

# BANKSWITCHING



NUTTIG GEBRUIK VAN DE

GEHEUGENBANKEN IN DE

P 2000 T MET 64 K

GEHEUGENUITBREIDING

CHARLES VAN DER LINDEN

Sinds februari 1985 houd ik mij o.a. bezig met te onderzoeken hoe de geheugenbanken, in de P-2000 met 64 K uitbreiding, gebruikt kunnen worden voor gegevensopslag.

In eerste instantie werd de uitbreiding door MINIWARE verkocht zonder enige toelichting, pas later verscheen een Aanvullende Gebruiksaanwijzing.

In de praktijk blijkt het dat de meesten nog weinig plezier hiervan hebben door gebrek aan kennis van de beginselen van machinetaal.

Pas toen ik mij zelf met machinetaal ging bezighouden, werd alles langzamerhand veel duidelijker en was ik in staat routines te maken om succesvol met de banken om te gaan.

Het is mij nu mogelijk om van alles in de banken weg te bergen, naar cassette te schrijven, terug in te lezen en uit de banken weer op te vragen.

Zo kunnen de banken een voor een van cassette worden gevuld met meerdere Basic-programma's, arrays, strings en video-beelden.

Eenmaal ingeladen zijn ze in een fractie van een seconde binnen handbereik.

Om het onderwerp Bankswitching aan iedereen duidelijk te kunnen maken, zullen we eerst enige begrippen die hierbij belangrijk zijn bespreken.

Indien men namelijk geen kennis van een bepaald begrip heeft, zijn alle zaken die in het verlengde ervan liggen bij voorbaat onbegrijpelijk.

Om zelf, als programmeur, gebruik te kunnen maken van BANKSWITCHING, met het geheugen te kunnen spelen en de geheugenbanken te vullen, zijn de volgende zaken van belang :

- 1e. Begrip van de elementaire principes van een computer.
- 2e. Vertrouwdheid met de geheugenindeling van de P-2000.
- 3e. Enig idee over de wijze van opslepen van Basic-regels, variabelen, arrays en strings in het geheugen.
- 4e. Kennis betreffende de werking van het beeldscherm en het Video-geheugen.
- 5e. Op de hoogte van de beginselen van machinetaal.
- 6e. Notie hebben van enige basis-machinetaalroutines.

Waar het om ging was manieren te vinden en methodes te ontwikkelen die ervoor zorgen dat een bepaald blok gegevens naar en uit een bank kan worden getransporteerd.

Om te beginnen kijken we naar de relatie van de computer met cijfers.

Daarna de geheugenindeling :

Om Basic-programma's, arrays, strings of video-beelden naar een bank te verplaatsen, zal eerst geweten moeten worden waar die zich bevinden.

Vervolgens het een en ander over de Z-80 microprocessor.

De Blockmove-instruktie zorgt voor het kopieeren van een reeks geheugenplaatsen in een ander deel van het geheugen.

Met behulp van Cassette-routines is men in staat aaneengesloten stukken geheugen naar cassette te schrijven en later weer in een bepaald geheugengebied in te lezen.

De computer kent slechts twee getallen n.l. 0 en 1.  
 Nul voor geen stroom, is er wel stroom dan betekent dat "1".  
 Op deze wijze worden cijfers omgezet in elektrische stroomstootjes.  
 Bij nul wordt de stroom gestopt, bij 1 laat men de stroom doorgaan.  
 Volgens het binair stelsel kunnen nu getallen geformeerd worden.  
 Het tweetallige stelsel gebruikt alleen de cijfers 0 en 1.

Om een reeks binaire getallen te maken, slaat men alle andere decimale getallen, die dan dus niet bestaan, doodgewoon over :

bin.	0	1	10	11	100	101	110	111	1000	1001	1010	1011	1100	1101	1110	1111	10000
dec.	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
hex.	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10

Hexadecimaal is volgens het zestientallig stelsel.

De getallen tien tot vijftien worden voorgesteld door de letters A-F.

tot de macht	0	1	2	3	4	5	6	7	8
decimaal	10	1	10	100	1000	10000	100000	1000000	10000000
binair	2	1	2	4	8	16	32	64	128
hexadec.	16	1	16	256	4096	65536	1048576	enz.	256

Een byte bestaat uit acht bits, het grootste getal onder te brengen in een byte is 255, n.l. binair 11111111.

De Z-80 microprocessor binnendoor heeft een bereik van 64 Kilobytes. D.w.z. er zijn  $64 \times 1024 = 65536$  ( 0-65535 ) geheugenplaatsen mogelijk.

65535 komt overeen met het 16 bits-getal 1111111111111111.

Elke plaats ( byte ) kan een getal van max. 255 bevatten.  
 Wanneer we een getal groter dan 255 willen vastleggen, dan zijn er meer plaatsen nodig.

Een getal A%, b.v. 25927 wordt opgeslagen in twee bytes.  
 De tweede ( hoge ) byte wordt gevuld met het getal dat ontstaat door met 256 te delen en naar beneden af te ronden ( integer-deling ).  
 De lage ( eerste ) byte gaat het restant van de deling bevatten ( modulo ).

HB = A%  $\sim$  256      of    25927  $\sim$  256      = 101      of    HB = INT ( 25927/256 )      = 101  
 LB = A% MOD 256      of    25927 MOD 256 = 71      LB = 25927 - ( 256 \* H ) = 71

Werken met hexadecimale getallen maakt alles eenvoudiger :  
 Het getal 25927 is hexadecimaal 6547 n.l. HEX\$ ( 25927 ) = &H 6547

&H 65 is de hoge byte, &H 47 de lage byte :  
 LB = HEX\$ ( 71 ) = &H 47  
 HB = HEX\$ ( 101 ) = &H 65

Een hexadecimale getal wordt dus omgekeerd gesplitst :  
 &H 625C wordt &H 5C en &H 62.

Een groot getal weer terug vormen uit twee bytes gaat als volgt :  
 A = inhoud lage byte + 256 x inhoud hoge byte      of  
 A = PEEK ( LB ) + 256 \* PEEK ( HB )

Bijvoorbeeld :  
 A = PEEK ( &H 6405 ) + 256 \* PEEK ( &H 6406 )

## G E H E U G E N I N D E L I N G

T.009

&H dec. I-----I  
 I-----I MONITOR in ROM ( read only memory = uit dit geheugen  
 I-0000 - 0 -I-- kan alleen worden gelezen en niets worden veranderd )  
 I-----I Routines voor Cassette, Disk en beeldscherm ( Video )  
 I-1000 - 4096-I-- BASIC-insteekmodule in sleuf 1. De BASIC-INTERPRETER  
 I-----I maakt het mogelijk om te programmeren in BASIC.  
 I-----I Het vertaalt de codewoorden van de Basic-instructies  
 I-----I in een aantal handelingen en voert ze uit. ( in ROM ).  
 I-5000 - 20480-I-- Het VIDEOGEHEUGEN. Linker- en rechter beeldscherm.  
 I-----I 24 regels en 80 kolommen, links en rechts 40 kolommen.  
 I-----I Met 40/80 karakterkaart alle kolommen op het scherm.  
 I-----I Het tweede Videogeheugen vanaf &H 5800 is niet aanwezig  
 I-----I op de P-2000 T, wel op het M-model.  
 I-6000 - 24576-I-- In gebruik door de Monitor voor registerruimten  
 I-----I en bufferblokken.  
 I-6200 - 25088-I-- Administratie- en bufferruimte voor  
 I-----I Philips Cassetter-Basic.  
 I-6547 - 25927-I-- Basic-ruimte : begin te lezen uit &H 625C en &H 625D.  
 I-----I Gevuld met regelns., regelaanwijzers, BASIC-tokens enz.  
 I-----I Variabelenruimte : begin te lezen uit &H 6405 en &H 6406.  
 I-----I integer-, string variabelen, enkels en dubbele precisie.  
 I-----I Array-ruimte : begin te lezen uit &H 6407 en &H 6408.  
 I-----I Ruimte voor arrays reserveren d.m.v. DIMmen.  
 I-----I Vrije ruimte : begin te lezen uit &H 6409 en &H 640A.  
 I-----I Ruimte afhankelijk van grootte andere ruimten.  
 I-----I STACK = stapel : einde te lezen uit &H 6258 en &H 6259.  
 I-----I Bewaring van o.a. terugspringadressen van GOSUB, FOR-NEXT.  
 I-----I String-ruimte : normaal 50 bytes, in te stellen met b.v.  
 I-----I CLEAR 4000  
 I-----I Gereserveerd : begin -1 te lezen uit &H 63B8 en &H 63B9.  
 I-----I in te stellen met b.v.: CLEAR 50, &H DFFF.  
 I-FFFF - 65535-I-- Einde geheugen

## G E H E U G E N B A N K E N

&H	dec.	I-----I	Einde geheugen :
- 1000 - 4096 -		I-----I	
		I-----I	16 K = &H 9FFF
		I-----I	32 K = &H DFFF
		I-----I	48 K = &H FFFF
- 5000 - 20480 -		I-----I	
		I-----I	te lezen in &H 605C :
- 6000 - 24576 -		I-----I	
		I-----I	1 = 16 K
- 6547 - 25927 -		I-----I	2 = 32 K
		I-----I	3 = 48 K
		I-----I	
		I-----I	
		I-----I	
- E0000 - 57344 ---		I-----I-----I-----I-----I-----I-----I-----I-----I-----I-----I	
CLEAR 50, &H DFFF		I-----I-----I-----I-----I-----I-----I-----I-----I-----I-----I	
8192 bytes		I-----I-----I-----I-----I-----I-----I-----I-----I-----I-----I	
gereserveerd		I-----I-----I-----I-----I-----I-----I-----I-----I-----I-----I	
		I-----I-----I-----I-----I-----I-----I-----I-----I-----I-----I	
		I-----I-----I-----I-----I-----I-----I-----I-----I-----I-----I	
- FFFF - 65535 ---		I-----I-----I-----I-----I-----I-----I-----I-----I-----I-----I	
banknummer --	0      1      2      3      4      5		
B A N K S W I T C H I N G		Charles van der Linden & Zn.	
11-11-86		Broederhof 11, 5504 JC Veldhoven.	

De Z-80 processor werkt met geheugenplaatsen die registers worden genoemd.  
Van belang zijn de registers:

A ( accumulator ), F ( vlagregister ), B, C, D, E, H, L ( secundaire registers )  
SP ( stackpointer ), PC ( program counter ), IX, IY ( index-registers )

De secundaire registers worden ook in paren gebruikt, om getallen groter dan 255 te kunnen opslaan : B-C, D-E, H-L.

De processor voert operaties uit aan de hand van de Z-80 Code.  
Een machinetaalprogramma is een reeks van instructies in binaire getallen.  
In de praktijk wordt gewerkt met hexadecimale getallen.

Het programmeren gaat gemakkelijker door te werken met Assembleertaal.  
Deze taal hanteert afkortingen ( Mnemonics ) van de uiteindelijke operatie die van de Z-80 verwacht wordt, b.v.

LD A,B	laad in register A het getal uit register B
LD HL,5000	laad in registerpaar HL het getal 5000, d.w.z. 00 in L, 50 in H
LD C,(HL)	laad in C de inhoud van de plaats die het getal in HL aanwijst
LD (E000),HL	laad in geheugenplaats &H E000 + E001 het getal uit HL
LD A,(650D)	laad in register A het getal uit geheugenplaats &H 650D
LD (HL),L	laad in de geheugenplaats die HL aanwijst het getal uit register L
INC DE	increase = verhoog het getal in register DE met 1
ADD HL,BC	addition = tel de inhoud van BC op bij de inhoud van HL
SBC HL,DE	subtract = trek de inhoud van DE af van de inhoud van HL
JP 9000	jump = spring naar de instructies vanaf plaats 9000
CALL 1059	call = roep de subroutine vanaf 1059 aan en keer terug
PUSH BC	zet de inhoud van BC even weg op de STACK
POP DE	vul DE weer met de inhoud die op de stapel is gezet
CP 01	compare = vergelijk het getal in register A met het getal 01
RET	return = terug ( naar BASIC )

Deze instructies worden d.m.v. het ASSEMBLER-programma vertaald in hexadecimale getallen. Deze getallen worden uiteraard aan de processor binair aangeboden.  
Bijvoorbeeld :

LD BC,0800	01 00 08	OUT 94	D3 94
LD (DE),A	12	SBC HL,SP	ED 72
DEC H	25	LD IX,7000	DD 21 00 70

Omgekeerd, vertalen van Z-80 Operation Code naar Assembleertaal heet Disassemble.

Blockmove is het d.m.v. een machinetaal-routine verplaatsen van een blok gegevens vanaf de ene geheugenplaats naar een andere.  
In feite wordt er niets verplaatst, maar gekopieerd, op de oude plaats zijn na afloop nog alle gegevens te vinden.  
Het HL register moet worden gevuld met het startadres.  
In register DE het bestemmingsadres.  
Register BC dient het aantal te "verplaatsen" bytes te bevatten.  
Deze waarden kunnen worden opgeslagen in de ongebruikte ruimte vanaf &H 6070, om later via de routine te worden uitgelezen.

LD HL , ( 6071 )	2A 71 60	beginadres in HL uit plaatsen &H 6071 en &H 6072
LD DE , ( 6073 )	ED 5B 73 60	bestemming in DE uit plaatsen &H 6073 en &H 6074
LD BC , ( 6075 )	ED 4B 75 60	aantal bytes in BC uit &H 6075 en &H 6076
LDIR	ED B0	blockmove
RET	C9	terug naar Basic

Indien de P-2000 over het volledige geheugen beschikt, zijn 64 K adresseerbaar.  
 Echter diverse zaken eisen geheugenruimte op :

MONITOR	4 K
BASIC-INTERPRETER	16 K
VIDEOGEHEUGEN gereserveerd	4 K
Administratie-en bufferruimten	1351 bytes
Stringruimte	50
Registers	16
TOTAAL	25993 bytes

OVER aan vrij werkgeheugen 65535 - 25993 = 39542 bytes.

Indien men meer gegevens kwijt zou willen, moet het toevlucht gezocht worden tot Bankswitching.

De laatste 8 K van het geheugen zijn te verwisselen met een andere geheugenset. Het voordeel is dat men 8 K per bank meer opslagcapaciteit heeft.

### V I D E O G E H E U G E N

0	1	2	3	4	5	6	7
0123456789012345678901234567890123456789012345678901234567890123456789							
linker scherm	-						rechter scherm

Het videogeheugen, de geheugenplaatsen die corresponderen met het beeldscherm, bevindt zich van &H 5000-57FF ( 20480-22527 ).

Het geheugen is opgebouwd uit 24 regels en 80 kolommen.

&H 5000 : De linkerhelft, 40 kolommen, vormt het normale linker beeldscherm.

I-----I... : Het beeldscherm kan opgenomen worden door de inhoud van de geheugenplaatsen vast te leggen als array,  
 I VIDEO I : als strings, in een bank of weg te schrijven naar  
 I GEHEUGEN I : cassette.  
 I-----I... : Opslaan in een bank is mogelijk door telkens de eerste  
 I-----I -5800 : 40 plaatsen van het beeld via blockmove achter elkaar  
 I-----I -6000 : in een bank te kopiëren.  
 I-----I -6547 : Een scherm bestaat uit  $24 \times 40 = 960$  bytes,  
 I-----I -6547 : zodat in een bank 8 beelden ( $8 \times 960 = 7680$ ) kunnen worden opgeslagen.  
 I Basic I : Indien 22 regels i.p.v. 24 dan 9 beelden ( $\times 880$ ).  
 I Varia- I : De inhoud van een bank kan weer worden weggeschreven  
 I belen I : naar cassette en later weer terug van cassette naar  
 I-----I : een bank.  
 I Arrays I : Vanuit de bank kunnen de beelden weer achter elkaar  
 I-----I : op het scherm worden geprojecteerd.  
 I Vrij I : Met een 40/80 karakterkaart is het mogelijk met het  
 I-----I : volledige beeldscherm van 80 kolommen zichtbaar  
 I Stack I : te werken.  
 I-----I : Een dergelijk beeld kan in een klap naar een bank  
 I Strings I : worden gebracht. Deze beelden bedragen 1920 bytes,  
 I-----I-----I : zodat hiervan 4 beelden in een bank passen.  
 I-----I 1 I : Wanneer het normale geheugen en de banken  
 I-----I 2 I : volledig worden gevuld, is het mogelijk om  
 I-----I 3 I : 40 beelden op te slaan.  
 I-----I 4 I : Dit komt overeen met 15 A-4 pagina's van  
 I-----I 5 I : 64 regels,  $64 \times 80 = 5120$  bytes = 5 K.  
 I-----I 6 I : 15 pagina's  $\times 5120 = 76800$  bytes.  
 I-----I 7 I : 40 beelden  $\times 1920 = 76800$  bytes  
 I-----I 8 I : Op deze manier is het mogelijk grote hoeveel-  
 I evt.9 I : heden tekst, direct zichtbaar te maken  
 I-----I-----I : en achter elkaar uit te laten printen.

O bank 1 enz.

## ROUTINE PROGRAMMING

8,921

In dit kader is het ook mogelijk om diverse programma's achter elkaar in de Basicruimte te zetten.

Opvragen van een programma kan dan door de wijzer van het begin van de Basicruimte n.l. de plaatsen &H 625C en 625D naar het bewuste programma te laten wijzen.

Zo zijn er wel meer mogelijkheden te bedenken als je ervoor gaat zitten.

