3006           BMI XQDATF   ;=>2A & 3A CHARACTERS
4110 042E C960           CMP #$60
4120 0430 300A           BMI XQDATG   ;=>CAPITAL
4130 0432 49FF           EOR #$FF     ;COMPLEMENT
4140 0434 299F   XQDATF  AND #%10011111
4150 0436 8D6323         STA AHOLD
4160 0439 4C4E04         JMP XQDATI
4170 043C C95F   XQDATG  CMP #$5F
4180 043E D005           BNE XQDATH   ;=>NO #
4190 0440 A923           LDA #'#
4200 0442 8D6323         STA AHOLD
4210 0445 C97F   XQDATH  CMP #$7F
4220 0447 D005           BNE XQDATI   ;=>NO BLOCK
4230 0449 A980           LDA #$80
4240 044B 8D6323         STA AHOLD
4250 044E 206D04 XQDATI  JSR PRINT
4260 0451 60             RTS
4270              ;
4280 0452 60     START   RTS          ;START OF DOWNLOADER
4290              ;
4300 0453 60     STOP    RTS          ;END OF DOWNLOADER
4310              ;DOWNLOAD ROUTINE NOT YET IMPELMENTED.
4320              ;
4330 0454 8D2AE4 DOSPRT  STA SAVEA
4340 0457 8E2BE4         STX SAVEX
4350 045A 8C2CE4         STY SAVEY    ;SAVE REGISTERS
4360 045D 20D103         JSR XQDATA   ;PRINT CHARACTER
4370 0460 AD2AE4         LDA SAVEA
4380 0463 8D6323         STA AHOLD    ;RESTORE AHOLD
4390 0466 AE2BE4         LDX SAVEX
4400 0469 AC2CE4         LDY SAVEY    ;RESTORE REGISTERS
4410 046C 60             RTS
4420              ;
4430              ;
4440              ;PRINT PRINTS AHOLD ON THE SCREEN. WHILE
4450              ;BACKGROUND IS NOT BLACK IT REVERSE.
4460              ;IN A LINE AFTER DOUBLE HEIGHT NOTHING IS
4470              ;PRINTED
4480              ;
4490 046D AE08E4 PRINT   LDX HEIGHC
4500 0470 301A           BMI PRINT3   ;=>PREVIOUS LINE DOUBLE
4510 0472 AE04E4         LDX CONFLG
4520 0475 D005           BNE PRINT1   ;=>NO CONCEAL
4530 0477 A920           LDA #$20
4540 0479 8D6323         STA AHOLD
4550 047C AE06E4 PRINT1  LDX COLBAC
4560 047F F008           BEQ PRINT2   ;=>BLACK BACKGROUND
4570 0481 AD6323         LDA AHOLD
4580 0484 4980           EOR #$80     ;REVERSE CHARACTER
4590 0486 8D6323         STA AHOLD
4600 0489 209004 PRINT2  JSR TVPUT    ;PUT CHAR. ON SCREEN
4610 048C 204205 PRINT3  JSR HT       ;CURSOR RIGHT
4620 048F 60             RTS
4630              ;
4640 0490 18     TVPUT   CLC
4650 0491 ADCFE7         LDA CLN
4660 0494 6DD8E7         ADC COLUMN
4670 0497 856C           STA RAMPTR     ;COMPUTE LSB
4680 0499 ADD0E7         LDA CLN+1
4690 049C 6900           ADC #$00       ;ADD CARRY IF SET
4700 049E 2907           AND #$07       ;$07FF IS MAX ADDRESS
4710 04A0 6DDCE7         ADC RAMBEG+1
4720 04A3 856D           STA RAMPTR+1
4730 04A5 0904           ORA #%00000100
4740 04A7 AA             TAX            ;IN X SECOND PAGE
4750 04A8 AD6323         LDA AHOLD
4760 04AB A000           LDY #$00
4770 04AD 916C           STA (RAMPTR),Y ;SET FIRST PAGE
4780 04AF 8D11E4         STA LASTCH     ;SET LAST CHARACRER
4790 04B2 866D           STX RAMPTR+1
4800 04B4 916C           STA (RAMPTR),Y ;SET SECOND PAGE
4810 04B6 AD03E4         LDA FLAFLG
4820 04B9 F009           BEQ TVPUT2     ;=>NO FLASH
4830 04BB AD06E4         LDA COLBAC
4840 04BE F002           BEQ TVPUT1     ;=>BLACK
4850 04C0 A980           LDA #$80
4860 04C2 916C   TVPUT1  STA (RAMPTR),Y
4870 04C4 60     TVPUT2  RTS
4880              ;
4890              ;ADJUST POSITIONED THE CURSOR
4900              ;
4910 04C5 08     ADJUST  PHP            ;SAVE STATUS
4920 04C6 78             SEI            ;NO TIMER INTERRUPT
4930 04C7 18             CLC
4940 04C8 ADCFE7         LDA CLN        ;CURRENT LINE
4950 04CB 6DD8E7         ADC COLUMN
4960 04CE 8DD5E7         STA CURSOR
4970 04D1 ADD0E7         LDA CLN+1
4980 04D4 6900           ADC #$00
4990 04D6 2907           AND #$07       ;
5000 04D8 8DD6E7         STA CURSOR+1
5010 04DB A20E           LDX #14        ;CURSOR REGISTER
5020 04DD 8E40E1         STX CRTC
5030 04E0 AD41E1         LDA CRTC+1     ;GET OLD POSITION
5040 04E3 2904           AND #%00000100
5050 04E5 4DD6E7         EOR CURSOR+1
5060 04E8 8D41E1         STA CRTC+1     ;SET MSB
5070 04EB E8             INX            ;X=15
5080 04EC ADD5E7         LDA CURSOR
5090 04EF 8E40E1         STX CRTC
5100 04F2 8D41E1         STA CRTC+1     ;SET MSB
5110 04F5 28             PLP            ;RESTORE STATUS
5120 04F6 60             RTS
5130              ;
5140              ;C1PRT IS THE TEST FOR THE GRAFIC-HOLD MODE
5150              ;
5160 04F7 48     C1PRT   PHA
5170 04F8 AE02E4         LDX GRAHOC
5180 04FB F00E           BEQ C1PRT1     ;=>NO HOLD GRAFIC
5190 04FD AE00E4         LDX GRAFLG     ;
5200 0500 F009           BEQ C1PRT1     ;=>NO GRAFIC MODE
5210 0502 2908           AND #%00001000
5220 0504 D005           BNE C1PRT1     ;=>NO COLOR CHANGE
```

```
5230 0506 AD11E4          LDA LASTCH  ;GET LAST CHARACTER
5240 0509 D002            BNE C1PRT2  ;=>ALWAYS
5250 050B A920   C1PRT1   LDA #$20
5260 050D 8D6323 C1PRT2   STA AHOLD
5270 0510 206D04          JSR PRINT
5280 0513 68              PLA
5290 0514 8D6323          STA AHOLD
5300 0517 60              RTS
5310              ;
5320              ;*** CO EXECUTION ROUTINES ***
5330              ;
5340 0518 20FD05 NULL     JSR RESMOD
5350 051B 202206          JSR RESDH   ;RESET DOUBLE HEIGHT
5360 051E 202E06          JSR RESIB   ;RESET INPUT BUFFER
5370 0521 60              RTS
5380              ;
5390 0522 A907   BELL     LDA #$07
5400 0524 8D6323          STA AHOLD
5410 0527 2000F0          JSR VIDEO
5420 052A 60              RTS
5430              ;
5440 052B AED8E7 BS       LDX COLUMN
5450 052E F009            BEQ BS1     ;=>FIRST COLUMN
5460 0530 CED8E7          DEC COLUMN
5470 0533 20C504          JSR ADJUST  ;SET CURSOR
5480 0536 4C4105          JMP BS2
5490 0539 A927   BS1      LDA #VDTCPL ;
5500 053B 8DD8E7          STA COLUMN  ;LAST COLUMN
5510 053E 208E05          JSR VT1     ;ONE LINE UP
5520 0541 60      BS2     RTS
5530              ;
5540 0542 AED8E7 HT       LDX COLUMN
5550 0545 E027            CPX #VDTCPL
5560 0547 F009            BEQ HT1     ;=>LAST COLUMN
5570 0549 EED8E7          INC COLUMN
5580 054C 20C504          JSR ADJUST  ;SET CURSOR
5590 054F 4C5A05          JMP HT2
5600 0552 A900   HT1      LDA #$00
5610 0554 8DD8E7          STA COLUMN
5620 0557 205E05          JSR LF1     ;NEXT LINE
5630 055A 60      HT2     RTS
5640              ;
5650 055B 204006 LF       JSR TSTBG   ;TEST BACKGROUND
5660 055E AED7E7 LF1      LDX INLINE
5670 0561 E017            CPX #VDTLPS
5680 0563 F00B            BEQ LF2     ;=>LAST LINE
5690 0565 201106          JSR TSTDH   ;TEST HEIGHT
5700 0568 20E7F0          JSR CRLDN   ;ONE LINE DOWN
5710 056B EED7E7          INC INLINE  ;
5720 056E 1014            BPL LF3     ;=>ALWAYS
5730 0570 ADCDE7 LF2      LDA FLN
5740 0573 8DCFE7          STA CLN
5750 0576 ADCEE7          LDA FLN+1
5760 0579 8DD0E7          STA CLN+1   ;FLN=CLN
5770 057C A900            LDA #$00
5780 057E 8DD7E7          STA INLINE  ;FIRST LINE
5790 0581 202206          JSR RESDH   ;NORMAL HEIGHT
5800 0584 20C504 LF3      JSR ADJUST  ;
5810 0587 20FD05          JSR RESMOD  ;NORMAL MODES
5820 058A 60              RTS
5830              ;
5840 058B 204006 VT       JSR TSTBG
5850 058E AED7E7 VT1      LDX INLINE
5860 0591 F008            BEQ VT2     ;=>FIRST LINE
5870 0593 20A8F0          JSR CRLUP   ;
5880 0596 CED7E7          DEC INLINE
5890 0599 1011            BPL VT3     ;=>ALWAYS
5900 059B ADD1E7 VT2      LDA LLN
5910 059E 8DCFE7          STA CLN
5920 05A1 ADD2E7          LDA LLN+1
5930 05A4 8DD0E7          STA CLN+1
5940 05A7 A217            LDX #VDTLPS
5950 05A9 8ED7E7          STX INLINE
5960 05AC 20C504 VT3      JSR ADJUST
5970 05AF 20FD05          JSR RESMOD  ;RESET MODES
5980 05B2 202206          JSR RESDH   ;RESET DOUBLE HEIGHT
5990 05B5 60              RTS
6000              ;
6010 05B6 206806 FF       JSR HOME
6020 05B9 AE09E4          LDX REVFLG
6030 05BC D003            BNE FF1     ;=>REVEAL MODE
6040 05BE 208C06          JSR ERTEOS
6050 05C1 20FD05 FF1      JSR RESMOD  ;RESET MODES
6060 05C4 202206          JSR RESDH   ;RESET DOUBLE HEIGHT
6070 05C7 60              RTS
6080              ;
6090 05C8 204006 CR       JSR TSTBG   ;TEST BACKGROUND
6100 05CB A900   RETURN   LDA #$00
6110 05CD 8DD8E7          STA COLUMN  ;FIRST COLUMN
6120 05D0 20C504          JSR ADJUST
6130 05D3 20FD05          JSR RESMOD  ;RESET MODES
6140 05D6 60              RTS
6150              ;
6160 05D7 A900   CURON    LDA #$00
6170 05D9 F002            BEQ SETCUR  ;=>ALWAYS
6180 05DB A920   CUROF    LDA #$20
6190 05DD A20A   SETCUR   LDX #$0A
6200 05DF 8E40E1          STX CRTC
6210 05E2 8D41E1          STA CRTC+1
6220 05E5 60              RTS
6230              ;
6240 05E6 20CB05 CAN      JSR RETURN
6250 05E9 208006          JSR ERTEOL  ;CLEAR LINE
6260 05EC 60              RTS
6270              ;
6280 05ED A9FF   ESC      LDA #$FF
6290 05EF 8D0AE4          STA ESCFLG  ;SET ESCAPE FLAG
6300 05F2 60              RTS
6310              ;
6320 05F3 206806 RS       JSR HOME
6330 05F6 20FD05          JSR RESMOD  ;RESET MODI
6340 05F9 202206          JSR RESDH   ;RESET DOUBLE HEIGHT
6350 05FC 60              RTS
6360              ;
6370              ;*** CO EXECUTION SUB-ROUTINES ***
6380              ;
6390 05FD A207   RESMOD   LDX #$07
6400 05FF BDB107 RESMO1   LDA RESTAB,X
6410 0602 9D00E4          STA GRAFLG,X
6420 0605 CA              DEX
6430 0606 10F7            BPL RESMO1
6440 0608 20D705          JSR CURON   ;CURSOR ON
6450 060B A200   RESESC   LDX #$00
6460 060D 8E0AE4          STX ESCFLG  ;NO C1 COMMANDS
6470 0610 60              RTS
6480              ;
6490 0611 AE08E4 TSTDH    LDX HEIGHC
6500 0614 F00B            BEQ TSTDH3  ;NO DOUBLE HEIGH
6510 0616 3004            BMI TSTDH1  ;=>PREVIOUS DOUBLE
6520 0618 A2FF            LDX #$FF
6530 061A D002            BNE TSTDH2  ;=>ALWAYS
6540 061C A200   TSTDH1   LDX #$00
6550 061E 8E08E4 TSTDH2   STX HEIGHC
6560 0621 60     TSTDH3   RTS
6570              ;
6580 0622 A200   RESDH    LDX #$00
6590 0624 8E08E4          STX HEIGHC
6600 0627 60              RTS
6610              ;
6620 0628 A200   RESRVF   LDX #$00
6630 062A 8E09E4          STX REVFLG  ;RESET REVEAL FLAG
6640 062D 60              RTS
6650              ;
6660 062E A900   RESIB    LDA #IB     ;SOP AND EOP
6670 0630 8D0BE4          STA SOP     ;ARE RESET TO
6680 0633 8D0DE4          STA EOP     ;SOP
6690 0636 A918            LDA #IB/256
6700 0638 8D0CE4          STA SOP+1
6710 063B 8D0EE4          STA EOP+1
6720 063E 60              RTS
6730              ;
6740 063F 60     NILL     RTS
6750              ;
6760 0640 AD06E4 TSTBG    LDA COLBAC
6770 0643 F022            BEQ TSTBG3  ;=>BLACK BACKGROUND
6780 0645 A900            LDA #$00
6790 0647 8D03E4          STA FLAFLG  ;NON FLASHING
6800 064A ADD8E7          LDA COLUMN
6810 064D 48              PHA         ;SAVE CURRENT CURSOR
6820 064E A9A0            LDA #$A0    ;REVERSE SPACE
6830 0650 8D6323          STA AHOLD
6840 0653 209004 TSTBG1   JSR TVPUT   ;PRINT SPACE
6850 0656 A227            LDX #VDTCPL ;CHARACTERS/LINE
6860 0658 ECD8E7          CPX COLUMN  ;LAST COLUMN
6870 065B F006            BEQ TSTBG2  ;=>LINE DONE
6880 065D EED8E7          INC COLUMN
6890 0660 4C5306          JMP TSTBG1  ;=>NEXT
6900 0663 68     TSTBG2   PLA
6910 0664 8DD8E7          STA COLUMN  ;RESTORE COLUMN
6920 0667 60     TSTBG3   RTS
6930              ;
6940 0668 ADCDE7 HOME     LDA FLN
6950 066B 8DCFE7          STA CLN
6960 066E ADCEE7          LDA FLN+1
```

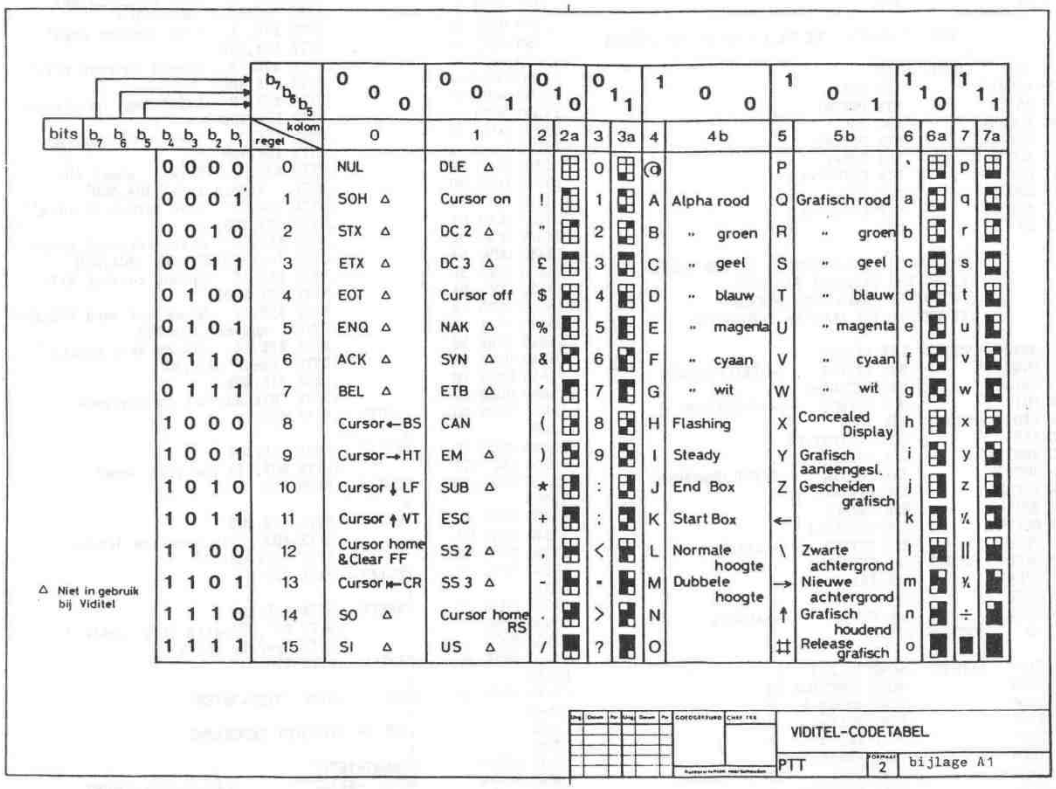
```
6970 0671 8DD0E7          STA CLN+1     ;CLN=FIRST LINE
6980 0674 A900            LDA #$00
6990 0676 8DD8E7          STA COLUMN
7000 0679 8DD7E7          STA INLINE    ;CURSOR HOME
7010 067C 20C504          JSR ADJUST
7020 067F 60              RTS
7030                ;
7040 0680 ADD8E7 ERTEOL   LDA COLUMN
7050 0683 48              PHA           ;SAVE CURRENT CURSOR
7060 0684 20C706          JSR ERLIN     ;
7070 0687 68              PLA
7080 0688 8DD8E7          STA COLUMN
7090 068B 60              RTS
7100                ;
7110 068C ADD8E7 ERTEOS   LDA COLUMN
7120 068F 48              PHA
7130 0690 ADD7E7          LDA INLINE
7140 0693 48              PHA           ;SAVE CURSOR POSITION
7150 0694 ADCFE7          LDA CLN
7160 0697 48              PHA
7170 0698 ADD0E7          LDA CLN+1
7180 069B 48              PHA           ;SAVE CURRENT LINE
7190 069C 20C706 ERTEO1   JSR ERLIN     ;EREASE LINE
7200 069F ADD7E7          LDA INLINE
7210 06A2 C917            CMP #VDTLPS
7220 06A4 F00D            BEQ ERTEO2    ;=>SCREEN CLEAR
7230 06A6 20E7F0          JSR CRLDN     ;LINE DOWN
7240 06A9 EED7E7          INC INLINE
7250 06AC A900            LDA #$00
7260 06AE 8DD8E7          STA COLUMN
7270 06B1 F0E9            BEQ ERTEO1    ;=>NEXT LINE
7280 06B3 68        ERTEO2 PLA
7290 06B4 8DD0E7          STA CLN+1
7300 06B7 68              PLA
7310 06B8 8DCFE7          STA CLN       ;SET CURRENT LINE
7320 06BB 68              PLA
7330 06BC 8DD7E7          STA INLINE
7340 06BF 68              PLA
7350 06C0 8DD8E7          STA COLUMN    ;RESET CURSOR POS.
7360 06C3 20C504          JSR ADJUST
7370 06C6 60              RTS
7380                ;
7390 06C7 A920      ERLIN LDA #$20      ;SPACE
7400 06C9 8D6323          STA AHOLD
7410 06CC 209004 ERLIN1   JSR TVPUT     ;PRINT SPACE
7420 06CF A227            LDX #VDTCPL   ;CHARACTERS/LINE
7430 06D1 ECD8E7          CPX COLUMN    ;LAST COLUMN
7440 06D4 F006            BEQ ERLIN2    ;=>LINE DONE
7450 06D6 EED8E7          INC COLUMN
7460 06D9 4CCC06          JMP ERLIN1    ;=>NEXT
7470 06DC 60        ERLIN2 RTS
7480                ;
7490                ;*** C1 EXECUTIN ROUTINES ***
7500                ;
7510 06DD 4A        COLOR LSR A
7520 06DE AD6323          LDA AHOLD
7530 06E1 B004            BCS SEGRCO
7540 06E3 A200      SEALCO LDX #$00
7550 06E5 F002            BEQ COLOR1    ;=>ALWAYS
7560 06E7 A210      SEGRCO LDX #$10
7570 06E9 2907      COLOR1 AND #$07     ;COMPUTE COLOR
7580 06EB 8D05E4          STA COLCHA
7590 06EE 8E00E4          STX GRAFLG    ;SET MODE
7600 06F1 60              RTS
7610                ;
7620 06F2 A980      FLASH LDA #$80
7630 06F4 8D03E4          STA FLAFLG    ;SET FLASH FLAG
7640 06F7 60              RTS
7650                ;
7660 06F8 A200      STEADY LDX #$00
7670 06FA 8E03E4          STX FLAFLG    ;RESET FLASH FLAG
7680 06FD 60              RTS
7690                ;
7700 06FE 60        ENDBOX RTS
7710                ;
7720 06FF 60        STRBOX RTS
7730                ;
7740 0700 A200      NORHIG LDX #$00
7750 0702 8E07E4          STX HEIFLG    ;RESET HEIGHT FLAG
7760 0705 60              RTS
7770                ;
7780 0706 A210      DUBHIG LDX #$10
7790 0708 8E07E4          STX HEIFLG    ;SET DOUBLE HEIG
7800 070B 8E08E4          STX HEIGHC    ;DOUBLE IN THIS LINE
7810 070E 60              RTS
7820                ;
7830 070F AE09E4 CONDIS   LDX REVFLG    ;
7840 0712 D005            BNE CONDI1    ;=>REVEAL MODE
7850 0714 A200            LDX #$00
7860 0716 8E04E4          STX CONFLG    ;SET CONCEAL MODE
7870 0719 60        CONDI1 RTS
7880                ;
7890 071A A200      GRANOR LDX #$00
7900 071C 8E01E4          STX SEPFLG
7910 071F 60              RTS
7920                ;
7930 0720 A2FF      GRASEP LDX #$FF
7940 0722 8E01E4          STX SEPFLG
7950 0725 60              RTS
7960                ;
7970 0726 A200      BACKBL LDX #$00
7980 0728 8E06E4          STX COLBAC    ;SET BACKGROUND COLOR
7990 072B 60              RTS
8000                ;
8010 072C AE05E4 BACKNW   LDX COLCHA    ;LOAD CHARACTER COLOR
8020 072F 8E06E4          STX COLBAC    ;NEW BACKGROUND
8030 0732 60              RTS
8040                ;
8050 0733 A210      GRAHOL LDX #$10
8060 0735 8E02E4          STX GRAHOC    ;SET GRAHOLD MODE
8070 0738 60              RTS
8080                ;
8090 0739 A200      GRAREL LDX #$00
8100 073B 8E02E4          STX GRAHOC    ;RESET HOLD MODE
8110 073E 60              RTS
8120                ;
8130                ;*** COMMAND JUMP TABLES ***
8140                ;CO IS THE CONTROL CHARACTER SET
8150                ;
8160 073F 1705      CO    .WORD NULL-1      ;START NEW FRAME
8170 0741 3E06            .WORD NILL-1      ;$00
8180 0743 3E06            .WORD NILL-1      ;$02
8190 0745 3E06            .WORD NILL-1      ;$03
8200 0747 3E06            .WORD NILL-1      ;$04
8210 0749 3E06            .WORD NILL-1      ;$05
8220 074B 3E06            .WORD NILL-1      ;$06
8230 074D 2105            .WORD BELL-1      ;$07
8240 074F 2A05            .WORD BS-1        ;$08
8250 0751 4105            .WORD HT-1        ;$09
8260 0753 5A05            .WORD LF-1        ;$0A
8270 0755 8A05            .WORD VT-1        ;$0B
8280 0757 B505            .WORD FF-1        ;$0C HOME & CLEAR
8290 0759 C705            .WORD CR-1        ;$0D
8300 075B 3E06            .WORD NILL-1      ;$0E
8310 075D 3E06            .WORD NILL-1      ;$0F
8320                ;
8330 075F 3E06            .WORD NILL-1      ;$10
8340 0761 D605            .WORD CURON-1     ;$11 CURSOR ON
8350 0763 3E06            .WORD NILL-1      ;$12
8360 0765 3E06            .WORD NILL-1      ;$13
8370 0767 DA05            .WORD CUROF-1     ;$14 CURSOR OFF
8380 0769 3E06            .WORD NILL-1      ;$15
8390 076B 3E06            .WORD NILL-1      ;$16
8400 076D 3E06            .WORD NILL-1      ;$17
8410 076F E505            .WORD CAN-1       ;$18
8420 0771 3E06            .WORD NILL-1      ;$19
8430 0773 3E06            .WORD NILL-1      ;$1A
8440 0775 EC05            .WORD ESC-1       ;$1B
8450 0777 3E06            .WORD NILL-1      ;$1C
8460 0779 3E06            .WORD NILL-1      ;$1D
8470 077B F205            .WORD RS-1        ;$1E
8480 077D 3E06            .WORD NILL-1      ;$1F
8490                ;
8500                ;
8510                ;C1 IS THE DISPLAY-ATTRIBUTE CONTROLE
8520                ;CODES SET (ESCAPE SEQUENCES)
8530                ;
8540 077F F106      C1    .WORD FLASH-1
8550 0781 F706            .WORD STEADY-1
8560 0783 FD06            .WORD ENDBOX-1
8570 0785 FE06            .WORD STRBOX-1
8580 0787 FF06            .WORD NORHIG-1
8590 0789 0507            .WORD DUBHIG-1
8600 078B 3E06            .WORD NILL-1
8610 078D 3E06            .WORD NILL-1
8620                ;
8630 078F 0E07            .WORD CONDIS-1
8640 0791 1907            .WORD GRANOR-1
8650 0793 1F07            .WORD GRASEP-1
8660 0795 3E06            .WORD NILL-1
8670 0797 2507            .WORD BACKBL-1     ;BLACK BACKGROUND
8680 0799 2B07            .WORD BACKNW-1     ;NEW BACKGROUND
8690 079B 3207            .WORD GRAHOL-1     ;HOLD GRAFICS
8700 079D 3807            .WORD GRAREL-1     ;RELEASE GRAFICS
```

