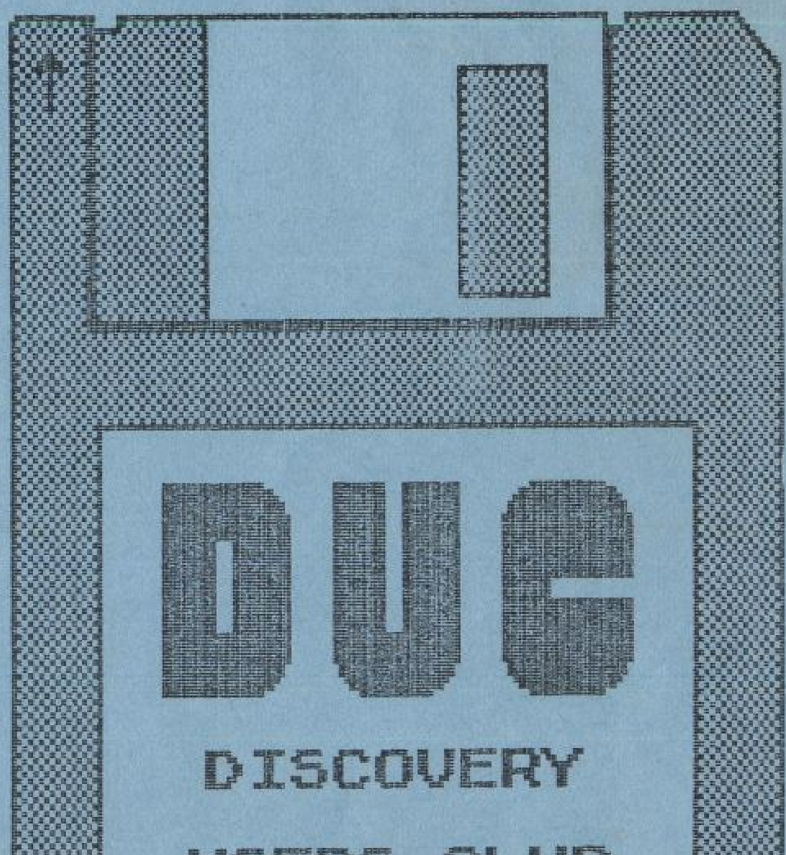
DISCOVERY Users Club Magazine

Feb. '87 No. 9



D.U.C.

Het magazine van de Nederlandse DISCOVERY USERS CLUB

BESTUUR

VOORZITTER : R.O. AALDERS (RUDIE)
SECRETARIS : D.C. KRUITHOF (DICK)
LEDEN : M.G. KUKLEWSKI (MIKE)
: P.J.M. HOPMANS (PETER)

SECRETARIAAT & LEDENADMINISTRATIE

p/a P.J.M. Hopmans, Ina Boudier-Bakkerlaan 41-1, 3582 VG Utrecht

REDACTIE CLUBBLAD & ADVERTENTIEADRES

p/a D.C. Kruithof, Boeierkade 6, 2725 CH Zoetermeer

KORRESPONDENT BELGIE

Peter Hopmans, Ina Boudier-Bakkerlaan 41-I, 3582 VG Utrecht

HULPLIJN & SERVICE-ADRESSEN

R. Macare(19.00-23.30)	R.O. Aalders	D.C. Kruithof
Gaesbeekstraat 77b	Furkabaan 625	Boeierkade 6
3081 NE Rotterdam	3524 ZE Utrecht	2725 CH Zoetermeer
tel: 010-4862184	(schriftelijk)	(schriftelijk)

Na 1 April: niet meer naar Rob bellen, maar naar John Koster, Repel 93, 3224 VE Hellevoetsluis, tel. 01883 - 12475

BETALINGEN

Alle betalingen dienen te worden overgemaakt op gironummer: 5241782 t.n.v. :R.O. Aalders, Furkabaan 625, 3524 ZE Utrecht.

NABESTELLINGEN CLUBBLAD

Oude nummers kunnen worden besteld door overmaking van f6,-- voor DUC #1 en f7,50 voor alle andere nummers. Portokosten f1,85 per blad!

ADVERTENTIES

In het magazine kunnen onder de volgende voorwaarden advertenties worden opgenomen:
-1- ze moeten betrekking hebben op SINCLAIR/OPUS produkten;
-2- kommerciele advertenties dienen vooruit te worden betaald, uitgaande van f35,-- per pagina A5 (abonnementen mogelijk);
-3- clubleden mogen gratis een oproep plaatsen onder de rubriek 'DISCOVERIES'.