## A R R A Y

			byte	
TYPE	I	2	I	1      integer
NAAM	I	90	I	2      2
	I	65	I	3      A
	I			aantal bytes
LENGTE	I	205	I	4      + 2 * aantal
vanaf hier	I	0	I	5      dimensies +1
DIMENSIE	I	1	I	6
Aantal elementen	I	101	I	7      101 elementen
	I	0	I	8
Element 0	I	12	I	9      Waarde ZA%(0)
	I	23	I	10     = 5900
Element 1	I	45	I	11     Waarde ZA%(1)
	I	19	I	12     = 4909
Element 2	I	127	I	13     Waarde ZA%(2)
	I	33	I	14     = 8575

## Voorbeeld integer array ZA%(100)

T.010

## TYPE :

2 = integer array  
 3 = string-array  
 4 = enkele precisie  
 8 = dubbele precisie

In de arrayruimte wordt ruimte gereserveerd door DIMmen b.v. :  
 DIM ZA% (100)

- A! = VARPIR(AA%(0))  
 - de geheugenplaats van de eerste byte van element 0

Wegschrijven naar cassette :  
 CSAVE \* ZA% : onder naam ZA%  
 CSAVE \* ZA% @ "Array" :  
 onder naam "Array"

## S T R I N G A R R A Y

## Voorbeeld AB\$(250)

			byte	
TYPE	I	3	I	1      string
NAAM	I	65	I	2      A
	I	66	I	3      B
	I			aantal bytes
LENGTE	I	244	I	4      + 2 * aantal
	I	2	I	5      dimensies +1
DIMENSIE	I	1	I	6
Aantal elementen	I	251	I	7      251 elementen
	I	0	I	8
Element 0	I	2	I	9      Lengte van string 0
	I	254	I	10     Plaats van string 0 : inhoud lage byte + 256 *
	I	255	I	11     in de stringruimte : inhoud hoge ( tweede ) byte = 65534
Element 1	I	6	I	12     lengte 1
	I	248	I	13     Plaats in de stringruimte
	I	255	I	14     = 65528
Element 2	I	3	I	15     lengte 2
	I	245	I	16     plaats 2
	I	255	I	17     = 65525
Element 3	I	7	I	18     lengte 3
	I	238	I	19     plaats 3
	I	255	I	20     = 65518

## S T R I N G - V A R I A B E L E

## Voorbeeld A2\$ = "plaats"

			byte	
	I	1	I	Type : string
	I	2	I	A
	I	3	I	2
	I	4	I	lengte
	I	5	I	plaats in stringruimte
	I	6	I	stringruimte
	I	7	I	
	I	8	I	
	I	9	I	
	I	10	I	
	I	11	I	
	I	12	I	
	I	13	I	
	I	14	I	
	I	15	I	
	I	16	I	
	I	17	I	
	I	18	I	
	I	19	I	
	I	20	I	
	I	21	I	
	I	22	I	
	I	23	I	
	I	24	I	
	I	25	I	
	I	26	I	
	I	27	I	
	I	28	I	
	I	29	I	
	I	30	I	
	I	31	I	
	I	32	I	
	I	33	I	
	I	34	I	
	I	35	I	

B A N K S W I T C H I N G  
 11-11-86

Charles van der Linden & Zn.  
 Broederhof 11, 5504 JC Veldhoven.

I-----I &H  
 I I 6547 Een array is een reeks variabelen, met dezelfde variabele-naam.  
 I BASIC I Voordat een array aangemaakt wordt, moet in het geheugen een  
 I-----I ruimte worden gereserveerd worden groot genoeg om de gegevens  
 I VARIA- I en de elementen van het array te bevatten.  
 I BELEN I Dit gebeurt d.m.v. DIMmen, b.v. DIM A%(120)  
 I-----I ... De plaats in de array-ruimte van een element is op te vragen  
 I I : met de VARPTR, bijvoorbeeld : A! = VARPTR(A%(0)).  
 I ARRAY I : ... A! geeft hier de geheugenplaats van de eerste byte van  
 I RUIMTE I : : element 0, dus het begin van het blok gegevens.  
 I I : : De lengte is twee maal het aantal elementen, in dit geval  
 I-----I : : 121 x 2 = 242. In element 0 wordt deze lengte vastgelegd.  
 I VRIJ I : M.b.v. blockmove worden deze 242 bytes een klap in een bank  
 I STACK I : gebracht. Vanuit de bank kan het array naar cassette worden  
 I-----I : ..... geschreven en later weer terug van cassette naar  
 I STRING I : een bank.  
 I-----I-----I ... : Meerdere arrays kunnen zodoende opgeslagen worden  
 I I : : in afwachting van gebruik, terwijl er maar voor  
 I I : : een array ruimte gereserveerd hoeft te worden.  
 I I : Ook is het mogelijk de array-gegevens rechtstreeks  
 I I : uit de bank te halen, zodat de eigenlijke array-  
 I-----I-----I- ruimte vrij blijft.

O 1 enz. -- banknummer

## S T R I N G - A R R A Y S

I-----I &H  
 I I 6547 Een stringarray wordt als volgt opgeslagen :  
 I BASIC I In de arrayruimte wordt een ruimte gereserveerd ( DIM )  
 I-----I voor het aantal voorziene elementen.  
 I VARIA- I Elk element heeft drie bytes nodig .  
 I BELEN I Deze bytes heten de stringdescriptoren ( aanwijzers ).  
 I-----I .... A! = VARPTR(A\$(0)) wijst naar byte 1 van A\$(0).  
 I DIM I : Deze eerste byte geeft de lengte van de string aan.  
 I I : Byte 2 en 3 wijzen naar de plaats van de string in de  
 I ARRAY- I : : stringruimte. B! = PEEK (A!+1) + 256 \* PEEK (A!+2).  
 I RUIMTE I : Via blockmove kunnen de strings een voor een vanuit de  
 I-----I : : stringruimte in een bank worden gekopieerd.  
 I VRIJ I : Telkens wordt de stringdescriptor in de arrayruimte zodanig  
 I STACK I : gewijzigd dat deze de string in de bank gaat aanwijzen.  
 I-----I : Nadat alle strings zijn overgeheveld wordt  
 I STRING I : ....string.... het blok descriptoren uit de arrayruimte  
 I : nr. descriptoren : achter het stringarray in de bank geplaatst.  
 I RUIMTE I- 5..... : Voorin de bank wordt de totale lengte en  
 I I- 4..... : het aantal strings vastgelegd en alles is  
 I I- 3..... : klaar om naar cassette te schrijven.  
 I I- 2..... : Terug van cassette gaat vervolgens :  
 I I- 1..... : Van cassette in een bank laden, voor het  
 I-----I : : : : aantal strings een ruimte reserveren in de  
 I I lengte I : : : : arrayruimte, de stringdescriptoren via  
 I I aantal I : : : : blockmove van achter de strings in de bank  
 I I 1- I.1 : : : : naar de geDIMde arrayruimte brengen.  
 I I 2- I...2 : : : : De strings worden nu uit de bank gelezen en  
 I I 3- I....3 : : : zodoende kan de normale stringruimte  
 I I 4- I.....4 : : gespaard worden.  
 I I 5- I.....5 : Wanneer de string wordt verbeterd, komt  
 I enz. .... de veranderde string ( met hetzelfde element-  
 I I : : nummer ) in de stringruimte te zitten.  
 I I : : Door elke string apart weer uit te lezen en  
 I I : : stringdes- in een andere bank op te slaan, kunnen  
 I-----I-----I criptoren de verbeterde versies uit die bank weer  
 naar cassette worden geschreven.

B A N K S W I T C H I N G

11-11-86

Charles van der Linden & Zn.

Broederhof 11, 5504 JC Veldhoven.

In de P2000 INTERPRETER kan de schrijfroutine aangeroepen worden op adres &H 105C, de leesroutine op &H 1059.  
 De registers moeten als volgt worden gevuld :  
 In register A de beginletter van de naam ( in ASCII-code ) waaronder het programma op cassette moet worden opgeslagen.  
 In registerpaar HL het startadres.      In registerpaar DE het einde + 1.  
 Deze waarden kunnen naar behoeftte verschillen en moeten tijdelijk ergens worden opgeslagen om ze daar te laten uitlezen.  
 Een bruikbare ruimte bevindt zich op de plaatsen &H 6070 - &H 6086,  
 eigenlijk bestemd voor de DISK-ROUTINE.

De Cassette-routine komt er als volgt uit te zien :

```
LD A , ( 6070 )    3A 70 60      beginletter in register A uit &H 6070
LD HL, ( 6071 )    2A 71 60      beginadres in HL uit &H 6071 en &H 6072
LD DE, ( 6073 )    ED 5B 73 60    eind +1 in paar DE uit &H 6073 en &H 6074
CALL 1059            CD 59 10      leesroutine
RET                  C9              terug naar Basic
```

Om te schrijven wordt CALL 105C gebruikt : CD 5C 10.

```
10 DATA 3A,70,60,2A,71,60,ED,5B,73,60,
   CD,59,10,C9                         Vanaf plaats &H 6075 kan de
20 FOR I = 0 TO 13 : READ A$ :
   POKE &H 6075 + I , VAL ( "&H" + A$ )
30 NEXT : DEFUSR 1 = &H 6075               Om een adres in twee bytes op te
40 DEF FN B ( A ) = INT ( A / 256 ) :
   DEF FN A ( A ) = A - 256 * FN B ( A )      slaan wordt gebruik gemaakt van
50 PRINT CHR$ ( 12 ) "1 = lezen                nevenstaande functies.
   2 = schrijven " ; : INPUT A                 Keuze : lezen of schrijven ?
60 ON A GOTO 70 , 80                         Lezen.
70 POKE &H 6080 , &H 59 : GOTO 90               Schrijven.
80 POKE &H 6080 , &H 5C                         Op plaats &H 6070 de naam
90 PRINT "Naam" ; : INPUT NS                 van het programma.
100 POKE &H 6070 , ASC ( NS )
110 PRINT "Startadres " ; : INPUT AS
120 PRINT "Eindadres " ; : INPUT BS
130 POKE &H 6071 , FN A ( A ) :
   POKE &H 6072 , FN B ( A )
140 POKE &H 6073 , FN A ( B ) :
   POKE &H 6074 , FN B ( B )
150 AS = USR 1 ( NS )                         Alvorens de cassette-routine
                                             aan te roepen moeten de
                                             uitleesadressen worden
                                             opgevuld.
```

Aanroep van de cassette-routine.

Op deze wijze kan elk willekeurig deel van het geheugen naar cassette worden geschreven of van cassette in een willekeurig stuk geheugen worden geladen.

#### ACHTERWAARDS WISSEN : ( naar P2C2 6.5.1 )

Het kan voorkomen dat een cassette " rommel " bevat en daardoor onbruikbaar lijkt. De cassette kan gewist worden terwijl de recorder achteruit loopt.

FOR I = 1 TO 40000 : OUT 16 , 70 : NEXT

Gewist wordt van einde tot begin band. Het begin van de band wordt niet gedetecteerd, daarom vlug uitnemen als de cassette stopt.

#### WISSEN VAN PROGRAMMA'S ( naar P2C2 6.5.2. )

Het kan zijn dat er halverwege een cassette de rest van de programma's verwijderd moeten worden. Dit kan als volgt :

1. CLOAD het programma dat als laatste moet blijven en CSAVE het weer.
2. Typ in : FOR I = 0 TO 1000 : OUT 16 , 74 : NEXT

Het bandje loopt nu ca. 3 seconden en wist daarmee de rest.

Het besturingssysteem ( OPERATING SYSTEM ) van de mini-cassettorecorder.

Alle functies beginnen op startadres &H 0018 in de MONITOR.

Via slechts vier bytes machinetaal zijn ze allen bereikbaar voor de programmeur.

LD A,(HL)	7E	haal het functienummer op
RST 18	DF	aanroep cassette-besturing
LD (HL),A	77	geef foutcode terug
RET	C9	terug naar Basic

De machinetaal kan in het geheugen worden gezet b.v. op de plaatsen die bestemd zijn voor de DISK-routine n.l. van &H 6070 - &H 6086. Stel vanaf &H 6083.

Het Basicprogramma luidt dan :

```
10 DATA 7E,DF,77,C9
20 FOR I = 0 TO 3 : READ A$ : POKE &H 6083 + I , VAL ( " &H " + A$ ) : NEXT
30 DEFUSR = &H 6083
```

Na RUNnen van dit programma kunnen de cassettefuncties worden aangeroepen via :  
PRINT USR ( getal ). Het getal moet corresponderen met de gevraagde opdracht.

#### Opdracht Betekenis

- 0 Initialiseer cassettereorder ( dit gebeurt steeds bij het aanzetten van de P-2000 ).
- 1 Spoel terug naar het begin van de band.
- 2 Spoel N blokken vooruit. Het getal N moet eerst op plaats &H 604F worden gePOKEt.  
B.v. POKE &H604F , 10 : PRINT USR ( 2 ) ( spoel 10 blokken vooruit ).
- 3 Spoel N blokken terug .  
B.v. POKE &H604F , 19 : PRINT USR ( 3 ) ( spoel 19 blokken terug ).
- 4 Schrijf " einde band ". De P-2000 weet dat vanaf dit punt de rest van de cassette leeg is.
- 5 Opnemen : Een stuk geheugen naar de cassette schrijven.  
Het startadres moet op de plaatsen &H 6030 en &H 6031 worden gePOKEt.  
De lengte in bytes op de adressen &H 6032 en &H 6033, alsmede op &H 6034 en &H 6035.  
De eerste 8 letters van de naam vanaf plaats &H 6036.  
De tweede 8 tekens van &H 6047 - &H 604E.  
Voorbeeld : Een deel geheugen vanaf 34968 ter lengte van 2634 bytes naar cassette onder naam "CASSETTE-ROUTINE"  
Eerst getallen naar hexadecimaal vertalen :  
HEX \$ ( 34968 ) = &H 8898 ;    HEX \$ ( 2634 ) = &H 0A4A  
Daarna de TAPE-HEADER adressen vullen :  
POKE &H 6030 , &H 98 : POKE &H 6031 , &H BB  
POKE &H 6032 , &H 4A : POKE &H 6033 , &H OA  
POKE &H 6034 , &H 4A : POKE &H 6035 , &H OA  
A\$ = "CASSETTE-ROUTINE"  
FOR I = 1 TO 8 : POKE &H 6036 + I - 1 , ASC ( MID\$ ( A\$ , I )) : NEXT  
FOR I = 1 TO 8 : POKE &H 6047 + I - 1 , ASC ( MID\$ ( A\$ , I+8 )) : NEXT  
De plaatsen &H 603E - &H 6040 duiden de soort file aan. BAS = BASIC  
Op plaats &H 6041 kan het type van de file worden gezet.  
B.v. B = Basic. Na dit alles PRINT USR ( 5 ) en er wordt geschreven.
- 6 Lees een stuk geheugen van de band. Als bij 5.  
De plaatsen &H 6030 en &H 6031 worden gevuld met het laadadres waarnaartoe de gegevens van cassette moeten worden getransporteerd.
- 7 Status van de cassettereorder. Werkt niet met bewuste machinetaal.

De TAPEHEADER is het blok van 32 bytes die elk programma op cassette voorafgaat.  
Bij schrijven worden de adressen &H 6030-&H 604F uitgelezen naar cassette.  
Bij laden worden deze adressen vanaf cassette gevuld.

## B A N K S W I T C H I N G

0.025

Indien de P-2000 over het volledige geheugen beschikt, zijn 64 K adresseerbaar.  
 Echter diverse zaken eisen geheugenruimte op:  
 MONITOR 4 K, BASIC-INTERPRETER 16 K, VIDEOGEHEUGEN 4 K, administratie- en  
 bufferruimten 1351 bytes, Stringruimte 50, Registers 16, Totaal 25993 bytes.  
 Over aan werkgeheugen 65535 - 25993 = 39542 bytes.  
 Indien men meer gegevens kwijt zou willen, kan gebruik worden gemaakt van  
 BANKSWITCHING.

I-----I &H 0000	
I MONITOR	De P-2000 I / 102 en de P-2000 met 64 K uitbreiding
I-----I 1000	beschikken over 5 extra geheugenbanken van elk 8 K
I BASIC- I	voor gegevensopslag.
I INTER- I	
I PRETER I	Deze banken kunnen verwisseld worden met de laatste 8 K van
I-----I 5000	het normale geheugen ( bank 0 ).
I VIDEO I	Hierin bevinden zich de adressen &H E000 - &H FFFF
I-----I 6000	( 57344-65535 ). Met de instructie OUT &H 94,N wordt bank 0
I-----I 6547	vervangen door bank N ( 1-5 ).
I I BASIC	
I I VARIAB.	Aangezien de STACK zich normaal gesproken ook achterin het
I I ARRAY	geheugen bevindt, dient eerst de laatste 8 K
I I URIJ	gereserveerd te worden d.m.v. CLEAR 50 , &H DFFF.
I I STACK	In dit geval blijven 32 K geheugen over en 6 banken.
I I STRINGS	
I-----I-----I-----I-----I-----I-----I-----I-----I &H E000 57344	
I I I I I I I I	
I O I 1 I 2 I 3 I 4 I 5 I	banken
I I I I I I I I	
I-----I-----I-----I-----I-----I-----I-----I &H FFFF 65535	

Bankswitching geeft een extra dimensie aan de P-2000.

Eenmaal in de banken geladen vanaf cassette, heeft men direct toegang tot diverse Basic-programma's, opgeslagen strings, arrays en Video-beelden. Men kan over ca.30 pagina's A-4 aan tekst beschikken en achter elkaar uitprinten. Het is mogelijk om de inhoud van de Basic-ruimte ( het Basic-programma ) in een bank op te slaan en op afroep weer naar de Basic-ruimte te verplaatsen. Een array kan uit de Array-ruimte in een bank gekopieerd worden en vice versa. De inhoud van het Video-geheugen ( het beeldscherm ) kan naar een bank worden getransporteerd, meerdere beelden kunnen achter elkaar in de bank komen te staan en later weer in een klap op het beeldscherm gedeponeerd. Om in een bank gegevens te schrijven, kan in BASIC gebruik worden gemaakt van POKE en PEEK. B.v. POKE bankadres, getal : POKE &H E000, 12 POKE bankadres, PEEK ( geheugenplaats ) : POKE &H F000, PEEK ( &H 6547 ) Dit is : Zet in geheugenplaats &H F000 het getal dat zich nu in geheugenplaats &H 6547 bevindt.