```
8710                    ;
8720                    ;
8730 079F 80      CRTTAB  .BYTE $80,40,$50,$08,$22,$00
8730 07A0 28
8740 07A5 18              .BYTE $18,$1C,$00,$08,$00,$09
8750 07AB 00              .BYTE $00,$00,$00,$00,40,$18
8760                    ;
8770 07B1 00      RESTAB  .BYTE $00         ;ALFANUMERIC
8780 07B2 00              .BYTE $00         ;NOT SEPERATE
8790 07B3 00              .BYTE $00         ;NO GRAFIC HOLD
8800 07B4 00              .BYTE $00         ;NO FLASHING
8810 07B5 FF              .BYTE $FF         ;NO CONCEALD PRINTING
8820 07B6 07              .BYTE $07         ;WHITE CHARACTERS
8830 07B7 00              .BYTE $00         ;BLACK BACKGROUND
8840 07B8 00              .BYTE $00         ;NORMAL HEIGHT
8850                    ;
8860 07B9 AD6323  XQUSER  LDA AHOLD     ;GET CHARACTER
8870 07BC C920            CMP #$20
8880 07BE 300D            BMI XQUSEB    ;=>CONTROL COMMAND
8890 07C0 C923            CMP #'#
8900 07C2 D005            BNE XQUSEA
8910 07C4 A95F            LDA #$5F
8920 07C6 8D6323          STA AHOLD
8930 07C9 209E03  XQUSEA  JSR PUTMOD    ;TRANSMIT CHARACTER
8940 07CC 60              RTS
8950 07CD 0A      XQUSEB  ASL A         ;ACCU*2
8960 07CE AA              TAX
8970 07CF BDD80A          LDA COMVEC+1,X
8980 07D2 48              PHA
8990 07D3 BDD70A          LDA COMVEC,X ;LOAD USERCOMMAND
9000 07D6 48              PHA           ;AND PUSH ROUTINE ADRES
9010 07D7 60              RTS           ;EXECUTE THE COMMAND
9020                    ;
9030 07D8 78      DOS     SEI
9040 07D9 A93A            LDA #$3A
9050 07DB 8D7A3A          STA SOF+1
9060 07DE A97E            LDA #$7E
9070 07E0 8D793A          STA SOF
9080 07E3 38              SEC
9090 07E4 AD7B3A          LDA EOF
9100 07E7 ED793A          SBC SOF
9110 07EA AD7C3A          LDA EOF+1
9120 07ED ED7A3A          SBC SOF+1
9130 07F0 4A              LSR A
9140 07F1 4A              LSR A
9150 07F2 4A              LSR A
9160 07F3 18              CLC
9170 07F4 6901            ADC #$01
9180 07F6 8D7D3A          STA EOF+2     ;COMPUTE FILE LENGTH
9190 07F9 207F08          JSR ERASE     ;CLEAR AND HOME
9200 07FC A93E            LDA #'>
9210 07FE 8D6323          STA AHOLD
9220 0801 206D04          JSR PRINT     ;PRINT DOS-PROMT
9230 0804 209B2C          JSR $2C9B     ;FILL INPUTBUFFER
9240 0807 A92E            LDA #$2E
9250 0809 85E2            STA $E2
9260 080B A91E            LDA #$1E
9270 080D 85E1            STA $E1       ;RESET BUFFER
9280 080F 20842A          JSR XQDOSC    ;EXECUTE COMMAND
9290 0812 58              CLI
9300 0813 60      DOSEX   RTS
9310                    ;
9320 0814 18      HELP    CLC
9330 0815 A917            LDA #HLPTXT
9340 0817 85FE            STA PRTPNT
9350 0819 E922            SBC #HLPTXE
9360 081B 8D12E4          STA PRTCNT
9370 081E A90B            LDA #HLPTXT/256
9380 0820 85FF            STA PRTPNT+1
9390 0822 E90D            SBC #HLPTXE/256
9400 0824 8D13E4          STA PRTCNT+1
9410 0827 20B00A          JSR PRTSCR
9420 082A 202103  HELP1   JSR GETKB
9430 082D 90FB            BCC HELP1     ;=>WAIT FOR KEY
9440 082F 207F08          JSR ERASE     ;CLEAR THE SCREEN
9450 0832 204208          JSR PRTCP     ;PRINT CURRENT PAGE
9460 0835 60              RTS
9470                    ;
9480 0836 A2FF    REVEAL  LDX #$FF
9490 0838 8E09E4          STX REVFLG    ;SET REVEAL FLAG
9500 083B 204208          JSR PRTCP     ;REPRINT CURRENT PAGE
9510 083E 202806          JSR RESRVF    ;RESET REVEAL FLAG
9520 0841 60              RTS
9530                    ;
9540 0842 20980A  PRTCP   JSR COMPL     ;COMPUTE PAGE LENTH
9550 0845 20B00A          JSR PRTSCR
9560 0848 60              RTS
9570                    ;
9580 0849 68      QUIT    PLA
9590 084A 68              PLA
9600 084B 20D705          JSR CURON
9610 084E A940            LDA #$40
9620 0850 8D0EE1          STA VAIER     ;RESET TIMER
9630 0853 68              PLA
9640 0854 8D1223          STA OUTV1+1
9650 0857 68              PLA
9660 0858 8D1123          STA OUTV1     ;RESET OUTPUT VECTOR
9670 085B 60              RTS
9680                    ;
9690 085C A918    CANCEL  LDA #$18      ;LOAD CTRL-X
9700 085E D00A            BNE TMCHR     ;=>ALWAYS
9710 0860 A92A    CHRINS  LDA #'*
9720 0862 D006            BNE TMCHR     ;=>ALWAYS
9730 0864 A95F    CRET    LDA #$5F
9740 0866 D002            BNE TMCHR     ;=>ALWAYS
9750 0868 A91B    ESCAPE  LDA #$1B
9760 086A 8D6323  TMCHR   STA AHOLD
9770 086D 209E03          JSR PUTMOD    ;SEND CHARACTER
9780 0870 60              RTS
9790                    ;
9800 0871 A200    DUMCOD  LDX #$00
9810 0873 BD7F0D  DUMCOE  LDA CODE,X
9820 0876 206A08          JSR TMCHR     ;TRANSMIT CHARACTER
9830 0879 E8              INX
9840 087A E00A            CPX #10
9850 087C D0F5            BNE DUMCOE    ;=>CODE NOT FINISHED
9860 087E 60              RTS
9870                    ;
9880 087F 20F305  ERASE   JSR RS        ;
9890 0882 202806          JSR RESRVF    ;RESET REVEAL MODE
9900 0885 208C06          JSR ERTEOS    ;CLEAR SCREEN
9910 0888 60              RTS
9920                    ;
9930 0889 202103  VIDIBU  JSR GETKB     ;READ COMMAND
9940 088C 90FB            BCC VIDIBU    ;WAIT FOR KEY
9950 088E AD6323          LDA AHOLD
9960 0891 C90D            CMP #$0D
9970 0893 F051            BEQ SENDNR    ;=>SEND NUMBER
9980 0895 48              PHA           ;SAVE A
9990 0896 20F305  READNR  JSR RS        ;HOME
10000 0899 208B05         JSR VT
10010 089C 208006         JSR ERTEOL    ;ERASE LINE
10020 089F 68             PLA           ;PULL CHARACTER
10030 08A0 8D6323         STA AHOLD
10040 08A3 A200           LDX #$00
10050 08A5 F012           BEQ READN3    ;=>ALWAYS
10060 08A7 8E0FE4 READN1  STX XHOLD
10070 08AA 202103 READN2  JSR GETKB
10080 08AD 90FB           BCC READN2    ;=>NO KEY
10090 08AF AE0FE4         LDX XHOLD
10100 08B2 AD6323         LDA AHOLD
10110 08B5 C90D           CMP #$0D
10120 08B7 F027           BEQ READN5    ;=>LAST
10130 08B9 C97F   READN3  CMP #DEL
10140 08BB D012           BNE READN4    ;=>NOT DELETE
10150 08BD 202B05         JSR BS
10160 08C0 AE0FE4         LDX XHOLD     ;RESTORE POSITION
10170 08C3 CA             DEX
10180 08C4 30C3           BMI VIDIBU    ;=>TRY AGAIN
10190 08C6 8E0FE4         STX XHOLD
10200 08C9 208006         JSR ERTEOL    ;ERASE LINE
10210 08CC 4CAA08         JMP READN2
10220 08CF 9D1AE4 READN4  STA NUMBER,X
10230 08D2 8E0FE4         STX XHOLD
10240 08D5 206D04         JSR PRINT     ;PRINT NUMBER
10250 08D8 AE0FE4         LDX XHOLD     ;RESTORE X
10260 08DB E8             INX
10270 08DC E010           CPX #$10
10280 08DE 30C7           BMI READN1    ;=>BUFFER NOT FULL
10290 08E0 A95F   READN5  LDA #$5F      ;LOAD '#
10300 08E2 9D1AE4         STA NUMBER,X ;CLOSE BUFFER
10310 08E5 60             RTS
10320 08E6 A200   SENDNR  LDX #$00
10330 08E8 BD1AE4         LDA NUMBER,X
10340 08EB C95F           CMP #$5F      ;TEST END
10350 08ED F014           BEQ SENDN2    ;=>NO NUMBER ENTERED
10360 08EF 8D6323 SENDN1  STA AHOLD
10370 08F2 209E03         JSR PUTMOD
10380 08F5 E8             INX
10390 08F6 BD1AE4         LDA NUMBER,X
10400 08F9 C95F           CMP #$5F
10410 08FB D0F2           BNE SENDN1    ;=>NOT LAST
10420 08FD 8D6323         STA AHOLD
10430 0900 209E03         JSR PUTMOD
```

```
10440 0903 60      SENDN2  RTS
10450                      ;
10460 0904 18      NEWFIL  CLC
10470 0905 A923            LDA #NEWTXT
10480 0907 85FE            STA PRTPNT
10490 0909 E938            SBC #NEWTXE
10500 090B 8D12E4          STA PRTCNT
10510 090E A90D            LDA #NEWTXT/256
10520 0910 85FF            STA PRTPNT+1
10530 0912 E90D            SBC #NEWTXE/256
10540 0914 8D13E4          STA PRTCNT+1
10550 0917 20B00A          JSR PRTSCR    ;'ARE YOU SURE?
10560 091A 202103 NEWFIM        JSR GETKB
10570 091D 90FB            BCC NEWFIM    ;=>WAIT FOR KEY
10580 091F AD6323          LDA AHOLD
10590 0922 295F            AND #%01011111
10600 0924 C959            CMP #'Y
10610 0926 D006            BNE NEWFIN    ;=>NO NEWFILE
10620 0928 207D0A          JSR INISOF    ;RESET START OF FILE
10630 092B 20880A          JSR INIEOF    ;RESET END OF FILE
10640 092E 207F08 NEWFIN   JSR ERASE     ;CLEAR THE SCREEN
10650 0931 204208          JSR PRTCP     ;PRINT CURRENT PAGE
10660 0934 60              RTS
10670                      ;
10680 0935 207F08 GETSCR   JSR ERASE     ;CLEAR
10690 0938 202E06          JSR RESIB     ;RESET INPUT BUFFER
10700 093B AD0BE4          LDA SOP
10710 093E 85FE            STA PRTPNT
10720 0940 AD0CE4          LDA SOP+1
10730 0943 85FF            STA PRTPNT+1  ;INIT DEST POINTER
10740 0945 AD793A          LDA SOF
10750 0948 856C            STA RAMPTR
10760 094A AD7A3A          LDA SOF+1
10770 094D 856D            STA RAMPTR+1  ;INIT SOURCE POINTER
10780 094F A200            LDX #$00
10790 0951 A16C    GETSCS  LDA (RAMPTR,X)
10800 0953 F041            BEQ GETSCX    ;=>END OF FILE
10810 0955 C90C            CMP #$0C
10820 0957 D008            BNE GETSCT    ;=>START OF SCREEN
10830 0959 E66C            INC RAMPTR
10840 095B D0F4            BNE GETSCS
10850 095D E66D            INC RAMPTR+1
10860 095F D0F0            BNE GETSCS    ;=>ALWAYS
10870 0961 A90C    GETSCT  LDA #$0C
10880 0963 81FE            STA (PRTPNT,X) ;START WITH FF
10890 0965 D010            BNE GETSCV    ;=>ALWAYS
10900 0967 A16C    GETSCU  LDA (RAMPTR,X)
10910 0969 F014            BEQ GETSCW    ;=>END OF FILE
10920 096B C90C            CMP #$0C
10930 096D F010            BEQ GETSCW    ;=>END OF SCREEN
10940 096F 81FE            STA (PRTPNT,X) ;STORE IN INPUT BUF
10950 0971 E66C            INC RAMPTR
10960 0973 D002            BNE GETSCV
10970 0975 E66D            INC RAMPTR+1
10980 0977 E6FE    GETSCV  INC PRTPNT
10990 0979 D0EC            BNE GETSCU
11000 097B E6FF            INC PRTPNT+1
11010 097D D0E8            BNE GETSCU    ;=>ALWAYS
11020 097F A56C    GETSCW  LDA RAMPTR
11030 0981 8D793A          STA SOF
11040 0984 A56D            LDA RAMPTR+1
11050 0986 8D7A3A          STA SOF+1     ;SET LOAD POINTER
11060 0989 A5FE            LDA PRTPNT
11070 098B 8D0DE4          STA EOP
11080 098E A5FF            LDA PRTPNT+1
11090 0990 8D0EE4          STA EOP+1     ;SET EOP POINTER
11100 0993 204208          JSR PRTCP     ;PRINT THE SCREEN
11110 0996 60      GETSCX  RTS
11120                      ;
11130                      ;SAVSCR STARTS PLACING $0C IN THE FILE
11140                      ;BECAUSE THIS IS USED AS SCHREEN SEPERATOR,
11150                      ;AND MAY NOT BE PART OF THE SCHREEN. IT ENDS
11160                      ;WITH PLACING 00 THIS IS THE END OF FILE MARK
11170                      ;
11180 0997 20980A SAVSCR   JSR COMPL     ;COMPUTE LENGTH
11190 099A AD12E4          LDA PRTCNT    ;GET COUNT
11200 099D D005            BNE SAVSCS    ;=>NOT EMPTY SCREEN
11210 099F AD12E4          LDA PRTCNT
11220 09A2 F061            BEQ SAVSCX    ;=>EMPTY SCHEEN
11230 09A4 AD7B3A SAVSCS   LDA EOF
11240 09A7 856C            STA RAMPTR
11250 09A9 AD7C3A          LDA EOF+1
11260 09AC 856D            STA RAMPTR+1  ;INIT TRANSF. POINTER
11270 09AE A000            LDY #$00
11280 09B0 A90C            LDA #$0C
11290 09B2 916C            STA (RAMPTR),Y ;SET SCREEN SEPERATOR
11300 09B4 E66C            INC RAMPTR
11310 09B6 D002            BNE SAVSCT
11320 09B8 E66D            INC RAMPTR+1
11330 09BA B1FE    SAVSCT  LDA (PRTPNT),Y ;GET CHARACTER
11340 09BC 916C            STA (RAMPTR),Y ;PALCA IN FILE
11350 09BE E66C            INC RAMPTR
11360 09C0 D002            BNE SAVSCU
11370 09C2 E66D            INC RAMPTR+1
11380 09C4 EE12E4 SAVSCU   INC PRTCNT
11390 09C7 D005            BNE SAVSCV
11400 09C9 EE13E4          INC PRTCNT+1
11410 09CC F029            BEQ SAVSCW    ;=>LAST CHARACTER
11420 09CE E6FE    SAVSCV  INC PRTPNT
11430 09D0 D0E8            BNE SAVSCT
11440 09D2 E6FF            INC PRTPNT+1
11450 09D4 AD0023          LDA HIMEM
11460 09D7 E903            SBC #$03      ;LITLE SPACE FOR DIR
11470 09D9 C5FF            CMP PRTPNT+1
11480 09DB 30DD            BMI SAVSCT    ;=>NOT MEMORY FULL
11490 09DD C6FE            DEC PRTPNT
11500 09DF C6FF            DEC PRTPNT+1
11510 09E1 18              CLC
11520 09E2 A939            LDA #FULTXT
11530 09E4 85FE            STA PRTPNT
11540 09E6 E95A            SBC #FULTXE
11550 09E8 8D12E4          STA PRTCNT
11560 09EB A90D            LDA #FULTXT/256
11570 09ED 85FF            STA PRTPNT+1
11580 09EF E90D            SBC #FULTXE/256
11590 09F1 8D13E4          STA PRTCNT+1
11600 09F4 20B00A          JSR PRTSCR    ;SET WARNING FULL!
11610 09F7 A900    SAVSCW  LDA #$00
11620 09F9 916C            STA (RAMPTR),Y ;SET END OF FILE
11630 09FB A56C            LDA RAMPTR
11640 09FD 8D7B3A          STA EOF
11650 0A00 A56D            LDA RAMPTR+1
11660 0A02 8D7C3A          STA EOF+1
11670 0A05 60      SAVSCX  RTS
11680                      ;
11690 0A06 AD793A UPLOAD   LDA SOF
11700 0A09 85FE            STA PRTPNT
11710 0A0B AE7A3A          LDX SOF+1
11720 0A0E 86FF            STX PRTPNT+1
11730 0A10 CD7B3A          CMP EOF
11740 0A13 D005            BNE UPLOAE    ;=>NOT LAST CHAR.
11750 0A15 EC7C3A          CPX EOF+1
11760 0A18 F02A            BEQ UPLOAI    ;=>EMPTY FILE
11770 0A1A A000    UPLOAE  LDY #$00
11780 0A1C B1FE            LDA (PRTPNT),Y ;LOAD CHARACTER
11790 0A1E 8D6323          STA AHOLD
11800 0A21 EE793A          INC SOF
11810 0A24 D003            BNE UPLOAF
11820 0A26 EE7A3A          INC SOF+1
11830 0A29 C90A    UPLOAF  CMP #$0A
11840 0A2B F0D9            BEQ UPLOAD    ;=>LINE FEED
11850 0A2D C90D            CMP #$0D
11860 0A2F D005            BNE UPLOAG    ;=>NO RETURN
11870 0A31 A95F            LDA #$5F      ;VIDIOTEX CODE #
11880 0A33 8D6323          STA AHOLD
11890 0A36 209E03 UPLOAG   JSR PUTMOD    ;SEND CHARACTER
11900 0A39 205503 UPLOAH   JSR GETMOD
11910 0A3C 90C8            BCC UPLOAD
11920 0A3E 20D103          JSR XQDATA    ;XQ CHARACTER
11930 0A41 4C390A          JMP UPLOAH
11940 0A44 60      UPLOAI  RTS
11950                      ;
11960                      ;KEY READ ONE KEY AND SEND IT UNCODED
11970 0A45 202103 KEY      JSR GETKB     ;READ KEY
11980 0A48 90FB            BCC KEY       ;=>NO KEY
11990 0A4A 209E03          JSR PUTMOD    ;TRANSMIT CHARACTER
12000 0A4D 60              RTS
12010                      ;
12020                      ;PARITY DIS/EN ABELES PARITY CHECK
12030 0A4E 18      PARITY  CLC
12040 0A4F A95B            LDA #PARTXT
12050 0A51 85FE            STA PRTPNT
12060 0A53 E97E            SBC #PARTXE
12070 0A55 8D12E4          STA PRTCNT    ;SET PRINT POINTERS
12080 0A58 A90D            LDA #PARTXT/256
12090 0A5A 85FF            STA PRTPNT+1
12100 0A5C E90D            SBC #PARTXE/256
12110 0A5E 8D13E4          STA PRTCNT+1
12120 0A61 20B00A          JSR PRTSCR    ;PRINT MESSAGE
12130 0A64 202103 PARIT1   JSR GETKB     ;GET ANSWER
12140 0A67 90FB            BCC PARIT1    ;=>NO KEY
12150 0A69 A001            LDY #PARMAS   ;PARITY MASKER
12160 0A6B AD6323          LDA AHOLD     ;GET ANSWER
12170 0A6E 295F            AND #%01011111 ;ONLY CAPITAL
```

```
12180 0A70 C944          CMP #'D
12190 0A72 D002          BNE PARIT2        ;=>ENABLE CHECK
12200 0A74 A000          LDY #$00          ;DISABLE CHECK
12210 0A76 8C17E4 PARIT2 STY PARCHK        ;SET MASKER
12220 0A79 204208        JSR PRTCP         ;PRINT PAGE
12230 0A7C 60            RTS
12240              ;
12250 0A7D A97E   INISOF LDA #SOF+5        ;RESET FILE
12260 0A7F 8D793A        STA SOF
12270 0A82 A93A          LDA #SOF+5/256
12280 0A84 8D7A3A        STA SOF+1
12290 0A87 60            RTS
12300              ;
12310 0A88 A97E   INIEOF LDA #SOF+5        ;RESET FILE
12320 0A8A 8D7B3A        STA EOF
12330 0A8D A93A          LDA #SOF+5/256
12340 0A8F 8D7C3A        STA EOF+1
12350 0A92 A900          LDA #$00
12360 0A94 8D7E3A        STA SOF+5         ;SET EOF MARK
12370 0A97 60            RTS
12380              ;
12390              ;COMPL COMPUTES THE PAGE LENGTH FOR PRTSCR
12400              ;
12410 0A98 38     COMPL  SEC
12420 0A99 AD0BE4        LDA SOP
12430 0A9C 85FE          STA PRTPNT
12440 0A9E ED0DE4        SBC EOP
12450 0AA1 8D12E4        STA PRTCNT
12460 0AA4 AD0CE4        LDA SOP+1
12470 0AA7 85FF          STA PRTPNT+1
12480 0AA9 ED0EE4        SBC EOP+1
12490 0AAC 8D13E4        STA PRTCNT+1
12500 0AAF 60            RTS
12510              ;
12520              ;PRTSCR PRINT CHARACTERS ON THE SCREEN
12530              ;USING THE VIDIOTEX PROTOCOL
12540              ;PRTPNT IS CHARACTER POINTER
12550              ;PRTCNT IS THE COUNTER (NEGATIVE)
12560              ;
12570 0AB0 AE12E4 PRTSCR LDX PRTCNT
12580 0AB3 D005          BNE PRTSC1     ;=>PRTCNT<>00
12590 0AB5 AE13E4        LDX PRTCNT+1
12600 0AB8 F01C          BEQ PRTSC3     ;=>PRTCNT=00
12610 0ABA A200   PRTSC1 LDX #$00
12620 0ABC A1FE          LDA (PRTPNT,X)
12630 0ABE 8D6323        STA AHOLD
12640 0AC1 20D103        JSR XQDATA     ;PRINT CHARACTER
12650 0AC4 EE12E4        INC PRTCNT
12660 0AC7 D005          BNE PRTSC2
12670 0AC9 EE13E4        INC PRTCNT+1
12680 0ACC F008          BEQ PRTSC3     ;=>READY
12690 0ACE E6FE   PRTSC2 INC PRTPNT
12700 0AD0 D0E8          BNE PRTSC1
12710 0AD2 E6FF          INC PRTPNT+1
12720 0AD4 D0E4          BNE PRTSC1     ;=>ALWAYS
12730 0AD6 60     PRTSC3 RTS
12740              ;
12750 0AD7 2105   COMVEC .WORD BELL-1    ;
12760 0AD9 7008          .WORD DUMCOD-1 ;A
12770 0ADB 2D06          .WORD RESIB-1  ;B
12780 0ADD 2105          .WORD BELL-1   ;C-
12790 0ADF D707          .WORD DOS-1    ;D
12800 0AE1 7E08          .WORD ERASE-1  ;E
12810 0AE3 2105          .WORD BELL-1   ;F-
12820 0AE5 3409          .WORD GETSCR-1 ;G
12830 0AE7 1308          .WORD HELP-1   ;H
12840 0AE9 7C0A          .WORD INISOF-1 ;I
12850 0AEB 2105          .WORD BELL-1   ;J-
12860 0AED 440A          .WORD KEY-1    ;K
12870 0AEF 4108          .WORD PRTCP-1  ;L
12880 0AF1 6308          .WORD CRET-1   ;M
12890 0AF3 0309          .WORD NEWFIL-1 ;N
12900 0AF5 2105          .WORD BELL-1   ;O-
12910              ;
12920 0AF7 2DDF          .WORD $DF2D    ;P  HardCopy routine
12930 0AF9 4808          .WORD QUIT-1   ;Q
12940 0AFB 3508          .WORD REVEAL-1 ;R
12950 0AFD 9609          .WORD SAVSCR-1 ;S
12960 0AFF 880D          .WORD TERMI-1  ;T
12970 0B01 050A          .WORD UPLOAD-1 ;U
12980 0B03 8808          .WORD VIDIBU-1 ;V
12990 0B05 2105          .WORD BELL-1   ;W-
13000 0B07 5B08          .WORD CANCEL-1 ;X
13010 0B09 4D0A          .WORD PARITY-1 ;Y
13020 0B0B 5F08          .WORD CHRINS-1 ;Z
13030 0B0D 6708          .WORD ESCAPE-1
13040 0B0F 2105          .WORD BELL-1   ;-
13050 0B11 2105          .WORD BELL-1   ;-
13060 0B13 2105          .WORD BELL-1   ;-
13070 0B15 2105          .WORD BELL-1   ;-
13080              ;
13090 0B17 0C     HLPTXT .BYTE $0C,$0A
13100 0B19 4F            .BYTE 'OPTIONAL COMMANDS:'
13110 0B2B 0A            .BYTE $0A,$0D,$0A
13120 0B2E 5E            .BYTE $5E,'A   Send access-codes'
13130 0B44 0A            .BYTE $0A,$0D
13140 0B46 5E            .BYTE $5E,'B   Reset page buffer'
13150 0B5C 0A            .BYTE $0A,$0D
13160 0B5E 5E            .BYTE $5E,'D   DOS-Command',$0A,$0D
13170 0B70 5E            .BYTE $5E,'E   Erase the screen'
13180 0B85 0A            .BYTE $0A,$0D
13190 0B87 5E            .BYTE $5E,'G   Get screen from file'
13200 0BA0 0A            .BYTE $0A,$0D
13210 0BA2 5E            .BYTE $5E,'I   Reset start of file'
13220 0BBA 0A            .BYTE $0A,$0D
13230 0BBC 5E            .BYTE $5E,'K   Read and send key '
13240 0BD3 75            .BYTE 'uncoded',$0A,$0D
13250 0BDC 5E            .BYTE $5E,'L   List current page'
13260 0BF2 0A            .BYTE $0A,$0D
13270 0BF4 5E            .BYTE $5E,'N   Clears current file'
13280 0C0C 0A            .BYTE $0A,$0D
13290 0C0E 5E            .BYTE $5E,'P   Print page on printer'
13300 0C28 0A            .BYTE $0A,$0D
13310 0C2A 5E            .BYTE $5E,'Q   Leave the programme '
13320 0C43 0A            .BYTE $0A,$0D
13330 0C45 5E            .BYTE $5E,'R   Reveal: shows the'
13340 0C5B 20            .BYTE ' hidden info',$0A,$0D
13350 0C69 5E            .BYTE $5E,'S   Save screen in file'
13360 0C81 0A            .BYTE $0A,$0D
13370 0C83 5E            .BYTE $5E,'T   Start terminal proto'
13380 0C9C 63            .BYTE 'col 40/80 col.',$0A,$0D
13390 0CAC 5E            .BYTE $5E,'U   Upload currend file'
13400 0CC4 0A            .BYTE $0A,$0D
13410 0CC6 5E            .BYTE $5E,'V   Store and send VIDIBUS'
13420 0CE1 20            .BYTE ' number',$0A,$0D
13430 0CEA 5E            .BYTE $5E,'Y   Dis/En-able parity '
13440 0D02 63            .BYTE 'check',$0A,$0D
13450 0D09 1E            .BYTE $1E,$0B
13460 0D0B 48            .BYTE 'Hit any key to continue'
13470 0D22 20     HLPTXE .BYTE $20
13480              ;
13490 0D23 1E     NEWTXT .BYTE $1E,$0B
13500 0D25 07            .BYTE $07,'Is the file save? '
13510 0D38 20     NEWTXE .BYTE $20
13520              ;
13530 0D39 1E     FULTXT .BYTE $1E,$0B
13540 0D3B 07            .BYTE $07,'File overflow Screen '
13550 0D51 6E            .BYTE 'not saved'
13560 0D5A 20     FULTXE .BYTE $20
13570              ;
13580 0D5B 1E     PARTXT .BYTE $1E,$0B
13590 0D5D 07            .BYTE $07,'E)nable or D)isable '
13600 0D72 70            .BYTE 'parity check'
13610 0D7E 20     PARTXE .BYTE $20
13620              ;
13630 0D7F 30     CODE   .BYTE '0123456789'
13640              ;
13650              ;END OF VIDIOTEX PROGRAMME
13660              ;
13670 E7DF=       FORMAT=$E7DF
13680 F3E1=       MOVE  =$F3E1        ;($F435)OLD MOVCRT
13690              ;
13700 0D89 18     TERMI  CLC
13710 0D8A A9FA          LDA #TERTXT
13720 0D8C 85FE          STA PRTPNT
13730 0D8E E930          SBC #TERTXE
13740 0D90 8D12E4        STA PRTCNT
13750 0D93 A90D          LDA #TERTXT/256
13760 0D95 85FF          STA PRTPNT+1
13770 0D97 E90E          SBC #TERTXE/256
13780 0D99 8D13E4        STA PRTCNT+1  ;SET TEXT POINTERS
13790 0D9C 20B00A        JSR PRTSCR    ;PRINT TERMI TEXT
13800 0D9F 202103 TERMI1 JSR GETKB     ;READ ANSWER
13810 0DA2 90FB          BCC TERMI1    ;=>NO KEY
13820 0DA4 AD6323        LDA AHOLD
13830 0DA7 C90D          CMP #$0D
13840 0DA9 F008          BEQ TERMI2    ;=>40 COLOMN
13850 0DAB A900          LDA #$00
13860 0DAD 8DDFE7        STA FORMAT
13870 0DB0 20E1F3        JSR MOVE
13880 0DB3 A940   TERMI2 LDA #$40
13890 0DB5 8D0EE1        STA VAIER     ;RESET TIMER
13900 0DB8 A900          LDA #$00
13910 0DBA 8D17E4        STA PARCHK    ;NO PARITY CHECK
```

```
13920 ODBD 202FF3          JSR RESET      ;RESET SCREEN
13930 ODC0 205503 TERMI3   JSR GETMOD
13940 ODC3 9016            BCC TERMI5     ;=>NO MODEM INPUT
13950 ODC5 AD6323          LDA AHOLD      ;GET CHARACTER
13960 ODC8 C90A            CMP #$0A
13970 ODCA F00F            BEQ TERMI5     ;=>LINE FEED
13980 ODCC C90D            CMP #$0D
13990 ODCE D008            BNE TERMI4     ;=>NO CR
14000 ODD0 2000F0          JSR VIDEO      ;SEND CR
14010 ODD3 A90A            LDA #$0A       ;LOAD LF
14020 ODD5 8D6323          STA AHOLD
14030 ODD8 2000F0 TERMI4   JSR VIDEO      ;PRINT
14040 ODDB 202103 TERMI5   JSR GETKB      ;READ KB
14050 ODDE 90E0            BCC TERMI3     ;=>NO KEY
14060 ODE0 AD6323          LDA AHOLD      ;GET CHARACTER
14070 ODE3 C916            CMP #$16
14080 ODE5 D006            BNE TERMI6     ;NO CTRL V
14090 ODE7 68              PLA
14100 ODE8 68              PLA
14110 ODE9 78              SEI            ;DISABLE INTERRUPTS
14120 ODEA 4C1802          JMP INITAM     ;BACK TO VIDIOTEX
14130 ODED C911    TERMI6  CMP #$11
14140 ODEF F006            BEQ TERMI7     ;=>CTRL Q
14150 ODF1 209E03          JSR PUTMOD     ;SEND CHARACTER
14160 ODF4 4CC00D          JMP TERMI3
14170 ODF7 4C4908 TERMI7   JMP QUIT       ;LEAVE
14180                      ;
14190 ODFA 1E      TERTXT  .BYTE $1E,$0B,$0B
14200 ODFD 07              .BYTE $07,'Type <ret> for 40 or '
14210 OE13 0A              .BYTE $0A,$0D
14220 OE15 61              .BYTE 'an other key for 80 columns'
14230 OE30 20      TERTXE  .BYTE $20
```



Literature: (1) Viditel techniek voor de abonnee apparatuur
Opgesteld door: J.J.M. Blokland
Een uitgave van VIDITEL T&O

DISKETTES 40 TRS, SS,DD. (To order, mention the format) Because OHIO-DOS is part of the system on bootable disks and is not placed in the public domain you must prove you bought it yourself, by sending copy of the invoice, before we can deliver the diskettes.

Bootable DISK with Micro-ADE V2.0A.
The Micro-ADE program cleaned up by relocating and removing of all unnecessary 'garbage' and upgraded to Version 2.0A (separated manual and complete commented assembly source-listing of V2.0 with special track zero and review of all commands available in the paperware service, ask for prices).
Send empty diskette with label and R/W prot. and cheque of Hfl. 22,00 (eurocheque 12,50) to W.L. van Pelt.

For EC65 alias OCTOPUS alias SAMSOM65 and JUNIOR with OS-65D OHIO Disk Operating System
VIDEOTEX is the international name for what is called VIDITEL by the Dutch PTT.
In issue 47 of Dec 1986 Coen Boltjes published the first part of the article to give you the possibility to comunicate with Videotex databanks, like Elektor's bulletin board and others.
In issue 48 of Febr 1987 he published the VIDEOTEX-programme with available commands like Acces, Buffer, Dos, Erase, Get, Help, Init, Key, List, New, Print, Quit, Reveal, Save, Terminal, Vidibus, etc. A download programme is to be developed.
SAVE YOUR TIME AND ORDER THE SOFTWARE !
Send empty diskette with label and R/W-prot. and cheque of Hfl. 22,00 (eurocheque = Hfl. 12,50) to W.L. van Pelt.