AANSPRAKELIJKHEID

Redactie noch bestuur kunnen aansprakelijk worden gesteld voor schade ontstaan door ingezonden programma's, tips etc. Bestuur noch redactie zijn verantwoordelijk voor het via het clubblad overtreden van COPYRIGHT door de leden.

COPYRIGHT

INHOUDSOPGAVE

COLOFON pagina 1
Let op: er is het een en ander veranderd!!!

REDACTIONEEL pagina 3
Onrust in DISCOVERY land

INGEZONDEN BRIEVEN pagina 4
Tips en vragen van leden

DUBBELDIK pagina 8
Een routine om characters op scherm dikker te maken

ALGEMEEN SCREEN$COPY (CORRECTIE) pagina 9
Dit programma wordt nu met recht algemeen

ALGEMEEN MACHINE CODE LADER (2) pagina 11
Reactie op een artikel in het vorig nummer

RANDOM ACCESS FILES pagina 12
Joop van Brummelen legt uit hoe dat gaat...

CAT MOVER pagina 15
Een snelle manier om gewenste files te MOVEn

CAT IN DRIE KOLOMMEN pagina 16
Een keurige CAT op het scherm

BREAK-OUT pagina 22
Nu kunt u uit alle programmas BREAKen

PROGRAMEREN MET PASCAL pagina 28
Rik Simons wijst ons de weg...

GESTRUCTUREERD PROGRAMEREN (DEEL 2) pagina 31
Norbert Simoens vervolgt zijn verhaal...

ULTI-MATE pagina 31
Het ultieme RUN programma?

CALPHY OPNIEUW BEZOCHT pagina 37
Rudie vertelt nog het een en ander over zijn routine

PROGRAMMA-BESPREKING pagina 43
Een editor/assembler/disassembler uit Duitsland

SPEELHOEK: BESPREKING pagina 46
Een jeugdig lid bespreekt zijn favorite spelletjes

PROGRAMMABANK NIEUWS pagina 48
Wat er is, wat het doet, en hoe u er aankomt.

DISCOVERIES pagina 50

OPUS DISCOVERY: DOOD OF LEVEND? Redactioneel)

Ik wet zeker dat we allemaal wel IETS gehoord hebben over de perikelen rond de DISCOVERY. Een paar maanden geleden dachten wij allen dat de DISCOVERY een nieuw standaard opslag-medium zou gaan worden voor de SPECTRUM. De prijs was aantrekkelijk geworden, er kwam steeds meer software, en zelfs SAGA -bekend om hun mooie toetsenborden- besloot om een kompleet systeem rond de DISCOVERY te bouwen..... 'THE COMPLEMENT'. Het zou niet zo zijn. Er kwam ongein. De eerste gerucht luidde dat de produktie van de DISCOVERY in handen zou komen van SAGA of TRANSFORM. Toen was er opeens een 'Spaanse Konnektie'; INVESTRONICA, de mensen die de eerste 128 SPECTRUM op de kontinent introduceerde, zouden op het punt staan de produktie van de DISCOVERY in licentie te krijgen. Ondertussen werd onze favorite disc drive een verzamelaars-item; moeilijk om aan te komen. Anders betrouwbare leveranciers klaagden over slechte levering uit Engeland, en hadden geen voorraad meer. Toen was het bruine spul pas écht aan de knikker! De geruchten bleken geen geruchten te zijn. Officiële berichten van OPUS luidde dat produktie inderdaad was stilgezet, en dat de voorraad verkocht werd. Meteen stegen in de winkels de prijzen, en lege beloften van leveranciers werden gemeengoed. Ik vraag me af wat het laatste woord in deze OPUS sage (grapje!) zal zijn. Het laatste wat wij hoorden was dat een of ander idioot met een Copyright op een rommetje de zaak tegen hield. Wist hij maar wat een markt er is voor deze machine.....

Ondertussen zijn er binnen het bestuur ook verwikkelingen geweest. Onze hulplijn en buitenlands-korrespondent, Rob, heeft besloten zijn systeem aan de wilgen te hangen en met ander materiaal verder te gaan (overloper!). Wij wensen hem voorspoed en plezier met zijn niuew speelgoed, en danken hem voor bewezen diensten (!) Het overige bestuur besloot om meteen wat taken anders te verdelen. Mike, samensteller van het blad, gaat nu ALLE Engels-talige werk van de club verrichten. Het vertalen van het blad, buitenlandse leden en kontakten met hard- en software-makers rusten nu op zijn schouders. Dick neemt de redactie over. LET DUS AUB. OP HET GEWIJZIGDE COLOFON!!!