Een reeks van C gegevens kan via FOR-NEXT van A naar B worden overgebracht :  
 FOR I = 0 TO C-1 : POKE B + I, PEEK ( A + I ) : NEXT I  
 Voor een grote reeks getallen gaat dit lang duren, vandaar dat de toevlucht moet worden gezocht tot Machinetaal, dat veel sneller is.  
 De Z-80 microprocessor kent de Blockmove-instructie, deze zorgt ervoor dat een blok gegevens van de ene plaats van het geheugen in een minimum van tijd in een ander deel van het geheugen wordt gekopieerd.  
 Om dus een aaneengesloten aantal gegevens uit het geheugen in een bank te zetten moet alleen geweten worden waar het beginpunt zich in het geheugen bevindt. Via een Cassette-routine kan de inhoud van een bank naar cassette worden geschreven. Later kunnen de gegevens weer van cassette in een bank worden gelezen en vandaaruit b.v. weer naar een ander deel van het geheugen. Op die manier is het mogelijk zes programma's of blokken gegevens, die niet groter dan 8 K zijn een voor een van cassette in de zes banken te laden. Ook kan omgekeerd de inhoud van de zes banken achter elkaar, zonder terugspoelen, naar cassette worden geschreven.

B A N K S W I T C H I N G  
 11-11-86

Charles van der Linden & Zn.  
 Broederhof 11 - 5504 JC Veldhoven

## B A S I C P R O G R A M M A ' S in een bank laden. 0.026

I -----I &H  
I I 6547 Een Basic-programma begint op de plaats die aangeduid wordt  
I Basic- I in de adressen &H 625C en &H 625D, normaal staan hier de  
I pro- I waarden &H 47 en &H 65, wijzend naar &H 6547.  
I gram- I Het einde van een programma is te vinden in de geheugen-  
I ma I plaatsen &H 6405 en &H 6406, volgens de formule :  
I I PEEK ( &H 6405 ) + 256 \* PEEK ( &H 6406 ).  
I I De lengte van een programma is dus te berekenen door  
I -----I- van het eindadres het beginadres af te trekken.  
I Vrij I D.m.v. blockmove naar een bankadres, kan zo het  
I voor I hele Basic-programma in een bank worden geparkeerd.  
I varia- I De lengte van het programma wordt in bepaalde geheugen-  
I belen I plaatsen vastgelegd.  
I arrays I Op deze wijze zijn alle banken te vullen met verschillende  
I en I programma's, mits de lengte niet meer dan 8 K bedraagt.  
I stringsI  
I -----I-----I-----I-----I-----I &H E000 57344  
I I I 4 K I I I I  
I 7 K I 6 K I.....I 8 K I 6 K I 7 K I  
I .....I I I I .....I I  
I .....I I I I .....I I  
I -----I-----I-----I-----I-----I &H FFFF 65535  
0 1 2 3 4 5 -- banknummer

Terug uit de bank naar de Basic-ruimte is zo ook eenvoudig via blockmove te realiseren, op voorwaarde dat het eindadres van het programma, op de plaatsen &H 6405 en &H 6406 eerst wordt afgebakend.

Wanneer we deze zes Basic-programma's achter elkaar naar cassette willen schrijven, kan de Cassette-routine op plaats &H 001B in de Monitor worden gebruikt.

Een volledige Cassette-zijde kan worden beschreven, mits de programma's samen niet meer dan 42 blokken ( van 1 K ) beslaan.

Elk programma op cassette moet worden voorafgegaan door een TAPE-HEADER, dit zijn 32 bytes ter identificatie van naam, lengte, soort file enz.

Deze gegevens worden gelezen uit de TAPE-HEADER-adressen &H 6030 tot &H 604F.

Om alles te vergemakkelijken worden ook de eerste 32 plaatsen in elke bank ( vanaf &H E000 ) gebruikt om deze cassette-gegevens op te slaan, het eigenlijke Basic-programma volgt daarna vanaf &H E020.

De procedure ziet er dan als volgt uit :

De cassette wordt naar het begin van de band gespoeld.

De eerste 32 bytes in bank 0 worden in &H 6030 - &H 604F gekopieerd.

Routine &H 001B leest deze adressen en schrijft ze naar de band, gevolgd door het Basic-programma dat zich vanaf &H E020 in de bank bevindt. Vervolgens wordt bank 1 ingeschakeld ( OUT &H 94,1 ), de 32 bytes worden gelezen enz.

Na programma 6 wordt een END OF TAPE MARK gezet ( ca. 3 seconden doorspoelen ).

Opnieuw inlezen van zes programma's van cassette naar de banken gaat als volgt :

De cassette wordt naar het begin van de band teruggespoeld.

Via de routine &H 001B wordt de TAPE-HEADER gelezen en vanaf &H 6030 opgeslagen.

Deze 32 bytes worden naar het begin van bank 0 geBlockmoved ( v.a. &H E000 ).

Daarna wordt het Basic-programma van cassette in de bank vanaf &H E020 geladen.

Bank 1 wordt ingeschakeld, de cassette spoelt door, de tweede TAPE-HEADER wordt gelezen enz.

De Basic-programma's kunnen op afroep in de Basic-ruimte worden geladen, verbeterd, verkleind, vergroot ( niet > 8 K ), veranderd van naam enz. om daarna weer in een bank te worden bewaard en na afloop eventueel te worden opgenomen.

De programma's kunnen individueel natuurlijk altijd vanuit de Basic-ruimte op de normale wijze via CSAVE "Naam" naar cassette worden geschreven.

## L A D E N   M A C H I N E T A A L   V O O R   B A N K S W I T C H I N G   P.009

```

10 CLEAR 50, &H DDD0
20 DATA 2A,0D,65,23,5E,23,56,1A,47,AF,F5,D3,94,3A,06,E0,B8      BANKINHOUD NAAR
21 DATA 2B,07,F1,3C,FE,06,C8,18,FO,F1,2A,5C,62,E5,ED,4B,04      BASIC-RUIMTE
22 DATA E0,7B,B1,28,1E,09,22,05,64,22,07,64,22,09,64,D1,21      USR 0 ("NAAM")
23 DATA 20,E0,ED,BO,21,00,60,36,80,21,OC,60,36,01,C9,E1,C9
24 DATA 06,08,21,36,60,3E,20,77,23,10,FC,06,08,21,47,60,77
25 DATA 23,10,FC,2A,0D,65,AF,47,4F,46,BO,C8,05,AF,BO,C8,05      -----
26 DATA AF,BO,C8,23,5E,23,56,1A,D6,36,DO,C6,06,D3,94,13,21      BASICPROGRAMMA
27 DATA 36,60,13,1A,77,23,OC,79,B8,28,OF,FE,08,28,06,FE,10      NAAR BANK
28 DATA 28,07,18,ED,21,47,60,18,E8,2A,05,64,ED,5B,5C,62,DS      USR 1 ("N,NAAM")
29 DATA A7,ED,52,22,32,60,22,34,60,E5,01,E0,1F,ED,42,30,31
30 DATA C1,E1,C5,11,20,E0,ED,BO,E1,11,00,04,2B,AF,3C,ED,52
31 DATA 30,FB,32,4F,60,06,05,21,3E,60,11,E2,DD,1A,77,23,13
32 DATA 10,FA,21,30,60,11,00,E0,01,20,00,ED,BO,18,02,E1,D1
33 DATA CD,C7,DE,C9,42,41,53,42,4E
34 DATA 2A,0D,65,23,5E,23,56,05,1A,FE,31,28,03,CD,38,15,D1
35 DATA 13,13,1A,D6,31,DB,D6,06,DO,C6,07,32,70,60,13,13,1A      -----
36 DATA D6,30,D8,D6,06,DO,C6,06,F5,D3,94,ED,4B,04,E0,78,B1      WEGSCHRIJVEN
37 DATA 28,1A,21,00,E0,11,30,60,01,20,00,ED,BO,21,20,E0,22      UIT BANK(EN)
38 DATA 30,60,CD,D3,15,3E,05,DF,B7,20,16,3A,70,60,3D,B7,28      NAAR CASSETTE
39 DATA 09,32,70,60,F1,3C,FE,06,20,CB,3E,04,DF,F1,C9      USR 2 ("X,Y,Z")
40 DATA 2A,0D,65,23,5E,23,56,05,1A,FE,31,28,03,CD,38,15,D1
41 DATA 13,13,1A,D6,31,DB,D6,06,DO,C6,07,32,70,60,13,13,1A
42 DATA D6,30,D8,D6,06,DO,C6,06,F5,D3,94,ED,4B,04,E0,78,B1
43 DATA 28,1A,21,00,E0,11,30,60,01,20,00,ED,BO,21,20,E0,22
44 DATA 30,60,CD,D3,15,3E,05,DF,B7,20,16,3A,70,60,3D,B7,28
45 DATA 09,32,70,60,F1,3C,FE,06,20,CB,3E,04,DF,F1,C9
46 DATA 2A,0D,65,23,5E,23,56,05,1A,FE,31,28,03,CD,38,15,D1
47 DATA 13,13,1A,D6,31,DB,D6,06,DO,C6,07,32,70,60,13,13,1A
48 DATA D6,30,D8,D6,06,DO,C6,06,F5,D3,94,CD,08,15,CD,D3,15
49 DATA 2A,34,60,01,E0,1F,ED,42,38,0D,3A,4F,60,3D,32,4F,60
50 DATA 3E,02,DF,F1,18,E0,21,30,60,11,00,E0,01,00,20,ED,BO      -----
51 DATA 3E,01,32,4F,60,3E,03,DF,21,20,E0,22,30,60,3E,06,DF      LADEN VAN
52 DATA B7,20,11,3A,70,60,3D,B7,28,0A,32,70,60,F1,3C,FE,06      CASSETTE IN
53 DATA C8,18,BO,F1,C9      BANK(EN)
54 DATA AF,F5,D3,94,21,00,E0,11,30,60,01,20,00,ED,BO,CD,D3      USR 3 ("X,Y,Z")
55 DATA 15,F1,3C,FE,06,C8,18,E8
56 DATA 3A,0D,65,FE,06,20,01,AF,32,70,60,21,00,E0,06,20,D3
57 DATA 94,36,00,23,10,FB,21,06,E0,11,24,DF,1A,FE,FE,20,06
58 DATA 21,17,E0,13,18,F5,FE,FF,28,05,77,23,13,1B,EC,3A,0D
59 DATA 65,FE,06,20,08,3A,70,60,3C,FE,06,20,CB,CD,C7,DE,C9
60 DATA 62,61,6E,6B,FE,6C,65,65,67,FF
61 DATA 3A,0D,65,D6,06,DO,C6,06,D3,94,2A,5C,62,E5,ED,4B,04
62 DATA E0,7B,B1,28,11,09,22,05,64,22,07,64,22,09,64,D1,21
63 DATA 20,E0,ED,BO,C9,E1,C9
64 FOR I = 0 TO 597 : READ A$ : POKE &H DDO1 + I, VAL ("&H" + A$) : NEXT
65 DEF USR 0 = &H DDO1 : DEF USR 1 = &H DD45 : DEF USR 2 = &H DDE7
66 DEF USR 3 = &H DE4B : DEF USR 4 = &H DEC7 : DEF USR 5 = &H DEEO
67 DEF USR 6 = &H DF2E

```

Na intypen en RUNnen van bovenstaand Basic-programma heeft men de volgende mogelijkheden ter beschikking :

PRINT USR ("naam") : Het Basic-programma genaamd "naam" wordt uit de bank waarin het zich bevindt naar de Basic-ruimte gebracht.

PRINT USR 1 ("N,naam") : Een Basic-programma genaamd "naam" wordt uit de Basic-ruimte in banknummer N ( 0-5 ) gekopieerd.

PRINT USR 2 ("X,Y,Z") : Wegschriften naar cassette van de inhoud van Y banken, te beginnen bij bank Z. X=0, cassette naar begin. X=1, wegschriften vanaf het punt waar de band zich bevindt. Laden van Y programma's vanaf bank nr. Z X=1 doorspoelen, X=0 terug naar begin band.

PRINT USR 3 ("X,Y,Z") : Inhoudsopgave der banken

PRINT USR 4 ( 0 ) : CLEAR bank nummer N

PRINT USR 5 ( N ) : N=6 : uit alle 6 banken wordt het programma verwijderd.

PRINT USR 6 ( N ) : Het programma dat zich in bank N bevindt, wordt in de Basic-ruimte gekopieerd.

B A N K S W I T C H I N G  
11-11-86

Charles van der Linden & Zn.  
Broederhof 11 - 5504 JC Veldhoven

## OPBERGEN MACHINETAAL BANKSWITCHING

M.008

&amp;H

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
DD00		-2A-	0D	65	23	5E	23	56	1A	47	AF	F5	D3	94	3A	06
10	EO	B8	28	07	F1	3C	FE	06	C8	18	FO	F1	2A	5C	62	E5
20	ED	4B	04	EO	78	B1	28	1E	09	22	05	64	22	07	64	22
30	09	64	D1	21	20	EO	ED	B0	21	00	60	36	80	21	0C	60
40	36	01	C9	E1	C9	-06-	08	21	36	60	3E	20	77	23	10	FC
50	06	08	21	47	60	77	23	10	FC	2A	0D	65	AF	47	4F	46
60	80	C8	05	AF	BO	C8	05	AF	BO	C8	23	SE	23	56	1A	D6
70	36	DO	C6	06	D3	94	13	21	36	60	13	1A	77	23	0C	79
80	B8	28	OF	FE	08	28	06	FE	10	28	07	18	ED	21	47	60
90	18	EB	2A	05	64	ED	5B	5C	62	D5	A7	ED	S2	22	32	60
A0	22	34	60	E5	01	EO	1F	ED	42	30	31	C1	E1	C5	11	20
B0	EO	ED	BO	E1	11	00	04	2B	AF	3C	ED	S2	30	FB	32	4F
CO	60	06	05	21	3E	60	11	E2	DD	1A	77	23	13	10	FA	21
DO	30	60	11	00	EO	01	20	00	ED	BO	18	02	E1	D1	CD	C7
EO	DE	C9	42	41	53	42	4E	-2A-	0D	65	23	5E	23	56	D5	1A
FO	FE	31	28	03	CD	38	15	D1	13	13	1A	D6	31	D8	D6	06

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
DE00	DO	C6	07	32	70	60	13	13	1A	D6	30	D8	D6	06	DO	C6
10	06	F5	D3	94	ED	4B	04	EO	78	B1	28	1A	21	00	EO	11
20	30	60	01	20	00	ED	BO	21	20	EO	22	30	60	CD	D3	15
30	3E	05	DF	B7	20	16	3A	70	60	3D	B7	28	09	32	70	60
40	F1	3C	FE	06	20	CB	3E	04	DF	F1	C9	-2A-	0D	65	23	5E
50	23	56	D5	1A	FE	31	28	03	CD	38	15	D1	13	13	1A	D6
60	31	D8	D6	06	DO	C6	07	32	70	60	13	13	1A	D6	30	D8
70	D6	06	DO	C6	06	F5	D3	94	CD	08	15	CD	D3	15	2A	34
80	60	01	EO	1F	ED	42	38	0D	3A	4F	60	3D	32	4F	60	3E
90	02	DF	F1	18	EO	21	30	60	11	00	EO	01	00	20	ED	BO
A0	3E	01	32	4F	60	3E	03	DF	21	20	EO	22	30	60	3E	06
B0	DF	B7	20	11	3A	70	60	3D	B7	28	0A	32	70	60	F1	3C
CO	FE	06	C8	18	BO	F1	C9	-AF-	FS	D3	94	21	00	EO	11	30
DO	60	01	20	00	ED	BO	CD	D3	15	F1	3C	FE	06	CB	18	E8
EO	-3A-	OD	65	FE	06	20	01	AF	32	70	60	21	00	EO	06	20
FO	D3	94	36	00	23	10	FB	21	06	EO	11	24	DF	1A	FE	FE

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
DF00	20	06	21	17	EO	13	18	F5	FE	FF	28	05	77	23	13	18
10	EC	3A	0D	65	FE	06	20	08	3A	70	60	3C	FE	06	20	C8
20	CD	C7	DE	C9	62	61	6E	6B	FE	6C	65	65	67	FF	-3A-	OD
30	65	D6	06	DO	C6	06	D3	94	2A	5C	62	E5	ED	4B	04	EO
40	7B	B1	28	11	09	22	05	64	22	07	64	22	09	64	D1	21
50	20	EO	ED	BO	C9	E1	C9									
60																
70																

80	USR	O	68	&H	DD01	BANK NAAR BASIC ( NAAM )								
90		1	157+5		DD45	BASIC NAAR BANK								
A0		2	100		DDE7	OPNEMEN								
B0		3	124		DE4B	LADEN								
CO		4	25		DEC7	INHOUDSOPGAVE								
DO		5	68+10		DEEO	CLEAR								
EO		6	41		DF2E	BANK NAAR BASIC								
FO														

B A N K I N H O U D    N A A R    B A S I C - R U I M T E    M.009  
U S R    0 ( "naam" )

	LD HL, (650D)	2A 0D 65	laadt in HL de inhoud van &H 650D en 650E
	INC HL	23	1 plaats verder
	LD E,(HL)	5E	laadt de inhoud van HL in register E
	INC HL	23	1 verder : in DE nu de
	LD D,(HL)	56	laadt de inhoud van HL in D:plaats in de
	LD A,(DE)	1A	in A de inhoud van DE :stringruimte
	LD B,A	47	in B de inhoud van A = 1e karakter
	XOR A	AF	vul register A met getal 0 = banknummer
START	PUSH AF	F5	bewaar de inhoud even op de stack
	OUT 94	D3 94	schakel bank in waarvan nr. in A
	LD A,(E006)	3A 06 EO	laadt in A de inhoud van &H E006 = naam
	CP B	BB	vergelijk met te zoeken naam in B
	JR Z LADEN	28 07	indien gevonden dan naar LADEN
	POP AF	F1	zoniet haal het banknr. van stack
	INC A	3C	verhoog het nummer met 1
	CP 06	FE 06	kijk of het getal dan 6 wordt
	RET Z	C8	in dit geval stop, er zijn maar 6 banken
	JR START	18 F0	zoniet dan de volgende bank vanaf START
LADEN	POP AF	F1	stack in balans brengen
	LD HL,(625C)	2A 5C 62	laadt in HL het beginadres van Basic
	PUSH HL	E5	zet dit even op de stack
	LD BC,(E004)	ED 4B 04 EO	laadt in BC de inhoud van &H E004= lengte
	LD A,B	78	verifieer of de lengte
	OR C	B1	niet 0 is
	JR Z EINDE	28 1E	zoja dan naar EINDE
	ADD HL,BC	09	zoneen tel dan lengte op bij beginadres
	LD (6405),HL	22 05 64	laadt het eindadres in &H 6405/06
	LD (6407),HL	22 07 64	en in 6407/08
	LD (6409),HL	22 09 64	en in 6409/0A
	POP DE	D1	laadt beginadres vanaf stack in DE
	LD HL,E020	21 20 EO	adres waar programma in bank begint
	LDIR	ED B0	kopieer programma uit bank in Basic-
	LD HL,6000	21 00 60	zet adres inputbuffer in HL ruimte
	LD (HL),80	36 B0	laadt hierin &H 80 = 128 ( voor START )
	LD HL,600C	21 0C 60	laadt in HL het adres van de teller
	LD (HL),01	36 01	zet hierin het getal 1
	RET	C9	terug naar Basic
EINDE	POP HL	E1	stack in balans
	RET	C9	terug

Bij de aanroep PRINT USR ("naam") komt in de adressen &H 650D en &H 650E de plaats te staan waar de stringdescriptor van string "naam" zich in het geheugen bevindt. Een stringdescriptor bestaat uit 3 bytes, de eerste byte geeft de lengte van de string aan , byte 2 en 3 wijzen naar de plaats van de string in de stringruimte.