```
Microware OS-9/68000 Resident Macro Assembler V1.5  86/12/30  19:36  Page     1
 expression.a
expression - comment
00001                           nam       expression
00002                           ttl       comment
00003
00010
00011  00000001 Edition     equ       1
00012  00000101 Typ_Lang    equ       (Prgrm<<8)+Objct
00013  00008000 Attr_Rev    equ       (ReEnt<<8)+0
00014
00015                           psect     Expression,Typ_Lang,Attr_Rev,Edition,0,Main
00016
00017 ********************************************************************************
00018 *
00019 *                         Calculate floating-point expressions
00020 *
00021 *                         ***************************************
00022 *                         *           G. van Opbroek            *
00023 *                         *           Bateweg 60                *
00024 *                         *           2481 AN Woubrugge         *
00025 *                         *           Tel: 01729-8638           *
00026 *                         ***************************************
00027 * History:
00028 *       1980 P.v.d.Zee,G.v.Opbroek: First version in Pascal for Cyber 170/760
00029 * Aug 1986 G.V.Opbroek           : 68000-assembler for mc68000 with monitor
00030 * Dec 1986 G.v.Opbroek  V1.1     : 68000-assembler for OS9/68K
00031 *
00032 * Dit programma berekent floating-point uitdrukkingen die aan de onder-
00033 * staande syntax voldoen. Het programma is geschreven op een mc 68000
00034 * systeem met OS9/68K als operating systeem. Om de floating-point bereke-
00035 * ningen uit te voeren, maakt het programma gebruik van system calls naar de
00036 * bij OS9/68K behorende Math Module.
00037 * Een te berekenende uitdrukking wordt ingevoerd vanaf het toetsenbord, en
00038 * het antwoord wordt op het beeldscherm afgedrukt. Een uitdrukking mag meer-
00039 * dere niveaus haakjes bevatten. Een uitdrukking wordt afgesloten met <CR>.
00040 *
00041 * This program calculates floating-point expressions that have the
00042 * following syntax:
00043 *
00044 * <expression>               ::= <term> |
00045 *                                <expression> <adding operator> <term>
00046 * <adding operator>          ::= + | -
00047 * <term>                     ::= <factor> |
00048 *                                <term> <multiplying operator> <factor>
00049 * <multiplying operator>     ::= * | /
00050 * <factor>                   ::= <primary> |
00051 *                                <factor> <exponentiation operator> <primary>
00052 * <exponentiation operator>  ::= ^
00053 * <primary>                  ::= <real> | <p_expression> |
00054 *                                <function name> <p_expression>
00055 * <function name>            ::= SIN  | COS  | TAN  | ASIN | ACOS | ATAN |
00056 *                                LN   | LOG  | SQRT | EXP
00057 * <p_expression>             ::= ( <expression> )
00058 * <real>                     ::= <unsigned real> | <sign> <unsigned real>
00059 * <sign>                     ::= + | -
00060 * <unsigned real>            ::= <mantissa> | <mantissa> <exp_char> <exponent>
00061 * <exp_char>                 ::= e | E
00062 * <mantissa>                 ::= <constant> | <point> <constant> |
00063 *                                <constant> <point> |
00064 *                                <constant> <point> <constant>
00065 * <point>                    ::= .
00066 * <exponent>                 ::= <constant> | <sign> <constant>
00067 * <constant>                 ::= <digit> | <digit> <constant>
00068 * <digit>                    ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
00069 *
00070 * An expression is terminated by <CR>.
00071 *
00072 * The program uses the following OS9-system calls:
00073 *
00074 *   I$ReadLn        I$Write
```

```
Microware OS-9/68000 Resident Macro Assembler V1.5  86/12/30  19:36  Page     2
 expression.a
expression - comment
00075 *   F$TLink          F$Exit
00076 *
00077 * These system calls are made by the macro os9.
00078 * This macro is a synonym for:
00079 *       trap    0
00080 *       dc.w    nnnn
00081 *
00082 * The Math Module (Floating-point library) is accessed by the following
00083 * user calls
00084 *
00085 *   T$DAdd  T$DSub  T$DMul  T$DDiv  T$Power
00086 *   T$Sin   T$Cos   T$Tan   T$Asn   T$Acs   T$Atn   T$Log   T$Log10
00087 *   T$Exp   T$Sqrt  T$AtoD  T$DtoA  T$LtoA
00088 *
00089 * Calls to the Math Module are made by the macro
00090 *       tcall   T$Math1,nnnn        or
00091 *       tcall   T$Math2,nnnn
00092 * These macro's are resp. synonym for:
00093 *       trap    15
00094 *       dc.w    nnnn                or
00095 *       trap    14
00096 *       dc.w    nnnn
00097 *
00098 * Note: In OS9, a floating-point library is included. For other systems, a
00099 *       floating-point library has to be added to this program. A HEX-dump
00100 *       of a library can be found in: mc 2/1985.
00101 *
00102 ********************************************************************************
00103
00104                         ttl     macro_definitions
00105
00106             Push_res    macro                   Push result of fp-calculation
00107                         move.l  d0,-(a7)
00108                         move.l  d1,-(a7)
00109                         endm
00110
00111             Pull_res    macro                   Pull fp-result from stack
00112                         move.l  (a7)+,d1
00113                         move.l  (a7)+,d0
00114                         endm
00115
00116             Copy_reg    macro                   Copy fp-registers d0,d1 to d2,d3
00117                         move.l  d0,d2
00118                         move.l  d1,d3
00119                         endm
00120
00121             Prec_load   macro                   Write precision to d2,d3
00122                         lea     Precision(a6),a2
00123                         move.l  (a2)+,d2
00124                         move.l  (a2)+,d3
00125                         endm
00126
00127                         ttl     data
00128                         vsect
00129
00130 0000 3d06 Precision    dc.l    $3D06849B       Precision in function calls: 1E-14
00131 0004 86a1             dc.l    $86A12B9C
00132
00133  00000050 IO_Len       equ     80              IO-buffer length
00134  00000000 IO_Buff      ds.b    IO_Len          80 Bytes IO-buffer
00135
00136  00000008             ends
00137
00138 0000 4d41 Math_name    dc      "MATH",0        Name of math module
00139  00000000 Inp_ch       equ     0               OS9 standard input channel
00140  00000001 Out_ch       equ     1               OS9 standard output channel
00141
00142 0006 4552 Err_msg      dc      "ERROR",$0d,$0a,0
```

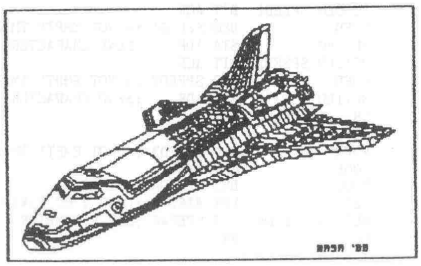
```
Microware OS-9/68000 Resident Macro Assembler V1.5  86/12/30  19:36  Page     3
 expression.a
expression - data
00143  0000000c Err_len    equ       *-Err_msg
00144
00145                      ttl       main_program
00146
00147 0012=303c Main       move.w    #T$Math1,d0     Trap number for math module
00148 0016 7200            moveq     #0,d1           No optional memory override
00149 0018 41fa            lea       Math_name(pc),a0
00150 001c=4e40            os9       F$TLink
00151
00152 0020=303c            move.w    #T$Math2,d0     Trap number for math module (functions)
00153 0024 7200            moveq     #0,d1           No optional memory override
00154 0026 41fa            lea       Math_name(pc),a0
00155 002a=4e40            os9       F$TLink
00156
00157 002e 7000 M_Loop     moveq     #Inp_ch,d0      Path_number
00158 0030 7250            moveq     #IO_Len,d1      Maximum numbers of bytes to be read
00159 0032 41ee            lea       IO_Buff(a6),a0 pointer on IO_Buffer
00160 0036=4e40            os9       I$ReadLn
00161 003a 6526            bcs.s     Main_end        On error, goto Main_end
00162
00163 003c 41ee            lea       IO_Buff(a6),a0 pointer on IO_Buffer
00164 0040 6124            bsr.s     Expression      Evaluate expression
00165 0042 6506            bcs.s     M_error         Carry set --> error
00166 0044 0c10            cmpi.b    #$0d,(a0)       End-of-line?
00167 0048 6712            beq.s     No_error        Yes, no error
00168
00169 004a 41fa M_error    lea       Err_msg(pc),a0 Write error message
00170 004e 7001            moveq     #Out_ch,d0      Path_number
00171 0050 223c            move.l    #Err_len,d1
00172 0056=4e40            os9       I$Write
00173 005a 60d2            bra.s     M_Loop
00174
00175 005c 6100 No_error   bsr       Wr_Real         Write answer
00176 0060 60cc            bra.s     M_Loop
00177
00178 0062=4e40 Main_end   os9       F$Exit
00179
00180                      ttl       subroutines
00181
00182           Expression
00183
00184 * Expression reads and calculates expression
00185
00186 0066 6100            bsr       Term            Read/calculate first term
00187 006a 6552            bcs.s     Exp_exit        On error; return
00188 006c 0c10 Exp_loop   cmpi.b    #'+',(a0)       Addition ?
00189 0070 661c            bne.s     T_Sub           When not, try subtraction
00190 0072 5288            addq.l    #1,a0           Skip +
00191                      Push_res
00192 0078 6100            bsr       Term            Read/calculate next term
00193 007c 6532            bcs.s     Exp1            On error, clear stack and return
00194                      Copy_reg
00195                      Pull_res
00196 0086=4e40            tcall     T$Math1,T$DAdd
00197 008a 64e0            bcc.s     Exp_loop        No error, continue
00198 008c 6030            bra.s     Exp_exit
00199 *
00200 008e 0c10 T_Sub      cmpi.b    #'-',(a0)       Subtraction ?
00201 0092 6626            bne.s     No_subion       When not, return
00202 0094 5288            addq.l    #1,a0           Skip -
00203                      Push_res
00204 009a 6100            bsr       Term            Read/calculate next term
00205 009e 6510            bcs.s     Exp1            On error, clear stack and return
00206                      Copy_reg
00207                      Pull_res
00208 00a8=4e40            tcall     T$Math1,T$DSub
00209 00ac 64be            bcc.s     Exp_loop        No error, continue
00210 00ae 600e            bra.s     Exp_exit
```

```
Microware OS-9/68000 Resident Macro Assembler V1.5  86/12/30  19:36  Page     4
 expression.a
expression - subroutines
00211                Exp1       Pull_res
00212 00b4 003c                 ori.b    #%00000001,ccr Set carry
00213 00b8 6004                 bra.s    Exp_exit       Exit factor
00214 00ba 023c No_subion       andi.b   #%11111110,ccr Clear carry
00215 00be 4e75 Exp_exit        rts
00216
00217                Term
00218
00219 * Term reads and calculates terms
00220
00221 00c0 6100                 bsr      Factor         Read/calculate first factor
00222 00c4 6552                 bcs.s    Ter_exit       On error; return
00223 00c6 0c10 Ter_loop        cmpi.b   #'*',(a0)      Multiplication ?
00224 00ca 661c                 bne.s    T_Div          When not, try division
00225 00cc 5288                 addq.l   #1,a0          Skip *
00226                           Push_res
00227 00d2 6100                 bsr      Factor         Read/calculate next factor
00228 00d6 656a                 bcs.s    Fac1           On error, clear stack and return
00229                           Copy_reg
00230                           Pull_res
00231 00e0=4e40                 tcall    T$Math1,T$DMul
00232 00e4 64e0                 bcc.s    Ter_loop       No error, continue
00233 00e6 6030                 bra.s    Ter_exit
00234 *
00235 00e8 0c10 T_Div           cmpi.b   #'/',(a0)      Division ?
00236 00ec 6626                 bne.s    No_divion      When not, return
00237 00ee 5288                 addq.l   #1,a0          Skip /
00238                           Push_res
00239 00f4 6100                 bsr      Factor         Read/calculate next factor
00240 00f8 6548                 bcs.s    Fac1           On error, clear stack and return
00241                           Copy_reg
00242                           Pull_res
00243 0102=4e40                 tcall    T$Math1,T$DDiv
00244 0106 64be                 bcc.s    Ter_loop       No error, continue
00245 0108 600e                 bra.s    Ter_exit
00246                Ter1       Pull_res
00247 010e 003c                 ori.b    #%00000001,ccr Set carry
00248 0112 6004                 bra.s    Ter_exit       Exit factor
00249 0114 023c No_divion       andi.b   #%11111110,ccr Clear carry
00250 0118 4e75 Ter_exit        rts
00251
00252                Factor
00253
00254 * Factor reads and calculates factors
00255
00256 011a 6100                 bsr      Primary        Read/calculate first primary
00257 011e 6530                 bcs.s    Fac_exit       On error; return
00258 0120 0c10 Fac_loop        cmpi.b   #'^',(a0)      exponentiation ?
00259 0124 6626                 bne.s    No_exption     When not, return
00260 0126 5288                 addq.l   #1,a0          Skip ^
00261                           Push_res
00262 012c 6100                 bsr      Primary        Read/calculate next primary
00263 0130 6510                 bcs.s    Fac1           On error, clear stack and return
00264                           Copy_reg
00265                           Pull_res
00266 013a=4e40                 tcall    T$Math2,T$Power
00267 013e 64e0                 bcc.s    Fac_loop       No error, continue
00268 0140 600e                 bra.s    Fac_exit
00269                Fac1       Pull_res
00270 0146 003c                 ori.b    #%00000001,ccr Set carry
00271 014a 6004                 bra.s    Fac_exit       Exit factor
00272 014c 023c No_exption      andi.b   #%11111110,ccr Clear carry
00273 0150 4e75 Fac_exit        rts
00274
00275                Primary
00276
00277 * Primary reads and calculates primaries
00278
```

```
Microware OS-9/68000 Resident Macro Assembler V1.5  86/12/30  19:36  Page     5
 expression.a
expression - subroutines
00279 0152 6100              bsr       Skip_blancs     Skip leading blancs
00280 0156=4e40              tcall     T$Math1,T$AtoD
00281 015a 6410              bcc.s     Prim_exit       Found real, return
00282 015c 0c10              cmpi.b    #'(',(a0)       Expression in parenthesis?
00283 0160 6606              bne.s     Prim1
00284 0162 6100              bsr       P_expres        Yes, read expression
00285 0166 6004              bra.s     Prim_exit       exit primary
00286 0168 6100 Prim1        bsr       Read_fie        Not an expression, try function
00287 016c 6504 Prim_exit    bcs.s     Prim_e1         On error, return
00288 016e 6100              bsr       Skip_blancs     Skip trailing blancs
00289 0172 4e75 Prim_e1      rts
00290
00291                Read_fie
00292
00293 * Read_fie reads and calculates functions
00294
00295 0174 2248              move.l    a0,a1           Save pointer
00296 0176 1818              move.b    (a0)+,d4        Get first character
00297 0178 7a02              moveq     #2,d5           Read another three caracters
00298 017a e18c R_fie1       lsl.l     #8,d4           Shift 1 byte
00299 017c 1818              move.b    (a0)+,d4        Read the next character
00300 017e b83c              cmp.b     #'A',d4         < 'A' ?
00301 0182 6d06              blt.s     R_fie3          no, continue
00302 0184 b83c              cmp.b     #'Z',d4         > 'Z' ?
00303 0188 6f06              ble.s     R_fie2          no, continue
00304 018a 5388 R_fie3       subq.l    #1,a0           Yes, adjust pointer
00305 018c 183c              move.b    #' ',d4         Add space to d4
00306 0190 51cd R_fie2       dbra      d5,R_fie1       Read until four characters in d4
00307
00308 0194 b8bc T_sin        cmp.l     #'SIN ',d4
00309 019a 6618              bne.s     T_cos
00310 019c 6100              bsr       P_expres        Calculate argument of function
00311 01a0 6500              bcs       Rf_exit         On error, exit Read_fie
00312                        Prec_load
00313 01ac=4e40              tcall     T$Math2,T$Sin
00314 01b0 6000              bra       Rf_exit
00315
00316 01b4 b8bc T_cos        cmp.l     #'COS ',d4
00317 01ba 6618              bne.s     T_tan
00318 01bc 6100              bsr       P_expres        Calculate argument of function
00319 01c0 6500              bcs       Rf_exit         On error, exit Read_fie
00320                        Prec_load
00321 01cc=4e40              tcall     T$Math2,T$Tan
00322 01d0 6000              bra       Rf_exit
00323
00324 01d4 b8bc T_tan        cmp.l     #'TAN ',d4
00325 01da 6618              bne.s     T_asin
00326 01dc 6100              bsr       P_expres        Calculate argument of function
00327 01e0 6500              bcs       Rf_exit         On error, exit Read_fie
00328                        Prec_load
00329 01ec=4e40              tcall     T$Math2,T$Tan
00330 01f0 6000              bra       Rf_exit
00331
00332 01f4 b8bc T_asin       cmp.l     #'ASIN',d4
00333 01fa 6618              bne.s     T_acos
00334 01fc 6100              bsr       P_expres        Calculate argument of function
00335 0200 6500              bcs       Rf_exit         On error, exit Read_fie
00336                        Prec_load
00337 020c=4e40              tcall     T$Math2,T$Asn
00338 0210 6000              bra       Rf_exit
00339
00340 0214 b8bc T_acos       cmp.l     #'ACOS',d4
00341 021a 6618              bne.s     T_atan
00342 021c 6100              bsr       P_expres        Calculate argument of function
00343 0220 6500              bcs       Rf_exit         On error, exit Read_fie
00344                        Prec_load
00345 022c=4e40              tcall     T$Math2,T$Acs
00346 0230 6000              bra       Rf_exit
```

```
Microware OS-9/68000 Resident Macro Assembler V1.5  86/12/30  19:36  Page     6
 expression.a
expression - subroutines
00347
00348 0234 b8bc T_atan      cmp.l    #'ATAN',d4
00349 023a 6618              bne.s    T_ln
00350 023c 6100              bsr      P_expres        Calculate argument of function
00351 0240 6500              bcs      Rf_exit         On error, exit Read_fie
00352                        Prec_load
00353 024c=4e40              tcall    T$Math2,T$Atn
00354 0250 6000              bra      Rf_exit
00355
00356 0254 b8bc T_ln        cmp.l    #'LN  ',d4
00357 025a 6618              bne.s    T_log
00358 025c 6100              bsr      P_expres        Calculate argument of function
00359 0260 6500              bcs      Rf_exit         On error, exit Read_fie
00360                        Prec_load
00361 026c=4e40              tcall    T$Math2,T$Log
00362 0270 6000              bra      Rf_exit
00363
00364 0274 b8bc T_log       cmp.l    #'LOG ',d4
00365 027a 6618              bne.s    T_sqrt
00366 027c 6100              bsr      P_expres        Calculate argument of function
00367 0280 6500              bcs      Rf_exit         On error, exit Read_fie
00368                        Prec_load
00369 028c=4e40              tcall    T$Math2,T$Log10
00370 0290 6000              bra      Rf_exit
00371
00372 0294 b8bc T_sqrt      cmp.l    #'SQRT',d4
00373 029a 6618              bne.s    T_exp
00374 029c 6100              bsr      P_expres        Calculate argument of function
00375 02a0 6500              bcs      Rf_exit         On error, exit Read_fie
00376                        Prec_load
00377 02ac=4e40              tcall    T$Math2,T$Sqrt
00378 02b0 6000              bra      Rf_exit
00379
00380 02b4 b8bc T_exp       cmp.l    #'EXP ',d4
00381 02ba 6618              bne.s    T_oth
00382 02bc 6100              bsr      P_expres        Calculate argument of function
00383 02c0 6500              bcs      Rf_exit         On error, exit Read_fie
00384                        Prec_load
00385 02cc=4e40              tcall    T$Math2,T$Exp
00386 02d0 6000              bra      Rf_exit
00387
00388 02d4 2049 T_oth       move.l   a1,a0           Restore pointer: no function match
00389 02d6 4e75 Rf_exit     rts
00390
00391           P_expres
00392
00393 * Calculate expressions within parenthesis
00394
00395 02d8 6100              bsr      Skip_blancs     Skip blancs
00396 02dc 0c18              cmpi.b   #'(',(a0)+      ( ?
00397 02e0 6610              bne.s    Arg_err         No, argument error
00398 02e2 6100              bsr      Expression      Calculate argument
00399 02e6 6100              bsr      Skip_blancs     Skip blancs
00400 02ea 0c18              cmpi.b   #')',(a0)+      ) ?
00401 02ee 6602              bne.s    Arg_err         No, argument error
00402 02f0 4e75              rts
00403 02f2 003c Arg_err     ori.b    #%00000001,ccr  Set carry
00404 02f6 4e75              rts
00405
00406           Wr_Real
00407
00408 * Convert real in d0,d1 to ASCII string
00409 * First position is for sign
00410
00411 02f8 41ee              lea      IO_Buff(a6),a0
00412 02fc 2248              move.l   a0,a1           Save start of mantissa
00413 02fe 10fc              move.b   #'+',(a0)+      Sign field
00414 0302 5288              addq.l   #1,a0           Save place for decimal point
```

```
Microware OS-9/68000 Resident Macro Assembler V1.5  86/12/30  19:36  Page     7
 expression.a
expression - subroutines
00415 0304 243c             move.l   #$7fff000a,d2  Format
00416 030a=4e40             tcall    T$Math1,T$DtoA
00417 *
00418 * T$DtoA gives the mantissa in the format .nnnnnnnnnn
00419 * we adjust the output to n.nnnnnnnnn
00420 * The exponent must be lowered by one afterwards
00421 *
00422 030e 6a04                bpl.s    Wr_mpos          Positive mantissa
00423 0310 12bc                move.b   #'-',(a1)        Adjust sign
00424
00425 0314 1150 Wr_mpos       move.b   (a0),-1(a0)      Shift first digit
00426 0318 10bc              move.b   #'.',(a0)        Write dicimal point
00427 031c d1fc              add.l    #10,a0           Now a0 points after last digit
00428 0322 10fc              move.b   #'E',(a0)+       Exponent field
00429
00430 0326 2248              move.l   a0,a1            Save start of exponent
00431 0328 10fc              move.b   #'+',(a0)+       Sign of exponent
00432 032c 5380              subq.l   #1,d0            Adjust exponent for first digit
00433 032e=4e40              tcall    T$Math1,T$LtoA
00434 0332 6a04              bpl.s    Wr_epos          Positive exponent
00435 0334 12bc              move.b   #'-',(a1)        Adjust sign of exponent
00436
00437 0338 5289 Wr_epos      addq.l   #1,a1            a1 now points to first digit of exp
00438 *
00439 * T$LtoA gives 10 digits; we want a three digit exponent;
00440 * a0 points to start of exponent-string
00441 *
00442 033a 5e88                addq.l   #7,a0            first significant digit
00443 033c 12d8                move.b   (a0)+,(a1)+      move digit
00444 033e 12d8                move.b   (a0)+,(a1)+      move digit
00445 0340 12d8                move.b   (a0)+,(a1)+      move digit
00446
00447 0342 12fc                move.b   #$0d,(a1)+       add CR
00448 0346 12fc                move.b   #$0A,(a1)+       add LF
00449 034a 4211                clr.b    (a1)             add End-of-string
00450
00451 034c 41ee                lea      IO_Buff(a6),a0   pointer on IO_Buffer
00452 0350 7001                moveq    #Out_ch,d0       Path_number
00453 0352 2209                move.l   a1,d1            Calculate number of bytes
00454 0354 9288                sub.l    a0,d1
00455
00456 0356=4e40                os9      I$Write
00457 035a 4e75                rts
00458
00459              SKip_blancs
00460 035c 0c10                cmpi.b   #' ',(a0)        a0 points to blanc?
00461 0360 6604                bne.s    SK_exit          no, exit
00462 0362 5288                addq.l   #1,a0            yes, skip blanc
00463 0364 60f6                bra.s    SKip_blancs      and skip following blancs
00464 0366 023c SK_exit        andi.b   #%11111110,ccr Clear carry
00465 036a 4e75                rts
00466  0000036c                ends
```



64/256K - SRAM BOARD

By : Bernd Dau, Germany

Do you already know? The IC1 of the board is working without power supply! You don't believe it? That's right, I do not either. Connect pin 20 with 1a-c from the buss.

Hexdump for ACORN ELECTRON

```
 10REM ************************
 20REM *** MACHINECODE DUMP ***
 30REM *** (HEX and  ASCII) ***
 40REM *** V1.7 june 9 1986 ***
 50REM ***  by  S.Voortman  ***
 60REM ***  Beatrixweg  28  ***
 70REM *** 3253 BB  OUDDORP ***
 80REM *** The  Netherlands ***
 90REM *** Tel.  01878-3113 ***
100REM ************************
110MODE6:*FX5,0
120Line=&70:End=&72:PRNT=&74:Lenght=&75
130Dump=&80
140oswrch=&FFEE
150Phex   =&B545:REM Print A as 2-digit hex number
160Phexsp=&B562:REM Print A as hexnumber + space
170Pnewl  =&BC25:REM Print CR/LF
180FOR pass=0 TO 2 STEP 2
190P%=&B00
200[OPT pass
210.init
220LDA PRNT     \Printer or screen?
230BEQ init8    \only screen
240LDA #&10     \Init for 16 bytes
250STA Lenght   \and store
260LDA #2       \Printer
270JSR oswrch   \on
280JMP start
290.init8
300LDA #14      \Code for
310JSR oswrch   \paged mode
320LDA #8       \Set counter
330STA Lenght   \and store
340.start
350LDX #6       \Print 6
360LDA #32      \spaces
370.l
380JSR oswrch
390DEX          \X=X-1
400BNE l        \X>0, next
410LDA #0       \Print 00, 01, etc.
420TAX          \Copy to X
430.loop
440JSR Phexsp
450INX          \X=X+1
460TXA          \Copy to A
470CMP Lenght   \A=Lenght?
480BNE loop     \no, print next
490JSR Pnewl    \Inserte line
500.newline
510JSR Pnewl    \Start on new line
520BIT &FF      \check for Escape
530BMI escape   \handle it
540LDA Line+1   \Print
550JSR Phex     \memory address
560LDA Line     \of
570JSR Phexsp   \current line
580LDA #32      \Print
590JSR oswrch   \space
600LDX Lenght   \Set
610LDY #0       \counters
620.dump
630LDA (Line),Y\Get byte of line
640STA Dump,Y   \store for dumping
650JSR Phexsp   \print in hex + space
660INY          \Update
670DEX          \counters
680BNE dump     \not zero, next
690.ascii
700LDX Lenght   \Set
710LDY #0       \counters
720LDA #32      \Print extra space
730JSR oswrch   \after hex
```

Tips and tricks for the EC65(K)/ Junior

By C.J. Boltjes
Nw. Plantage 9
2611 XH Delft

If you want to run your computer on 2 MC a parameter in a 10 milisecond loop must be changed. $267B contains this parameter, which is $31 or $62 for 1 MC and 2 MC respectively. To avoid manual adjusting of this parameter if you use an other speed than usual a little routine is developed which can be referred in the bootroutine, and will select the right value.