Zo onzeker als het voortbestaan van de DISCOVERY mag zijn, zo zeker en betrouwbaar blijft het DUC!!! Om dat te bewijzen, ligt vr u weer een dik nummer vol interessante artikelen. Voor beginners zijn er wat artikelen over programeren. Ook zijn er nieuwe toepassingsprogrammas van leden, programmabesprekingen en het laatste nieuws van het programmabank. Over deze service van de klub gesproken.... u kunt zich niet voorstellen hoeveel mensen niet weten hoe ze een programma moeten bestellen! Eenvoudig: maak het juiste bedrag over op onze girorekening, met vermelding van uw lidnummer en het gevraagde artikel. U ontvangt het bestelde zodra wij een bewijs van storting ontvangen van de giro.

Het volgende nummer zal voor het grootste deel bestaan uit besprekingen van hard- en software. Zo heeft bijv. BRADWAY SOFTWARE 4 programma's ter beschikking gesteld. Een bevriend handelaar in Gouda stelt ook het een en ander ter beschikking, dus dat word smullen!!! Tot dan.

BRIEVEN

Peter Adriaanse uit Eindhoven stuurde de volgende twee tips in.

1. Het herstellen van sectoren.
Gebeurt het u ook weleens dat een I/O error in een bepaalde file programma's en soms hele discs onleesbaar maakt? Met een tweetal utilities die onze club de leden aanbiedt, ben ik in staat geweest ettelijke files en zelf hele discettes die door een fout in de catalogue file onleesbaar waren geworden, toch weer te herstellen. Wat moet u doen?

Laad allereerst het programma 'SECTORTEST' in, dat op DUC-disc 1 staat. RUN dit met de te behandelen disc. Noteer vervolgens de sector waar een fout optreedt. U kunt dan proberen om met de uitstekende LINK II deze sector te herstellen. Reken daartoe de opgeschreven sector om in de door LINK vereiste notatie. Dat gaat als volgt. Bereken (eventueel in ZX BASIC) het tracknummer (=INT (sector/18)) en het blok in het berekende track (=sector-(INT(sector/18)). Deze twee getallen dan in de hexadecimale vorm omzetten en invoeren in de 'DISC EDITOR' van LINK II. Probeer vervolgens deze sector te lezen met de READ optie. Waarschijnlijk krijgt u dan een foutmelding. Blijf echter net zolang pogen totdat het wel lukt. Meestal slaag ik binnen tien pogingen. Schrijf dan het stukje code weg met behulp van het WRITE commando. Verlaat het programma en test de herstelde disc uit. De kans is nu zeer groot dat alles weer werkt!

2. MCODER II OP DE DISCOVERY.
We weten dat het programmeren in machinetaal grote voordelen biedt. Het grote pluspunt van BASIC, en dan met name gecompileerde BASIC, is de veel betere gebruikersvriendelijkheid. Ik probeer zelf altijd trage programma's te compileren alvorens ik in de 'bits & bytes' duik. De compiler MCODER II die ik voor het compileren van BASIC programmatuur gebruik, is al enige jaren verkrijgbaar, maar werkt niet geheel feilloos met de DISCOVERY. Bij het optreden van fouten tijdens het vertalen wil de boel weleens vastlopen. Daartoe schreef ik de volgende routine die de SPECTRUM in zo'n situatie weer aan de praat kan krijgen. Het is een interrupt-gestuurde routine die machinetaal programma's (en dus ook MCODER II) kan BREAKen. Zie ook artikel 'BREAK-OUT. Het kan geactiveerd worden door op de toetsen SYMBOL SHIFT en SPACE te drukken. De listing ziet er als volgt uit:

```
  6 REM ** Mcoder II BREAKER **
  7 REM adres 59367 tot 59919
  8
 10 RESTORE
 15
 17 REM * maak vectortabel aan **
 18
 20 FOR f=59648 TO 59904: POKE f,231: NEXT f
 30
 35 REM ** Interrupt service routine **
 36
```

```
40 FOR f=59367 TO 59413: READ a: POKE f,a: NEXT f
50 DATA 255, 243, 245, 197, 213, 229, 1, 254, 127, 237, 120
   254, 252, 4, 2, 24, 2, 24, 24, 62, 56, 50, 141, 92, 1,
   254, 127, 237, 120
55 DATA 254, 252, 40, 247, 251, 207, 20, 62, 9, 237, 71, 237
   94, 201, 255, 209, 193, 241, 251, 201
60
70 REM **Opstart routine **
75
80 FOR f=59910 TO 59916: READ a: POKE f,a: NEXT f
90 DATA 62, 233, 237, 71, 237, 94, 201
95
97 REM ** Aktiveer **
98
99 RANDOMIZE USR 59910
```