Het eerste karakter van de naam wordt even opgeslagen in register B. Nu worden de banken een voor een afgezocht of de naam van het programma in de bank overeenstemt met de naam van het gevraagde programma.

Het eerste teken van de naam van het programma in de bank bevindt zich in plaats &H E006. Door de inhoud in register A te laden en te vergelijken met de inhoud van register B, komt naar voren of het gevraagde programma in deze bank aanwezig is. Zodra de juiste bank is gevonden, wordt de lengte van het programma afgelezen uit de plaatsen &H E004/05. Door dit getal op te tellen bij het begin van de Basic-ruimte ontstaat het Basic-eindadres, wat in de plaatsen &H 6405/06 wordt vastgelegd. Evenzo in &H 6407/08 en &H 6409/0A om resp. het begin van de array- en vrije ruimte aan te geven. D.m.v. de blockmove-instruktie wordt het programma uit de bank in de Basic-ruimte gekopieerd.

Als laatste wordt de toetscode voor START ( &H 80 = 128 ) in de inputbuffer geladen, de teller hiervan op 1 gezet, het Basic-programma start nu automatisch.

B A N K S W I T C H I N G

11-11-86

Charles van der Linden & Zn.  
Broederhof 11 - 5504 JC Veldhoven

	LD B,08	06 08	laadt in B ( teller ) het getal 8
	LD HL,6036	21 36 60	laadt in HL het adres &H 6036
	LD A,20	3E 20	laadt in A het getal &H 20=32 voor spatie
START	LD (HL),A	77	laad een spatie in de geheugenplaats
	INC HL	23	1 plaats verder
	DJNZ START	10 FC	opnieuw, tot B plaatsen gevuld zijn
	LD B,08	06 08	register B weer vullen met getal 8
	LD HL,6047	21 47 60	in HL adres &H 6047
AANVANG	LD (HL),A	77	laadt vervolgens weer spatie
	INC HL	23	1 verder
	DJNZ AANVANG	10 FC	tot B plaatsen gevuld zijn
	LD HL,(650D)	2A 0D 65	laadt in HL de descriptor van "naam"
	XOR A	AF	zet inhoud van A op nul
	LD B,A	47	idem in B nul
	LD C,A	4F	idem in C nul
	LD B,(HL)	46	laadt in B de inhoud van HL=lengte string
	OR B	B0	indien lengte is nul
	RET Z	C8	dan terug naar Basic
	DEC B	05	trek van teller B het getal 1 af
	XOR A	AF	maak A weer nul
	OR B	B0	indien B = 0 ( lengte = 1 )
	RET Z	C8	dan terug
	DEC B	05	weer 1 eraf
	XOR A	AF	maak A weer 0
	OR B	B0	is B nu 0 ( lengte = 2 )
	RET Z	C8	dan terug
	INC HL	23	HL ophogen naar 2e byte van descriptor
	LD E,(HL)	5E	laadt in E de inhoud uit plaats HL
	INC HL	23	verhoog HL naar 3e byte descriptor
	LD D,(HL)	56	in D inhoud van HL : in DE plaats string
	LD A,(DE)	1A	1e karakter van string in A
	SUB 36	D6 36	trek af &H 36 (= 54 = cijfer 6 )
	REI NC	00	was getal > 6 dan terug ( 6 banken )
	ADD 06	C6 06	tel 6 erbij op = banknummer
	OUT 94	D3 94	schakel die bank in
	INC DE	13	DE 1 verder = 2e karakter
BEGIN	LD HL,6036	21 36 60	laadt in HL adres &H 6036
	INC DE	13	DE 1 verder = 3e kar.string = 1e kar.naam
	LD A,(DE)	1A	laadt de beginletter in A
	LD (HL),A	77	laadt deze letter in TAPE-HEADER-adres
	INC HL	23	volgende
	INC C	0C	verhoog karakterteller
	LD A,C	79	laadt dit getal in A
	CP B	B8	en vergelijk met totale naamlengte
	JR Z VERDER	28 0F	zijn ze gelijk dan klaar en verder
	CP 08	FE 08	indien lengte 8 is bereikt
	JR Z VOLG	28 06	dan vanaf &H 6047 verder
	CP 10	FE 10	indien lengte &H 10 = 16
	JR Z VERDER	28 07	dan is de naam maximaal en naar VERDER
	JR BEGIN	18 ED	zoniet dan terug naar BEGIN
VOLG	LD HL,6047	21 47 60	laadt in HL het adres &H 6047
	JR BEGIN	18 EB	en terug naar BEGIN
VERDER	LD HL,(6405)	2A 05 64	laadt in HL het beginadres van Basic
	LD DE,(625C)	ED 5B 5C 62	laadt in DE het eindadres
	PUSH DE	05	zet deze waarde op de stack
	AND A	A7	carry nul maken
	SBC HL,DE	ED 52	trek het beginadres van het eindadres af

VERDER OP PAGINA M.011

BANKSWITCHING  
11-11-86Charles van der Linden & Zn.  
Broederhof 11 - 5504 JC Veldhoven

	LD (6032),HL	22 32 60	laadt het verschil, dus de lengte van het Basic-programma in &H 6032/33 en 6034/35
	LD (6034),HL	22 34 60	lengte even naar de stack
	PUSH HL	E5	laadt in BC het getal &H 1FEO ( 8 K-32 )
	LD BC,1FEO	01 EO 1F	trek dit van de lengte af bytes
	SBC HL,BC	ED 42	terug indien programma groter dan 8 K
	JR NC END	30 31	lengte terug van stack
	POP BC	C1	beginadres terug van stack
	POP HL	E1	lengte nog eens bewaren
	PUSH BC	C5	laadt in DE het bankadres = begin opslag
	LD DE,E020	11 20 EO	Basic-programma naar de bank
	LDIR	ED B0	lengte van stack halen
	POP HL	E1	laadt in DE &H 0400 - 1 K = 1 cass.blok
	LD DE,0400	11 00 04	lengte 1 eraf
	DEC HL	2B	teller op 0
	XOR A	AF	teller 1 erbij
TELLER	INC A	3C	trek van de lengte 1 K af
	SBC HL,DE	ED 52	indien restant positief dan nog eens
	JR NC TELLER	30 FB	zoniet laadt in &H 604F de blokteller
	LD (604F),A	32 4F 60	laadt in teller B het getal 5
	LD B,05	06 05	laadt in HL adres &H 603E
	LD HL,603E	21 3E 60	laadt in DE het adres van TEKST
	LD DE,TEKST	11 E2 DD	in A het 1e karakter van de tekst
BASBN	LD A,(DE)	1A	zet dit karakter in HL
	LD (HL),A	77	hoog HL op met 1
	INC HL	23	idem DE
	INC DE	13	teller 1 eraf, niet nul dan nog eens
	DJNZ BASBN	10 FA	laadt in HL &H 6030 ( begin TAPE-HEADER )
	LD HL,6030	21 30 60	laadt in DE &H E000 ( begin bank )
	LD DE,E000	11 00 EO	in BC lengte &H 0020 = 32
	LD BC,0020	01 20 00	kopieer de TAPE-HEADER in begin van bank
	LDIR	ED B0	en eindig via de inhoudsopgave
	JR WEG	18 02	stack in balans
END	POP HL	E1	stack in balans
	POP DE	D1	stack in balans
WEG	CALL DIR	CD C7 DE	DIRECTORY = inhoudsopgave
	RET	C9	terug
TEKST	ASC BASBN	42 41 53 42 4E	BASBN ( soort file )

Om te beginnen worden de TAPE-HEADER administratie-adressen waar de 1e acht karakters ( &H 6036 - 603D ) plus de tweede 8 karakters ( &H 6047 - 604E ) van de programma-naam zich bevinden, met spaties gevuld. Uit de adressen &H 6050/E kan de geheugenplaats worden afgelezen waar de string-descriptor ( aanwijzer ) begint van de string binnen de USR-aanroep. B.v. A = USR 1 ( "1,programma-naam" ), de string is "1,programma-naam". De eerste byte van de descriptor geeft de lengte van de string aan, de volgende 2 bytes wijzen naar de plaats waar de string in de stringruimte te vinden is. Eerst wordt gekeken of de lengte van de string niet 0,1 of twee is, namelijk het 1e karakter van de programma-naam staat op de 3e plaats in de string. Het eerste karakter van de string geeft het banknummer aan, echter in ASCII, zodat &H 30 = 48 moet worden afgetrokken om een normaal cijfer te krijgen. De programma-naam wordt ter lengte van het aantal opgegeven karakters vastgelegd v.a. &H 6036. Bij meer dan 8 karakters, verder vanaf &H 6047 tot maximaal 16. De lengte van het Basic-programma wordt berekend door eindadres ( in &H 6405/6 ) en beginadres ( in &H 625C/5D ) van elkaar af te trekken en daarna op de plaatsen &H 6032/33 en &H 6034/35 vast te leggen. Het Basic-programma ( niet groter dan 8 K ) wordt in de bank gekopieerd ( v.a. &H E020 ). Het aantal cas-setteblokken wordt berekend en opgeslagen in &H 604F. Als laatste worden de 32 bytes van de TAPE-HEADER in het begin van de bank ( v.a. &H E000 ) gekopieerd.

PROGRAMMA'S VAN BANK NAAR CASSETTE M.012  
 USR 2 ("X,Y,Z") Y programma's vanaf bank Z, X=1 doorspoelen, X=0 terug

	LD HL,(650D)	2A 0D 65	* laadt in HL de inhoud van 650D=descriptor
	INC HL	23	* 1 plaats verder = 2e byte stringdescriptor
	LD E,(HL)	5E	* laadt in E de inhoud van HL
	INC HL	23	* 1 verder = 3e byte van descriptor, nu in
	LD D,(HL)	56	* laadt in D inhoud DE DE de plaats
	PUSH DE	DS	* waarde DE op stack van de string
	LD A,(DE)	1A	* laad in A inhoud DE = 1e karakter string
	CP 31	FE 31	* is de inhoud &H 31 = 49 = getal 1
	JR Z VERDER	28 03	* dan naar VERDER
	CALL 1538	CD 38 15	* zoniet terugspoelen naar begin band
VERDER	POP DE	D1	* waarde DE van stack halen
	INC DE	13	* 1 plaats verder = 1e komma
	INC DE	13	* 1 plaats verder = getal na komma
	LD A,(DE)	1A	* inhoud in A = aantal programma's in ASCII
	SUB 31	D6 31	* trek van dit getal &H 31 = 49 af
	RET C	D8	* indien verschil < 0 dan terug naar Basic
	SUB 06	D6 06	* trek nog eens 6 af
	RET NC	DO	* is dit verschil < 0 dan terug, > 6 banken
	ADD 07	C6 07	* zoniet tel er weer 7 bij op
	LD (6070),A	32 70 60	* laadt aantal programma's in &H 6070
	INC DE	13	* DE 1 plaats verder = 2e komma
	INC DE	13	* een plaats verder = getal na komma
	LD A,(DE)	1A	* laadt getal uit DE = banknummer in ASCII
	SUB 30	D6 30	* trek er &H 30 = 48 = getal 0 van af
	RET C	D8	* wordt verschil < 0 dan terug naar Basic
	SUB 06	D6 06	* zoniet trek nog eens 6 af
	RET NC	DO	* is getal nu < 0 dan terug, > 6 banken
	ADD 06	C6 06	* zoniet tel weer 6 erbij op
START	PUSH AF	F5	* zet de waarde op stack ( banknummer )
	OUT 94	D3 94	* schakel betreffende bank in
	LD BC,(E004)	ED 4B 04 EO	laadt in BC inhoud van &H E004 ( lengte )
	LD A,B	78	staat in B en
	OR C	B1	in C het getal 0 dus lengte nul
	JR Z EIND	28 1A	dan naar EIND
	LD HL,E000	21 00 EO	laadt in HL de geheugenplaats &H E000
	LD DE,6030	11 30 60	laadt in DE de geheugenplaats &H 6030
	LD BC,0020	01 20 00	laadt in BC het getal &H 20 = 32
	LDIR	ED B0	blockmove, 32 bytes van &H E000 naar 6030
	LD HL,E020	21 20 EO	laadt in HL de geheugenplaats &H E020
	LD (6030),HL	22 30 60	laadt in &H 6030/31 de waarden &H 20 + EO
	CALL 15D3	CD D3 15	subroutine : zet programma-naam op scherm
	LD A,05	3E 05	laadt getal 5 in A ( schrijven )
	RST 18	DF	aanroep Cassette-routine in Monitor
	OR A	B7	is cassette-fout nul ( geen fout )
	JR NZ SLOT	20 16	zoniet dan naar SLOT
EIND	LD A,(6070)	3A 70 60	waarde programma-teller uit &H 6070
	DEC A	3D	een eraf
	OR A	B7	is het nu nul
	JR Z END	28 09	zoja naar END
	LD (6070),A	32 70 60	zoniet nieuwe tellerstand in &H 6070
	POP AF	F1	banknummer van stack halen
	INC A	3C	nummer 1 omhoog
	CP 06	FE 06	is het al 6
	JR NZ START	20 CB	zoniet weer opnieuw vanaf START
END	LD A,04	3E 04	in A het getal 4 voor END OF TAPE MARK
	RST 18	DF	naar Cassette-routine
SLOT	POP AF	F1	stack in balans brengen
	RET	C9	terug naar Basic

BANKSWITCHING

11-11-86

Charles van der Linden & Zn.

Broederhof 11 - 5504 JC Veldhoven

## LADEN VAN CASSETTE IN BANK ( EN )

M.013

USR 3 ( "X,Y,Z" ) Y programma's vanaf bank Z, X=1 doorspoelen, X=0 terug

LD HL,650D	2A 0D 65	* Het begin is hetzelfde als bij Program-
INC HL	23	* ma's van Bank naar Cassette ( M.012 )
:::::::		* zie gedeelte voorafgegaan door '*'s
enZ.		*
:::::::		*
ADD 06	C6 06	*
START PUSH AF	FS	*
OUT 94	D3 94	*
CALL 150B	CD 08 15	lees de TAPE-HEADER
CALL 15D3	CD 03 15	zet de programma-naam op het beeldscherm
LD HL,(6034)	2A 34 60	laadt in HL waarde uit &H 6034 ( lengte )
LD BC,1FEO	01 E0 1F	laadt in BC het getal &H 1FEO = 8 K - 32
SBC HL,BC	ED 42	trek getallen van elkaar af
JR C DOOR	38 0D	indien verschil < 0 dan naar DOOR
LD A,(604F)	3A 4F 60	is programma dus > B K dan laadt
DEC A	3D	in A het aantal blokken, verminder
LD (604F),A	32 4F 60	het met 1 en laadt getal in &H 604F
LD A,02	3E 02	spoel dit aantal verder
RST 1B	DF	op cassette
POP AF	F1	stack in balans
JR START	18 E0	en ga verder met volgend programma
DOOR LD HL,6030	21 30 60	laadt in HL de geheugenplaats &H 6030
LD DE,E000	11 00 E0	laadt in DE de geheugenplaats &H E000
LD BC,0020	01 20 00	laadt in BC het getal &H 20 - 32
LDIR	ED B0	blockmove, 32 bytes van &H 6030 naar E000
LD A,01	3E 01	laadt in A het getal 1
LD (604F),A	32 4F 60	laadt in de blokteller dit getal
LD A,03	3E 03	laadt in A getal 3 voor terugspoelen
RST 1B	DF	Cassette-routine
LD HL,E020	21 20 E0	laadt in HL de geheugenplaats &H E020
LD (6030),HL	22 30 60	laadt in &H 6030/31 waarden &H 20 en E0
LD A,06	3E 06	laadt het getal 6 in A voor lezen
RST 1B	DF	van cassette
OR A	B7	is de cassette-fout 0 ( geen fout )
JR NZ EIND	20 11	zoniet dan naar EIND
LD A,(6070)	3A 70 60	geen fout dan in A getal uit &H 6070
DEC A	3D	trek van dit programma-aantal 1 af
OR A	B7	wordt het getal nu 0
JR Z EIND	28 0A	zoja dan naar EIND
LD (6070),A	32 70 60	programma-teller opslaan in &H 6070
POP AF	F1	banknummer van stack
INC A	3C	1 bank verder
CP 06	FE 06	wordt het nummer 6
RET Z	C8	dan terug ( niet > 6 banken )
JR START	18 B0	zoniet weer opnieuw vanaf START
EIND POP AF	F1	stack in balans
RET	C9	terug naar Basic

Wegschrijven van een bankinhoud naar cassette gaat als volgt :

De string uit de USR-aanroep : USR 2 ( "X,Y,Z" ), bevat drie waarden, X,Y en Z. Indien X=1 dan wordt geschreven vanaf het punt waar de cassette zich bevindt, bij X=0 spoelt de cassette naar het begin van de band. Y geeft het aantal programma's aan dat opgenomen moet worden en Z vanaf welk banknummer.

De eerste 32 bytes uit de bank ( v.a. &H E000 ) worden in de TAPE-HEADER-adressen ( v.a. &H 6030 ) gekopieerd. Het programma dat zich in de bank ( vanaf &H E020 ) bevindt, wordt ter lengte van de inhoud van &H 6034/35 naar cassette geschreven. Daarna wordt, indien gevraagd, de volgende bank ingeschakeld enz.