```
                ;
E010=           ACR   =$E010        ;ACIA CONTROL FLOPPY
E011=           ADR   =ACR+1
267B=           SPEPAR=$267B
                ;
     A231   SPEED    LDX #$31    ;LOAD 1 MC PARAMETER
     A000            LDY #$00
     A231   SPEED    LDX #$31    ;LOAD 1 MC PARAMETER
     A000            LDY #$00    ;INIT WAITING LOOP
     A902            LDA #$02    ;INIT REGISTER MASK
     2C10E0 SPEED1   BIT ACR
     F0FB            BEQ SPEED1 ;=>NOT EMPTY TM REG.
     8D11E0          STA ADR     ;SEND CHARACTER
     2C10E0 SPEED2   BIT ACR
     F0FB            BEQ SPEED2 ;=>NOT EMPTY TM REG.
     8D11E0          STA ADR     ;SEND CHARACTER
     C8     SPEED3   INY
     2C10E0          BIT ACR
     F0FA            BEQ SPEED3 ;=>NOT EMPTY TM REG
     C00F            CPY #$0F
     3002            BMI SPEED4 ;=>1 MC
     A262            LDX #$62    ;LOAD 2 MC PARAMETER
     8E7B26 SPEED4   STX SPEPAR ;SET PARAMETER
     60              RTS
```

```
740.loop2
750LDA Dump,Y   \Load character
760CMP #&7F     \A < &7E ?
770BCS replace  \no
780CMP #&20     \A > &1F ?
790BCS over     \yes, skip over
800.replace
810LDA #&2E     \Load code '.'
820.over
830JSR oswrch   \Print it
840INY          \Update
850DEX          \counters
860BNE loop2    \X not 0, next character
870.next
880LDA Lenght   \Load lenght of line
890CLC          \clear carry
900ADC Line     \add to Line
910STA Line     \store it
920BCC check2   \Check Line
930INC Line+1   \Update Line+1
940.check2
950LDA Line     \Check Line
960CMP End      \for
970BMI newline  \Line<End?
980.check1
990LDA Line+1
1000CMP End+1   \for
1010BMI newline \Line+1<End+1?
1020BEQ end     \Line+1=End+1?
1030BCS end     \Line+1>End+1?
1040.escape
1050JSR end     \Stop printing
1060BRK         \Generate
1070EQUB 17     \'Escape' error
1080EQUS "Escape"
1090EQUB 0
1100
1110.end
1120JSR Pnewl   \New line
1130LDA #3      \Printer
1140JSR oswrch  \off
1150RTS
1160J:NEXT
1170CLS
1180INPUT"Startaddress &"start$
1190ST=EVAL("&"+start$):!Line=ST
1200INPUT"End  address &"end$
1210EN=EVAL("&"+end$):!End=EN
1220IF EN<ST GOTO1170
1230PRINT"PRINTER on (1) or off (0) ";
1240D%=GET-48:IF D%=1 THEN ?PRNT=D%:*FX5,1
1250IF D%=1 MODE3:PRINT': ELSE MODE6
1260VDU2,1,14:PRINT"HEXDUMP &";~ST;" - &";~EN':VDU3
1270CALL init
```

Tuning procedures for Octopus interface card. Coen Boltjes. 015-136812

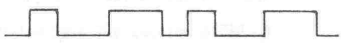
Although the tuning of the interfaces on the interface card of EC2 are not very critical, a description of the tuning procedures is described below.

The tuning of the KIM-interface:

-Record modulated logic one's and zero's on a cassette. When you use the Marc Lachaert KIMTAP utilitys load the code in your system and change the following locations.

```
D2D4  LDA #$55    A9 55
D2D9  JMP $D2D4   4C D4 D2
```

Start the program at $D28B and record the output signal for some minutes on a tape.
Note: your clockspeed must be 1 MHz.
-Replay the recorded tape and tune P1 so that on an osciloscope the signal on pen 7 of IC 14 has the folowing shape:



Your PLL is now optimal tuned.

The tuning of the BASICODE-interface:

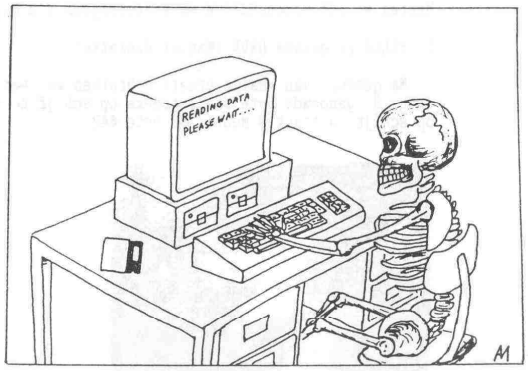
-Remove IC 10 and connect pen 6 of the foot of IC 10 to logic zero. Now record the 1200 Hz output signal for some minutes.
-Connect your osciloscope with pen 8 of IC 17 and replay the recorded tape. Tune P3 so that the pulse width is 3/4 of the pulse period.

Your MMV is now optimal tuned.

New OHIO DOS command. Coen Boltjes

This command enables the user of the OHIO DOS-extentions of Gert Klein (issue 39) to change the track range of previous defined files. The command RD must be added to the command table.

```
20 91 E4 REDEF   JSR FINDFILE  ;FIND FILE
8A               TXA
29 F8            AND #$F8      ;GET START
85 E6            STA TEMP4     ;SAVE START
4C DC E2         JMP CRA       ;ADJ. TRACKS
```



NOGMAALS "TASC" (APPLE)

Ruud Uphoff

Even een reactie op het artikel in 6502 KENNER 43 (blz. 25) over het opstarten van TASC gecompileerde programma's. Ten eerste wordt daarin opgemerkt dat het voor APPLE bezitters zeker de moeite waard is om deze compiler van Microsoft aan te schaffen. De naam Microsoft staat hier zeker garant voor kwaliteit, want ik heb al heel wat omvanrijke programma's met TASC gecompileerd zonder ooit een bug aan te treffen ! Voor gebruikers van het nieuwe APPLE DOS, "PRODOS" genaamd, is die bug er echter wel en wat voor een: Het werkt niet meer. Dat komt omdat APPLE (Het moet maar eens gezegd) er in is geslaagd om zijn PROfesional DOS te voorzien van een link naar de basicinterpreter die daar op een andere dan de tot nu toe gebruikelijke methode op CTRL-D voor DOS controleerd. Het gevolg is dat printen van een CTRL-D niet meer werkt zodat compileren van DOS-I/O onmogelijk is geworden. Dit gecombineerd met de gigantische geheugenconsumptie hebben mij PRODOS doen weggooien. Geen TASC met PRODOS dus.

Dan de eigenlijke reden van deze reactie: Het opstartprobleem bij gecompileerde software. Er zijn namelijk heel wat simpelere methoden bedacht om dit te verwezenlijken:

A. BASIC oplossing (Stel dat uw programma de naam PGM draagt)

```
100 RU$=CHR$(4)+"BLOAD RUNTIME"+CHR$(13)+CHR$(4)+"BRUN PGM"
110 PRINT RU$
```

B. Machinetaal oplossing (Uw gecompileerde programma heet weer PGM)

Op uw TASC diskette vindt U een tekstfile ADR welke we hier gebruiken. Tik in 'direct mode': BLOAD PGM en na het laden EXEC ADR waarna de APPLE zal afdrukken:

A6064,L.....

We noteren nu de afgedrukte lengte L en tellen daar 4016 bij op. Nu doen we nog: BLOAD RUNTIME en we gaan in de monitor om op adres $0800 een directe JMP te zetten: JMP $17B0 (800: 4C B0 17) en we schrijven ons programma weg met BSAVE PGM,A2048,L..... waarvoor we achter L de berekende lengte invullen. Voortaan dus gewoon in een keer BRUN PGM

C. HELLO programma BRUN (Slave diskette)

Ja hoor dat kan heel eenvoudig. We gaan in de monitor en veranderen een enkele locatie in DOS 3.3 namelijk $9E42. Daar staat $06 en daarvan maken we $34. Nu gaan we met CTRL-C terug naar BASIC en we tikken:

```
]NEW
]INIT PGM
]DELETE PGM
```

Waarna we PGM samenstellen en wegschrijven als bij methode B

D. HELLO programma BRUN (Master diskette)

Na gebruik van master create gebruiken we een disk monitor om het onder B. genoemde byte rechtstreeks op schijf te veranderen. De locatie op schijf is track 0 sector $C byte $42

How to use the Gert Klein OHIO DOS-Extentions (issue 39) on the Octopus.
Coen Boltjes. 015-136812

Originally the extentions were located from $E000, but since this is the Octopus I/O room we have to use an other location. In this example we will use the locations starting at $D000.

1)Copy system-disk 5a, and boot with this disk.

2)Assemble the source with origin $D000

3)Save the extentions on track 14 on the new disk. (SA 14,1=D000/8)

4)Load an empty directory from disk, (CA 2E79=11,3) and enter the monitor. (RE M)

5)Change the next locations:

```
2E79     4F 53 36 35 44 33 00 14
2A4B     20 12 D5
2A8B     BD 00 D0
2A95     BD 01 D0
2A9C     BD 03 D0
2AA0     BD 02 D0
```

6)Return to the DOS-Kernal, (.OS65D) and save the directory (SA 12,1=2E79/1) and save the adjusted track 1. (SA 01,1=2A00/8)

7)Enter the new bootroutine with one adjustment. The LDA #$E0 must be change in LDA #$D0. Assemble this source but locate the object code in an other location. For example $A200. When you use the OSI assembler you can use the M8000 option.

8)Load the track0 R/W option (CA 0200=06,4) and start the program (GO 0200) Load track 0 into $8200 (R8200) and leave the program. (E)

9)Enter the monitor (RE M) and move the new Boot-routine into the track0 copy (8200=A200,A300).

10)Return to DOS-Kernal, (.OS65D) and restart the track0 R/W program (GO 0200). Save the new track0 on disk (W8200=2200/8) and leave the program.

Your disk is ready now.

When you use the new disk you will notice that after a DIR or HOME command the drive will be selected. To deselect the drive we adjust the old HOME routine, and locate it at the old command table area. Boot with the new disk and adjust the next locations. Then return to the kernal and save track 0 and track 1.

```
2663     4C 30 2E
2E30     20 54 27
         20 8A 26
         20 66 26
         4C 61 27
```

Evaluatie van de resultaten van de enquete uit nummer 45.

(For a summary of the results of the enquiry and a conclusion in English, see at the end of the article).

Allereerst een dankwoord aan al diegenen die de moeite hebben genomen de enquete in te vullen en aan het bestuur te retourneren. In totaal hebben 68 leden de enquete ingevuld. Bij een totaal ledenbestand van 380 betekent dit, dat 18 procent van de leden heeft deelgenomen.
Uit het buitenland kwamen 11 reacties, 1 uit Denemarken, 1 uit Frankrijk, 1 uit Portugal en 8 uit België.
De Engelstalige enquete werd door 1 deelnemer aangevraagd en ook teruggestuurd.
Het verloop van de binnenkomst van de ingevulde formulieren (slechts vier mensen maakten een kopie en gebruikten deze) was als volgt:

Week	Aantal
1	12
2	8
3	13
4	2
5	12
6	7
7	7
8	4
9+	3

Zoals te zien is, kwamen de meeste reacties kort na het verschijnen van de enquete binnen; de vierde week, halverwege de geldige inzendtermijn droogde de stroom reacties vrijwel op. Daarna volgde een kleine opleving in week 5 gevolgd door een constante doch kleinere stroom in de laatste drie weken. Drie reacties kwamen ruim na de sluitingsdatum (1 oktober) binnen. Deze zijn wel meegeteld.

Dan nu de resultaten bij de beantwoording van de vragen. Om te beginnen heeft geen enkele deelnemer de enquete anoniem ingestuurd, ofschoon dit was toegestaan. Ofschoon de helft van de wereldbevolking duidelijk vrouwelijke kenmerken vertoont, is de club aangaande dit aspect niet representatief te noemen: alle reacties waren van mannen, ofschoon 1 deelnemer hier twijfelde en beide hokjes had ingevuld!

Aangaande de opgegeven beroepen kan worden opgemerkt dat 10 personen nog studeerden. Indien er werd opgegeven wat er gestudeerd werd, had dit in alle gevallen sterk met computers te maken. (electrotechniek, informatica). 42 deelnemers gaven een beroep op dat zeer waarschijnlijk met computers te maken heeft, of in ieder geval van technische aard was. 5 deelnemers gaven geen beroep op. De overigen (9) hadden een beroep van niet-technische aard. De club blijkt huisartsen, tandartsen, advokaten musici, ambtenaren en agrarisch georienteerde leden te hebben.

De leeftijdverdeling is als volgt:

jonger dan 15 jaar	: 0	tussen 35 en 45 jaar	: 22
tussen 15 en 25 jaar	: 14	tussen 45 en 55 jaar	: 6
tussen 25 en 35 jaar	: 22	tussen 55 en 65 jaar	: 3
		ouder dan 65 jaar	: 1

Hieruit blijkt, dat tweederde van de leden zich in de leeftijdsgroep van 25 tot 45 jaar bevindt.
Nu we toch met de factor tijd bezig zijn: vraag 2B over de tijd dat men een computer bezit, werd als volgt beantwoord:

Korter dan een jaar	3	5 á 7 jaar	29
1 á 2 jaar	5	8 á 9 jaar	3
3 á 4 jaar	24	10 jaar en meer	4

Uit de verdeling blijkt overduidelijk dat het overgrote deel der clubleden kan worden aangeduid als 'ouwe rotten in het vak'. Er zijn opvallend weinig nieuwelingen, terwijl er nog steeds mensen van het eerste uur in de gelederen te vinden zijn.

Van de 67 Nederlandstalige reflectanten bleken er 44 (66%) goed Engels te kunnen lezen/spreken, 22 (34%) vonden dat zij dit redelijk konden. Slechts 1 lid bleek geen Engels te beheersen.
De enige Engelstalige reflectant bleek absoluut geen Nederlands te spreken.

De tweede vraag ging over het computer systeem dat de leden gebruiken. De verdeling was als volgt (het aantal is hoger dan 68 daar een aantal (44) deelnemers meer dan 1 computersysteem blijken te bezitten).

6502-computers:

Junior	35
OCTOPUS/SAMSON65/EC65	23
DOS65	16
Commodore 64	7
Acorn Atom	5
Acorn Electron	4
BBC	3
KIM	3
Apple][	3
SYM	2
PET	2
CBM 30XX/40XX/8032	2
Geheel zelfgebouwd	3
Proton	1
AIM	0

In totaal: 143 Computers. 23 stuks hebben een andere CPU dan een 6502 of aanverwant. Meer dan de helft van de deelnemers heeft een Junior, al wordt deze in sommige gevallen nauwelijks meer gebruikt.

Andere Computers:

Commodore C-16		5
Commodore Plus/4		1
Atari 520ST	(68000)	1
Atari 600XL		2
OSI Superboard		1
OSI C1P		1
OSI C24P		1
MC 68000	(68000)	1
Elektuur EC68K	(68000)	1
SC/MP	(SC/MP)	1
Robby	(6809)	1
Zelfgebouwd 6802/6809		1
MC CP/M	(Z80)	1
MPF II	(Z80)	2
Philips P2000T	(Z80)	1
Philips P3100	(8086)	1
Sinclair ZX81	(Z80)	4
Casio PB-410	(???)	1
Sanyo MBC-85/2	(Z80?)	1
Sharp MZ-80K	(Z80)	1
MSX	(Z80)	1
IBM-PC Kloon	(8088)	2
Advance 86B	(8086)	1
IBM-PC/XT	(8088)	1

Eenderde van de deelnemers gebruikt OHIO OS-65D op de standaard configuratie die oorspronkelijk onder de naam Octopus werd gepubliceerd door Elektuur. De tegenhanger, DOS65, wordt door een kwart van de deelnemers gebruikt, waarvan tenminste 1 met een Junior, de rest met de CPU-kaart configuratie.

Van alle genoemde computers is 52% een zelfgebouwd exemplaar, ontworpen en gepubliceerd door Elektuur (inclusief DOS65).

Samenhangend met de computer is het operating system dat gebruikt wordt.

OHIO OS-65D	27	Proton DOS	1
DOS65	16	TOS (Atari 520ST)	1
Koen van Nieuwehove's DOS	0	Apple DOS	1
ROM-based	25	MPF-DOS	1
MS-DOS	4	CP/M	1
Zelfgemaakt OS	2		
Totaal	79		

Ook hier is de som groter dan 68 vanwege de meerdere computersystemen die de meeste deelnemers gebruiken.
Onder de noemer ROM-based vallen in principe alle computers die geen schijfgeheugens hebben, alsmede alle Commodore machines met disks; deze hebben het DOS in ROM.
Het blijkt verder, dat er tenminste 4 Junior computers moeten zijn, die onder OS-65D draaien. Dit operating systeem wordt door meer dan een derde van de deelnemers gebruikt.
DOS65 maakt 20% van het totaal uit. Het tweede en ook oudere in de club ontwikkelde DOS, dat van Koen van Nieuwehove, blijkt geen gebruikers te hebben gevonden.
Voor zover dit uit de formulieren was op te maken, is er 1 deelnemer die gaat omschakelen van OHIO OS-65D naar DOS65.

Verantwoording prijzen enquete.

10 sets van 2 5.25 floppy disks gingen naar:

K.J.A. Koning, Maasland
G.J. van Norel, Epe
B.J. van Bemmel, Nieuwkoop
R. Manders, Wageningen
A. v.d. Beukel, Delft
C. Kleipool, Cogolin, Frankrijk
dhr. van Huffel, Brussel, Belgie
G. Vandenbulcke, Brugge, België
H.A.J. Quast, Huizen (NH)
M. Leemans, Edegem, België

De volgende materiële zaken maken deel uit van de systemen van de deelnemers:

Hex display en -keyboard	34	Winchesterdisk	2
Monitor + ASCII toetsenbord	62	Cassetterecorder	47
Terminal	11	Modem	20
Teletype	5	EPROMprogrammer	44
Matrixprinter	46	BASICode interface	2
Daisywheelprinter	11	Kleurenmonitor	1
Diskdrive(s), 5.25 inch	54	RTTY morse decoder	1
Diskdrive(s), 8 inch	1	FORTH module	1
Diskdrive(s), 3/3.5 inch	4	Muis	1
		Sprekende kaart	2

Nagenoeg iedereen (91%) communiceert met zijn systeem via een monitor met een typemachine-achtig toetsenbord, de rest doet dit met een terminal, hetgeen vergelijkbaar is. Ook hebben de meesten (87%) de mogelijkheid met schijven te werken, waarbij 5.25 inch nog steeds verreweg de populairste maat is. Alle single-board computers (Junior, SYM en KIM) blijken te zijn voorzien van een cassetterecorder als opslagmedium. Slechts 2 deelnemers kunnen zich de luxe van een harde schijf permitteren. Bijna tweederde van de deelnemers kan zijn eigen EPROMs programmeren.
84% van de deelnemers heeft tenminste 1 printer, de meeste een matrixprinter.

De computers van de deelnemers worden meestal gebruikt voor het ontwikkelen van software, met tekstverwerking als goede tweede:

Administratie, bedrijf	5	Ontwikkelen van software	57
Administratie, prive	14	Ontwikkelen van hardware	25
Tekstverwerking	39	Besturingen	15
Bestandsbeheer	15	Spelen van spelletjes	2
		Imponeren van de buurman	1
Studie	2		
Niet ingevuld	1		

5/6 van de deelnemers gebruikt zijn systeem om software op te schrijven. Dit houdt in, dat de meeste systemen niet gebruikt worden: er wordt niet alleen bestaande software gebruikt maar vooral eigen nieuwe software gemaakt.
1 deelnemer imponeerde liever zijn buurvrouw (ofschoon niet onderzocht): ik kan hem geen ongelijk geven.

Over welke hulpmiddelen al die software gemaakt wordt, moge uit de derde vraag blijken. De eerste vraag diende om het niveau waarop het gemiddelde clublid zich aangaande software bevindt, te bepalen.

Alleen hardware	1	Schrijft af en toe zelf	37
Kopieert alleen	1	Schrijft het meeste zelf	24
Verbetert software	1	Schrijft alles zelf	4

Bij de beoordeling van deze vraag is er van uitgegaan dat indien een bepaald antwoord werd aangekruist, dat dan ook activiteiten op de 'lagere' niveaux ontplooid worden.
De antwoorden op deze vraag weerspiegelen inderdaad het actieve beeld van de club uit de vorige vraag: vrijwel iedereen schrijft software.

In welke taal of talen dat schrijven gebeurt blijkt zo te liggen:

Hex	6	PASCAL	0
Assembler	36	C	0
BASIC	21	dBase III	1
FORTH	1	Hangt van probleem af	2
		Niet ingevuld	1

De helft van de deelnemers blijkt voornamelijk in assembler te programmeren, overeenkomstig het karakter van de club. Ondanks het feit dat voor sommige systemen een legale BASIC niet beschikbaar is, programmeert een derde van de deelnemers voornamelijk in deze taal.

Naast de meestgebruikte taal blijken de deelnemers de volgende talen te beheersen:

BASIC	60	C	6	Simula67	1
Assembler	53	Algol60/68	6	DIBOL	1
Hex	41	dBase II(I)	3	RTL-2	1
PASCAL	22	COMAL80	2	APL	1
FORTH	17	Symphony	1	LISP	1
FORTRAN	9	PL1	1	LOGO	1
COBOL	9	Modula2	1	Niet ingevuld	2

BASIC is hier winnaar: 88% beheerst het. Op grond van eenvoud en beschikbaarheid is dit geen vreemde uitkomst. 78% van de deelnemers beheerst (een) assembler.
Een derde van de deelnemers blijkt PASCAL te beheersen ofschoon niemand het als hoofdprogrammeertaal gebruikt.

De redenen om een bepaalde taal te bevoordelen lagen als volgt:

Assembler	26	BASIC	12
Snelheid en vrijheid	20	Enige dat beschikbaar is	4
Compactheid	1	Snel resultaat	3
Samen met FORTH kan alles	1	Lekker uitgebreid	1
Computer geschikt	2	Goed te begrijpen/leze	1
Ken de rest niet goed	1	Gemakkelijk bruikbaar	1
Schrijf op OS niveau	1	Iedereen heeft/kent het	2
FORTH	6	PASCAL	5
Kompakt, snel, flexibel	1	Superieure structuren	2
Geschikt voor besturingen	2	Geen reden opgegeven	1
Gestructureerd	2	Krijgt er les in	1
		Gedwongen top-down aanpak	1
FORTRAN	2	C	2
Meest mee bekend	2	Krachtig, portable	1
		Gebruikersvriendelijk	1
dBase III	1	Geen voorkeur	3
Krachtig	1	Applicatieafhankelijk	3
Niet ingevuld	8		

Het meeste wordt in assembler geprogrammeerd: hiervoor heb je uiteraard een assembler nodig. De vraag of er voorkeur voor een bepaalde assembler was, leverde de volgende bloemlezing op:

		Enige			Enige
Niet ingevuld	16		Devpac ST (520ST)	1	0
Micro-Ade	15	9	RAE-1	1	1
AS uit DOS65	8	0	CBM-assembler	1	1
OHIO-assembler	6	3	Micro-ware	1	0
Moser (ASM/TED, MAE)	4	0	Monitor CBM-C16	1	1
Geen voorkeur	4		Acorn Electron	1	1
Geen ervaring	3		Profi-Ass	1	1
Fate	2	0	Toolkit	1	0
Merlin	2	0	ALDS	1	0
ASS114	2	0			

Opvallend is, dat in ongeveer een derde van de gevallen waar een voorkeur wordt uitgesproken, deze een assembler geldt, die men als enige kent.
Een kwart van de deelnemers heeft de vraag niet beantwoord, eveneens een kwart gebruikt Micro-Ade, waarvan de club de rechten bezit. De assembler die jarenlang als state-of-the-art te boek stond, Moser, heeft slechts een geringe aanhang.
Eenderde van de DOS65 gebruikers gebruikt de nieuwe assembler AS. Merkwaardig genoeg zijn er twee DOS65 gebruikers die Moser prefereren boven AS, ondanks een gigantisch verschil in snelheid en het bestaan van een Moser naar AS omzetter.

Verantwoording prijzen enquete.

10 EPROMs 2764 gingen naar:

B. van Tiel, Gouda
G.P.M. Stappers, Overloon
S.J. Voortman, Ouddorp
P. de Visser, Veldhoven
M. van Dorp, Leiden
L.H. van Velzen, Zeist
H. Speksnijder, Rotterdam
C. Boltjes, Delft
T. Smits, Oudenbosch
J.W. van Dongen, Rotterdam

DE 6502 KENNER

Met de kennis van andere CPU's en het vermogen om deze in machinetaal/assembler te programmeren staat het zo:

Niet ingevuld	23	8085	2
Z80	16	SC/MP	1
6800	12	IBM-370	1
Geen andere	11	PDP-11/VAX-11	1
6809	7	Ja, geen opgave	1
68000	6	Nagenoeg alles	1
8086/8088	5	Met geduld alles	1
2650	3	1802	0

Gaat men ervan uit, dat niet ingevuld 'nee' betekent, dan beheerst de helft van de leden alleen de 6502. Dit ondanks het feit dat het relatief eenvoudig is een andere '6'-CPU te leren kennen als men eenmaal de 6502 kent. De Z80 scoort bij een kwart van de leden, vermoedelijk via de werkkring.

Hoe goed men in machinetaal is vond men:

Slecht	7
Ik kan me ermee redden	19
Voldoende	20
Ik ben er goed in	20
Niet ingevuld	2

59% van de deelnemers geeft zichzelf een voldoende of hoger als het om machinetaal gaat. Slechts 10% vindt dat zij het slecht doen. Deze leden blijken meestal in BASIC te programmeren, 4 onder OS65D, de rest met een Commodore machine.

Zonder software geen hardware, en andersom. De vierde vraag moest dus wel over hardware gaan. Allereerst het niveau waarop men bezig is:

Ik koop alles	0
Bouw wel eens een printje	12
Systeem gebouwd	17
Ontwerpt hardware	34
Heeft systeem zelf ontworpen	5

De helft van de deelnemers ontwerpt zelf hardware, en kan dus als electronicus worden aangemerkt. Iedereen weet met soldeerbout en schroevedraaier om te gaan. 7.5% ontwerpt zelfs complete systemen beginnend met niets.

Ofschoon iedereen kan solderen en schroeven, wil bijna de helft assemblagewerk voor de club doen:

Nee 37 Ja 29 Niet 2

Met de printjesfabrieken is het zo gesteld:

Nee	9	Etst	5
Te veel werk	27	Plakt en etst	24
Plakt	2	CAD/CAM	1

9 deelnemers (13%) willen en kunnen zelf geen print maken. Bijna de helft kan dat wel, maar vindt het teveel werk. De rest (ook bijna de helft) is actief bezig.

De belangrijkste ontwerpgrondstof, het IC-datasheet, krijgt de volgende behandeling:

Begrijpt het niet : 1	Pin-out: 3
Begrijpt het meeste: 31	Begrijpt alles: 33

Ook hieruit blijkt opnieuw, dat we een club van electronici zijn, vrijwel iedereen weet met een data-sheet om te gaan.

Het belangrijkste gezicht naar de buitenwereld is voor de club het clubblad, De 6502 Kenner.
Ofschoon de vraag anders getiteld was, ging de vijfde vraag hierover. Allereerst de waardering voor het blad:

4	2	6	2	8	28	10	1
5	1	7	24	9	3	niet	7

Het gemiddelde is 7,4.

Over de verschillende onderdelen van het blad wordt als volgt geoordeeld:

	Goed	Gem.	Slecht	Niet
Redactionele praatje	42	14	0	12
Uitnodiging	38	15	1	14
Algemene artikelen	26	31	1	10
Assembler listings	43	12	3	10
BASIC listings	18	31	8	11
FORTH listings	17	19	6	26
Hardware ontwerpen	25	29	4	11
Reclame clubartikelen	16	31	8	13
Opmaak	30	25	6	7

Overeenkomstig het gemiddelde cijfer wordt hier weinig 'slecht' toegekend. Duidelijk positief zijn de onderdelen Redactioneel, Uitnodiging, assembler listings en opmaak. Uitgesproken gemiddeld zijn de BASIC-listings.
Met de FORTH-listings weet niet iedereen raad; ruim een derde heeft geen mening, bijna een derde vindt het maar gemiddeld.

Met de schrijfactiviteit voor het blad is het dunnetjes gesteld, bijna de helft schrijft nooit iets, ongeveer een derde met enige regelmaat:

Nooit	28	Regelmatig	4
Weinig	11	Nog niet	4
Af en toe	19	Niet ingevuld	2

De taal waarin het clubblad gesteld is al menigmaal aan de orde geweest. De leden denken er zo over:

Nederlands/Engels	20
Engels	16
Van alles door elkaar	14
Nederlands en een beetje Engels	9
Alleen Nederlands	5
1 taal (Nederlands OF Engels)	2
Niet ingevuld	1
Geen mening/voorkeur	1
Nederlands en een beetje Duits	0
Nederlands/Duits	0
Duits	0

Op Duits zit niemand expliciet te wachten, er is echter een sterke voorkeur voor Engels. Globaal gesproken mag volgens deze verdeling de helft van het blad in het Engels gesteld zijn. Een aantal leden vinden wel dat indien er van Engels gebruik gemaakt wordt, dit goed Engels moet zijn. Zij hebben liever Nederlands dan steenkolen-Engels.

Volgens de deelnemers moet het volgende wat minder verschijnen:

Niet ingevuld	38
BASIC programma's	9
Lange listings	7
Spelletjes	5
Lange listings met kleine toepasbaarheid	3
Schaakprogramma's	3
Junior promoting artikelen	2
Onduidelijke listings	1
Artikelen in het Engels	1
Algemene verhalen	1
De leden maken het blad	1
Artikelen die verouderd zijn	1
Disassemblers	1
Listings met 'grote gaten' (no.43 31-43)	1
'Kale' listings	1
Zeer grote modificaties van programma's	1
Meer over gekocht minder over zelfbouw	1

Voor de BASIC programma's kwam iemand met een goed argument: er valt meestal niets van te leren en dat is tegen de doelstelling van de club.

Verantwoording prijzen enquete.

5 SRAMs, 8kx8 gingen naar:

H. van Engelen, Utrecht
J.W. Oostman, Haarlem
P. Tratsaert, Gent, België
E.R. Elderenbosch, Amsterdam
H.J. Gits, Eindhoven

Het volgende moet meer in het blad verschijnen:

Niet ingevuld	16
Software	10
Hardware	12
Algemene artikelen in informele stijl	6
DOS65 artikelen	3
Kleine programma's met grote gevolgen	2
Tips	2
Interfacing (modems, AD/DA enz)	2
Assembler listings	2
BASIC listings	2
Universele software	2
Cybernetica/robotics	2
Programma's in nette talen (PASCAL)	1
De leden maken het blad	1
Artikelen v d beginner/kleine systemen	1
Octopus/Samson artikelen algemeen	1
Bespreking mogelijkheden programma's	1
Verbetering/-atches	1
Niet helemaal uitgewerkte ideeen	1
Artikelen over programmeermethoden	1
Marktvergelijkingen randapparatuur	1
Subroutine bibliotheek	1
Combinaties van hard- en software	1
Print layout's	1
Uitleg hardware	1
Artikelen van mijzelf (van Engelen)	1
Junior software	1
Uitleg verschil systemen (Junior/EC65)	1
Info over ?'s	1
Boekbesprekingen	1
Marktontwikkelingen	1
Beetje PASCAL	1
Besturingen met microprocessors	1
68000	1
Cursus machinetaal	1
16-bit club systeem	1
Listings van utilities	1
Hoger werk (UNIX, CP/M)	1
Apple [illegible]	1

Verdere opmerkingen over het clubblad waren:

Opmaak kan veel beter 6x
Drukkwaliteit moet beter 5x
Houden zo!!! 4x
Standaard kop voor alle artikelen 4x
Meer commentaar in listings 3x
Minder Engelse verhalen, anders zeg ik op 2x
Nieuwe rubriek: problemen met de Octopus
Adressenlijst van Belgische leden
Artikelen niet door elkaar afdrukken
We ondersteunen teveel systemen
Engelse inleiding/samenvatting bij artikelen
Auteur met adres bij ALLE artikelen vermelden
Maak eerst een goedkope barkode-lezer,
Dan: alle listings in barkode
Meer DOS65 artikelen
Blad moet dikker
Blad moet in 1 taal, maakt niet uit welke
Selfmodifying code: streng verboden!
Enthousiasme is prima
Aandacht voor gezamenlijke inkoop hardware
Voor publicatie eerst correctheid verifieren
DOS65 is een reden om te blijven
Paperware veel te duur, bij AH goedkoper
Hardwaretekeningen onvolledig/slecht
Veelal geen inleiding bij artikelen
Listings eerst verkleinen, dan in blad
Afdrukken in 2 of 3 kolommen
Forecastpagina: wie werkt waaraan?
Tijd tussen schrijven en publicatie verkorten
Meer uitwisseling Octopus software
Andere kleur kaft

De zesde vraag bevatte de 'hamvragen' van het bestuur, waarbij de naamgeving de belangrijkste is.
De eerste vraag diende om de 'activiteitsbereidheid' te peilen. Dat zit zo:

Lid voor het blad	3	In het bestuur zitten	3
Te weinig tijd	35	Spreekbeurten houden	3
Artikelen schrijven	9	Vertaalwerk doen	8
Hardware ontwikkelen	15	Niet ingevuld	3
Software ontwikkelen	18	Hardware bouwen	1
Bijeenkomst organiseren	3		

Twee beantwoorders die in het bestuur zouden willen zitten, waren reeds bestuurslid. De derde zal uiteraard benaderd worden.
De helft van de inzenders blijkt geen tijd te hebben voor clubgerichte activiteiten. Ongeveer een kwart wil wel ontwikkelingswerk in soft- en/of hardware doen.

De hamvraag, of de club van naam moet veranderen werd als volgt beantwoord:

	Totaal	(leeftijd)					
		15	25	35	45	55	65
Geen mening	26	0	5	10	8	3	0	1
Nee	11	0	3	5	2	0	1	0
Ja	29	0	6	7	12	3	2	0

	Totaal	(tijd dat men een computer heeft)					
		-1	1-2	3-4	5-7	8-9	10+
Geen mening	26	3	5	8	9	0	2
Nee	11	0	0	4	7	0	0
Ja	29	0	0	12	13	3	2

De Nee-zeggers zijn duidelijk in de minderheid. De leeftijdsgroep 25-35 is duidelijk de 'geen mening'-groep, de leeftijdsgroep 35-45 zijn de duidelijke ja-zeggers.
Indien men 3 tot 7 jaar een computer heeft, reageert men ongeveer als het gemiddelde. Korter dan 3 jaar heeft men geen mening, langer dan 7 jaar zijn meestal ja-zeggers.
De redenen die zoal werden opgegeven waren:

Vlag dekt lading niet meer	20
Buitenstaanders begrijpen naam niet	1
We doen meer dan alleen KIM	4
Nadruk op praktijk	1
Aansluiten bij HCC	1
Verandering van spijs doet eten	1
Naam in buitenland onbruikbaar	1
Misleidend, onduidelijk	1

De hoofdreden is duidelijk: De KIM is er niet meer en we doen veel meer dan dat, dus klopt de naam niet meer.

De volgende namen werden voorgesteld:

Niet ingevuld	28
De 6502 Kenners	10
FDC(C)	9
Club65+	8
ENCC	2
De 65XX(X) Kenners	2
The 6502 users	1
De 65+ Kenners	1
Computerclub 65+	1
Dutch micro60 club	1
Micro60 hobby club	1
Club 65X	1
Het zesde alternatief	1
First Dutch processor club	1
Hobby computer knowledge club	1
Bit Burger	1
Chip Chat	1
De 6502 hobby club	1
United processor club	1
RMP (Rockwell/Motorola Processors)	1
CPM (Central Processing Magazine)	1
Computer user club Holland	1
CCE (Computer Club Europe)	1
Multilayer	1

Op de vraag of er nog andere processors dan de 6502 in de club moeten worden toegelaten werd door 51 deelnemers (75%) positief geantwoord. Een zesde deel (11 dus) vond van niet. 6 deelnemers hadden hierover geen mening.

Of de club de gebondenheid met een bepaalde processor of processorfamilie moet loslaten of niet was ook duidelijk: 46 (68%) deelnemers vinden van niet. Ook hier hebben 6 deelnemers geen mening.

De volgende redenen werden opgegeven:

Door de ja-zeggers bij 6D/E	
Blijf in de 6XXXX serie	21
We moeten met de tijd meegaan	5
Beperking is het einde van de club	4
6502 te smalle basis	3
Andere CPU bevordert kennisuitw.	3
8-bitters maken plaats voor 16	2
KIM had 6502, waarom geen andere?	1
Meer over 68000	1
Ook interesse in Z80 en 8088	1
Door de nee-zeggers bij 6D/E	
Meer CPU's: moeilijker kennisuitw.	4
Meer CPU's: die clubs zijn er al	3
Algemeen: blad wordt zootje	3
Geen reden meer om lid te blijven	1
Er zal kliekvorming optreden	1
Doorgaan met 6502: steeds mooier	2
Het is nu al een rommeltje	1
Nog te vroeg voor opvolgers	1

Over opvolgers voor de 6502 gesproken: dit zijn ze volgens de deelnemers:

65C02	12	6809	2
65802/65816	34	Geen	5
6800	0	8088	1
68000	22	Niet ingevuld	9

De helft ziet de 65802/65816 als belangrijkste opvolger, een derde de 68000. De meerderheid ziet 16-bit dus als volgende stap.

Met het bijeenkomstenbezoek is het als volgt gesteld:

Nooit: 28	Alleen in de buurt: 10	Af en toe: 13	
Altijd: 14	Niet ingevuld: 3		

De ene helft komt nooit (de buitenlanders vanwege de afstand), de andere helft gemiddeld genomen af en toe.

Met de kwaliteit van de bijeenkomsten en de lezingen is het volgens de leden zo:

Bijeenkomst		Lezing	
Goed	33	Goed	28
Matig	0	Matig	4
Slecht	1	Slecht	0
Niet ingevuld	34	Niet ingevuld	36

Diegenen die naar de bijeenkomsten komen, blijken ze goed te vinden. Ook de lezingen vallen in de smaak, zij het wat minder. De nooit-komers vulden deze vragen uiteraard niet in.
Tenslotte werden de volgende suggesties ter verbetering van de bijeenkomsten gedaan:

Waarom altijd zo ver weg? (3x)
Lezingen op te hoog niveau (2x)
Jaarlijks 1x in Belgie of Duitsland (2x)
Verzamelen/verspreiden public domain software
Niet op zaterdag
Ontplooi regionale activiteiten
Meer tam-tam
Gratis toegang en contributieverhoging
Beter geluid
Vriendelijker maken voor buitenstaanders
Beter gestructureerd forum

Conclusies.

1. Een deelname van 10% van de leden geldt als goed; derhalve heeft een hoog percentage van de leden deelgenomen. Een vervolgconclusie is dan, dat de enqueteresultaten de mening van alle leden redelijk nauwkeurig weerspiegelt.
2. Wij zijn een club technisch georienteerde mensen, die voorkeur hun computersysteem zelf bouwen, en op dit systeem in machinecode kunnen programmeren. Vrijwel iedereen schrijft eigen nieuwe software op zijn systeem.
3. De talen assembler en BASIC dienen de meeste aandacht in het clubblad te krijgen op grond van hun gebruiksgraad.
4. Over de taal FORTH zijn de meningen verdeeld: er is een kleine kern van fanatieke gebruikers/aanhangers, maar een veel groter deel van de leden vindt deze taal oninteressant of weet er geen raad mee. Minder aandacht in het blad is dan ook op zijn plaats.
5. De kennis van andere processors dan de 6502 valt tegen; slechts de helft van de leden beheerst nog andere processors.
6. Eenderde van de leden gebruikt het OHIO OS65D disk operating systeem, een kwart gebruikt het in de club ontwikkelde DOS65. Meer dan helft van de computers is derhalve opgebouwd rond de Elektuur CPU-kaart.
7. Sterke kanten van het clubblad zijn het redactionele praatje, de uitnodiging voor de bijeenkomst en assembler listings, die echter niet te lang mogen zijn.
8. Zwakke punten van het blad zijn BASIC listings en listings van spelletjes (in welke taal dan ook). Ook de opmaak werd hier en daar bekritiseerd.
9. De kennis van de Engels onder de leden is over algemeen goed. De leden hebben er gemiddeld genomen weinig bezwaar tegen als een gedeelte van het clubblad in het Engels gesteld is, mits dit Engels van redelijke kwaliteit is.
10. De leden vinden dat de club haar naam moet wijzigen, omdat de huidige naam de lading duidelijk niet meer dekt. Indien de enquete doorslaggevend voor de nieuwe naam moet zijn, dan luidt deze: 'De 6502 Kenners'. Dit is thans de naam van het blad. Als alternatief kan gelden: First Dutch Computer-club, afgekort tot FDC.
11. Er is over het algemeen weinig bezwaar tegen dat de club zich in de toekomst met andere processors gaat bezighouden. Men ziet de 65816 als opvolger, met als goede tweede de 68000-familie.
12. De bijeenkomsten worden redelijk bezocht, en vallen over het algemeen goed in de smaak, zij het dat de plaats nog wel eens wat kritiek ondervindt. Een centraal punt geniet een lichte voorkeur. De lezingen zouden wat eenvoudiger kunnen.

For the English-speaking readers.

First of all, the executive committee would like to thank those of you that participated in the enquiry. To save space, and because the majority of our members can read Dutch, only the conclusions drawn from the results of the enquiry are translated below. If you have any questions on the Dutch part of the article, do not hesitate to write to the organizer of the enquiry.

The conclusions drawn were:

1. A partipation level of 10% is usually good. We did better, so the results reflect the opinion of the average member pretty good.
2. The average member of the club is a technician, who prefers to build and program his system by himself. Programming is usually done in machine language.
3. Most members use BASIC and machine language in their programming; the magazine should reflect this.
4. The language FORTH is used by a small group of usually enthousiastic people, but the majority has no interest or kwowledge in it. The magazine should therefore contain less FORTH articles.
5. About half of the members can program another CPU beside the 6502 in machine language. This is less than usually assumed by the committee.
6. OHIO OS65D DOS is the most commonly used DOS: about one third of the members uses it. DOS65, developed in the club by a few members, is used by a quarter of the members. This means that the majority of the systems is based on Elektor's CPU-card.
7. The best parts of the magazine are: the editorial, the invitation for the meetings and the lisstings in assembler. The latter should not be too long.
8. Weak points of the magazine are BASIC listings and listing of games (in whatever laguage). Sometimes the lay-out was critisized.
9. Most members can read and speak English, have little objections against a part of the magazine being written in English. The language used should be of sound quality.
10. The members feel, that the name of the club should change, because it is no longer a valid name. According to the results of the enquiry, the new (Dutch) name should be 'De 6502 Kenners'. An alternative can be: First Dutch Computer-club, abbreviated to FDC.
11. In future, the club should also deal with the 65816 CPU, as most members think this will be the successor of the 6502. The 68000-family can form a good alternative.
12. The meetings are visited regularly by the average member, who has a good feeling about them. Some feel that the meetings are in places too far away and should be held in a more central spot. The lectures are a bit too difficult.

TEXT ON SCREEN IN GRAPHICS 8

FOR ATARI 600 XL

AUTHOR: Henk Speksnijder, The Netherlands

VARIABLES:
H = HORIZONTAL POSITION OF THE STRING
V = VERTICAL POSITION OF THE STRING
T$ = STRING TO BE PRINTED
B = ADDRESS OF THE VIDEO RAM
M = MODE M = 0 NORMAL
M = 1 VERTICAL TEXT

```
 10 P=0
 12 FOR I=0 TO 23                        MACHINECODE IN
 14 READC:P=P+C:POKE 1536+I,C            WORKAREA
 16 NEXT I
 18 IF P<>2851 THEN STOP                 TEST CHECKSUM
 20 DATA 104,104,133,213,104
 22 DATA 133,212,160,7,177
 24 DATA 212,162,7,10,126
 26 DATA 24,6,202,16,249
 28 DATA 136,16,242,96

100 GRAPHICS 8                           INIT GRAPHICS MODE
110 C=PEEK(560)+256*PEEK(561)            AND CALCULATE FROM
120 B=PEEK(C+4)+256*PEEK(C+5)            BEGIN OF VIDEO RAM
                                         IN B
200 DIM T$(30)
210 T$="HORIZONTAL"                      T$ IS THE STRING
                                         NEEDED BY THE SUBR
220 H=40:V=20:M=0
230 GOSUB 1000                           H = HOR.POS. FIRST
                                         CHAR.
240 T$="VERTICAL"                        V = VERT.POS.FIRST
250 H=100:V=120:M=1                      CHAR.
260 GOSUB 1000
300 END

1000 REM SUBROUTINE TEXT IN GRAPHICS 8
1010 FOR J=1 TO LEN(T$)                  REPEAT FOR ALL
                                         CHARS OF THE
                                         STRING
1020 IF H<0 OR H>319 THEN STOP           TEST ON H AND V
                                         NOT EXCEED THE
                                         LIMITS
1030 IF V<0 OR V>159 THEN STOP
1040 P=B+INT(H/8)+40*V                   CALCULATE THE DEST
                                         ADDRESS OF THE
                                         CHAR. IN VIDEO RAM
1050 C=ASC(T$(J,J))
1060 IF C>96 THEN C=C+32                 CONVERT ASCII TO
1070 IF C<32 THEN C=C+96                 ADDR WHERE CHAR IS
1080 C=8*C+57088                         IN ROM
1090 IF M THEN C=USR(1536,C):C=1560      IF VERT. THEN CALL
1100 FOR I=0 TO 7                        MACHINE CODE
1110 POKE (P+40*I),PEEK(C+I)             WRITE CHAR TO
1120 NEXT I                              SCREEN
1130 IF M=0 THEN H=H+8                   IF HOR. THEN
                                         INCREMENT H
1140 IF M THEN V=V-8                     IF VERT. THEN
                                         INCREMENT V
1150 NEXT J
1160 RETURN
```

Lines 10 - 28 gather. Lines 100 - 120 gather. Watch the B that must hold the calculated value the whole running. Lines 200 - 300 give an example; one can change without any risk. In the string everything is allowed: ALPHABET, alphabet, 0 until 9, graphical characters, etc. Lines 1000 - 1160 is the subroutine. Lines 1020 and 1030 prevent serious errors in the computer; in this subroutine 8 * a poke will be executed. If no need of vertical text, lines 1090 en 1140 can be cancelled and line 1130 will be changed to 1130 H=H+8. The lines 10 - 28 also to be cancelled.

ASCII-DUMP WITH OS-65D V3.3 EXTENDED MONITOR

It is sometimes very useful to have an ASCII-dump together with a HEX-dump.
This extension of the OS-65D Extended Monitor produces, when the D-command is entered, an ASCII-dump on the right of the HEX-dump on the screen.
It can be used by JUNIOR-owners with an 80 columns screen. The extension is placed in memory from $1F3E which was not used originally.
The D-command must be entered in the same way as before.

Gerrit van Woerkom
Appelaar 55
4907 KD Oosterhout
The Netherlands

```
10                      ;Dump gives HEX- and ASC-dump
20              ;
30 00CC=                STADR =$CC
40 00CE=                ENDADR=$CE
50              ;
60 2325=                KPDO  =$2325
70 1A6B=                GETPAR=$1A6B
80 1A56=                CRLF  =$1A56
90 1BAF=                PRSP  =$1BAF
100 17A0=               PRASC =$17A0
110 19F2=               PRNIBL=$19F2
120 19E9=               PRBYTE=$19E9
130 1A85=               NXTADR=$1A85
140 1A53=               RETEM =$1A53
150 19C8=               PRSPS =$19C8
160             ;
170 1C3D                *=$1C3D
180             ;
190 1C3D 206B1A         JSR GETPAR      ;get start- and
200 1C40 A900           LDA #0          ;and endaddr
210 1C42 8D2523         STA KPDO
220 1C45 20561A         JSR CRLF        ;print CR,LF
230 1C48 206F1C         JSR SPACES      ;print 5 spaces
240 1C4B A6CC           LDX STADR
250 1C4D A010           LDY #$10
260 1C4F 20AF1B   NEXT1 JSR PRSP        ;print 2 spaces
270 1C52 20AF1B         JSR PRSP
280 1C55 8A             TXA
290 1C56 20F219         JSR PRNIBL      ;print low bytes
300 1C59 E8             INX             ;of addresses on
310 1C5A 88             DEY             ;top of columns
320 1C5B D0F2           BNE NEXT1
330 1C5D 206F1C         JSR SPACES      ;print 5 spaces
340 1C60 A6CC           LDX STADR
350 1C62 A010           LDY #$10        ;16 columns for
360 1C64 8A       NEXT2 TXA             ;ASC-dump
370 1C65 20F219         JSR PRNIBL
380 1C68 E8             INX
390 1C69 88             DEY
400 1C6A D0F8           BNE NEXT2
410 1C6C 4C411F         JMP DUMP        ;make HEX- and
420                     ;ASC-dump
```