Laden van cassette in bank ( en ) geschiedt op eenzelfde wijze. De TAPE-HEADER wordt in de TAPE-HEADER-adressen gelezen en daarna vanaf &H E000 in de bank gekopieerd. Het eigenlijke programma volgt vanaf &H E020, enz.

B A N K S W I T C H I N G

11-11-86

Charles van der Linden & Zn.  
Broederhof 11 - 5504 JC Veldhoven

## INHOUDSOPGAVE EN CLEAR BANK (EN)

M.014

## USR 4 (0)

## INHOUDSOPGAVE = DIR(ECTORY)

DIR	XOR A	AF	zet inhoud van register A op nul
START	PUSH AF	F5	zet inhoud van A op stack ( banknummer )
	OUT 94	D3 94	schakel het banknummer in
	LD HL,E000	21 00 EO	laadt in HL de geheugenplaats &H E000
	LD DE,6030	11 30 60	laadt in DE de geheugenplaats &H 6030
	LD BC,0020	01 20 00	laadt in BC het getal &H 20 = 32
	LDIR	ED B0	blockmove, 32 bytes van &H E000 naar 6030
	CALL 15D3	CD D3 15	zet programma-naam enz. op het scherm
	POP AF	F1	banknummer van stack
	INC A	3C	1 bank verder
	CP 06	FE 06	wordt getal 6
	RET 2	C8	dan terug naar Basic ( niet > 6 banken )
	JR START	18 E8	zoniet opnieuw volgende bank vanaf START

## USR 5 (N)

## CLEAR BANK NUMMER N ( N=6 : alle banken )

	LD A,(650D)	3A 0D 65	laadt in A het getal uit &H 650D = N
	CP 06	FE 06	kijk of het 6 is : CLEAR alle banken
	JR NZ AANVANG	20 01	zoneen dan naar AANVANG
	XOR A	AF	zet banknummer op nul
AANVANG	LD (6070),A	32 70 60	laadt banknummer in &H 6070
	LD HL,E000	21 00 EO	laadt in HL de geheugenplaats &H E000
	LD B,20	06 20	laadt in B de teller &H 20 = 32
	OUT 94	D3 94	schakel de desbetreffende bank in
BEGIN	LD (HL),00	36 00	laadt in geheugenplaats het getal nul
	INC HL	23	een verder
	DJNZ BEGIN	10 FB	indien teller niet nul dan nogmaals
	LD HL,E006	21 06 EO	laadt in HL de geheugenplaats &H E006
	LD DE,TEKST	11 24 DF	laadt in DE de plaats aangeduid met TEKST
BACK	LD A,(DE)	1A	laadt in A karakter uit woord : bank enz.
	CP FE	FE FE	is dit byte FE
	JR NZ volg	20 06	zoniet dan naar volg
	LD HL,E017	21 17 EO	anders laadt in HL de plaats &H E017
	INC DE	13	DE 1 verder
	JR BACK	18 F5	weer naar BACK voor het volgend woord
volg	CP FF	FE FF	is het byte FF
	JR Z VOLG	28 05	dan naar VOLG
	LD (HL),A	77	zoniet zet 1 karakter van woord in HEADER
	INC HL	23	1 plaats verder
	INC DE	13	volgend karakter
	JR BACK	18 EC	indien niet klaar dan naar BACK
VOLG	LD A,(650D)	3A 0D 65	laadt in A nogmaals het getal uit &H 650D
	CP 06	FE 06	was dit 6
	JR NZ VERDER	20 08	zoneen dan verder
	LD A,(6070)	3A 70 60	zoja dan tellerstand uit &H 6070
	INC A	3C	verhoog banknummer met een
	CP 06	FE 06	wordt het dan zes
VERDER	JR NZ AANVANG	20 C8	zoniet dan terug naar AANVANG
	CALL DIR	CD C7 DE	na afloop vraag INHOUDSOPGAVE op
	RET	C9	en terug naar Basic
TEKST	ASC bank	62 61 6E 6B	woord bank
	BYTE FE	FE	einde 1e woord
	ASC leeg	6C 65 65 67	woord leeg
	BYTE FF	FF	einde tekst

De inhoudsopgave wordt gerealiseerd door de eerste 32 bytes uit de bank in de TAPE-HEADER-adressen te kopiëren om daarna op het scherm te laten printen.  
 CLEAR van een of alle banken geschiedt door in de eerste 32 bytes van de bank nullen te laden en daarna de tekst : bank leeg, op de naam-plaatsen te zetten.

B A N K S W I T C H I N G

11-11-86

Charles van der Linden &amp; Zn.

Broederhof 11 - 5504 JC Veldhoven

Half augustus 1984 kwam ik in het bezit van een 64 K-uitbreiding met 5 extra geheugenbanken ( van 8 K ) van Miniware. Ondanks alle moeite, zowel persoonlijk als telefonisch en schriftelijk, kon mij niet worden verteld hoe die 5 banken te gebruiken waren. Ten einde raad besloot ik het zelf te gaan uitzoeken. Na enkele bezoeken aan een deskundige op het gebied van machinetaal programmeren ( Hans van der Veer ), ben ik aan het experimenteren geslagen. De laatste 8 K van het geheugen ( &H E000 - &H FFFF ) ( bank 0 ), kan worden verwisseld met een bank die dezelfde geheugenadressen heeft. Door middel van CLEAR 50,&H DFFF wordt 8 K gereserveerd. De instructie OUT &H 94,N ( banknummer ) laat het begin van de bank ( &H E000 ) aansluiten op plaats &H DFFF van het geheugen. Na 2 1/2 maand gepuzzel is het voorlopige resultaat 10 machinetaal-routines : USR 0 (N) = Een Basic-programma ( max. 8 K ) wordt uit bank N naar de Basic-ruimte gebracht. USR 1 (N) = Een programma wordt vanuit de Basic-ruimte in bank N gekopieerd. USR 2 ("n") = Een programma wordt vanuit een bank naar cassette geschreven onder de naam "n". USR 3 ("n") = Een programma met naam "n" wordt van cassette gelezen in een bank. USR 4 (N) = De inhoud van het beeldscherm ( Video-geheugen ) wordt naar bank N gebracht. Diverse beelden achter elkaar. USR 5 (N) = Uit bank N wordt de beeldscherm-inhoud achter elkaar op het scherm geprojecteerd. USR 6 (0) = De blockmove routine ; van plaats naar plaats ter lengte van. USR 7 (0) = De cassette-routine : van plaats tot plaats. USR 8 (U) = Verplaats een array naar een bank en omgekeerd. U = VARPIR (array-element (1)). USR 9 (U) = Vertaald een string-array uit de stringruimte naar een bank. B = VARPIR (string-array-element (1)).

Deze routines ( ruim 300 bytes ) bevinden zich in een startprogramma van 2 K dat eerst moet worden geRUNd. Het is zodoende mogelijk van alles in de

de banken op te bergen, terug te vragen, weg te schrijven en in te lezen.

I-----I &H 6547	
I Voor- I ( 25927 )	Op de bijeenkomst van 25 april 1985 van
I programma uitvoering	afdeling Eindhoven werd een demonstratie-
I-----I &H 6D47	programma met 70 schermbladzijden tekst
I Opening I ( 27975 )	en 21 pagina's overzichten getoond.
I 1 I Tekst 1-5	De 6 banken en praktisch het hele geheugen
I I uit bank 1-5	werden gevuld. Het startprogramma laadt de
I I	machinetaal en achter elkaar de banken 1 t/m 5.
I-----I &H 8D47	De cassette wordt omgedraaid. Het hoofdprogramma
I Opening I ( 36167 )	wordt geRUNd, dit programma beslaat 18 K,
I 2 I Tekst 6 + 3	waarvan 2 K uitvoerend programma en 2 x 8 K
I I Overzichten	open ruimten.
I I	Opening 2 bevat de eindtekst en drie overzichten
I-----I &H AD47	Het uitvoerend programma laadt overzicht 6 van
I I ( 44359 )	cassette in bank 0 en overzicht 7 vult opening 3.
I Vrij I Variabelen	In opening 1 wordt de inhoud van bank 1-5 naar
I geheugen I Array	behoefte via blockmove gekopieerd.
I I Stack	Het voorprogramma laat zodoende elke gewenste
I-----I &H C000	tekstbladzijde of overzichtspagina op het scherm
I Opening I ( 50176 )	verschijnen. Het laden van een cassette-zijde
I 3 I Overzicht 7	duurt ca. 1 minuut, eenmaal in de banken geladen
I I Z-80 Code	kan elk soort programma direct opgeroepen worden.
I-----I-----I-----I-----I-----I-----I-----I-----I-----I &H E000	
I bank 0 I bank 1 I bank 2 I bank 3 I bank 4 I bank 5 I ( 57344 )	
I overz. 6 I I I I I I	
I RAM-adres I TEKST 1 I TEKST 2 I TEKST 3 I TEKST 4 I TEKST 5 I ( 65535 )	
I-----I-----I-----I-----I-----I-----I-----I-----I-----I &H FFFF	

M A G I K M I J E U E N V O O R S T E L L E N ? ? ? I.004  
M I J N N A A M I S . . .  
Charles van der Linden (augustus 1985)

Sinds februari 1984 houd ik mij bezig met het fenomeen Informatica en in het bijzonder met de P-2000 I.  
Momenteel ben ik vooral in de weer met de mogelijkheden van extra geheugenbanken met behulp van BANKSWITCHING, zowel in Basic als met gebruik van machinetaal.  
Omdat ik in een relatief korte periode diverse stadia van programmeren doorlopen heb met alle problemen van dien, wil ik enthousiaste P-2000 bezitters, met voorlopig nog minder ervaring, mee laten profiteren van de opgedane kennis.  
Via de TRON en eventuele programma's op cassette zal ik benaderd informeren over mijn gevolgde weg, de tussenliggende obstakels en de uiteindelijke resultaten.  
Verder ben ik graag bereid serieuze belangstellenden bij mij thuis wegwijs te maken, na een schriftelijk verzoek daartoe.

Tot voor kort was ik zelf nog een volslagen leek op computergebied.  
Wel was er langzamerhand bij mij een steeds groter wordende sluimerende belangstelling ontstaan voor de Computer. Voor leken is dit een magisch woord.  
Het onbekende geeft iets mysterieus, voor sommigen iets beangstigends.  
Er is een hele hoge muur waarover je niet heen kan kijken naar Computerland.

Heel wat moet er gebeuren om de drempelvrees te overwinnen.  
Wanneer je niet beroepsmatig min of meer gedwongen met de computer te maken krijgt, is het moeilijk een initiatief te nemen om kennis te maken met deze onbekende wereld. In mijn geval was er enerzijds de interesse ontstaan door de groeiende confrontatie met computers via krant en T.V.  
Anderzijds was ik zakelijk komen te staan voor een enorm informatie-aanbod, dat met de traditionele middelen niet onder eigen controle te brengen was, hetgeen een groot gevoel van onmacht gaf.

Uiteindelijk was het mijn dochter, van toen 13 jaar, die mij over de muur heentilde. Zij kwam thuis met de mededeling: "Ik wil een computer!".  
Op mijn vraag waarom, vertelde ze computerles op school te gaan krijgen en er dan thuis ook wel een nodig zou hebben om te oefenen.

Ofschoon haar over-enthousiasme snel wegebde, was het voor mij de directe aanleiding om aktie te ondernemem.

Ik wilde absoluut de aansluiting met de toekomst niet missen.  
Na enige informatie inwinnen belandde ik bij een computerclub, de Eindhovenese Computer Associatie, kortweg E.C.A., waar ik na een korte proefperiode lid werd.  
Ik maakte daar kennis met de P-2000 en startte met het BASIC-Probeerboek.  
Een nieuwe wereld ging voor mij open, die echter ook een aaneenschakeling van doolhofs bleek te zijn.

Er waren zoveel merken computers, computertalen, hardware, software, vakjargon alleen voor ingewijden. De eerste periode onderging ik als een lichte hersenspoeling, je hele denkwereld wordt een slag gedraaid.

Ik was praktisch op mezelf aangewezen, hulp van anderen bleef tot een minimum beperkt. Bij vragen had men geen tijd, men wist het zelf niet of de vraag was dom. De meeste mensen in het computerwereldje zijn zo vertrouwd met alles, dat ze zich niet meer kunnen voorstellen hoe het was zonder deze kennis. Dat ze zelf niet met de kennis zijn geboren, het ooit eens niet wisten en het ook hadden moeten leren, zijn ze vergeten.

Door gebruiksaanwijzingen lijken te zijn geschreven voor personen die het al weten. Het schijnt voor de schrijvers moeilijk zich in te denken in het geestelijk bevattingsvermogen van een beginner.

Meestal pas wanneer iemand het uitlegt of voordoet, komt de geschreven tekst tot leven. Vaak echter ga je dingen pas echt snapen na een bepaalde tijd, met meer ervaring, bij nog eens en nog eens doorlezen.  
Tot nu toe ben ik in staat geweest alle voorkomende problemen bij het programmeren vroeg of laat op te lossen. De juiste dosis wilskracht, concentratie en de tijd ervoor nemen zijn hierbij de belangrijkste ingredienten voor succes.

B A N K S W I T C H I N G - Charles van der Linden & Zn. - Broederhof 11  
11-11-86 5504 JC Veldhoven.

In 1.000 werd besproken welke mogelijkheden extra geheugenbanken bieden, er werd echter niet verteld hoe alles exact in elkaar zit.

Uit de vele reakties is gebleken dat alle P-2000 bezitters met 64 K-uitbreiding problemen hebben met het niet kunnen gebruiken van de banken.

Om ieder in staat te stellen eindelijk iets met deze 5 x 8 K extra te gaan doen, zal een tipje van de sluier opgelicht worden.

Besproken worden 4 machine-taal routines : 1e Basic naar Bank  
 2e Bank naar Basic 3e Bank naar Cassette 4e Cassette naar bank  
 Om dit alles te kunnen volgen is het onderstaande van belang :  
 Een bank kan de laatste 8 K van het geheugen ( bank 0 ) vervangen.  
 De geheugen-adressen lopen van &H E000 tot &H FFFF ( 57344 - 65535 ).  
 D.m.v. de instruktie OUT &H 94,N ( N = 1 - 5 ) wordt bank 0 verwisseld met  
 bank N. Vooraf dient deze 8 K gereserveerd te worden door: CLEAR 50,&H DFFF.  
 Een Basic-programma begint op de plaats die aangeduid wordt in de adressen  
 &H 625C en &H 625D, normaal &H 47 en &H 65, wijzend naar &H 6547.  
 Het einde van een programma is te vinden bij het begin van de variabelen-  
 ruimte via de geheugenplaatsen &H 6405 en &H 6406, zoals boven of volgens de  
 formule: inhoud &H 6405 + 256 x inhoud &H 6406 = PEEK(&H 6405)+ 256\*PEEK(&H 6406)  
 Aanroep van een machinetaalprogramma gaat via A = USR(X(Y) of PRINT USR(X(Y),  
 X = 0 - 9 , Y kan een getal of string zijn. In de plaatsen &H 650D en &H 650E  
 komt dit getal te staan.  
 Bij een string betekent dit cijfer het adres van de stringdescriptor.

#### BASIC naar BANK :

Een Basic-programma van max. 8 K wordt in een bank gekopieerd.

Gekozen is voor de aanroep A = USR1(N), waarbij N = het banknummer.

LD A,(650D)	3A 0D 65	laadt in register A het getal in &H 650D
OUT 94	D3 94	schakel bank N in
LD DE,(625C)	ED 5B SC 62	laadt in registerpaar DE inhoud &H 625C / 62
LD HL,(6405)	2A 05 64	laadt in registerpaar HL inhoud &H 6405 / 64
SBC HL,DE	ED 52	trek getal in DE af van getal in HL, verschil
LD (E000),HL	22 00 EO	laadt in adressen E000 / 01 getal uit HL - lengte
LD B,H	44	laadt in register B getal uit H : lengte nu
LD C,L	4D	laadt in register C getal uit L : in BC
EX DE,HL	EB	verwissel getallen in HL en DE met elkaar
LD DE,E002	11 02 EO	laadt in DE het getal E002
LDIR	ED B0	blockmove = verplaats = kopieer
RET	C9	terug uit machine-taal

Blockmove is de instructie om een reeks getallen ( een block ) te kopieeren van de ene geheugenplaats naar de andere :

Registerpaar HL dient het adres te bevatten vanaf waar gekopieerd moet worden, in DE het adres waarnaartoe en in BC het aantal te verplaatsen bytes.

In het bovenstaande geval wordt in het begin van de bank de lengte van het Basicprogramma bewaard ( in E000 en E001 ), vanaf &H E002 staat het block getallen die samen het programma vormen.

#### BANK naar BASIC :

Om een programma uit een bank weer terug in de Basicruimte te brengen is de aanroep A = USR0(N) gekozen, kort A = USR(N).

Een ruimte voor het programma wordt gereserveerd door bij het normale beginadres &H 6547 de lengte van het programma op te tellen ( uit &H E000 / 01 ) en de eindwijzer te vullen met de som (= begin variabelenruimte ).

LD A,(650D)	3A 0D 65	banknummer in register A
OUT 94	D3 94	bank N ingeschakeld
LD DE,6547	11 47 65	in DE beginadres Basic, waarnaartoe block
LD HL,(E000)	2A 00 EO	in HL lengte uit adressen E000 / 01
ADD HL,DE	19	tel lengte op bij beginadres
LD (6405),HL	22 05 64	laadt in &H 6405 / 64 het eindadres
LD HL,E002	21 02 EO	in HL adres vanaf waar
LD BC,(E000)	ED 4B 00 EO	in BC de lengte uit E000 / 01
LDIR	ED B0	blockmove
RET	C9	terug

B A N K S W I T C H I N G - Charles van der Linden & Zn. - Broederhof 11  
 11-11-86 5504 JC Veldhoven.

## BANK naar CASSETTE :

Nadat een Basic-programma in een bank is opgeslagen, kan het vanuit de bank naar cassette worden geschreven. Gebruik gemaakt wordt van de machinetaal aanwezig in de Basic-module op de plaatsen &H 1059 ( lezen ) en &H 105C ( schrijven ). Registerter A wordt gevuld met de ASCII-waarde van de beginletter van de naam van het programma. Registerpaar HL dient het adres te bevatten van de eerste byte van het op te nemen blok, in DE het adres van de laatste byte + 1. De routine wordt aangeroepen met AS\$ = USR2("n"). De stringdescriptor, het adres van de eerste van drie bytes in de variabelenruimte, komt nu in adres &H 650D en &H 650E te staan. Deze byte geeft de lengte van de string, bytes 2 + 3 wijzen naar het adres van de eerste byte van string "n" in de stringruimte.

LD HL,(650D)	2A 0D 65	in HL adres stringdescriptor uit &H 650D / 0E
INC HL	23	verhoog HL met 1, = 1e byte adres stringruimte
LD E,(HL)	SE	laadt in E de inhoud van adres in HL
INC HL	23	HL + 1, = 2e byte adres in stringruimte
LD D,(HL)	56	laadt in D de inhoud van 2e byte
LD A,(DE)	1A	laadt in A 1e letter van string
LD DE,E000	11 00 EO	in DE begin te schrijven blok
LD BC,(E000)	ED 4B 00 EO	in BC lengte blok
LD HL,E002	21 02 EO	in HL plaats waar Basic-gegevens beginnen
ADD HL,BC	09	tel BC ( lengte ) op bij HL ( begin )
INC HL	23	eind + 1
EX DE,HL	EB	verwissel DE en HL, begin in HL, eind in DE
RET	C9	terug

## CASSETTE naar BANK : Aanroep AS\$ = USR3("n")

LD HL,(650D)	2A 0D 65	stringdescriptor van "n" in HL
INC HL	23	1 adres verder
LD E,(HL)	SE	)
INC HL	23	) adres van "n" in stringruimte
LD D,(HL)	56	)
LD A,(DE)	1A	inhoud adres "n" in A, naam
LD HL,E000	21 00 EO	in HL begin bank start inlezen
LD DE,0000	11 00 00	einde bank + 1, FFFF + 1 = 0000
CALL 1059	CD 59 10	leesroutine in Basic-module
RET	C9	terug

De 4 machinetaal-routines moeten nu in het geheugen worden vastgelegd om ze op afroep te kunnen gebruiken. Een lege ruimte in het geheugen bevindt zich tussen &H 6150 en &H 61EF.

Het volgende Basic-programma zet de routines op zijn plaats, wijst waar welke routine begint en reserveert de 8 K voor de banken.

```

10 DATA 3A,0D,65,D3,94,11,47,65,2A,00,EO,19,22,05,64,21,02,EO,ED,4B,00,
    EO,ED,BO,C9:           REM : USR0(N)      &H 6150 - &H 6168
20 DATA 3A,0D,65,D3,94,ED,5B,5C,62,2A,05,64,ED,52,22,00,EO,44,4D,EB,11,
    02,EO,ED,BO,C9:       REM : USR1(N)      &H 6169 - &H 6182
30 DATA 2A,0D,65,23,SE,23,56,1A,11,00,EO,ED,4B,00,EO,21,02,EO,09,23,EB,
    CD,5C,10,C9:          REM : USR2("n")    &H 6183 - &H 619B
40 DATA 2A,0D,65,23,SE,23,56,1A,21,00,EO,11,00,00,CD,59,10,C9:
    REM : USR3("n")    &H 619C - &H 61AD
50 CLEAR 50,&H DFFF
60 FOR I = 0 TO 93 : READ AS$ : POKE &H 6150+I, VAL("&H"+AS$) : NEXT I
70 DEFUSR = &H 6150 : DEFUSR1 = &H 6169 : DEFUSR2 = &H 6183 : DEFUSR3 = &H 619C

```

Na inlezen van het programma zijn ook na NEW de routines aanroepbaar.

Naar aanleiding van het vorig artikel zijn er diverse problemen naar voren gekomen.

Een bestaand programma (<8 K) in een bank brengen gaat als volgt:

Het Startprogramma wordt geRUNd, de machinetaal is dan geladen.  
Het normale programma wordt ingelezen via CLOAD "Programma".

Het programma wordt vanuit de Basic-ruimte m.b.v. PRINT USR1(N) in bank N geladen, b.v. PRINTUSR1(1).

Daarna uit de bank opnemen op cassette via : OUT&H94,1 : PRINT USR2("A")  
A = de letter waaronder de bankinhoud op cassette opgeslagen wordt.

Kopieren van het startprogramma op een andere cassette is mogelijk.  
De andere programma's zijn weggeschreven bankprogramma's en kunnen niet zonder meer gekopieerd worden.

Het is alleen mogelijk door via USR3("naam") van cassette in een bank te laden en daarna uit de bank op een andere cassette te schrijven m.b.v. ? USR2("naam").

Een programma als b.v. Snelzoekboek zal eigen machinetaal hebben en wellicht daardoor de bankswitching-machinetaal door de war gooien of de DEF USR-adressen op een andere plaats zetten.

In dit geval kan alleen een kans van slagen hebben indien de Bankswitch-routines via de Funktietoetsen aangeroepen kunnen worden i.p.v. met USR's.  
In een volgend nummer zal ik hierop verder ingaan.

Dit uitgave zal benut worden om te laten zien hoe strings en videobeelden in een bank kunnen worden opgeslagen.

Machinetaal voor de programma's STRINGS MAKEN (1) en VIDEOTEXT (2).

Onderstaand STARTPROGRAMMA eerst runnen, daarna het programma 1 of 2 apart laden.

```

10 CLEAR 50,&HDE7F
20 DATA 3A,0D,65,D3,94,ED,5B,70,60,3A,72,60,2A,73,60,01,28,00,C5,ED,B0,
   C1,09,3D,B7,C8,1B,F6           REM:USR4(N)

30 DATA 3A,0D,65,D3,94,2A,70,60,3A,72,60,ED,5B,73,60,01,28,00,C5,ED,B0,
   EB,C1,09,EB,3D,B7,C8,1B,F4     REM:USR5(N)

40 DATA 2A,75,60,ED,5B,77,60,ED,4B,79,60,ED,B0,C9     REM:USR6(O)

50 DATA 3A,7B,60,2A,7C,60,ED,5B,7E,60,CD,59,10,C9     REM:USR7("n")

60 DATA 2A,E7,DE,44,4D,2A,0D,65,C5,22,EB,DE,3A,E9,DE,D3,94,AF,47,4E,23,
   5E,23,56,EB,11,00,DF,D5,C5,ED,B0,3A,EA,DE,D3,94,C1,D1,EB,ED,5B,
   ED,DE,D5,ED,B0,ED,53,ED,DE,2A,EB,DE,23,D1,73,23,72,23,C1,0B,78,
   B1,20,C6,2A,E5,DE,22,00,EO,2A,E7,DE,22,02,EO,29,ED,4B,E7,DE,09,
   44,4D,2A,0D,65,ED,5B,ED,DE,ED,B0,C9           REM:USR9(V)

70 FORI=0TO57:READA$:POKE&H61AE+I,VAL("&H"+A$):NEXTI
80 FORI=0TO27:READA$:POKE&H6360+I,VAL("&H"+A$):NEXTI
90 FORI=0TO95:READA$:POKE&HDE80+I,VAL("&H"+A$):NEXTI
100 DEFUSR4 = &H61AE:DEFUSR5 = &H61CA:DEFUSR6 = &H6360:DEFUSR7 = &H636E:
      DEFUSR9 = &HDE80

```

USR 0(N) BANK N naar BASIC	USR 1(N) BASIC naar BANK N	USR 2("n") BANK naar CASSEITIE naam"n"
LD A,(650D)	3A 0D 65	LD HL,(650D) 2A 0D 65
OUT 94	D3 94	OUT 94 D3 94 INC HL 23
LD DE,6547	11 47 65	LD DE,(625C) ED 5B 5C 62 LD E,(HL) SE
LD HL,(E000)	2A 00 EO	LD HL,(6405) 2A 05 64 INC HL 23
ADD HL,DE	19	SBC HL,DE ED 52 LD D,(HL) 56
LD (6405),HL	22 05 64	LD (E000),HL 22 00 EO LD A,(DE) 1A
LD HL,E002	21 02 EO	LD B,H 44 LD DE,E000 11 00 EO
LD BC,(E000)	ED 4B 00 EO	LD C,L 40 LD BC,(E000) ED 4B 00 EO
LDIR	ED BO	EX DE,HL EB LD HL,E002 21 02 EO
RET	C9	LD DE,E002 11 02 EO ADD HL,BC 09
		LDIR ED BO INC HL 23
	RET	C9 EX DE,HL EB
		CALL 105C CD 5C 10
		RET C9

USR 3("n") CASSETTE NAAR BANK naam"n"	USR 4(N) VIDEO-BEELD naar BANK N	USR 5(N) BANK N naar VIDEO-BEELD
--	-------------------------------------	-------------------------------------

LD HL,(650D)	2A 0D 65	LD A,(650D) 3A 0D 65
INC HL	23	OUT 94 D3 94
LD E,(HL)	SE	LD A,(6070) 3A 70 60
INC HL	23	LD HL,(6071) 2A 71 60
LD D,(HL)	56	LD DE,(6073) ED 5B 73 60
LD A,(DE)	1A	LD BC,0028 01 28 00
LD HL,E000	21 00 EO	start:PUSH BC CS
LD DE,0000	11 00 00	LD DE,(6073) ED 5B 73 60
CALL 1059	CD 59 10	LD BC,0028 01 28 00
RET	C9	start:PUSH BC CS
		LDIR ED BO
		POP BC C1
		ADD HL,BC 09
		DEC A 3D
		OR A B7
		RETI Z CB
		JR :start 18 F6
		POP BC C1
		ADD HL,BC 09
		DEC A 3D
		OR A B7
		RETI Z CB
		JR : start 18 F4

USR 6(O)  
BLOCKMOVE

USR 8(V)  
ARRAY naar BANK , V=VARPIR(G%(0))

LD HL,(6071)	2A 71 60	LD HL,(650D) 2A 0D 65
LD DE,(6073)	ED 5B 73 60	PUSH HL E5
LD BC,(6075)	ED 4B 75 60	DEC HL 2B
LDIR	ED BO	DEC HL 2B
RET	C9	DEC HL 2B
		DEC HL 2B

USR 7(O)  
CASSETTE-ROUTINE

LD A,(6070)	3A 70 60	LD B,HL 46
LD HL,(6071)	2A 71 60	DEC HL 2B
LD DE,(6073)	ED 5B 73 60	LD C,HL 4E
CALL 1059	CD 59 10	DEC BC OB
RET	C9	LD (E000),BC ED 43 00 EO
		DEC BC OB
		DEC BC OB
		LD DE,E002 11 02 EO
		POP HL E1
		LDIR ED BO
		RET C9

## USR 9(V)

STRINGS naar BANK , V=VARPTR(U\$(1))

LD HL,(DEE7)	2A E7 DE	in HL aantal strings uit plaats &H DEE7
LD B,H	44	in reg. B inhoud van reg. H
LD C,L	4D	in C inhoud L : dus in BC inhoud HL
LD HL,(650D)	2A 0D 65	in HL inhoud van &H 650D = VARPTR(U(1))
start:PUSH BC	C5	bewaar BC op stack
LD (DEEB),HL	22 EB DE	bewaar HL in plaats &H DEEB ( VARPTR )
LD A,(DEE9)	3A E9 DE	in A banknr. uit &H DEE9
OUT 94	D3 94	bank ingeschakeld
XOR A	AF	inhoud A wordt 0
LD B,A	47	in B inhoud van A (= 0)
LD C,(HL)	4E	in C inhoud van HL
INC HL	23	hoog HL met 1 op
LD E,(HL)	5E	in E inhoud HL
INC HL	23	hoog HL met 1 op
LD D,(HL)	56	in D inhoud HL
EX DE,HL	EB	verwissel DE met HL
LD DE,DFOO	11 00 DF	in DE getal &H DFOO
PUSH DE	DS	bewaar DE op stack
PUSH BC	C5	bewaar BC op stack
LDIR	ED BO	blockmove
LD A,(DEEA)	3A EA DE	in A banknr. waarnaartoe
OUT 94	D3 94	bank ingeschakeld
POP BC	C1	BC van stack
POP DE	D1	DE van stack
EX DE,HL	EB	verwissel DE met HL
LD DE,(DEED)	ED 5B ED DE	in DE inhoud van &H DEED ( plaats )
PUSH DE	DS	bewaar DE op stack
LDIR	ED BO	blockmove
LD (DEED),DE	ED 53 ED DE	in &H DEED inhoud DE
LD HL,(DEEB)	2A EB DE	in HL inhoud &H DEEB ( VARPTR )
INC HL	23	hoog HL met 1 op
POP DE	D1	DE van stack
LD (HL),E	73	in plaats door HL aangewezen, inhoud E
INC HL	23	hoog HL met 1 op
LD (HL),D	72	in plaats door HL aangewezen, inhoud D
INC HL	23	hoog HL met 1 op
POP BC	C1	BC van stack
DEC BC	0B	verlaag BC met 1
LD A,B	78	in A inhoud B
OR C	B1	is teller 0
JR NZ start	20 C6	indien niet 0 dan terug naar start
LD HL,(DEE5)	2A E5 DE	in HL inhoud van &H DEE5 ( lengte )
LD (EOOO),HL	22 00 EO	in &H EOOO inhoud van HL
LD HL,(DEE7)	2A E7 DE	in HL inhoud van &H DEE7 ( aantal )
LD (EOO2),HL	22 02 EO	in &H EOO2 inhoud van HL
ADD HL,HL	29	tel inhoud HL op bij inhoud HL
LD BC,(DEE7)	ED 4B E7 DE	in BC inhoud van &H DEE7 ( aantal )
ADD HL,BC	09	tel inhoud BC op bij inhoud HL
LD B,H	44	in B inhoud H
LD C,L	4D	in C inhoud L
LD HL,(650D)	2A 0D 65	in HL inhoud &H 650D
LD DE,(DEED)	ED 5B ED DE	in DE inhoud &H DEED ( plaats )
LDIR	ED BO	blockmove
RET	C9	terug

## S T R I N G S

Een stringarray wordt als volgt opgeslagen:

In de arrayruimte wordt een ruimte gereserveerd d.m.v. DIMmen.

B.v. DIM A\$(100) betekent dat er plaats is voor 101 elementen ( 0 - 100 ). Elk element heeft 3 bytes nodig om aan te wijzen waar de string zich in de Stringruimte bevindt, de stringdescriptoren.

De plaats in de arrayruimte wordt aangeduid door de VARPTR.

A! = VARPTR(A\$(1)) wijst naar de 1e byte van A\$(1).

Byte 1 geeft de lengte aan, byte 2 en 3 samen de plaats, volgens de formule : PEEK ( A! + 1 ) + 256 \* PEEK ( A! + 2 ).

De Stringruimte bevindt zich achter in het geheugen. De eerste string wordt achteraan opgeslagen de volgende strings telkens daarvoor.

D.m.v. blockmove is het mogelijk elke string vanuit de stringruimte in een bank te kopiëren, van voren naar achteren.

Telkens wordt de stringdescriptor in de arrayruimte zodanig gewijzigd dat deze naar de string in de bank gaat wijzen.

Na afloop worden de stringdescriptoren achteraan de strings in de bank geschreven.

Voorin de bank worden de totale lengte en het aantal strings vastgelegd.

Op de in IRON 4 besproken wijze kan de bankinhoud naar cassette worden geschreven en terug op dezelfde manier :

Van cassette in een bank laden, voor het aantal strings een ruimte DIMmen in de arrayruimte ,de stringdescriptoren via blockmove van achter de strings in de bank naar de arrayruimte brengen.

De strings worden nu uit de bank gelezen en zodoende kan de stringruimte gespaard worden.

Wanneer deze strings worden veranderd of verbeterd komt deze nieuwe string in de stringruimte te zitten.

Door elke string apart weer uit te lezen en in een andere bank op te slaan kunnen de verbeterde versies uit die bank weer naar cassette worden geschreven.

STRINGS MAKEN : programma 1.

10 CLEARB200,&HDE7F	stringruimte op B200
200 PRINICHR\$(12)	schoon scherm
210 DEFINTA-I	variabelen A - I zijn integers
220 DEFSTRU-Z	variabelen U - Z zijn strings
230 DEFFNA!(A!)=PEEK(A!)+256*PEEK(A!+1)	maakt 16 bits getal
240 DEFFNB(A!)=INT(A!/256)	splitst 16 bits getal,
250 DEFFNA(A!)=A!-256*FNB(A!)	in hoge en lage byte
260 Z!=57344	begin bank
270 PRINT"max.aantal strings";	te verwachten max.
280 INPUTA	
290 DIMU(A)	ruimte reserveren
300 GOSUB910	naar menu
310 PRINT"hoeveel strings maken";	
320 INPUT H	aantal strings
330 K=0	
340 FORI=1TOH	
350 PRINT"string" I	invoer strings
360 LINEINPUTU(I)	indien alleen return
370 IFU(I)=""THEN350	lengte strings tesamen
380 K=K+LEN(U(I))	
390 NEXTI	

B A N K S W I T C H I N G  
S T R I N G S vervolg.  
VERTALEN NAAR BANK :

410 GOSUB810 overzicht strings  
420 GOSUB710 welke bank  
430 K=K+4+3\*X lengte totale File  
440 M=N uit stringruimte  
450 GOTO530  
510 PRINT"naar bank";  
520 INPUTM uitleesadressen,  
530 POKE&HDEE5,FNA(K):POKE&HDEE6,FNB(K) invullen  
550 POKE&HDEE7,FNA(H):POKE&HDEE8,FNB(H) van bank  
570 POKE&HDEE9,N naar bank  
580 POKE&HDEEA,M opslan v.a.&H E004  
590 POKE&HDEED,&H04:POKE&HDEEE,&HE0 stringroutine  
610 A=USR9(VARPTR(U(1)))  
620 N=M  
630 GOSUB810 overzicht strings  
640 GOTO910 naar keuze  
710 PRINT"welke bank";  
720 INPUTN banknummer  
730 OUT&H94,N inschakelen bank  
740 RETURN  
810 FORI=1TOH  
820 PRINTI;  
830 PRINTUCI);  
840 PRINT" ";  
850 NEXTI  
860 PRINT:RETURN  
910 PRINT"1=strings maken "; menu  
920 PRINT"2=nakijken"  
930 PRINT"3=vertalen naar bank ";  
950 PRINT"4=opnemen ";  
960 PRINT"5=laden";  
970 INPUTA  
980 ONAGOTO310,1010,510,1210,1230  
STRINGS UIT BANK :  
1010 GOSUB710 bank  
1020 A=0  
1030 K=FNAI(Z1) lengte file  
1040 H=FNAI(Z1+2) aantal strings  
1050 Y1=Z1+K-3\*X plaats van descriptoren  
1060 G1=VARPTR(U(1)) plaats van stringdescriptor,  
1070 IFG1<OTHENG1=G1+65536 in arrayruimte  
1080 POKE&H6077,FNA(G1):POKE&H6078,FNB(G1) vastleggen uitleesgegevens  
1100 POKE&H6075,FNA(Y1):POKE&H6076,FNB(Y1)  
1120 POKE&H6079,FNA(H\*3):POKE&H607A,FNB(K\*3)  
1140 A=USR6(0) blockmove descriptoren  
1150 GOSUB810 overzicht strings  
1160 GOTO910 naar menu  
OPNEMEN EN LADEN :  
1210 POKE&H6379,&H5C:GOTO1250 byte voor schrijven  
1230 POKE&H6379,&H59 byte voor lezen van cassette  
1240 K=8191 gehele bank  
1250 GOSUB710 bank  
1310 PRINT"welke naam"; 1e letter van naam  
1320 INPUTN\$ vullen van door de machinestaal  
1330 POKE&H607C,FNA(Z1):POKE&H607D,FNB(Z1) uit te lezen adressen  
1350 POKE&H607E,FNA(Z1+K):POKE&H607F,FNB(Z1+K) naam  
1370 POKE&H607B,ASC(N\$) cassetteroutine

## V I D E O B E E L D E N

Het videogeheugen, de geheugenplaatsen die corresponderen met het beeldscherm, bevindt zich van &H 5000 - &H 57FF ( 20480 - 22527 ).