```
430 1D6F A205       SPACES LDX #5
440 1D71 4CC819            JMP PRSPS
450                ;
460 1C74 3004       OUTPUT BMI NOCHAR   ;is byte ) $7F?
470 1C76 C920              CMP #' '
480 1C78 B002              BCS CHAR     ;is byte ( $20?
490 1C7A A92E       NOCHAR LDA #'.'     ;no char. so
495                                     ;print '.'
500 1C7C 4CA017     CHAR   JMP PRASC    ;print the char.
510                ;
520 1F3E                   *=$1F3E
530                ;
540 1F3E 4C531A     BREAK  JMP RETEM    ;back to EM
550                ;
560 1F41 AD2523     DUMP   LDA KPDO
570 1F44 C903              CMP #$03     ;BREAK-key hit?
580 1F46 F0F6              BEQ BREAK
590 1F48 20561A            JSR CRLF     ;print CR,LF
600 1F4B A5CD              LDA STADR+1  ;get startaddr-H
610 1F4D 48                PHA          ;save it
620 1F4E 20E919            JSR PRBYTE   ;and print it
630 1F51 A5CC              LDA STADR    ;get startaddr-L
640 1F53 48                PHA          ;save it
650 1F54 20E919            JSR PRBYTE   ;and print it
660 1F57 A210              LDX #$10     ;columncntr=16
670 1F59 20AF1B            JSR PRSP
680 1F5C 20AF1B     NEXTBT JSR PRSP
690 1F5F B1CC              LDA (STADR),Y get byte to
695                                     ;print
700 1F61 20E919            JSR PRBYTE   ;and print it
710 1F64 E6CC              INC STADR    ;incr. startaddr
720 1F66 D002              BNE NONUL
730 1F68 E6CD              INC STADR+1
740 1F6A 38         NONUL  SEC
750 1F6B A5CC              LDA STADR
760 1F6D E5CE              SBC ENDADR
770 1F6F A5CD              LDA STADR+1
780 1F71 E5CF              SBC ENDADR+1
790 1F73 CA                DEX
800 1F74 B002              BCS HEXRDY   ;HEX-dump ready?
810 1F76 D0E4              BNE NEXTBT   ;16 col. ready?
820 1F78 F00A       HEXRDY BEQ ASCDMP   ;16 columns in
830 1F7A 8A                TXA          ;HEX-dump?
840 1F7B A8                TAY
850 1F7C A203       FILL   LDX #3       ;if not then
860 1F7E 20C819            JSR PRSPS    ;fill room with
870 1F81 88                DEY          ;spaces
880 1F82 D0 F8             BNE FILL
890 1F84 206F1C     ASCDMP JSR SPACES
900 1F87 68                PLA
910 1F88 85CC              STA STADR    ;restore
920 1F8A 68                PLA          ;startaddr-L
930 1F8B 85CD              STA STADR+1  ;- startaddr-H
940 1F8D A210              LDX #$10     ;columncntr=16
950 1F8F B1CC       NEXTCH LDA (STADR),Y get character
960 1F91 20741C            JSR OUTPUT   ;and print it
970 1F94 20851A            JSR NXTADR   ;incr startaddr
980                                     ;and go to EM
985                                     ;if ready
990 1F97 CA                DEX
1000 1F98 D0F5             BNE NEXTCH
1010 1F9A F0A5             BEQ DUMP
```

L_E_T_T_E_R_ T_O_ T_H_E_ E_D_I_T_O_R_

R. Baarslag
Beatrixstraat 3
5161 HR Sprang-Capelle
The Netherlands.

I have found a bug in Elektor Computing No. 1, on page 66. The BIOS-subroutine ACIOUT + $F77E.
The character to be transmitted must be in AHOLD + $2363 instead of the accu before jumping to the subroutine ACIOUT.

I discovered another bug when I tried to use the instruction SED in a self written program. When I use this instruction in a program, and I run it, the monitor gives an anusual picture. I looked if this instruction has an effect on the CRTC. I must give a hardware reset to re-initialise the CRTC. Is there anyone who has the same problems or can anyone give me a solution?

Fig-FORTH DISK 10.
When I loaded disk 10 with Fig-fORTH and I tried VLIST, an unusual picture came on the screen of the monitor. From each word the last letter is changed in a graphic-sign. And the string editing commands are not working. Has anyone a solution for these problems?

==========

T_E_L_E_A_C_ C_U_R_S_U_S_S_E_N_ 1_9_8_7_

Vanaf donderdag 12 maart 1987 via radio 5 van 22.00 - 22.30 uur geeft TELEAC een veertiental lessen Structuur in Basic voor hen die 'netjes' willen leren programmeren en voor beginners. Er wordt gebruik gemaakt van GW Basic, een 'dialect', dat echter goed bruikbaar is voor mensen die een computer hebben met een ander dialect dan GW Basic.

Vanaf dinsdag 10 maart 1987 via radio 5 van 20.30 - 21.00 uur geeft TELEAC een twaalftal lessen Personal Computer 2 voor hen die al enige tijd met een computer werken maar in hun werkomgeving effektiever gebruik van willen maken van de computer. De kennis uit de cursus Personal Computer 1 wordt verondersteld bekend te zijn.
De cursus Personal Computer 2 wordt eveneens gestart op de televisie via Nederland 2 vanaf maandag 9 maart 1987 van 23.05 - 23.35 uur, eveneens twaalf lessen.

Inlichtingen:
Stichting TELEAC, Buro Voorlichting,
Postbus 2414, 3500 GK Utrecht.
Telefoon: 030 - 95 69 11

==========

B_O_E_K_I_N_F_O_R_M_A_T_I_E_

Praktijkboek Videotex
Dirk de Grooff (red), Marc Anne', Patrick Engelen, Frits van Dissel.
Uitg.: Maarten Kluwer. Antwerpen - Apeldoorn.
ISBN 90 6215 102 7, 1984, 147 p., met adressenlijst. 39,50
Met deze 'cursus' leert de lezer in een kort tijdsbestek zelf videotextpagina's aan te maken en op te slaan in de computer.

WORKING WITH MICRO-ADE and BANK SWITCHING on the JUNIOR COMPUTER

* *HARDWARE CONSIDERATIONS*

When I began testing with the project issued in DE 6502 KENNER, last April (^{10}), all was well until I started using floppy disks to read and write the memory banks in a fast manner: by that time, the system went down often, with, I discovered later, timing problems.

I had two sets of 4164 chips, one of 250ns access time in one board, the other with 200ns. So, I bought a new set of 8x 4164 - 150ns, and tested again: it was not enough.

In my system, the Φ_1 clock has a rising time (^{8}) too great (fig.1), and so the timing chain in the refreshing section of the dyn. RAM cards began too late, and the time left to access the chips was too short; and this was, of course, aggravated by the timing overhead of the cascade of gates N27,28 N29, 'LS138, N103,107... ((^{10}) fig IA and IB), necessary to decode the $\overline{SEL}$ lines to the banks. To cure this, I've used Φ_2, instead of $\overline{\Phi_1}$, entering N2. IC10, in the Junior's main board, to which belongs N6 (see fig.1) is overloaded: first, it is R5; second, this line drives directly all the dynamic RAM chips in every card; the 'LS version has a MAX low-level output current of only 8 mA, against 20 of either the 'S or the 'H versions: preferably, the 74S01 should be used. Also, I've got a more simple way to generate the signals $\overline{0 \rightarrow D}$ (original board with bank.0) and $\overline{4 \rightarrow B}$ (new board(s)) in the place of IC11 and associated gates N27, 28 and 29. More, I was offered with a better, neater and faster solution, reducing the amount of TTL IC's needed: a PROM generates the required $\overline{SEL}$ lines to the various boards.

The PROM, fig.2, drives directly the $\overline{SEL}$ line of the refreshing section of each dyn. RAM card: its Q outputs will only be *low* in the address range of $4000 through BFFF in the selected bank, except for the original board (system RAM + bank.0), to which the $\overline{SEL}$ line must be active always in the ranges $0000 → 3FFF and $C000 → DFFF (system RAM) whatever bank be selected. Then N31, (an or-gate) in the dyn. RAM boards, adds the $\overline{SEL}$ line with the Φ_1 clock, so the RAM chips are enabled only if the two lines are logical low.

The programming of the PROM (fig.3) is straigthforward:

address (MSNibble)
bank

bank	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00 =	FE	FE	FE	FE	FE	FE	FE	FE	FE	FE	FE	FE	FE	FE	FF	FF
10 =	FE	FE	FE	FE	FD	FD	FD	FD	FD	FD	FD	FD	FE	FE	FF	FF
20 =	FE	FE	FE	FE	FD	FD	FD	FD	FD	FD	FD	FD	FE	FE	FF	FF
30 =	FE	FE	FE	FE	FB	(FB)	FB	FB	FB	FB	FB	FB	FE	FE	FF	FF
40 =	FE	FE	FE	FE	FB	FB	FB	FB	FB	FB	FB	FB	FE	FE	FF	FF
50 =	FE	FE	FE	FE	F7	F7	F7	F7	F7	F7	F7	F7	FE	FE	FF	FF
60 =	FE	FE	FE	FE	F7	F7	F7	F7	F7	F7	F7	F7	FE	FE	FF	FF
70 =	FE	FE	FE	FE	EF	EF	EF	EF	EF	EF	EF	EF	FE	FE	FF	FF
80 =	FE	FE	FE	FE	EF	EF	EF	EF	EF	EF	EF	EF	FE	FE	FF	FF
90 =	FE	FE	FE	FE	DF	DF	DF	DF	DF	DF	DF	DF	FE	FE	FF	FF
A0 =	FE	FE	FE	FE	DF	DF	DF	DF	DF	DF	DF	DF	FE	FE	FF	FF
B0 =	FE	FE	FE	FE	BF	BF	BF	BF	BF	BF	BF	BF	FE	FE	FF	FF
C0 =	FE	FE	FE	FE	BF	BF	BF	BF	BF	BF	BF	BF	FE	FE	FF	FF
D0 =	FE	FE	FE	FE	7F	7F	7F	7F	7F	7F	7F	7F	FE	FE	FF	FF
E0 =	FE	FE	FE	FE	7F	7F	7F	7F	7F	7F	7F	7F	FE	FE	FF	FF
F0 =	FE	FE	FE	FE	FF	FF	FF	FF	FF	FF	FF	FF	FE	FE	FF	FF

data-out lines:

bit7 banks D/E
bit6 banks B/C
bit5 banks 9/A
bit4 banks 7/8
bit3 banks 5/6
bit2 banks 3/4
bit1 banks 1/2
bit0 bank.0 and system RAM

Memory adresses:	0,1,2,3XXX always accessed on the original board	4,5,6,7,8,9,A,BXXX addressed on the various banks	C,DXXX always accessed original board	E,FXXX never accessed (not RAM)

fig.3 - programming table of the TBP 28L22 (256x8 bits PROM), as the ruler of 16 banks.

The first 4 bytes in every row are $FE (1111.1110): the system RAM is always selected at addresses 0,1,2 and 3XXX; the following 8 bytes vary according to the bank selected; as an example, the selection of bank.3 at address 5XXX, reads location $35 of the PROM whose data is $FB or 1111.1011, where bit 2 selects, in fact, bank.3. Columns 13 and 14 correspond to addresses C and DXXX, system RAM, active in the original board (whatever selected bank). At last, columns 15 and 16 in the table, are programmed as $FF (1111.1111), so none bank is selected at addresses E and FXXX, because this range is not dyn. RAM.

I've been working with bank.0 (the original 64K card where is also the system RAM) and bank.1 & 2 (another 64K board). They permitted the edition of an 88K source file.

Two PROMs were tested. The TBP 24S10 (74S287) with 4 bits, addresses 7 banks (0 → 6). While the TBP 28L22 with 8 bits has provision for the use of 15 banks (0 through E, not counting with the system RAM) or even 16 (if the SEL line for bank.F is decoded from the bank-register's 1,2,3,4-Q lines - when these are all high), I found that it was not much feasible the use of only 64K dynamic RAM cards: with a 7-connector bus (9) I have space for banks 0, 1-2, 3-4 (3 RAM cards), bank.5 (an 8K RAM + 16 EPROM card) fig.4, the FDC card, the VDU card, and a R.T.Clock card. Bank.5 is a good goal, because an old card is reutilized with small effort, fig.5, and an extensive (E)PROM space is put at hand for frequently used programs and utilities.

16 banks (32K RAM each) amount to 512K bytes. It shall come in hand the RAM disk board promised by Erwin Visschedyk (11).

* *SOFTWARE*

I have now MICRO ADE extensively re-edited and assembled and the main routines are already tryed-out with banks. Following are the listings of some of them: INCWSP, DECWSP, SETBK0, SWAPDN, SAVBNK, RSTBNK and SETBNK.

Passing by, I have fixed some slow routines (FIX, DELETE, INSERT...), while implementing new features of this editor/assembler.

The memory map is now a litle more organized. Fig.6 is the basic version without banks; the workspace is extended to a maximum, leaving enough space for the tables. Fig.7, the bank-switching-mode map, has enlarged symbol and cross-reference tables, to be able to assemble large programs, as MICRO ADE itself (8k object code).

All the tables (symbol, X-reference, object code) are in bank.0. The source extends from bank.0 up to the last bank. These tables are needed *only* to *ASSEMBLE* the source, not to the editing process. So, the lines in the source are accessed one at a time, and stored on the MA buffer at $0100; after this reading of the source, the number of the present bank, the workspace pointer and the bank delimiters (different on bank.0, intermediate bank(s) and on last bank), are saved onto the stack, and bank.0 is activated (called), to let the tables be accessed and updated by the assembly process, i.e., the line in the buffer is read, interpreted and assembled.

Then, the previous bank is restored, and also its respective pointers and delimiters, and the process is repeated until the EOF character is reached, either in PASS.1 or in PASS.2 .

The source's reading is effected via Y-indirect indexed addressing (LDAIY...) and is pointed to by the WorkSpace PoiNTer in the current bank. The routines INCrement and DECrement the WSPNTer, check for the crossing of the banks' boundaries. At the end of every bank (except in the last one) is kept a 1-page buffer to keep pace with the use of the indexed instructions, which could attempt to read or write beyond each bank's limits. When the pointer is incremented and the new address falls in this buffer ($BF00), swapping is made to other zone (temporary), the bank is switched (incremented), and the first page (at the base) of the (next) bank is loaded with the 256 bytes from the swap zone.

The other way around, when the pointer is decremented, reaching the address before the start of the source-file (beyond the base of the current bank), the first page of the source in this bank is swapped (through the swap zone) to the last page (the buffer) at the top of the previous bank, after its activation (call). Please see fig.IIIB in the first article ([10]).

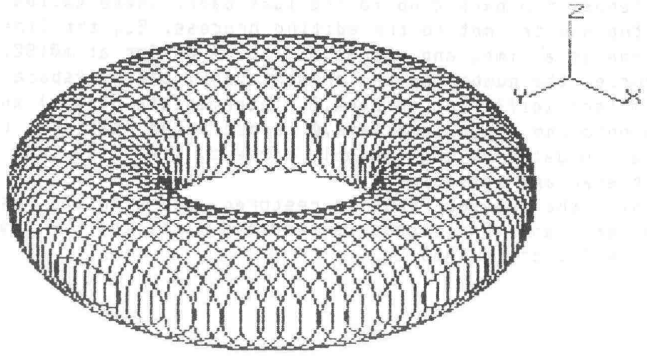
The 2k DOS (Kernel) at $2200 in the memory maps, is an adapted and reduced version of the disk drive software in OS65D V3.3, with almost the same commands as the original.

Ask the Paperware Service for the complete source-listings and more information, if you're interested.

References:

1. *junior computer (fig.2)* - elektor MAY.80
2. *8K RAM + 4, 8 or 16K EPROM on a single card* - elektor SEP.80
3. *the junior computer memory card* - elektor OCT.80
4. *8K RAM card (missing link)* - elektor OCT.80
5. *dynamic RAM card* - elektor APR.82
6. *floppy-disk interface for the junior (fig.8)* - elektor DEC.82
7. *64K on the 16K Dynamic RAM Card* - elektor SEP.83
8. *R650X and R651X MICROPROCESSORS (CPU)* - Rockwell DataSheet N.D39 Rev.5 AUG.83
9. *64-way busboard* - elektor DEC.83
10. *bank switching for the junior computer* - DE 6502 KENNER #43, Apr.86
11. *DOS 65 Corner* - DE 6502 KENNER #44, Jun.86

(F.LOPES and A.A.Ferreira, Colégio dos Orfãos, L. Baltazar Guedes,
4300 PORTO PORTUGAL)



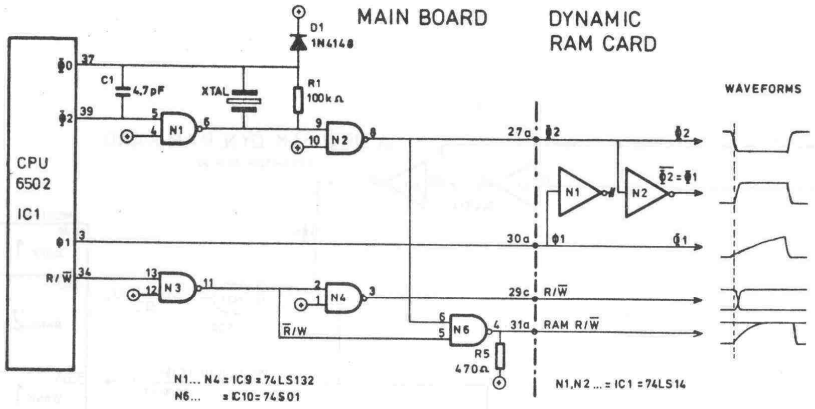


fig.1 - the problem of the delay in the rising edge of the Φ_1 clock. The relevant parts of the Junior's main board (1) and dynamic RAM card are redesigned here for clarity, with the respective waveforms; a modification is shown to obtain a sharper "Φ_1".

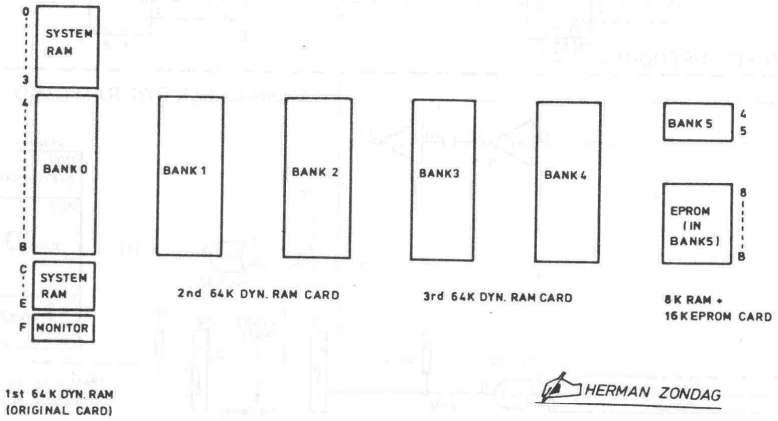


fig.4 - memory organization for a 6-banks, euracard-system (3x 64K dyn. RAM cards), in a 7-connector bus board.

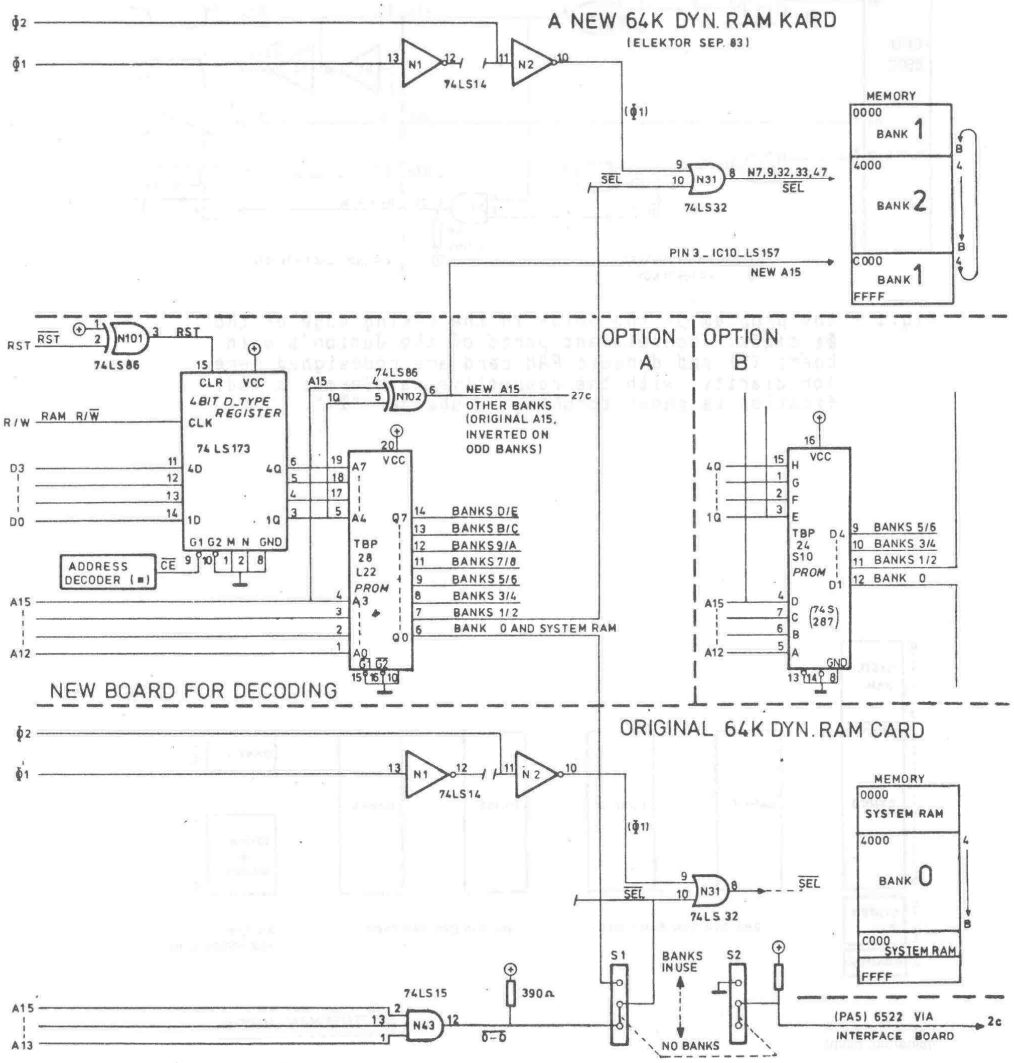


fig.2D- an overall view of the system: the decoding board, now with a PROM, and two 64K dyn. RAM cards (5,6,7). Much of the ICs of the original project disappeared, and even some others in the dynamic cards are now also unnecessary.

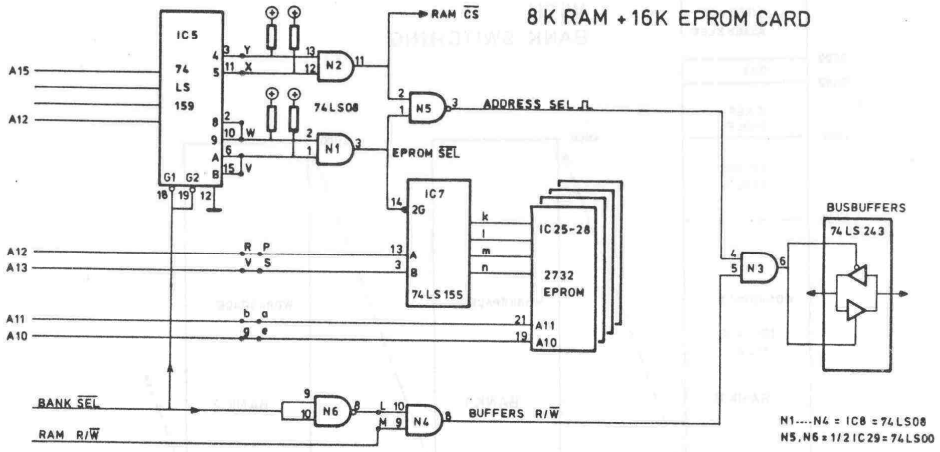


fig.5 - the RAM + EPROM card (2,3,4), unused, is picked-up again and put to work as the last bank, with RAM ($4000→5FFF) and EPROM ($8000→BFFF).

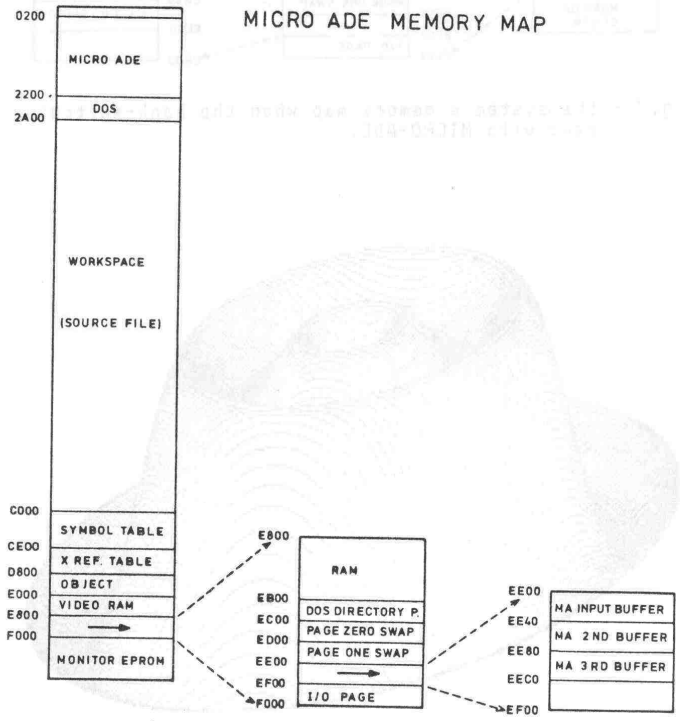


fig.6 - memory map, without the use of banks.

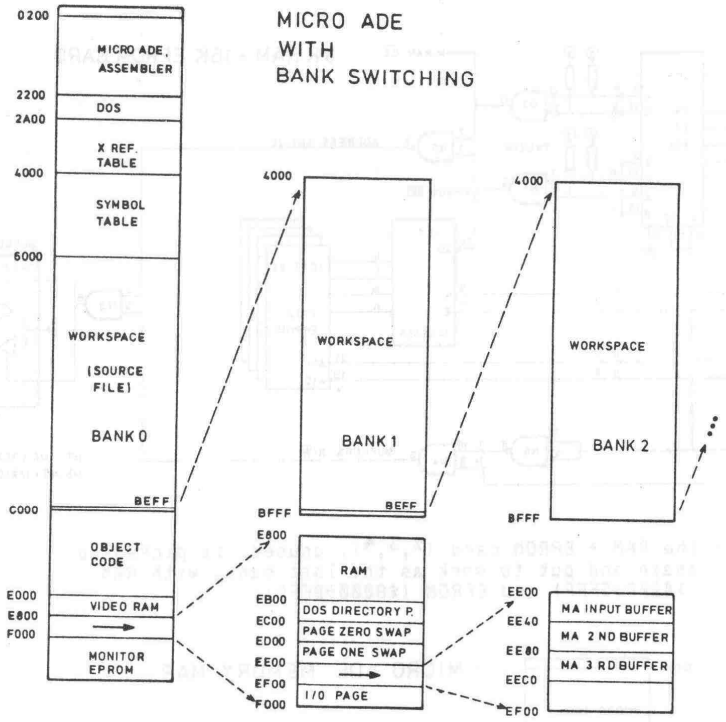
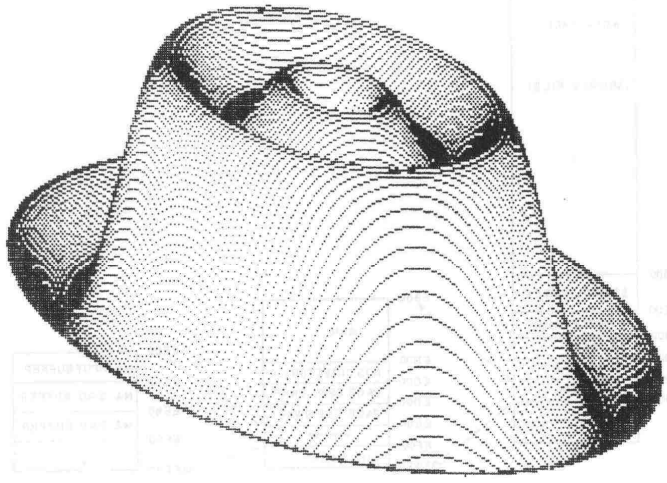


fig.7 - the system's memory map when the bank-switching is used with MICRO-ADE.



```
16670:
16680:                  *** DECREMENT WORKSPACE POINTER *** (23E6)
16690:
16691:                  ...
16700:                  Bank.2 ($4000) → bank.1 ($BEFF); SOURCF ($C0 → BF)
16710:                  Bank.1 ($4000) → bank.0 ($BEFF)
16720:                  Bank.0 ($6000) → bank.0 ($6000) C=1
16730:                  BankFF ($2A00) → bankFF ($2A00) C=1
16740:
16750: 0AB5 48          DECWSP PHA          save Acc
16760: 0AB6 18                 CLC          C=0
16770: 0AB7 A5 10              LDAZ  WSPNTL page boundary?
16780: 0AB9 D0 03              BNE   DECBU
16790: 0ABB 38                 SEC          WSPNT=XX00
16800: 0ABC C6 11              DECZ  WSPNTH changed, C=1 *
16810:
16820: 0ABE C6 10       DECBU  DECZ  WSPNTL
16830: 0AC0 90 2D              BCC   DECRTN if only WSPNTL changed.
16840: 0AC2 8A                 TXA          WSPNT=(XX-1)FF; C=1
16850: 0AC3 48                 PHA          save X
16860: 0AC4 AE 5B EF           LDX   BANKRG which bank?
16870: 0AC7 08                 PHP          (C=1)*
16880: 0AC8 AD 58 10           LDA   SOURCE start of workspace
16890: 0ACB E9 01              SBCIM $01    (Base - 1) → $5F,3F,29 (carry is set)
16900: 0ACD C5 11              CMPZ  WSPNTH BASE<CMP>WSPNT * is pointer under BASE?
16910: 0ACF B0 04              BCS   DECB   if already below start of workspace.
16920: 0AD1 28                 PLP
16930: 0AD2 18                 CLC          flag for O.K.
16940: 0AD3 90 18              BCC   DECRT  always: still into workspace; C=0
16950:
16960: 0AD5 E6 10       DECB   INCZ  WSPNTL WSPNT=BASE-1 (FF); increment → BASE (00)
16970: 0AD7 28                 PLP          again, which bank? (C=1)*
16980: 0AD8 F0 0F              BEQ   DECR   if bank was 0.
16990: 0ADA 30 0A              BMI   DEC    if no banks in use (FF).
17000: 0ADC C6 10              DECZ  WSPNTL bank was intermediate; WSPNTL → TOP-1 (/FF)
17010: 0ADE 20 AF 0F           JSR   SWAPDN move buffer from base to top of previous bank
17020: 0AE1 18                 CLC          C=0, ok; now bank=0 or 1
17030:
17040: 0AE2 A2 BF              LDXIM TOPINT / top of intermediate banks
17050: 0AE4 CA                 DEX          WSPNT → $BEFF
17060: 0AE5 2C                 =     BIT
17070:
17080: 0AE6 A2 2A       DEC    LDXIM BASENB / there are no banks in use; WSPNT= $2A00
17090: 0AE8 2C                 =     BIT      (C=1)*
17100:
17110: 0AE9 A2 60       DECR   LDXIM BASEB0 / bank was already 0; WSPNT= $6000;(C=1)*
17120: 0AEB 86 11              STX   WSPNTH = $2A00/6000/BEFF
17130:
17140: 0AED 68          DECRT  PLA          C=1 or C=0
17150: 0AEE AA                 TAX          restore X
17160:
17170: 0AEF 68          DECRTN PLA          restore Acc
17180: 0AF0 60                 RTS
17190:
17200:
```

```
19540:                ***** INCREMENT WORKSPACE POINTER ***** (250D)
19550:
19560:                Acc, X and Y  reg. are saved
19570:
19580:                Bank.FF ($BFFF)  →  Bank.FF ($2A00) C=1 (no banks in use)
19590:                INTERMEDIATE BANK ($BEFF)  →  NEXT BANK ($4000)
19610:                LAST BANK ($BFFF)  →  BANK.0 ($6000) C=1
19620:
19630:                Return with C=0 if still into workspace
19640:                 or with C=1 if the top of source-space was reached
19650:
19660: 0C30 18        INCWSP CLC
19670: 0C31 E6 10             INCZ  WSPNTL
19680: 0C33 D0 59             BNE   INCRTN with C=0
19690: 0C35 48                PHA           save Acc. WSPNTL=00
19700: 0C36 E6 11             INCZ  WSPNTH
19710: 0C38 A5 11             LDAZ  WSPNTH
19720: 0C3A CD 59 10          CMP   SOURCF top of workspace? (top of bank?) C0 or BF
19730: 0C3D 90 4E             BCC   INCRT   if still in workspace return with C=0.
19740: 0C3F 8A                TXA           WSPNTH = TOP OF WORKSPACE (this bank); C=1
19750: 0C40 48                PHA           save index X
19760: 0C41 AE 5B EF          LDX   BANKRG banks in use?
19770: 0C44 30 41             BMI   INC     if no banks (FF): TOP of WORKSPAC. Return with C=1
19780: 0C46 EC 5E EF          CPX   LASTBK BANKS IN USE: last bank?
19790: 0C49 90 05             BCC   INCSWP if bank was not the last, with C=0.
19800:                                      last bank: change to bank.0; C=1
19810: 0C4B 20 E1 0F  INCSPD  JSR   DOWNS   swap down (bank.X 4→3 3→2 2→1 1→0; carry=)
19820: 0C4E B0 34             BCS   INCLDX always (C=1)
19830:
19840: 0C50 98        INCSWP  TYA           C=0
19850: 0C51 48                PHA
19860: 0C52 A0 00             LDYIM $00     the bank was not the last; C=0
19870:
19880: 0C54 B9 00 BF  INCSWA  LDAAY TOPINT ($BF00)
19890: 0C57 99 00 EB          STAAY SWAPZN swap zone ($EB00)
19900: 0C5A C8                INY           swap buffer
19910: 0C5B D0 F7             BNE   INCSWA
19920: 0C5D 20 37 10          JSR   INXBNK (INX,STX,STX); switch to next bank
19930: 0C60 EC 5E EF          CPX   LASTBK last bank now?
19940: 0C63 90 08             BCC   INCP
19950: 0C65 18                CLC           falling into last bank: no buffer at top
19960: 0C66 A9 C0             LDAIM TOPLBK /
19970: 0C68 8D 59 10          STA   SOURCF intermediate bank → last bank: change SOURCF
19980: 0C6B D0 09             BNE   INCWS   always with C=0
19990:
20000: 0C6D A2 40     INCP    LDXIM BASINT / intermediate bank
20010: 0C6F 8E 58 10          STX   SOURCE
20020: 0C72 CA                DEX           bank.0 → bank.1: change SOURCE
20030: 0C73 8E 57 10          STX   SOURCM C=0
20040:
20050: 0C76 B9 00 EB  INCWS   LDAAY SWAPZN swap buffer back
20060: 0C79 99 00 40          STAAY BASINT ($4000)
20070: 0C7C C8                INY
20080: 0C7D D0 F7             BNE   INCWS   ;Y=00
20090: 0C7F 68                PLA
20100: 0C80 A8                TAY
20110: 0C81 A2 40             LDXIM BASINT / bank=1, 2, 3 or 4; WSPNT= $4000; C=0
20120: 0C83 2C                =     BIT     NV-Z flags
```

```
20130:
20140: 0C84 A2 60     INCLDX LDXIM BASEB0 / set bank.0 source-file base address
20150: 0C86 2C               =     BIT    NV-Z flags              (C=1)*
20160:
20170: 0C87 A2 2A     INC    LDXIM BASENB / no banks in use: WSPNT=$2A00; (C=1)*
20180: 0C89 86 11            STXZ  WSPNTH
20190:
20200: 0C8B 68        INCR   PLA
20210: 0C8C AA               TAX           restore index X
20220:
20230: 0C8D 68        INCRT  PLA           restore Acc
20240:
20250: 0C8E 60        INCRTN RTS
20260:

25760:                *** SET BANK ZERO *** SET THE MEMORY MAPPING ***
25770:
25780: 0F68 A2 00     SETBK0 LDXIM $00
25790: 0F6A 8E FF F7         STX   BANKSW select always bank.0
25800: 0F6D AC 5B EF         LDY   BANKRG which bank? FF or 00, 01, 02...
25810: 0F70 10 09            BPL   SETMEM if banks in use.
25820:
25830: 0F72 CA               DEX           (FF) *** NO BANKS IN USE ***
25840: 0F73 8E 5B EF         STX   BANKRG MA memory-map is already set: was read from dsk
25850: 0F76 8E 5E EF         STX   LASTBK
25860: 0F79 D0 25            BNE   RESB    always.
25870:                                     (00) *** BANKS IN USE ***
25880: 0F7B 8E 5B EF  SETMEM STX   BANKRG save bank number (00) !
25890: 0F7E A2 60            LDXIM BASEB0 / set bank.0 memory map
25900: 0F80 8E 5B 10         STX   SOURCE source = $60oo
25910: 0F83 8E 5B 10         STX   SYMF   symbol up to $6000
25920: 0F86 CA               DEX           (5F)
25930: 0F87 8E 57 10         STX   SOURCM source - 1 = $5Fff
25940: 0F8A A2 C0            LDXIM TOPNBK /
25950: 0F8C 8E 5E 10         STX   OBJECT object code = $C0oo
25960: 0F8F CA               DEX           (BF)
25970: 0F90 8E 59 10         STX   SOURCF source up to $BFoo
25980: 0F93 A2 40            LDXIM BASINT /
25990: 0F95 8E 5A 10         STX   SYMBOL symbol = $40oo
26000: 0F98 8E 5D 10         STX   XRFMAX xref up to $40oo
26010: 0F9B A2 2A            LDXIM BASENB /
26020: 0F9D 8E 5C 10         STX   XREFST xref = $2Aoo
26030:
26040:
26050:                *** RESET WORKSPACE POINTER
26060:                    TO THE START OF THIS BANK
26070:                    AND LINE-NUMBER TO 0000   * (23F4)
26080:
26090: 0FA0 AE 57 10  RESB   LDX   SOURCM start of source-file minus 1 (29/5F/3F/3F)
26100: 0FA3 86 11            STX   WSPNTH workspace pointer = $xxFF
26110: 0FA5 A2 FF            LDXIM $FF
26120: 0FA7 86 10            STX   WSPNTL
26130: 0FA9 E8               INX           00
26140: 0FAA 86 15            STXZ  LNUMBL line number = 0000
26150: 0FAC 86 16            STXZ  LNUMBH
26160: 0FAE 60        RETURN RTS           X=00
26170:
26180:
```

```
26190:
26200:                  *** SWAP BUFFER FROM ONE BANK'S BASE
26210:                      TO THE PREVIOUS BANK'S TOP    ***
26220:
26221:                  X must come loaded with bank number
26230:                  X=X-1; Y is saved; V and C flags unchanged;
26240:                  N and Z flags set according to X reg. (bank.x)
26250:
26260: 0FAF 98          SWAPDN TYA
26270: 0FB0 48                 PHA
26280: 0FB1 A0 00              LDYIM $00     swap buffer at base to swapzone
26290: 0FB3 B9 00 40   DECSWA  LDAAY BASINT
26300: 0FB6 99 00 EB           STAAY SWAPZN
26310: 0FB9 C8                 INY
26320: 0FBA D0 F7              BNE   DECSWA
26330: 0FBC CA                 DEX           change (decrement) bank: X=3 or 2 or 1 or 0
26340: 0FBD 20 3A 10           JSR   STOBNK (STX,STX) C=
26350: 0FC0 F0 07              BEQ   DECSW  if bank.0 now.
26360: 0FC2 A9 BF              LDAIM TOPINT / last bank → intermediate bk. change top
26370: 0FC4 8D 59 10           STA   SOURCF
26380: 0FC7 D0 0B              BNE   DECSWP always.
26390:
26400: 0FC9 A0 60      DECSW   LDYIM BASEB0 / bank.1 → bank.0 : set base address
26410: 0FCB 8C 58 10           STY   SOURCE
26420: 0FCE 88                 DEY
26430: 0FCF 8C 57 10           STY   SOURCM
26440: 0FD2 A0 00              LDYIM $00
26450: 0FD4 B9 00 EB   DECSWP  LDAAY SWAPZN swap buffer back: from swapzone to top
26460: 0FD7 99 00 BF           STAAY TOPINT
26470: 0FDA C8                 INY
26480: 0FDB D0 F7              BNE   DECSWP
26490: 0FDD 68                 PLA
26500: 0FDE A8                 TAY
26510: 0FDF 8A                 TXA           set flags according to bank.X
26520: 0FE0 60                 RTS           carry unchanged
26530:
26540:
26550:
26560:                  *** SWAP DOWN EVERY BANK'S BUFFER ***
26570:                  Swap modified lines over the previous, unmodified ones.
26571:                  X must be loaded with bank number (EOF) beforehand
26580:
26590: 0FE1 20 AF 0F   DOWNS   JSR   SWAPDN swap buffer at base to buffer at top
26600:                                     carry unchanged; X=X-1
26610: 0FE4 F0 02      DNSWAP  BEQ   DSPRTN if bank.0 no more swaps; return.
26620: 0FE6 10 F9              BPL   DOWNS  if not yet bank.0.
26630: 0FE8 60         DSPRTN  RTS          no swap also, if no banks in use; C=.
26640:
26650:
26660:
26670:                  *** SAVE PRESENT BANK REGISTER
26680:                      AND ITS POINTERS ONTO THE STACK ***
26690:
26700: 0FE9 68          SAVBNK PLA           only N and Z flags change
26710: 0FEA 85 0C              STAZ  STACKA save return address
26720: 0FEC 68                 PLA
26730: 0FED 85 0D              STAZ  STACKB
26740: 0FEF AD 5B EF           LDA   BANKRG
26750: 0FF2 48                 PHA          save pointers onto the stack
26760: 0FF3 A5 10              LDAZ  WSPNTL
```

```
26770: 0FF5 48              PHA
26780: 0FF6 A5 11           LDAZ  WSPNTH
26790: 0FF8 48              PHA
26800: 0FF9 AD 57 10        LDA   SOURCM
26810: 0FFC 48              PHA
26820: 0FFD AD 58 10        LDA   SOURCE
26830: 1000 48              PHA
26840: 1001 AD 59 10        LDA   SOURCF
26850: 1004 48              PHA
26860: 1005 A5 0D     SAVRST LDAZ  STACKB restore return address
26870: 1007 48              PHA
26880: 1008 A5 0C           LDAZ  STACKA
26890: 100A 48              PHA
26900: 100B 60              RTS
26910:
26920:
26930:
26940:                *** RESTORE THE PREVIOUS BANK AND ITS POINTERS ***
26950:
26960: 100C 68        RSTBNK PLA
26970: 100D 85 0C           STAZ  STACKA save return address
26980: 100F 68              PLA
26990: 1010 85 0D           STAZ  STACKB
27000: 1012 68              PLA         restore pointers
27010: 1013 8D 59 10        STA   SOURCF
27020: 1016 68              PLA
27030: 1017 8D 58 10        STA   SOURCE
27040: 101A 68              PLA
27050: 101B 8D 57 10        STA   SOURCM
27060: 101E 68              PLA
27070: 101F 85 11           STAZ  WSPNTH
27080: 1021 68              PLA
27090: 1022 85 10           STAZ  WSPNTL
27100: 1024 68              PLA
27110: 1025 86 FD           STX   TEMPX  save index X
27120: 1027 AA              TAX         get bank from stack
27130: 1028 20 30 10        JSR   SETBNK set bank.X
27140: 102B A6 FD           LDX   TEMPX
27150: 102D 4C 05 10        JMP   SAVRST restore return address & return
27160:
27170:
27180:
27190:                *** SET BANKK.X ***
27200:
27210: 1030 10 08     SETBNK BPL   STOBNK if banks in use
27220: 1032 A2 FF           LDXIM $FF    if no banks
27230: 1034 8E 5B EF        STX   BANKRG flag it for shure!
27240:
27250: 1037 E8        INXBNK INX         (00)
27260: 1038 F0 03           BEQ   STOBNK +03 always.
27270:
27280: 103A 8E 5B EF  STOBNK STX   BANKRG bank register
27290: 103D 8E FF F7        STX   BANKSW bank switch
27300: 1040 60              RTS
27310:
27320:
```

```
27330:
27340:                  *** PRINT THE NUMBER OF THE BANK IN ACC OR X REG ***
27350:
27360: 1041 8A          PBANKX TXA
27370: 1042 48          PBANKA PHA
27380: 1043 20 A1 29           JSR   PRINTV "bank."
27390: 1046 62                 =     'b
27400: 1047 61                 =     'a
27410: 1048 6E                 =     'n
27420: 1049 6B                 =     'k
27430: 104A 2E                 =     '.
27440: 104B 03                 =     ETX
27450: 104C 68                 PLA
27460: 104D 20 7E F5           JSR   PRNIBL number
27470: 1050 20 A1 29           JSR   PRINTV "; "
27480: 1053 3B                 =     ';
27490: 1054 20                 =     SP
27500: 1055 03                 =     ETX
27510: 1056 60                 RTS
27520:
27530:
27540:
27550:
```

FATE 65 C Format Lister/Assembler/Tape-utilities/Editor for the CMOS-6502 on Elektor's JUNIOR with PM and TM.
Complete source-listing of the 12K version.
Send cheque of Hfl.119,50 to W.v.Pelt (eurocheque 110,00)

FATE Format Lister/Assembler/Editor for Elektor's JUNIOR-computer with DOS of Koen van Nieuwenhove, Belgium.
ASK FOR PRICES.
Information about the DOS is also obtainable.

TOKENIZED Microsoft Basic Keywords and addresses KIM-1 and Elektor's JUNIOR KB-9 Basic.
W.L. van Pelt, The Netherlands.
Send cheque of Hfl. 14,50 to W.L.v.Pelt (eurocheque 5,00)

TOKENIZED Microsoft Basic Keywords and addresses SYM-1 Basic.
W.L. van Pelt, The Netherlands.
Send cheque of Hfl. 14,50 to W.L.v.Pelt (eurocheque 5,00)

TOKENIZED Microsoft Basic Keywords and addresses AIM-65 Basic.
W.L. van Pelt, The Netherlands.
Send cheque of Hfl. 14,50 to W.L.v.Pelt (eurocheque 5,00)

TOKENIZED Microsoft Basic Keywords and addresses APPLEsoft
W.L. van Pelt, The Netherlands.
Send cheque of Hfl. 14,50 to W.L.v.Pelt (eurocheque 5,00)

TOKENIZED Microsoft Basic Keywords and addresses CBM 40XX and CBM 80XX. Commodore Basic 4.0.
Nico de Vries, The Netherlands.
Send cheque of Hfl. 14,50 to W.L.v.Pelt (eurocheque 5,00)

TOKENIZED Microsoft Basic Keywords and addresses C - 16
Commodore Basic 3.5.
Nico de Vries, The Netherlands.
Send cheque of Hfl. 14,50 to W.L.v.Pelt (eurocheque 5,00)

TOKENIZED Microsoft Basic Keywords and addresses CBM 30XX
Commodore asic 2.0.
Nico de Vries, The Netherlands.
Send cheque of Hfl. 14,50 to W.L.v.Pelt (eurocheque 5,00)

TOKENIZED Microsoft Basic Keywords and addresses VIC-20
Commodore Basic V2.
Nico de Vries, The Netherlands.
Send cheque of Hfl. 14,50 to W.L.v.Pelt (eurocheque 5,00)

CENTRONICS PRINTER INTERFACE DEVICE 4 OR 5 ON COMMODORE 64
Ruud Uphoff, The Netherlands. Machinecode.
Send cheque of Hfl. 15,00 to W.L.v.Pelt (eurocheque 5,50)

THE VDU CARD
Article to assure that Elektor's VDU card works well.
Software (machinecode) with introduction.
Many members of the club use this software instead of the software from Elektor.
Authors: J.J.A. and J.A.J. Janssen, The Netherlands.
Transl.: Willem van Asperen.
System : Elektor's JUNIOR computer with VDU card.
Publ. : DE 6502 KENNER 31, Apr. 1984, p.17-26.
Send cheque of Hfl. 19.00 to W.L.v.Pelt (eurocheque 9,50)

THE GRAPHIC DISPLAY
Article with schematic diagram to do graphical work (plotting of graphs etc.) together with Elektor's VDU card.
Authors: J.J.A. and J.A.J. Janssen
Transl.: Willem van Asperen
System : Elektor's JUNIOR computer with VDU card.
Publ. : DE 6502 KENNER 36, Febr. 1985, p.5-11
Send cheque of Hfl. 14,50 to W.L.v.Pelt (eurocheque 5,00)

EXTENSIONS ON OS-65D V3.3 FOR ELEKTOR'S JUNIOR-computer.
Gert Klein, The Netherlands.
Dutch version published in DE 6502 KENNER nr.39, Aug.'86.
With the new DOS-commands :
DI to supply the directory of a diskette, SD to replace the old DI command in OS-65D, FO to format a diskette and to write an empty directory to track 12, CR to write a filename into the directory with trackrange to match, DE to delete a filename from the directory, CH to change filename1 into filename2, NU to supply the number of tracks by the Basic of assembly sourcefile. With four commands for further extensions: FT to load the FORTH interpreter/compiler, WF to warmstart the FORTH, MA to load the macro assembler by C. Moser, WS to warmstart the Moser-assembler.
Complete English assembly source listing with introduction for users of JUNIOR, but easy to adapt on your OCTOPUS.
Send cheque of Hfl. 17,50 to W.L.v.Pelt (eurocheque 8,00)

PATCHES ON KIM-1 MICROSOFT BASIC FOR ELEKTOR'S JUNIOR-comp
Koen van Nieuwenhove, Belgium/W.L. van Pelt, The Netherl.
With PMODE, TRACE, STEP RESET, PRINTER PG., VIDEO, AUTO, RENUMBER, DATASAVE, EDIT LINE, APPEND.
Send cheque of Hfl. 24,50 to W.L.v.Pelt (eurocheque 15,00)


```
 10REM    RTTY TX/RX VAN
 20REM F.VERGOOSSEN (PA3ELB)
 30REM    TEL.04754-1972
 40REM TTL IN PC3 VAN 8255
 50REM (MARK="1" SPACE="0")
 60REM AFSK OUT PB7 VAN 6522
 70REM (BLOKGOLF)
 80REM PTT OUT PB6 VAN 6522
 90REM (TX="1" RX="0")
100P.$12;?#E1=0;P."BEDIENEN ALS VOLGT:"''
110P."m = MENU"''r = RX"''t = TX"'
120P."c = CW ID."''b = BUFFER"''
130P."EVEN GEDULD A.U.B."
140REM RESET WRCVEC
150?#209=#FE;?#208=#52
160C=1000000;REM CLOCK FREQ.
170DIMII40,TT40,RR40,ZZ40
180F.Z=0TO40
190IIZ=#FFFF;TTZ=#FFFF
200RRZ=#FFFF;ZZZ=#FFFF
210N.;DIMK255,T26,R82,Q-1
220$K="HIER UW CALL";REM CALL
230REM CW SNELHEID 10-30 WPM
240V=20
250J=C*3/(5*V)-1;IFJ/256>255E.
260REM BAUDOTTABEL
270!R=#120E1318;R!4=#050B1610;R!8=#091E1A0C
280R!12=#0D030607;R!16=#01140A1D;R!20=#17190F1C
290R!24=#1115
300REM MORSETABEL
310R!26=#00000052;R!30=#6D2D5E00;R!34=#61732A19
320R!38=#3E3F296A;R!42=#2030383C;R!46=#2F272321
330R!50=#31350047;R!54=#06004C2A;R!58=#02091511
340R!62=#04100B14;R!66=#07120D1E;R!70=#1B160F05
350R!74=#0C03080A;R!78=#1D190E18;R!82=#13
360GOS.1300;REM ASSEMBLEER
370REM VOORKEURSINSTELLING
380%B=45.45;S=170;F=1275;I=0;G=1;D=1;A=3;E=1;N=1
390GOS.810;GOS.950;GOS.1160
400LI.II0;REM INIT SNELLE VDU
410LI.II18;REM INIT VIA
420P.$3$15$12;?#E1=0;@=12
430P.$30"        MENU RTTY TX/RX"'
440P."1 BAUDRATE          "%B
450P."2 SHIFT             "S
460P."3 LAAGSTE FREQUENTIE"F
470P."4 POLARITEIT                ";IFI=0P."MLSH"
480IFI P."MHSL"
490P."5 BLANK TEKEN               ";IFG=0P."AAN"
500IFG P."UIT"
510P."6 AUTO CR/LF                ";IFD=0P."AAN"
520IFD P."UIT"
530P."7 UNSHIFT ON SPACE          ";IFA=0P."AAN"
540IFA P."UIT"
550P."8 STOPBIT                   ";IFE=0P."  1"
560IFE P."1.5"
570P."9 LEESTEKENS                ";IFN=1P."INT"
```

```
580IFN=2P." WU"
590IFN=3P."MIL"
600P.'"r = NAAR RX"
610P.'"t = NAAR TX"
620P.'"b = WIJZIG BUFFER"
630Z=0;LI.II17;REM LEES TOETS
640REM r=RX
650IFZ=114P.$12$2;LI.RR0;G.420
660REM t=TX
670IFZ=116P.$12$2;LI.TT0;G.420
680REM b=BUFFER
690IFZ=98G.1200
700REM 1-9
710IFZ<49ORZ>57G.630
720Z=Z-48;GOS.730;G.430
730IFZ>1G.830
740REM BAUDRATE
750FIF%B<50%B=50;G.810
760FIF%B<57%B=57;G.810
770FIF%B<75%B=75;G.810
780FIF%B<100%B=100;G.810
790FIF%B<110%B=110;G.810
800FIF%B>45.45%B=45.45
810B=C/%B-1;IFB/256>255G.750
820R.
830IFZ>2G.910
840REM FREQUENTIESHIFT
850IFS<170S=170;G.950
860IFS<300S=300;G.950
870IFS<425S=425;G.950
880IFS<850S=850;G.950
890IFS<1000S=1000;G.950
900IFS>85S=85;G.950
910IFZ>3G.1020
920REM LAAGSTE FREQUENTIE
930IFF<2125F=2125;G.950
940IFF>1275F=1275
950H=C/2/(F+S)-2;L=C/2/F-2
960IFL/256>255G.930
970REM POLARITEIT
980IFI=0Y=L;L=H;H=Y
990REM INIT VIA
1000IFI=8ANDZ=2ORZ=3ORZ=4LI.II20
1010R.
1020IFZ>8G.1140
1030REM POLARITEIT
1040IFZ=4I=8-I;G.950
1050REM BLANK TEKEN
1060IFZ=5G=1-G
1070REM AUTO CR/LF
1080IFZ=6D=1-D
1090REM UNSHIFT ON SPACE
1100IFZ=7A=3-A
1110REM STOPBIT
1120IFZ=8E=1-E
1130R.
1140REM LEESTEKENS
1150N=N+1;IFN>3N=1
1160$T="-?:e3eee8b().,9014'57=2/6+"
1170IFN>1;T?3=36;T?6=38;T?7=35;T?21=59;
    T?25=34
1180IFN=3;T?5=33;T?7=101;T?9=39;T?18=98
1190R.
1200P.$12"BUFFER: (m=MENU)"'
1210W=#8200
1220LI.II17;IFZ=127IFW>#8200W=W-1;P.$Z
1230IFZ=127G.1220
1240?W=Z;P.$Z;IFZ=13P.$10
1250IFZ=109G.420
1260W=W+1;IFW<#9FFFG.1220
1270?W=109;P."BUFFER VOL"
1280P.'"DRUK RETURN"
1290DOLI.II17;U.Z=13;G.420
1300P.$21;F.Y=0TO1;P=Q;[
1310:II0\WRCVEC
1320LDA@II1%256;STA#208
1330LDA@II1/256;STA#209
1340RTS
1350:II1\PRINTER
1360PHA\SAVE A
1370\CTRL B
1380CMP@2;BEQII2
1390\CTRL C
1400CMP@3;BEQII5
1410\NORMAAL LF
1420CMP#FE;BEQII4
1430\TEST NAAR PRINTER
1440LDA#B80C;AND@#E;BEQII4
1450\TEST PRINTER AAN
1460BIT#B801;BMIII4
1470PLA\LOAD A
1480\NAAR PRINTER
1490STA#B801
1500PHA\SAVE A
1510LDA#B80C
1520AND@#F0;ORA@#C
1530STA#B80C
1540ORA@2;BNEII3
1550:II2\PRINTER AAN
1560LDA@#7F;STA#B803
1570LDA#B80C
1580AND@#F0;ORA@#E
1590:II3\AAN OF UIT
1600STA#B80C
1610:II4\EINDE
1620PLA;JMPII7
1630:II5\PRINTER UIT
1640LDA#B80C
1650AND@#F0;BCSII3
1660:II7\SAVE REGISTERS
1670PHP;PHA;CLD;STY#E5;STX#E4
1680\SCHRIJFROUTINE
1690JSRII8
1700\LOAD REGISTERS
1710JMP#FE5F
1720:II8\TEST CTRL CODES
1730\TEST SCHERM AAN
```

```
1740CMP@6;BEQII13
1750\TEST SCHERM UIT
1760CMP@21;BEQII14
1770\TEST CURSOR POSITIE
1780LDY#E0;BMIII15
1790\TEST ESCAPE
1800CMP@27;BEQII13
1810\TEST PIEPTOON
1820CMP@7;BEQII16
1830\ZOEK CTRL-CODE OP
1840JSRII12;LDX@10;JSR#FEC5
1850\NIET GEVONDEN
1860BNEII9
1870\VOER CODE UIT
1880JMP#FEB7
1890:II9\TEST CODE
1900CMP@#20
1910\CONTROL CODE
1920BCCII12
1930\BEPAAL BEELDCODE
1940ADC@#1F;BMIII10;EOR@#60
1950:II10\KARAKTER OP SCHERM
1960STA(#DE),Y;INY
1970\TEST EINDE REGEL
1980CPY@#20;BCCII11
1990\VOLGENDE REGEL
2000JSR#FDEC;LDY@0
2010:II11\SAVE INDEX
2020STY#E0
2030:II12\CURSOR ON/OFF
2040PHA;LDA(#DE),Y;EOR#E1
2050STA(#DE),Y;PLA
2060RTS\TERUG
2070:II13\KARAKTERMODE
2080CLC;LDX@0;STX#B000
2090:II14
2100LDX@2;PHP;ASL#DE,X
2110PLP;ROR#DE,X
2120:II15RTS\TERUG
2130:II16\PIEPTOON
2140LDA@5;LDY@64;JMP#FD1D
2150:II17\TOETSLEESROUTINE
2160JSR#FFE3;STA#33B;RTS
2170:II18\INIT VIA
2180\INTERRUPTS UIT
2190LDA@#7F;STA#B80E
2200\PB6,7 OUTPUT
2210LDA@#C0;STA#B802
2220\CONT UITVOER PB7
2230STA#B80B
2240\CLEAR FLAGS
2250LDA@0;LDX@#10
2260:II19STA#80,X
2270DEX;BPLII19
2280\MSB VAN BUFFER
2290LDA@#60;STA#8E;STA#90
2300:II20\MARK
2310LDA#329;STA#B804
2320LDA#344;STA#B805
2330RTS\TERUG
2340:RR0\START RX
2350\PTT=0
2360LDA@0;STA#B800
2370\UNSHIFT
2380STA#82
2390\RX INDICATOR
2400LDA@146;STA#81FF
2410\POORT C INPUT
2420LDA@#8B;STA#B003
2430:RR2\READ KEY
2440JSRTT28
2450\LAAD TOETS
2460LDA#84
2470\TEST t=TX
2480CMP@116;BNERR4
2490LDA@13;STA#81\CR+LF
2500JSRRR27\DRUK AF
2510\CLEAR FLAG
2520LDA@0;STA#88
2530JMPTT0\TX
2540:RR4\TEST m=MENU
2550CMP@109;BNERR10
2560RTS\MENU IN BASIC
2570:RR10\METING
2580LDA#B002;AND@8
2590\POLARITEIT
2600EOR#32A
2610BNERR11\SPACE
2620\MARKINDICATOR
2630LDA@112;STA#81FD
2640BNERR2\TERUG
2650:RR11\BITTELLER
2660LDA@4;STA#80
2670:RR12\BAUDRATE TELLER
2680LDA#323;STA#B808
2690LDA#33E;STA#B809
2700\RESET DUURTELLER
2710LDA@128;STA#8A
2720\READ KEY
2730JSRTT28
2740:RR13\METING
2750LDA#B002;AND@8
2760\POLARITEIT
2770EOR#32A
2780BNERR14\SPACE
2790\MARK
2800INC#8A;BNERR16
2810DEC#8A;BNERR16
2820:RR14\SPACE
2830DEC#8A;BNERR16
2840INC#8A
2850:RR16\TEST EINDE TELLER
2860LDA#B80D;AND@#20;BEQRR13
2870\NEG=MARK POS=SPACE
2880LDA#8A;CMP@128
2890\SPACEINDICATOR
```

```
2900LDA@67;BCCRR18
2910\TEST EERSTE BIT MARK
2920LDA#80;CMP@4;BNERR17
2930BEQRR2\OPNIEUW
2940:RR17\MARKINDICATOR
2950LDA@112
2960:RR18\INDICATOR
2970STA#81FD
2980ROL#80\SCHUIF BITS
2990BCCRR12\VOLGENDE BITS
3000LDA#80\BAUDOT
3010\WORD SPACE
3020CMP@4;BNERR19
3030\UNSHIFT ON SPACE
3040LDA#82;AND#322;STA#82
3050\SPATIE
3060LDA@32;BNERR26
3070:RR19\CARRIAGE RETURN
3080CMP@2;BNERR20
3090\RETURN
3100LDA@13;BNERR26
3110:RR20\FIGURES
3120CMP@27;BNERR21
3130LDA@1;STA#82;JMPRR2
3140:RR21\LETTERS
3150CMP@31;BNERR22
3160LDA@0;STA#82;JMPRR2
3170:RR22\VERTALING
3180LDY@25\TELLER
3190:RR23\VERGELIJK
3200CMPR,Y;BEQRR24
3210DEY;BPLRR23;JMPRR2
3220:RR24\TEST FLAG
3230LDA#82;BEQRR25
3240LDAT,Y;JMPRR26
3250:RR25\ASCIIWAARDE
3260TYA;CLC;ADC@65
3270:RR26\DRUK AF
3280STA#81;JSRRR27
3290JMPRR2\TERUG
3300:RR27\CURSOR POSITIE
3310LDA#DE;CMP@#E0;BNERR28
3320LDA#DF;CMP@#81;BNERR28
3330\LAAD t OF r
3340LDY#81FF
3350\WIS SCHERM
3360LDA@12;JSR#FE55
3370\ZET t OF r TERUG
3380STY#81FF
3390:RR28\OSWRCH
3400LDA#81;JSR#FFE9
3410RTS\TERUG
3420:TT0\START TX
3430\POORT C OUTPUT
3440LDA@#8A;STA#B003
3450\PTT=1
3460LDA@#40;STA#B800
3470\GEEN (UN)SHIFT
3480LDA@3;STA#82
3490\TX INDICATOR
3500LDA@148;STA#81FF
3510\MARKINDICATOR
3520LDA@112;STA#81FD
3530:TT1\DUBBEL CODE
3540LDA#83;BEQTT2;STA#80
3550LDA@0;STA#83;JMPTT19
3560:TT2LDA#325;BNETT3
3570\TEST FLAG
3580LDA#88;CMP@64;BNETT3
3590\AUTOMATISCH RETURN
3600LDA@13;STA#81;JMPTT13
3610:TT3\READ KEY
3620JSRTT28
3630\LAAD TOETS
3640LDA#84
3650\TEST r=RX
3660CMP@114;BNETT4
3670LDA@13;STA#81\CR+LF
3680JSRRR27\DRUK AF
3690JMPRR0\RX
3700:TT4\TEST m=MENU
3710CMP@109;BNETT6
3720RTS\MENU IN BASIC
3730:TT6\TEST BUFFER
3740LDA#87;CMP#86;BNETT8
3750LDA#90;CMP#8E;BNETT8
3760\TEST FLAG
3770LDA#85;BEQTT7
3780\REPEAT KEY
3790BIT#B002;BVSTT7
3800\CLEAR FLAG
3810LDA@0;STA#85;BEQTT3
3820:TT7\BLANK TEKEN
3830LDA#328;BNETT3;JMPTT20
3840:TT8\VERHOOG INDEX
3850INC#86;BNETT9
3860\PAGINAGRENS
3870INC#8E;LDA#8E;CMP@#80
3880BNETT9
3890\BUFFERGRENS
3900LDA@#60;STA#8E
3910:TT9\LAAD BUFFER
3920LDY#86;LDA(#8D),Y
3930STA#81\SAVE ACCU
3940\TEST c=CW-ID
3950CMP@99;BNETT10
3960JSRZZ0\CW
3970JMPTT3\TERUG
3980:TT10\TEST b=BUFFER
3990CMP@98;BNETT11
4000JSRZZ11\BUFFER
4010JMPTT3\TERUG
4020:TT11\TEST SPATIE
4030CMP@32;BNETT12
4040\UNSHIFT ON SPACE
4050LDA#82;AND#322;STA#82
```

```
4060\WORD SPACE
4070LDA@4:STA#80:JMPTT19
4080:TT12\TEST RETURN
4090CMP@13:BNETT14
4100:TT13\FLAG RESET
4110LDA@#FF:STA#88
4120\CARRIAGE RETURN
4130LDA@2:STA#83
4140\LINE FEED
4150LDA@8:JMPTT20
4160:TT14\TEST TOETS
4170CMP@91:BMITT15
4180JMPTT6\TERUG
4190:TT15\TEST LETTERS
4200CMP@65:BMITT16
4210\LAAD CODE
4220TAY:LDAR-65,Y:STA#80
4230\TEST FLAG
4240LDA#82
4250BEQTT19\AL OP LETTERS
4260LDA@0:STA#82
4270LDA#80:STA#83
4280\LETTERS
4290LDA@31:JMPTT20
4300:TT16\TEST FIGURE
4310LDY@25\TELLER
4320:TT17\VERGELIJK
4330CMPT,Y:BEQTT18
4340DEY:BPLTT17
4350JMPTT6\TERUG
4360:TT18\LAAD CODE
4370LDAR,Y:STA#80
4380\TEST FLAG
4390LDA#82:CMP@1
4400BEQTT19\AL OP FIGURES
4410LDA@1:STA#82
4420LDA#80:STA#83
4430\FIGURES
4440LDA@27:JMPTT20
4450:TT19\DRUK AF
4460JSRRR27
4470\OPTELLEN FLAG
4480INC#88:LDA#80
4490:TT20\SCHUIF BITS
4500SEC:ROLA:ASLA:STA#80
4510:TT21\SCHUIF 1 BIT
4520ASL#80:BCCTT22
4530\MARK
4540LDA#329:STA#B806
4550LDA#344:STA#B807
4560LDA@112\MARK IND.
4570BCSTT23
4580:TT22\SPACE
4590LDA#320:STA#B806
4600LDA#348:STA#B807
4610LDA@67\SPACE IND.
4620:TT23\INDICATOR
4630STA#81FD
4640JSRTT26\WACHT 1 BIT
4650LDA#80:BNETT21
4660LDA#326:BEQTT24
4670JSRTT25
4680:TT24\TERUG
4690JMPTT1
4700:TT25\WACHT 1/2 BIT
4710LDA#33E:LSRA:TAY
4720LDA#323:RORA
4730STA#B808:STY#B809
4740JMPTT27\TEST EINDE
4750:TT26\WACHT 1 BIT
4760LDA#323:STA#B808
4770LDA#33E:STA#B809
4780:TT27\TEST EINDE
4790JSRTT28\READ KEY
4800LDA#B80D:AND@#20:BEQTT27
4810RTS\TERUG
4820:TT28\SCAN KEYS
4830JSR#FE71
4840BCSTT38\GEEN TOETS
4850\LOCK+CURSOR NEGEREN
4860CPY@8:BCSTT29
4870CPY@5:BCSTT38
4880:TT29\COPYTOETS NEGEREN
4890CPY@14:BEQTT38
4900\ASCIIWAARDE
4910JSRTT40
4920\TEST VORIGE TOETS
4930CMP#84:BNETT30
4940\TEST VASTHOUDEN
4950LDY#85:BNETT39
4960:TT30\TOETSWAARDE
4970STA#84
4980\TEST DELETE
4990CMP@#7F:BNETT33
5000:TT31\TEST BUFFER LEEG
5010LDY#87:CPY#86:BNETT32
5020LDY#90:CPY#8E:BEQTT37
5030:TT32\VERLAAG BUFFER
5040DEC#87:LDY#87:CPY@#FF
5050BNETT37
5060\PAGINAGRENS
5070DEC#90:LDY#90:CPY@#5F
5080BNETT37
5090\BUFFERGRENS
5100LDY@#7F:STY#90:BNETT37
5110:TT33\TEST BUFFER VOL
5120LDY#87:INY:BEQTT34
5130CPY#86:BNETT35\NIET VOL
5140LDY#90:CPY#8E
5150BNETT35:BEQTT38
5160:TT34\BUFFERGRENS
5170LDY#90:INY:CPY#8E
5180BNETT35\NIET VOL
5190LDY#87:INY:CPY#86
5200BEQTT38\VOL
5210:TT35\VERHOOG BUFFER
```