Het geheugen is opgebouwd uit 24 regels en 80 kolommen.

De linkerhelft, 40 kolommen, vormt het normale beeldscherm.

Het beeldscherm kan opgenomen worden door de inhoud van de geheugenplaatsen vast te leggen als array, als strings, in een bank of weg te schrijven naar cassette.

Opslaan in een bank is mogelijk door telkens de eerste 40 plaatsen van het beeld via blockmove achter elkaar in een bank te kopiëren.

Een scherm bestaat uit  $24 \times 40 = 960$  bytes, zodat in een bank van 8 K B beelden ( $8 \times 960 = 7680$ ) kunnen worden opgeslagen.

De inhoud van de bank kan worden opgenomen op cassette en later weer terug van cassette naar een bank.

Vanuit de bank kunnen de beelden weer achter elkaar op het scherm worden geprojecteerd.

## V I D E O T E X T

## programma 2.

```

110 DEFINTA-U
120 DEFSTRV-Z
130 DEFFNCS=CHR$(4)+CHR$(24)+CHR$(1)
140 DEFFNAI(A1)=PEEK(A1)+256*PEEK(A1+1)
150 DEFFNBA(A1)=INT(A1/256)
160 DEFFNAC(A1)=A1-256*FNB(A1)
170 Z1=57344
TEKST OP SCHERM
200 PRINTCHR$(12)"1=tekst maken ";
210 PRINT"2=bank naar beeld"
220 PRINT"3=opnemen 4=laden";:INPUTA
230 ONAGO10310,1010,2010,2110
310 GOSUB850
320 PRINTCHR$(12)
330 PRINTCHR$(2);
340 Q=20480
400 J=PEEK(Q)
410 U=PEEK(Q)
420 IF J>159THENJ=J-128
430 POKEQ,J+128
440 POKE&H600C,0
450 A=INP("")
460 IFA=35HENNA=95ELSEIFA=95HENNA=35
470 IFA>31ANDA<128HENJ=A:K=1:GOTO610
500 IFA=16HENK=-1
510 IFA=17HENK=-80
520 IFA=18HENK=80
530 IFA=19HENK=-1
540 IFA=7IHEN910
550 IFA=130THEN200
560 IFA=13IHENGOSUB710:GOSUB810:Q=00:
                           GOTO750
570 IFA=8IHENGOSUB710:Q=Q-1:POKEQ,32:
                           J=32:GOTO610
580 IFA=9IHENGOSUB710:Q=Q+8:GOTO750
600 J=U
610 POKEQ,J
620 Q=Q+K
630 IFQ<20480ORQ>22399HENQ=Q-K
640 IF(Q=20480)MOD80>39thenQ=Q+40
650 GOTO400

```

variabelen A - U zijn integers  
 variabelen V - Z zijn strings  
 regel 24 kolom 1  
 maakt 16 bits cijfer  
 splitst 16 bits cijfer,  
 in hoge en lage byte  
 &H E000 = begin bank

schoon scherm  
 keuze

welke bank  
 schoon scherm  
 normale cursor uit  
 startplaats linksboven &H 5000  
 inhoud cursorplaats  
 idem  
 indien karakter geinverteerd  
 karakter inverteren  
 inputbuffer leeg maken  
 toetsindruk in ASCII-code  
 omwisselen # en \_  
 indien letter of cijfer  
 cursortoets naar links  
 cursortoets omhoog  
 cursortoets omlaag  
 cursortoets naar rechts  
 OPN beeldscherm naar bank  
 DEF naar menu  
 RETURN regel omlaag vooraan regel

wis 1 plaats terug

TAB 8 plaatsen verder  
 karakter blijft gelijk  
 druk karakter af  
 cursor naar nieuwe plaats  
 indien buiten beeld, terug  
 indien rechterscherm  
 terug

V I D E O T E X T      vervolg.

710 K=0  
 720 POKEQ,U  
 730 RETURN  
 750 J=PEEK(Q)  
 760 GOT0610  
 810 D=INT(Q-20480)/80  
 820 IFQ<24THENOO=20480+(D+1)\*80:RETURN  
 830 OO=22320:RETURN  
 850 PRINT"welke bank";  
 860 INPUT N  
 870 OUT&H94,N  
 880 RETURN  
 BEELD NAAR BANK :  
 910 GOSUB 710  
 920 GOSUB1110  
 930 A=24  
 940 GOSUB 1210  
 950 AA=USR4(N)  
 960 GOT0340  
 1110 GOSUB 1310  
 1120 PRINTFNC\$;  
 1130 PRINTCHR\$(2);  
 1140 GOSUB1810  
 1150 GOSUB1410  
 1160 RETURN  
 1210 GOSUB1910  
 1220 POKE&H6070,FNA(A!):POKE&H6071,FNB(A!)  
 1240 POKE&H6072,A  
 1250 POKE&H6073,&H00:POKE&H6074,&H50  
 1270 RETURN  
 1310 A!=&H5730  
 1320 B!=&H6260  
 1330 POKE&H6075,FNA(A!):POKE&H6076,FNB(A!)  
 1340 POKE&H6077,FNA(B!):POKE&H6078,FNB(B!)  
 1350 POKE&H6079,&H50:POKE&H607A,&H00  
 1370 AA=USR6(O)  
 1380 RETURN  
 1410 A!=&H6260  
 1420 B!=&H5730  
 1430 GOT01330  
 1810 PRINT"welk beeld ? ";  
 1820 B=INP("")-48  
 1830 IFB<1THEN1820  
 1840 IFB>8THEN1820  
 1850 RETURN  
 1910 A!=Z!+2+(B-1)\*A\*40  
 1920 RETURN  
 BANK NAAR BEELD :  
 1010 GOSUB850  
 1020 PRINTCHR\$(12)  
 1030 GOSUB1110  
 1040 GOSUB1060  
 1050 GOT01510  
 1060 A=24  
 1070 GOSUB1210  
 1080 AA=USR5(N)  
 1090 RETURN  
 1510 GOSUB1310

cursor blijft op plaats  
 geinverteerd uit  
 karakter blijft gelijk  
 regelnummer  
 naar volgende regel  
 indien regel 24  
 banknummer  
 bank N wordt ingeschakeld

geinverteerd uit  
 welk beeld in bank  
 24 regels  
 uit te lezen adressen vullen  
 routine beeld naar bank  
 terug naar beeld  
 regel 24 even opslaan  
 op regel 24  
 cursor uit  
 welk beeld  
 regel 24 weer terug

berekening plaats in bank  
 vastleggen plaats  
 aantal regels  
 begin scherm &H 5000

regel 24,  
 naar inputbuffer  
 vastleggen om uit te lezen,  
 door machinetaalroutine  
 lengte &H 50 = 80 karakters  
 blockmove

uit inputbuffer,  
 naar regel 24  
 enz.

ASCII-code - 48 = cijfer  
 getal tussen 1,  
 en 8

plaats in bank

welke bank  
 schoon scherm  
 welk beeld  
 beeld uit bank

24 regels  
 vullen adressen  
 routine bank naar beeld

regel 24 opbergen

## V I D E O T E X T      vervolg.

```

1520 PRINTFNC$CHR$(2);
1540 PRINT"1=edit 2=bladeren";
1550 A=INP("")-48
1560 GOSUB1410
1570 IFA=2THEN1610
1580 GOTO330
1610 A=INP("")
1620 IFA=16THENB=B-1
1630 IFA=19THENB=B+1
1640 IFB=OTHENGOSUB1710
1650 IFB=9THENGOSUB1750
1660 IFA=32THENGOSUB1410:GOTO330
1670 GOSUB1060
1680 GOSUB1310
1690 PRINTFNC$N;B;
1700 GOTO1610
1710 N=N-1
1720 IFN<0IHENN=0:B=1:RETURN
1730 B=B
1740 RETURN
1750 N=N+1
1760 IFN>5IHENN=5:B=0:RETURN
1770 B=1
1780 RETURN

```

op regel 24, cursor uit  
keuze  
invoer cijfer  
regel 24 weer terug  
bladeren naar 1610  
EDIT, naar scherm  
toets in ASCII-code  
toets naar links, beeld terug  
toets naar rechts, verder  
naar vorige bank  
naar volgende bank  
spatie, naar EDIT  
beeld naar bank  
regel 24 opbergen  
nummer bank en beeld  
doorbladeren

bank 0 blijft  
beeld B

volgende bank  
bank 5 blijft  
beeld 1

## OPNEMEN EN LADEN :

```

2010 POKE&H6379,&H5C
2020 GOSUB2210
2030 GOTO340
2110 POKE&H6379,&H59
2120 GOSUB2210
2130 B=1:GOSUB1060
2140 GOTO340
2150 POKE&H607C,FNA(A!):POKE&H607D,FNB(A!)
2170 POKE&H607E,FNA(B!):POKE&H607F,FNB(B!)
2190 A$=USR7(N$)
2200 RETURN
2210 PRINT"1=beeld 2=bank";
2220 INPUTA
2230 GOSUB2510
2240 GOSUBB50
2250 POKE&H607B,ASC(N$)
2260 ONAGOTO2310,2410
2310 GOSUB1810
2320 PRINTB;
2330 GOSUB1910
2340 B!=A!+960
2350 GOSUB2150
2360 RETURN
2410 A!=57344
2420 B!=A!+7682
2430 GOSUB2150
2440 RETURN
2510 PRINT"welke naam";
2520 INPUTN$
2530 RETURN

```

byte voor opnemen  
cassetteroutine  
terug naar EDIT-stand  
byte voor laden  
cassetteroutine  
beeld 1 naar scherm  
naar EDIT-stand  
vastleggen om uit te lezen  
idem  
cassetteroutine

keuze om op te nemen

naam  
banknummer  
letter voor routine

beeldnummer

beginadres in bank  
eindadres block  
naar cassetteroutine

begin bank  
eindadres  
naar cassetteroutine

1e letter naam op cassette

## HEXADECIMAAL - BINAIR

T.001

&H	00	10	20	30	40	50	60	70	&H
0	00000000	00010000	00100000	00110000	01000000	01010000	01100000	01110000	0
1	00000001	00010001	00000001	00110001	01000001	01010001	01100001	01110001	1
2	00000010	00010010	00100010	00110010	01000010	01010010	01100010	01110010	2
3	00000011	00010011	00100011	00110011	01000011	01010011	01100011	01110011	3
4	00000100	00010100	00100100	00110100	01000100	01010100	01100100	01110100	4
5	00000101	00010101	00100101	00110101	01000101	01010101	01100101	01110101	5
6	00000110	00010110	00100110	00110110	01000110	01010110	01100110	01110110	6
7	00000111	00010111	00100111	00110111	01000111	01010111	01100111	01110111	7
8	00001000	00011000	00101000	00111000	01001000	01011000	01101000	01111000	8
9	00001001	00011001	00101001	00111001	01001001	01011001	01101001	01111001	9
A	00001010	00011010	00101010	00111010	01001010	01011010	01101010	01111010	A
B	00001011	00011011	00101011	00111011	01001011	01011011	01101011	01111011	B
C	00001100	00011100	00101100	00111100	01001100	01011100	01101100	01111100	C
D	00001101	00011101	00101101	00111101	01001101	01011101	01101101	01111101	D
E	00001110	00011110	00101110	00111110	01001110	01011110	01101110	01111110	E
F	00001111	00011111	00101111	00111111	01001111	01011111	01101111	01111111	F
&H	80	90	A0	B0	C0	D0	E0	F0	&H
0	10000000	10010000	10100000	10110000	11000000	11010000	11100000	11110000	0
1	10000001	10010001	10100001	10110001	11000001	11010001	11100001	11110001	1
2	10000010	10010010	10100010	10110010	11000010	11010010	11100010	11110010	2
3	10000011	10010011	10100011	10110011	11000011	11010011	11100011	11110011	3
4	10000100	10010100	10100100	10110100	11000100	11010100	11100100	11110100	4
5	10000101	10010101	10100101	10110101	11000101	11010101	11100101	11110101	5
6	10000110	10010110	10100110	10110110	11000110	11010110	11100110	11110110	6
7	10000111	10010111	10100111	10110111	11000111	11010111	11100111	11110111	7
8	10001000	10011000	10101000	10111000	11001000	11011000	11101000	11111000	8
9	10001001	10011001	10101001	10111001	11001001	11011001	11101001	11111001	9
A	10001010	10011010	10101010	10111010	11001010	11011010	11101010	11111010	A
B	10001011	10011011	10101011	10111011	11001011	11011011	11101011	11111011	B
C	10001100	10011100	10101100	10111100	11001100	11011100	11101100	11111100	C
D	10001101	10011101	10101101	10111101	11001101	11011101	11101101	11111101	D
E	10001110	10011110	10101110	10111110	11001110	11011110	11101110	11111110	E
F	10001111	10011111	10101111	10111111	11001111	11011111	11101111	11111111	F

## HEXADECIMAAL - DECIMAAL

&H	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0	&H
0	0	16	32	48	64	80	96	112	128	144	160	176	192	208	224	240	0
1	17	33	49	65	81	97	113	129	145	161	177	193	209	225	241	1	
2	18	34	50	66	82	98	114	130	146	162	178	194	210	226	242	2	
3	19	35	51	67	83	99	115	131	147	163	179	195	211	227	243	3	
4	20	36	52	68	84	100	116	132	148	164	180	196	212	228	244	4	
5	21	37	53	69	85	101	117	133	149	165	181	197	213	229	245	5	
6	22	38	54	70	86	102	118	134	150	166	182	198	214	230	246	6	
7	23	39	55	71	87	103	119	135	151	167	183	199	215	231	247	7	
8	24	40	56	72	88	104	120	136	152	168	184	200	216	232	248	8	
9	25	41	57	73	89	105	121	137	153	169	185	201	217	233	249	9	
A	26	42	58	74	90	106	122	138	154	170	186	202	218	234	250	A	
B	27	43	59	75	91	107	123	139	155	171	187	203	219	235	251	B	
C	28	44	60	76	92	108	124	140	156	172	188	204	220	236	252	C	
D	29	45	61	77	93	109	125	141	157	173	189	205	221	237	253	D	
E	30	46	62	78	94	110	126	142	158	174	190	206	222	238	254	E	
F	31	47	63	79	95	111	127	143	159	175	191	207	223	239	255	F	

T A B E L L E N

11-11-86

Charles van der Linden &amp; Zn.

Broederhof 11, 5504 JC Veldhoven.