```
5220INC#87:BNETT36
5230\PAGINAGRENS
5240INC#90:LDY#90:CPY@#80
5250BNETT36
5260\BUFFERGRENS
5270LDY@#60:STY#90
5280:TT36\KEY IN BUFFER
5290LDY#87:STA(#8F),Y
5300:TT37\ZET FLAG
5310LDA@1:STA#85:BNETT39
5320:TT38\CLEAR FLAG
5330LDA@0:STA#85
5340:TT39RTS\TERUG
5350:TT40\ASCIIWAARDE KEY
5360PHP:CLD:JMP#FEB1
5370:ZZ0\MORSEIDENTIFICATIE
5380LDA@13:STA#81\CR+LF
5390JSRRR27\DRUK AF
5400LDA@0:STA#8A
5410\PAUZE 4 TELLEN
5420LDA@8:JSRZZ9
5430:ZZ2\LEES KARAKTER
5440LDY#8A:LDAK,Y:STA#81
5450\TEST EINDE
5460CMP@13:BNEZZ3
5470LDA@3:STA#82\SHIFT AF
5480LDA@13:STA#81\CR+LF
5490JSRRR27\DRUK AF
5500RTS\TERUG NAAR RTTY
5510:ZZ3\TEST SPATIE
5520CMP@32:BNEZZ4
5530JSRRR27\DRUK AF
5540\SPATIE 7 TELLEN (NOG 4)
5550LDA@8:JSRZZ9
5560BEQZZ8\VOLGENDE
5570:ZZ4\TEST KARAKTER
5580CMP@91:BPLZZ8
5590CMP@34:BMIZZ8
5600\LEES MORSETEKEN
5610TAX:LDAR-8,X:STA#80
5620BEQZZ8\GEEN MORSETEKEN
5630JSRRR27\DRUK AF
5640:ZZ5\SCHUIF IN CARRY
5650LSR#80
5660\CONTROLEER EINDE
5670BEQZZ7
5680\TOON IS SPACE
5690LDA#32D:STA#B806
5700LDA#348:STA#B807
5710\INDICATOR
5720LDA@67:STA#81FD
5730LDA@2\PUNT 1 TEL
5740BCCZZ6
5750LDA@6\STREEP 3 TELLEN
5760:ZZ6\PAUZE
5770JSRZZ9
5780\PAUZETOON IS MARK
5790LDA#329:STA#B806
5800LDA#344:STA#B807
5810\INDICATOR
5820LDA@112:STA#81FD
5830LDA@2\PAUZE 1 TEL
5840JSRZZ9
5850JMPZZ5\VOLGENDE
5860:ZZ7\NOG 2 TELLEN
5870LDA@4
5880JSRZZ9\PAUZE
5890:ZZ8\VOLGENDE LETTER
5900INC#8A:JMPZZ2
5910:ZZ9\AANTAL EENHEDEN
5920STA#89
5930:ZZ10\PAUZE 1 TEL
5940LDA#32B:STA#B808
5950LDA#346:STA#B809
5960JSRTT27\TEST EINDE
5970\VERLAAG TELLER
5980DEC#89:BNEZZ10
5990RTS\TERUG
6000:ZZ11\BUFFER
6010LDA@#82:STA#8C
6020LDY@0
6030:ZZ12\ZOEK EINDE TEKST
6040LDA(#8B),Y:CMP@109
6050\EINDE GEVONDEN
6060BEQZZ13
6070\NOG NIET GEVONDEN
6080INY:BNEZZ12
6090\PAGINAGRENS
6100INC#8C:LDA#8C:CMP@#A0
6110BNEZZ12\VERDER
6120\NIET GEVONDEN
6130BEQZZ15
6140:ZZ13\COPY IN KEYBUFFER
6150DEY:CPY@#FF:BNEZZ14
6160\PAGINAGRENS
6170DEC#8C:LDA#8C:CMP@#81
6180\BUFFERBEGIN
6190BEQZZ15\TERUG
6200:ZZ14\LAAD UIT BUFFER
6210LDA(#8B),Y
6220\COPY IN BUFFER
6230STY#8A:LDY#86
6240STA(#8D),Y
6250LDY#8A
6260\PAGINAGRENS
6270DEC#86:LDA#86:CMP@#FF
6280BNEZZ13
6290\BUFFERGRENS
6300DEC#8E:LDA#8E:CMP@#5F
6310BNEZZ13
6320LDA@#7F:STA#8E
6330BNEZZ13
6340:ZZ15\TERUG
6350RTS
6360JN.:P.$6:R.
```

```
0010 ;
0020 ;*******************************************
0030 ;**********          RBC          **********
0040 ;**********                       **********
0050 ;**********    READ BASICODE-2    **********
0060 ;**********    DATE 07-03-86      **********
0070 ;**********                       **********
0080 ;**********    P.LASKER           **********
0090 ;**********    DRAKENSTEYN 291    **********
0100 ;**********    7608 TR ALMELO     **********
0110 ;**********    05490-72178        **********
0120 ;**********                       **********
0130 ;*******************************************
0140 ;
0150 ;
0160 ; BASICODE READ PROGRAM FOR DOS-65 COMPUTER
0170 ; MAX 32K $2000 - $9FFF
0180 ;
0190 ; THE DATA FORMAT ON TAPE IS ASCII:
0200 ; 5 SEC 2400 HZ<>STX<>DATA<> / <>DATA<>ETX<>CHECKSUM<>5 SEC 2400 HZ
0210 ; MOST SIGNIFICANT ASCII-BIT IS "1".
0220 ; 1 = 2 * 2400 HZ
0230 ; 0 = 1 * 1200 HZ
0240 ; THE TRANSPORT RATE OF THE DATA IS 1200 BAUD
0250 ; BYTE: 1 STARTBIT (0) <> 8*DATA <> 2 STOPBITS (1)
0260 ;
0270            .OS
0280            .CE
0290            .ES
0300            .BA $C800
0310            .MC $C800
0320 ;
0330 ;
0340 ;**** PIA FLOPPY-DISK ****
0350 ;
0360 PBD        .DE $C002    ;DATA + DATADIRECTION B
0370 ;
0380 ;
0390 ;**** 6522 VIA-2 REGISTERS ****
0400 ;
0410 VIA2       .DE $C110
0420 T1CL       .DE VIA2+$04        ;T1, LATCH LOW, COUNTER LOW
0430 T1CH       .DE VIA2+$05        ;T1 COUNTER HIGH
0440 ACR        .DE VIA2+$0B        ;AUXILIARY CONTROL REGISTER
0450 PCR        .DE VIA2+$0C        ;PERIPHERAL CONTROL REGISTER
0460 IFR        .DE VIA2+$0D        ;INTERRUPT FLAG REGISTER
0470 IER        .DE VIA2+$0E        ;INTERRUPT ENABLE REGISTER
0480 ;
0490 ;
0500 ;**** TEMPORARY DATA BUFFERS ****
0510 ;
0520 CHSUM      .DE $CA00    ;CHECKSUM
0530 X          .DE CHSUM
0540 PRCNTL     .DE X+$01    ;PERIOD COUNTER
0550 PRCNTH     .DE X+$02
0560 ZERO       .DE X+$03    ;PERIOD-TIME
0570 HLFPTM     .DE X+$04    ;HALF PERIOD-TIME
0580 SIVL       .DE X+$05
0590 SIVH       .DE X+$06
0600 TIMECNTL   .DE X+$07    ;TIMER
0610 TIMECNTH   .DE X+$08
0620 SAVEACCU   .DE X+$09    ;SAVE ACCU
0630 SAVEX      .DE X+$0A    ;SAVE X-REG
0640 SAVEY      .DE X+$0B    ;SAVE Y-REG
0650 RAMSTARTL  .DE X+$0C    ;STARTADDRESS RAM
0660 RAMSTARTH  .DE X+$0D
0670 LENGTHL    .DE X+$0E    ;LENGTH OF FILE
0680 LENGTHH    .DE X+$0F
0690 ;
0700 ;
0710 ;**** EXTERNAL SUBROUTINE ****
0720 ;
0730 PRCHA      .DE $E000    ;PRINT CHARACTER ROUTINE
0740 KBIT       .DE $E144    ;INIT KEYBORD
0750 PIEPER     .DE $EC87    ;BUZZER
0760 ;
0770 ;
0780 ;**** DOS SUBROUTINES ****
0790 ;
0800 WRITE1     .DE $B027
0810 CREATE1    .DE $B039
0820 CLOSE1     .DE $B04B
0830 ERMES      .DE $B0B7
0840 ;
0850 ;
0860 ;*********************************************************
0870 ;***** START *****
0880 ;*********************************************************
0890            STA SAVEACCU        ;SAVE FILE-NAME POINTER
0900            STY SAVEY
0910 ;
0920 ;***** CREATE FILE *****
0930 ;
C806- A2 E1     0940            LDX #$E1    ;CREATE FILE R/W/D/ASCII
C808- 20 39 B0  0950            JSR CREATE1
C80B- B0 14     0960            BCS BUERA   ;ERROR
C80D- 8E 0A CA  0970            STX SAVEX   ;SAVE FILE NR
                0980 ;
C810- A9 17     0990            LDA #$17    ;DESELECT DRIVES
C812- 0D 02 C0  1000            ORA PBD
C815- 8D 02 C0  1010            STA PBD
                1020 ;
C818- 78        1030            SEI  ;NO INTERRUPTS
C819- A0 12     1040            LDY #$12
C81B- 20 1F C9  1050            JSR MESSY   ;PRINT 'READ BCODE-2'
C81E- 4C 24 C8  1060            JMP READ
                1070 ;
C821- 4C BE C9  1080 BUERA      JMP BUER
                1090 ;
                1100 ;*************************************************
                1110 ;**** INITIALISE AND RECEIVE ****
                1120 ;*************************************************
                1130 ;
C824- A9 7F     1140 READ       LDA #$7F
C826- 8D 1E C1  1150            STA IER     ;SET INTERRUPT DISABLE MODE
C829- A9 3F     1160            LDA #%3F    ;PB7 DISABLED, FREE-RUN DISABLED
C82B- 2D 1B C1  1170            AND ACR
C82E- 8D 1B C1  1180            STA ACR
C831- A9 00     1190            LDA #$00
C833- 8D 1C C1  1200            STA PCR     ;SET CB1 NEGTIVE EDGE DETECT
C836- 8D 00 CA  1210            STA CHSUM   ;CLEAR CHECKSUM
C839- 8D 0C CA  1220            STA RAMSTARTL       ;SET RAM START ADDRESS $2000
C83C- 8D 74 C8  1230            STA STAIND+$01      ;AND SET RAM POINTER
C83F- A9 20     1240            LDA #$20
C841- 8D 0D CA  1250            STA RAMSTARTH
C844- 8D 75 C8  1260            STA STAIND+$02
                1270 ;
                1280 ;**** HEADER ****
                1290 ;
C847- A9 10     1300 HEADER     LDA #$10
C849- 8D 02 CA  1310            STA PRCNTH  ;SET PERIOD COUNTER
C84C- 20 B8 C8  1320 HDR        JSR PERIOD
C84F- C9 40     1330            CMP #$40
C851- 90 F4     1340            BCC HEADER  ;PERIODE <<2400HZ
C853- C9 88     1350            CMP #$88
C855- B0 F0     1360            BCS HEADER  ;PERIODE >>2400HZ
C857- EE 01 CA  1370            INC PRCNTL
C85A- D0 F0     1380            BNE HDR     ;NOT 256 PERIODES
C85C- CE 02 CA  1390            DEC PRCNTH
C85F- D0 EB     1400            BNE HDR     ;NOT 16*256&PERIODES OF 2400HZ
C861- 0A        1410            ASL A       ;PERIODETIME *2 (=1200HZ)
C862- 38        1420            SEC
C863- E9 19     1430            SBC #$19    ;1200HZ TIME-100US=ZERO
C865- 8D 03 CA  1440            STA ZERO
                1450 ;
                1460 ;**** START-BIT ****
                1470 ;
C868- 20 BB C8  1480 STARTBIT   JSR HLFPER  ;FIND STARTBIT
C86B- CD 03 CA  1490            CMP ZERO
C86E- 90 F8     1500            BCC STARTBIT        ;NO 1200HZ THEN BRANCH
                1510 ;
                1520 ;**** READ BYTE ****
                1530 ;
C870- 20 FD C8  1540            JSR RBYT    ;READ ONE BYTE
C873- 8D FF FF  1550 STAIND     STA $FFFF   ;CHARACTER TO TABLE
C876- C9 03     1560            CMP #$03
C878- F0 1E     1570            BEQ EOT     ;END OF TEXT ?
C87A- EE 74 C8  1580            INC STAIND+$01      ;INCREMENT POINTER
C87D- D0 03     1590            BNE EORAM
C87F- EE 75 C8  1600            INC STAIND+$02
                1610 ;
                1620 ;**** CHECK END OF RAM ****
                1630 ;
C882- A9 9F     1640 EORAM      LDA #$9F    ;RAM MAX $9FFF
C884- CD 75 C8  1650            CMP STAIND+$02
C887- D0 DF     1660            BNE STARTBIT        ;NOT END OF RAM ?
C889- A9 FF     1670            LDA #$FF
C88B- CD 74 C8  1680            CMP STAIND+$01
C88E- D0 D8     1690            BNE STARTBIT
                1700 ;
                1710 ;**** END OF RAM ****
C890- A0 00     1720            LDY #$00
C892- 20 1F C9  1730            JSR MESSY   ;PRINT 'OUT OF MEMORY'
C895- 4C 84 C9  1740            JMP SAVERDATA
                1750 ;
                1760 ;**** END OF TEXT, GET CHEKSUM ****
                1770 ;
C898- 20 BB C8  1780 EOT        JSR HLFPER
C89B- CD 03 CA  1790            CMP ZERO
C89E- 90 F8     1800            BCC EOT
```

```
C8A0- 20 FD C8  1810          JSR RBYT    ;READ CHEKSUM
C8A3- AD 00 CA  1820          LDA CHSUM
C8A6- F0 08     1830          BEQ OKE
C8A8- A0 31     1840          LDY #$31
C8AA- 20 1F C9  1850          JSR MESSY   ;PRINT 'CHECKSUM ERROR'
C8AD- 4C 84 C9  1860          JMP SAVERDATA
C8B0- A0 44     1870 OKE      LDY #$44
C8B2- 20 1F C9  1880          JSR MESSY   ;PRINT 'DATA RECEIVED'
C8B5- 4C 84 C9  1890          JMP SAVERDATA
                1900 ;
                1910 ;*************************************************
                1920 ;***** MEASURE ONE PERIODTIME *****
                1930 ;*************************************************
                1940 ;
C8B8- 20 BB C8  1950 PERIOD   JSR HLFPER
C8BB- A9 10     1960 HLFPER   LDA #$10
C8BD- 2C 1D C1  1970 HLF      BIT IFR
C8C0- F0 FB     1980          BEQ HLF     ;NO ACTIVE EDGE THEN BRANCH
C8C2- 8D 1D C1  1990          STA IFR     ;CLEAR CB1 FLAG
C8C5- A9 10     2000          LDA #$10
C8C7- 4D 1C C1  2010          EOR PCR
C8CA- 8D 1C C1  2020          STA PCR     ;OPPOSITE ACTIVE CB1 EDGE DETECT
C8CD- A9 FF     2030          LDA #$FF
C8CF- AA        2040          TAX
C8D0- 4D 15 C1  2050          EOR T1CH    ;GET ELAPSED TIME
C8D3- 8D 08 CA  2060          STA TIMECNTH
C8D6- 8A        2070          TXA
C8D7- 4D 14 C1  2080          EOR T1CL
C8DA- 8D 07 CA  2090          STA TIMECNTL
C8DD- A9 FF     2100          LDA #$FF
C8DF- 8D 14 C1  2110          STA T1CL    ;RESET TIMER
C8E2- 8D 15 C1  2120          STA T1CH
C8E5- 4E 08 CA  2130          LSR TIMECNTH        ;DIVIDE TIME BY 4
C8E8- 6E 07 CA  2140          ROR TIMECNTL        ;AND PUT IT IN TIMECNTL
C8EB- 4E 08 CA  2150          LSR TIMECNTH
C8EE- 6E 07 CA  2160          ROR TIMECNTL
C8F1- AD 07 CA  2170          LDA TIMECNTL
C8F4- AA        2180          TAX
C8F5- 18        2190          CLC
C8F6- 6D 04 CA  2200          ADC HLFPTM  ;FULL PERIOD T IN ACCU
C8F9- 8E 04 CA  2210          STX HLFPTM  ;SAVE HALF PERIOD T
C8FC- 60        2220          RTS
                2230 ;
                2240 ;*************************************************
                2250 ;***** READ ONE BYTE *****
                2260 ;*************************************************
                2270 ;
C8FD- A0 08     2280 RBYT     LDY #$08    ;SET BIT COUNTER
C8FF- 48        2290 RB       PHA
C900- 20 B8 C8  2300          JSR PERIOD
C903- CD 03 CA  2310          CMP ZERO
C906- B0 06     2320          BCS FNDZRO  ;1200HZ?
C908- 20 B8 C8  2330          JSR PERIOD  ;SECOND 2400HZ PERIOD ?
C90B- 38        2340          SEC
C90C- B0 01     2350          BCS SHIFT   ;BRANCH ALWAYS
C90E- 18        2360 FNDZRO   CLC
C90F- 68        2370 SHIFT    PLA
C910- 6A        2380          ROR A       ;NEXT BIT TO ACCU
C911- 88        2390          DEY
C912- D0 EB     2400          BNE RB      ;NOT 8 BITS ?
C914- 48        2410          PHA
C915- 4D 00 CA  2420          EOR CHSUM
C918- 8D 00 CA  2430          STA CHSUM   ;UPDATE CHEKSUM
C91B- 68        2440          PLA
C91C- 29 7F     2450          AND #$7F    ;CLEAR BIT 7
C91E- 60        2460          RTS
                2470 ;
                2480 ;*************************************************
                2490 ;***** OUTPUT MESSAGE *****
                2500 ;*************************************************
                2510 ;
C91F- B9 2E C9  2520 MESSY    LDA MESS,Y  ;LOAD CHARACTER
C922- C9 03     2530          CMP #$03
C924- F0 07     2540          BEQ MESEND  ;END OF TEXT ?
C926- 20 00 E0  2550          JSR PRCHA
C929- C8        2560          INY
C92A- 4C 1F C9  2570          JMP MESSY
C92D- 60        2580 MESEND   RTS
                2590 ;
                2600 ;**** MESSAGES ****
                2610 ;
C92E- 0D 0A 4F  2620 MESS     .BY $0D $0A 'OUT OF MEMORY' $0A $0D $03
C931- 55 54 20
C934- 4F 46 20
C937- 4D 45 4D
C93A- 4F 52 59
C93D- 0A 0D
C940- 4[illegible]  2630      .BY $0D $0A 'READ BCODE-2' $0A $0D $03
C943- [illegible]
C946- 2[illegible]
C949- 4F 44 45
C94C- 2D 32 0A
C94F- 0D 03
C951- 0D 0A 53  2640          .BY $0D $0A 'SAVE DATA' $0A $0D $03
C954- 41 56 45
C957- 20 44 41
C95A- 54 41 0A
C95D- 0D 03
C95F- 0D 0A 43  2650          .BY $0D $0A 'CHECKSUM ERROR' $0A $0D $03
C962- 48 45 43
C965- 4B 53 55
C968- 4D 20 45
C96B- 52 52 4F
C96E- 52 0A 0D
C971- 03
C972- 0D 0A 44  2660          .BY $0D $0A 'DATA RECEIVED' $0A $0D $03
C975- 41 54 41
C978- 20 52 45
C97B- 43 45 49
C97E- 56 45 44
C981- 0A 0D 03
                2670 ;
                2680 ;*************************************************
                2690 ;***** SAVE RECEIVED DATA ****
                2700 ;*************************************************
                2710 ;
C984- 20 44 E1  2720 SAVERDATA JSR KBIT     ;INIT. KEYBORD
C987- 20 87 EC  2730          JSR PIEPER
C98A- A0 23     2740          LDY #$23
C98C- 20 1F C9  2750          JSR MESSY   ;PRINT 'SAVE DATA'
C98F- 58        2760          CLI  ;INTERRUPT ENABLE
                2770 ;
                2780 ;**** SET POINTERS TO DATA ****
                2790 ;
C990- 38        2800          SEC  ;CALCULATE LENGTH OF FILE
C991- AD 75 C8  2810          LDA STAIND+$02      ;END OF DATA HIGH
C994- ED 0D CA  2820          SBC RAMSTARTH
C997- 8D 0F CA  2830          STA LENGTHH  ;DATA LENGTHH VALUE
C99A- AD 74 C8  2840          LDA STAIND+$01      ;END OF DATA LOW
C99D- ED 0C CA  2850          SBC RAMSTARTL
C9A0- 8D 0E CA  2860          STA LENGTHL  ;DATA LENTHL VALUE
C9A3- A9 CA     2870          LDA #H,RAMSTARTH     ;SET FILE DATA POINTER
C9A5- A0 0C     2880          LDY #L,RAMSTARTL
                2890 ;
                2900 ;**** WRITE DATA ****
                2910 ;
C9A7- AE 0A CA  2920          LDX SAVEX   ;GET FILE NR
C9AA- 20 27 B0  2930          JSR WRITE1  ;WRITE DATA TO FILE
C9AD- B0 03     2940          BCS BUERC   ;ERROR
                2950 ;
C9AF- 4C 4B B0  2960 EOWRITE  JMP CLOSE1
                2970 ;
C9B2- AC 0B CA  2980 BUERC    LDY SAVEY   ;RESTORE FILE-NAME POINTER
C9B5- AD 09 CA  2990          LDA SAVEACCU
C9B8- 20 B7 B0  3000          JSR ERMES   ;WRITE ERROR
C9BB- 4C 4B B0  3010          JMP CLOSE1
                3020 ;
C9BE- 4C B7 B0  3030 BUER     JMP ERMES   ;CREATE ERROR
                3040 ;
                3050          .EN
                0010 ;
                0020 ;*************************************************
                0030 ;**********          BCSUBR           **********
                0040 ;**********                           **********
                0050 ;**********   BASICODE-2 SUBROUTINES  **********
                0060 ;**********   DATE 11-04-86           **********
                0070 ;**********                           **********
                0080 ;**********     P.LASKER              **********
                0090 ;**********     DRAKENSTEYN 291       **********
                0100 ;**********     7640 TR ALMELO        **********
                0110 ;**********     05490-72178           **********
                0120 ;**********                           **********
                0130 ;*************************************************
                0140 ;
                0150 ;
                0160          .OS
                0170          .CE
                0180          .BA $C800
                0190          .MC $C800
                0200 ;
                0210 ;**** MONITOR SUBROUTINES****
                0220 ;
                0230 GETASKEY  .DE $E00C   ;GET A KEY
                0240 PRTEXTSTR .DE $E00F   ;PRINT A TEXTSTRING, LAST CHAR. =$00
                0250 GETEVTAS  .DE $E012   ;GET A KEY IF THERE IS ONE, OTHERWISE A=$00
                0260 ;
                0270 ;
                0280 ;**** MONITOR VARIABLES ****
                0290 ;
                0300 DEVMODOUT .DE $CE12   ;OUTPUT DEVICE MODE BITS
                0310 ;
```

```
                0320 ;
                0330 ;**************************************************
                0340 ;**** JUMP TABEL ****
                0350 ;**************************************************
                0360 ;
C800- 4C 0D C8  0370 CURSORP    JMP CURSORPOS       ;PLACE CURSOR AT NEW POSITION
C803- 4C 15 C8  0380 GETK       JMP GETKEY  ;GET A KEY
C806- 4C 1C C8  0390 WAITK      JMP WAITKEY ;WAIT FOR A KEY
C809- 4C 23 C8  0400 BUZZ       JMP BUZZER  ;BUZZ
                0410 ;
                0420 ;
C80C-           0430 KEY        .DS 1       ;SAVE PRESSED KEY
                0440 ;
                0450 ;**************************************************
                0460 ;**** BASICODE-2 LINE 100 ****
                0470 ;**************************************************
                0480 ;
C80D- 20 0F E0  0490 CURSORPOS  JSR PRTEXTSTR       ;PRINT TEXT-STRING
C810- 14        0500            .BY $14     ;DIRECT CURSOR ADDRESSING MODE
C811-           0510            .DS 1       ;HORZONTAL POSITION
C812-           0520            .DS 1       ;VERTICAL POSITION
C813- 00        0530            .BY $00     ;END OF TEXT-STRING
C814- 60        0540            RTS
                0550 ;
                0560 ;**************************************************
                0570 ;**** BASICODE-2 LINE 200 ****
                0580 ;**************************************************
                0590 ;
C815- 20 12 E0  0600 GETKEY     JSR GETEVTAS        ;GET KEY
C818- 8D 0C C8  0610            STA KEY
C81B- 60        0620            RTS
                0630 ;
                0640 ;**************************************************
                0650 ;**** BASICODE-2 LINE 210 ****
                0660 ;**************************************************
                0670 ;
C81C- 20 0C E0  0680 WAITKEY    JSR GETASKEY        ;WAIT FOR KEY
C81F- 8D 0C C8  0690            STA KEY
C822- 60        0700            RTS
                0710 ;
                0720 ;**************************************************
                0730 ;**** BASICODE-2 LINE 250 ****
                0740 ;**************************************************
                0750 ;
C823- 20 0F E0  0760 BUZZER     JSR PRTEXTSTR       ;PRINT TEXT-STRING
C826- 07        0770            .BY $07     ;BUZZER MODE
C827- 00        0780            .BY $00     ;END OF TEXT-STRING
C828- 60        0790            RTS
                0800 ;
                0810            .EN
```

```
 1 REM ****************************************
 2 REM **********      BCSTRT        **********
 3 REM **********  BASICODE-2 ROUTINES **********
 4 REM **********  DATE 18-04-86       **********
 5 REM **********  P. LASKER           **********
 6 REM **********  DRAKENSTEYN 291     **********
 7 REM **********  7648 TR ALMELO      **********
 8 REM **********  05490-72178         **********
 9 REM ****************************************
10 GOTO 1000
20 GOTO1010
100 HO=0:VE=0:PRINTCHR$(12);:RETURN
110 IF HO)79 OR VE)23 THEN RETURN
111 IF HO=11 THEN HO=HO+1:HF=1:REM MONITOR ADDRESSING MODE CODE
112 IF VE=11 THEN VE=VE+1:VF=1:REM MONITOR ADDRESSING MODE CODE
113 POKE51217,HO+1:REM X-VALUE TO $C811
114 POKE51218,VE+1:REM Y-VALUE TO $C812
115 SYS51200:REM GOSUB $C800 = PLACE CURSOR AT NEW POSITION
116 IF VF=1 THEN VE=VE-1:VF=0:PRINTCHR$(11);
117 IF HF=1 THEN HO=HO-1:HF=0:PRINTCHR$(8);
119 RETURN
120 HO=PEEK(52745):VE=PEEK(52746):RETURN:REM X=$CE09,Y=$CE0A
200 SYS51203:REM GOSUB $C803 = GET KEY
202 O=PEEK(51212):REM KEY=$C80C
204 IN$=CHR$(O):IF O=0THEN IN$=""
205 RETURN
210 SYS 51206:REM GOSUB $C806 = WAIT FOR KEY
212 O=PEEK(51212):REM KEY=$C80C
214 IN$=CHR$(O)
215 RETURN
250 SYS51209:REM GOSUB $C809 = BUZZER
260 RV=RND(1):RETURN
270 FR=FRE(0):RETURN
300 IF SR<.01AND SR>-.01 THEN SR=0
301 IF SGN(SR)=-1 THEN SR$=STR$(SR):RETURN
302 SR$=MID$(STR$(SR),2):RETURN
310 OS=ABS(SR)+.5*10^-CN:OI=INT(OS):OD=OS-OI+1
311 SR$=""
312 IF OS>=1E9 THEN 321
313 IF CN=0 THEN OD$="":GOTO317
314 IF OD=1 THEN OD$=".":GOTO316
315 OD$=MID$(STR$(OD),3,CN+1)
316 IF LEN(OD$)<CN+1 THENOD$=OD$+"0":GOTO316
317 SR$=MID$(STR$(OI),2)+OD$
318 IF SR<0 AND VAL(SR$)<>0 THEN SR$="-"+SR$
319 IF LEN(SR$)<CT THEN SR$=" "+SR$:GOTO 319
320 IF LEN(SR$)>CT THEN SR$=""
321 IF LEN(SR$)<CT THEN SR$=SR$+"*":GOTO321
322 RETURN
350 POKE52754,64:REM $CE12 = ONLY PRINTER ON
352 PRINTSR$;
355 RETURN
360 POKE52754,64:REM $CE12 = ONLY PRINTER ON
362 PRINT
363 POKE52754,128:REM $CE12 = PRINTER OFF
367 RETURN
```

```
**********************************
*                                *
*        ================        *
*        |  |  |  |  |  |        *
*        |  |  |  |  |  |        *
*        |  |  |  |  |  |        *
*        |  |  |  |  |  |        *
*        |  |  |  |  |  |        *
*        |  |  |  |  |  |        *
*        |  |  |  |  |  |        *
*        ================        *
*                                *
*             WANTED             *
*                                *
*          DEAD OR ALIVE!        *
*                                *
*      6809  Cross-Assembler     *
*      to run under DOS65 v2     *
*                                *
*    To be used with a planned   *
*      6809 second processor     *
*                                *
*             REWARD             *
*                                *
*        Hardware design         *
*      when and if completed     *
*                                *
**********************************
```

```
10;TRACK 0 FOR NON BOOTABLE DISK'S
20;FOR EC65 OR DOS-JUNIOR
30;COEN BOLTJES
40;
50NMIVEC=$E7CB
60INIDSK=$F464          ;HEAD UP
70AHOLD =$2363
80;
90*=$2200
100;
110BOOTUP  JSR STROUT    ;PRINT NEXT
120        JSR INIDSK
130        JMP (NMIVEC)
140;
150STROUT  LDX #$00
160STROUU  LDA TEXT,X
170        BEQ STROEX    ;=>END
180        STA AHOLD
190        JSR $F000     ;PRINT
200        INX
210        JMP STROUU
220STROEX  RTS
230;
240TEXT    .BYTE $0A,$0D,$0A,$0A
250        .BYTE '**********************'
260        .BYTE $0A,$0D
270        .BYTE 'Insert Systemdisk to'
280        .BYTE 'Boot up',$0A,$0D,$07
290        .BYTE $00
```