( c )

## HEXADECIMAAL - DECIMAAL

T.002

&H 000 100 200 300 400 500 600 700 800 900 A00 B00 C00 D00 E00 F00																
00	00	256	512	768	1024	1280	1536	1792	2048	2304	2560	2816	3072	3328	3540	3840
10	16	272	528	784	1040	1296	1552	1808	2064	2320	2576	2832	3088	3344	3600	3856
20	32	288	544	800	1056	1312	1568	1824	2080	2336	2592	2848	3104	3360	3616	3872
30	48	304	560	816	1072	1328	1584	1840	2096	2352	2608	2864	3120	3376	3632	3888
40	64	320	576	832	1088	1344	1600	1856	2112	2368	2624	2880	3136	3392	3648	3904
50	80	336	592	848	1104	1360	1616	1872	2128	2384	2640	2896	3152	3408	3664	3920
60	96	352	608	864	1120	1376	1632	1888	2144	2400	2656	2912	3168	3424	3680	3936
70	112	368	624	880	1136	1392	1648	1904	2160	2416	2672	2928	3184	3440	3696	3952
80	128	384	640	896	1152	1408	1664	1920	2176	2432	2688	2944	3200	3456	3712	3968
90	144	400	656	912	1168	1424	1680	1936	2192	2448	2704	2960	3216	3472	3728	3984
A0	160	416	672	928	1184	1440	1696	1952	2208	2464	2720	2976	3232	3488	3744	4000
B0	176	432	688	944	1200	1456	1712	1968	2224	2480	2736	2992	3248	3504	3760	4016
C0	192	448	704	960	1216	1472	1728	1984	2240	2496	2752	3008	3264	3520	3776	4032
D0	208	464	720	976	1232	1488	1744	2000	2256	2512	2768	3024	3280	3536	3792	4048
E0	224	480	736	992	1248	1504	1760	2016	2272	2528	2784	3040	3296	3552	3808	4064
F0	240	496	752	1008	1264	1520	1776	2032	2288	2544	2800	3056	3312	3568	3824	4080
																4096

&amp;H 0000 1000 2000 3000 4000 5000 6000 7000 &amp;H

000	000	4096	8192	12288	16384	20480	24576	28672	000
100	256	4352	8448	12544	16640	20736	24832	28928	100
200	512	4608	8708	12800	16896	20992	25088	29184	200
300	768	4864	8960	13056	17152	21248	25344	29440	300
400	1024	5120	9216	13312	17408	21504	25600	29696	400
500	1280	5376	9472	13568	17664	21760	25856	29952	500
600	1536	5632	9728	13824	17920	22016	26112	30208	600
700	1792	5888	9984	14080	18176	22272	26368	30464	700
800	2048	6144	10240	14336	18432	22528	26624	30720	800
900	2304	6400	10496	14592	18688	22784	26880	30976	900
A00	2560	6656	10752	14848	18944	23040	27136	31232	A00
B00	2816	6912	11008	15104	19200	23296	27392	31488	B00
C00	3072	7168	11264	15360	19456	23552	27648	31744	C00
D00	3328	7424	11520	15616	19712	23808	27904	32000	D00
E00	3584	7680	11776	15872	19968	24064	28160	32256	E00
F00	3840	7936	12032	16128	20224	24320	28416	32512	FOO

&amp;H 8000 9000 A000 B000 C000 D000 E000 F000 &amp;H

000	32768	36864	40960	45056	49152	53248	57344	61440	000
100	33024	37120	41216	45312	49408	53504	57600	61696	100
200	33280	37376	41472	45568	49664	53760	57856	61952	200
300	33536	37632	41728	45824	49920	54016	58112	62208	300
400	33792	37888	41984	46080	50176	54272	58368	62464	400
500	34048	38144	42240	46336	50432	54528	58624	62720	500
600	34304	38400	42496	46592	50668	54784	58880	62976	600
700	34560	38656	42752	46848	50944	55040	59136	63232	700
800	34816	38912	43008	47104	51200	55296	59392	63488	800
900	35072	39168	43264	47360	51456	55552	59648	63744	900
A00	35328	39424	43520	47616	51712	55808	59904	64000	A00
B00	35584	39680	43776	47872	51968	56064	60160	64256	B00
C00	35840	39936	44032	48128	52224	56320	60416	64512	C00
D00	36096	40192	44288	48384	52480	56576	60672	64768	D00
E00	36352	40448	44544	48640	52736	56832	60928	65024	E00
F00	36608	40704	44800	48896	52992	57088	61184	65280	FOO
									65536

T A B E L L E N

11-11-86

Charles van der Linden & Zn.  
Broederhof 11, 5504 JC Veldhoven.

( c )

&H	00	10	20	30	40	&H
O	NOP	DJNZ ..	JR NZ ..	JR NC ....	LD B,B	O
1	LD BC,....	LD DE,....	LD HL,....	LD SP,....	LD B,C	1
2	LD (BC),A	LD (DE),A	LD (....),HL	LD (....),A	LD B,D	2
3	INC BC	INC DE	INC HL	INC SP	LD B,E	3
4	INC B	INC D	INC H	INC (HL)	LD B,H	4
5	DEC B	DEC D	DEC H	DEC (HL)	LD B,L	5
6	LD B,..	LD D,..	LD H,..	LD (HL),..	LD B,(HL)	6
7	RLCA	RLA	DAA	SCF	LD B,A	7
8	EX AF	JR ..	JR Z ..	JR C ..	LD C,B	8
9	ADD HL,BC	ADD HL,DE	ADD HL,HL	ADD HL,SP	LD C,C	9
A	LD A,(BC)	LD A,(DE)	LD HL,(....)	LD A,(....)	LD C,D	A
B	DEC BC	DEC DE	DEC HL	DEC SP	LD C,E	B
C	INC C	INC E	INC L	INC A	LD C,H	C
D	DEC C	DEC E	DEC L	DEC A	LD C,L	D
E	LD C,..	LD E,..	LD L,..	LD A,..	LD C,(HL)	E
F	RRCA	RRA	CPL	CCF	LD C,A	F

&H	50	60	70	80	90	&H
O	LD D,B	LD H,B	LD (HL),B	ADD B	SUB B	AND B
1	LD D,C	LD H,C	LD (HL),C	ADD C	SUB C	AND C
2	LD D,D	LD H,D	LD (HL),D	ADD D	SUB D	AND D
3	LD D,E	LD H,E	LD (HL),E	ADD E	SUB E	AND E
4	LD D,H	LD H,H	LD (HL),H	ADD H	SUB H	AND H
5	LD D,L	LD H,L	LD (HL),L	ADD L	SUB L	AND L
6	LD D,(HL)	LD H,(HL)	HALT	ADD (HL)	SUB (HL)	AND (HL)
7	LD D,A	LD H,A	LD (HL),A	ADD A	SUB A	AND A
8	LD E,B	LD L,B	LD A,B	ADC B	SBC B	XOR B
9	LD E,C	LD L,C	LD A,C	ADC C	SBC C	XOR C
A	LD E,D	LD L,D	LD A,D	ADC D	SBC D	XOR D
B	LD E,E	LD L,E	LD A,E	ADC E	SBC E	XOR E
C	LD E,H	LD L,H	LD A,H	ADC H	SBC H	XOR H
D	LD E,L	LD L,L	LD A,L	ADC L	SBC L	XOR L
E	LD E,(HL)	LD L,(HL)	LD A,(HL)	ADC (HL)	SBC (HL)	XOR (HL)
F	LD E,A	LD L,A	LD A,A	ADC A	SBC A	XOR A

&H	BO	CO	DO	EO	FO	&H
O	OR B	REI NZ	REI NC	RET PO	RET P	O
1	OR C	POP BC	POP DE	POP HL	POP AF	1
2	OR D	JP NZ ....	JP NC ....	JP PO ....	JP P ....	2
3	OR E	JP ....	OUT ..	EX (SP),HL	DI	3
4	OR H	CALL NZ ....	CALL NC ....	CALL PO ....	CALL P ....	4
5	OR L	PUSH BC	PUSH DE	PUSH HL	PUSH AF	5
6	OR (HL)	ADD ..	SUB ..	AND ..	OR ..	6
7	OR A	RST 00	RST 10	RST 20	RST 30	7
8	CP B	RET Z	RET C	RET PE	RET M	8
9	CP C	RET	EXX	JP (HL)	LD SP,HL	9
A	CP D	JP Z ....	JP C ....	JP PE ....	JP M ....	A
B	CP E	diversen	IN	EX DE,HL	EI	B
C	CP H	CALL Z ....	CALL C ....	CALL PE ....	CALL M ....	C
D	CP L	CALL ....	diversen	diversen	diversen	D
E	CP (HL)	ADC ..	SBC ..	XOR ..	CP ..	E
F	CP A	RST 08	RST 18	RST 28	RST 38	F

## Z - 8 0 C O D E - I I .

T.004

ED ..	nn = ....	ED ..	ED ..	ED ..	ED ..	ED ..	ED ..
&H 40	50	60	70	AO	BO	BO	&H
O IN B,(C)	IN D,(C)	IN H,(C)	---	LDI	LDI	0	
1 OUT (C),B	OUT (C),D	OUT (C),H	---	CPI	CPIR	1	
2 SBC HL,BC	SBC HL,DE	SBC HL,HL	SBC HL,SP	INI	INIR	2	
3 LD (nn),BC	LD (nn),DE	---	LD (nn),SP	OUTI	OUTIR	3	
4 NEG	---	---	---	---	---	4	
5 RETN	---	---	---	---	---	5	
6 IM 0	IM 1	---	---	---	---	6	
7 LD I,A	LD A,I	RRD	---	---	---	7	
8 IN C,(C)	IN E,(C)	IN L,(C)	IN A,(C)	LDD	LDDR	8	
9 OUT (C),C	OUT (C),E	OUT (C),L	OUT (C),A	CPD	CPDR	9	
A ADC HL,BC	ADC HL,DE	ADC HL,HL	ADC HL,SP	IND	INDR	A	
B LD BC,(nn)	LD DE,(nn)		LD SP,(nn)	OUTD	OUTDR	B	
C ---	---	---	---	---	---	C	
D RETI	---	---	---	---	---	D	
E ---	IM 2	---	---	---	---	E	
F LD R,A	LD A,R	RLD	---	---	---	F	

DD .. d (d : - 128 tot + 127)

&H	FD .. d	IY = IX	FD i.p.v. DD : IY = IX
09	ADD IX,BC	71 LD (IX+d),C	DD CB d ..
19	ADD IX,DE	72 LD (IX+d),D	FD CB d ..
21	LD IX,....	73 LD (IX+d),E	&H
22	LD (....),IX	74 LD (IX+d),H	06 RLC (IX+d)
23	INC IX	75 LD (IX+d),L	0E RRC (IX+d)
29	ADD IX,IX	77 LD (IX+d),A	16 RL (IX+d)
2A	LD IX,(....)	7E LD A,(IX+d)	1E RR (IX+d)
2B	DEC IX	86 ADD (IX+d)	26 SLA (IX+d)
34	INC (IX+d)	8E ADC (IX+d)	2E SRA (IX+d)
35	DEC (IX+d)	96 SUB (IX+d)	36 ---
36	LD (IX+d),..	9E SBC (IX+d)	3E SRL (IX+d)
39	ADD IX,SP	A6 AND (IX+d)	46 BIT 0,(IX+d)
46	LD B,(IX+d)	AE XOR (IX+d)	4E BIT 1,(IX+d)
4E	LD C,(IX+d)	B6 OR (IX+d)	56 BIT 2,(IX+d)
56	LD D,(IX+d)	BE CP (IX+d)	5E BIT 3,(IX+d)
5E	LD E,(IX+d)	E1 POP IX	66 BIT 4,(IX+d)
66	LD H,(IX+d)	E5 PUSH IX	6E BIT 5,(IX+d)
6E	LD L,(IX+d)	E9 JP (IX)	76 BIT 6,(IX+d)
70	LD (IX+d),B	F9 LD SP,IX	7E BIT 7,(IX+d)
			FE SET 7,(IX+d)

CB ..

registers : B,C,D,E,H,L,(HL),A

&amp;H

00-07	RLC	register	80-87	RES 0,	register
08-OF	RRC		88-8F	RES 1,	
10-17	RL		90-97	RES 2,	
18-1F	RR		98-9F	RES 3,	
20-27	SLA		A0-A7	RES 4,	
28-2F	SRA		A8-AF	RES 5,	
30-37	?		B0-B7	RES 6,	
38-3F	SRL		B8-BF	RES 7,	
40-47	BIT 0,	register	C0-C7	SET 0,	register
48-4F	BIT 1,		C8-CF	SET 1,	
50-57	BIT 2,		D0-D7	SET 2,	
58-5F	BIT 3,		D8-DF	SET 3,	
60-67	BIT 4,		E0-E7	SET 4,	
68-6F	BIT 5,		E8-EF	SET 5,	
70-77	BIT 6,		F0-F7	SET 6,	
78-7F	BIT 7,		F8-FF	SET 7,	

Z - 8 0 C O D E

11-11-86

Charles van der Linden &amp; Zn.

Broederhof 11 - 5504 JC Veldhoven

## linkerscherf

kolom &H	0	1	2	3	regel
1 5000 20480	0123456789	0123456789	0123456789	0123456789	20519 5027 1
2 5050 20560	.....	.....	.....	.....	20599 5077 2
3 50A0 20640	.....	.....	.....	.....	20679 50C7 3
4 50F0 20720	.....	.....	.....	.....	20759 5117 4
5 5140 20800	.....	.....	.....	.....	20839 5167 5
6 5190 20880	.....	.....	.....	.....	20919 51B7 6
7 51E0 20960	.....	.....	.....	.....	20999 5207 7
8 5230 21040	.....	.....	.....	.....	21079 5257 8
9 5280 21120	.....	.....	.....	.....	21159 52A7 9
10 52D0 21200	.....	.....	.....	.....	21239 52F7 10
11 5320 21280	.....	.....	.....	.....	21319 5347 11
12 5370 21360	.....	.....	.....	.....	21399 5397 12
13 53C0 21440	.....	.....	.....	.....	21479 53E7 13
14 5410 21520	.....	.....	.....	.....	21559 5437 14
15 5460 21600	.....	.....	.....	.....	21639 5487 15
16 54B0 21680	.....	.....	.....	.....	21719 54D7 16
17 5500 21760	.....	.....	.....	.....	21799 5527 17
18 5550 21840	.....	.....	.....	.....	21879 5577 18
19 55A0 21920	.....	.....	.....	.....	21959 55C7 19
20 55F0 22000	.....	.....	.....	.....	22039 5617 20
21 5640 22080	.....	.....	.....	.....	22119 5667 21
22 5690 22160	.....	.....	.....	.....	22199 56B7 22
23 56E0 22240	.....	.....	.....	.....	22279 5707 23
24 5730 22320	.....	.....	.....	.....	22359 5757 24

## rechterscherf

	4	5	6	7	
	0123456789	0123456789	0123456789	0123456789	
1 5028 20520	.....	.....	.....	.....	20559 504F 1
2 5078 20600	.....	.....	.....	.....	20639 509F 2
3 50C8 20680	.....	.....	.....	.....	20719 50EF 3
4 5118 20760	.....	.....	.....	.....	20799 513F 4
5 5168 20840	.....	.....	.....	.....	20879 518F 5
6 51B8 20920	.....	.....	.....	.....	20959 51DF 6
7 5208 21000	.....	.....	.....	.....	21039 522F 7
8 5258 21080	.....	.....	.....	.....	21119 527F 8
9 52A8 21160	.....	.....	.....	.....	21199 52CF 9
10 52F8 21240	.....	.....	.....	.....	21279 531F 10
11 5348 21320	.....	.....	.....	.....	21359 536F 11
12 5398 21400	.....	.....	.....	.....	21439 53BF 12
13 53E8 21480	.....	.....	.....	.....	21519 540F 13
14 5438 21560	.....	.....	.....	.....	21599 545F 14
15 5488 21640	.....	.....	.....	.....	21679 54AF 15
16 54D8 21720	.....	.....	.....	.....	21759 54FF 16
17 5528 21800	.....	.....	.....	.....	21839 554F 17
18 5578 21880	.....	.....	.....	.....	21919 559F 18
19 55C8 21960	.....	.....	.....	.....	21999 55EF 19
20 5618 22040	.....	.....	.....	.....	22079 563F 20
21 5668 22120	.....	.....	.....	.....	22159 568F 21
22 56B8 22200	.....	.....	.....	.....	22239 56DF 22
23 5708 22280	.....	.....	.....	.....	22319 572F 23
24 5758 22360	.....	.....	.....	.....	22399 577F 24
25.5780 22400	r 57A8 22440	26.57D0 22480	r 57F8 22520-22527	(57FF)	

## A S C I I - C O D E

T.008

33	34	95	36	37	38	39	40	41	61	96	125	11	92	42	12	
27	49	50	51	52	53	54	55	56	57	48	45	123	8	45	43	15
CODE	1	2	3	4	5	6	7	8	9	0	-	wis	-	+		
25	81	87	69	82	84	89	85	73	79	80	94	91	129	7	14	
9	113	119	101	114	116	121	117	105	111	112	64	93	55	56	57	
TAB	q	w	e	r	t	y	u	i	o	p	@	]	7	8	9	
65	83	68	70	71	72	74	75	76	43	42	127		7	133	7	
97	115	100	102	103	104	106	107	108	59	58	35	13	52	53	54	
a	s	d	f	g	h	j	k	l	;	:	#	Return	4	5	6	
62	90	88	67	86	66	78	77	44	46	63			128	7	131	
60	122	120	99	118	98	110	109	44	46	47			49	50	51	
<	z	x	c	v	b	n	m	,	.	/			1	2	3	
29	31												130	5	3	
16	17												48	48	46	
-----cursortoetsen-----																
0													0	00	,	

CHR\$ ( ... )															
0 einde toonstring	15	wis tot cursorpunt	131	geel	147	gr.geel									
1 cursor aan	16	cursor naar links	132	blauw	148	gr.blauw									
2 cursor uit	17	cursor omhoog	133	paars	149	gr.paars									
3 OUT 48,0 + CHR\$(29)	18	cursor omlaag	134	cyaan	150	gr.cyaan									
4 pos.cursor+chr\$ R+K	19	cursor naar rechts	135	wit	151	graf.wit									
5 scherm naar printer	20	cursor op kolom k	136	knipperen											
6 cursorpunt huidige	21	wis tot einde regel	137	niet knipperen											
7 piep	22	wis - einde venster	140	normale hoogte											
8 cursor naar links	23	begin toonstring	141	dubbele hoogte											
9 horizontale TAB	24	cursor naar rechts	152	verbergen											
10 cursor omlaag	28	venster 24 r. 40 k.	153	grafisch aaneen											
11 wis karakter links	29	CHR\$(13) + CHR\$(10)	154	gescheiden grafisch											
12 wis venster	30	deel-PRINT uit	156	zwarte achtergrond											
13 cursor linker kant	31	cursor op cursorpunt	157	gekleurde achtergr.											
14 deel-PRINT	129	rood 145 graf.rood	158	herhaal graf.kar.											
	130	groen 146 gr.groen	159	zet uit CHR\$(158)											

## T O E T S - C O D E

104	118	135	76	79	77	73	78	126	112	117	119	140	116	115	114	112
32	46	63	4	7	5	1	6	54	41	45	47	68	44	43	42	40
CODE	1	2	3	4	5	6	7	8	9	0	-	wis	-	+		
80	75	107	108	111	109	105	110	145	121	125	127	132		123	122	120
8	3	35	36	39	37	33	38	70	49	53	55	60		51	50	48
TAB	q	w	e	r	t	y	u	i	o	p	@			7	8	9
106	83	84	87	85	81	86	134	137	141	143	92	124		139	138	136
34	11	12	15	13	9	14	62	65	69	71	20	52		67	66	64
a	s	d	f	g	h	j	k	l	;	:	#	Return		4	5	6
98	82	99	100	103	101	97	102	94	129	133				131	130	128
26	10	27	28	31	29	25	30	22	57	61				59	58	56
<	z	x	c	v	b	n	m	,	.	/				1	2	3
72	74													91	90	88
0	2													19	18	16
-----cursortoetsen-----																
0													0	00	,	