Dit programma moet geRUNd worden na het inladen van de compiler. Als u de computer NEWt om het te compileren programma in te laden of in te tikken, dient de RANDOMIZE USR 59910 nog een keer extra te worden uitgevoerd. Als tijdens het compileren een fout wordt gekonstateerd door Mcoder II, is de kans groot dat de zaak vastloopt. Op dat moment is het tijd voor de routine. Houd SYMBOL SHIFT ingedrukt en druk op SPACE. Het scherm wordt schoongemaakt en het vertrouwde cursor verschijnt weer. De fout in het BASIC-programma kan weer hersteld worden en de compiler weer opgestart. De werking van de routine kan ook worden gedemonstreerd door een zogenaamde automatische listing - verkregen door op ENTER te drukken- trachten te onderbreken. Ik hoop dat er nog Mcoder II gebruikers zijn, die van mijn routine profijt hebben.

Peter Adriaanse
Eindhoven.

Deze tips komen van B. Wesselius uit St. Maartensbrug.

- Als je over een groot toetsenbord beschikt, en je SPECTRUM zit niet aan je DISCOVERY vastgeschroefd, kijk dan alsjeblieft uit dat je niet wiebelt met je toetsenbord t.o.v. je drive, want dan kunnen er vitale delen stukgaan. Ook de foutmelding DISK I/O ERROR komt dan (als het goed is) niet meer op je scherm.

- Mocht die foutmelding toch komen, en je krijgt niets meer van de diskette af (of er komt een gebibber uit de drive), zet alles dan uit en maak de randkonnektor met een schone gum schoon. Weer monteren en opnieuw proberen.

- Ook kan het zijn dat de drive nog te koud is. Als de verwarming uit staat, en het is in je kamer ca. 10 graden Celsius dan moet je de drive eerst opwarmen. Zet alles gewoon aan en wacht een paar minuten. Niet alles meteen in de huiskamer zetten, want dan heb je kans op kondens op vitale delen in de drive. Ook na vervoer in de auto etc. (Ik was zo stom om de hele

boel in een fietskarretje achter mijn brommer te hangen). Wacht tot hij op temperatuur is alvorens je hem aanzet. Aanzetten en wachten tot-ie het doet.

- Als dat ook niets helpt zit de fout in de drive: De stappenmotor van de leeskop is door stoten (vervoer) verdraaid, zodat de leeskop te ver buiten de catalog (buitenste rand) van de schijf staat en dus geen CAT meer kan vinden. De schijf gedraagt zich dus als niet-geformatteerd. FORMAT 1;"naam" had dus niet een DESTROY "..."? tot gevolg, en een regelmatig getik zei me dat ik alles kwijt was, want de schijf werd opnieuw ingedeeld en voor zover ik weet is dan ook alles weggepoetst. Deze fout had ik opgelost door alles open te schroeven (weg garantie!) en de stappenmotor met het asje een zetje te geven naar het midden, terwijl ik met de andere hand CAT 1 intypte. DIT IS NIET GOED VOOR DE DRIVE!!! Beter is gewoon niets open te maken, maar een lege diskette erin te doen, en deze dan FORMATteren. Halverwege (na plm. 20 tikjes) kan je dit afbreken met BREAK, met als effekt dat de leeskop ergens in het midden van de diskette staat. Je kan hem ook door laten gaan, maar dat duurt alleen langer. Na dit proces haal je deze TEST-DISK eruit (ik heb dus een speciale schijf hiervoor) en stop je de pseudo-korrupte schijf er weer in. Na een CAT schuift de leeskop weer naar de rand en komt onderweg de CATALOG file tegen. De diskette is nu weer te lezen.

- Als ook dat niet helpt staat er inderdaad ergens een tik op je schijf, wat door een schroevendraaier (magnetisch) of vuil (sigaretterook!) kan komen. De oplossing is dan: stuk voor stuk de files proberen te MOVEn naar een veilige schijf. De rest is verloren... tenzij je een block-editor hebt.

- Om beschadiging van de programma's te voorkomen, kan je om te beginnen je schijven netjes in een bakje plaatsen. Als je wat eet of drinkt, houdt dan schone handen! Om ongelukken te vermijden: zet koffie of bier (komt vaker voor dan je denkt) niet in de buurt. Pak geen schijfjes met vette patat-handen uit de bak. Ook huisgenoten mogen geen etenswaren in de buurt hebben; een bakje koffie stoot je gauw om als je je omdraait om even op een ASSEMBLER-lijst te gluren wat ook al weer DEC BC was.

- Dan heb ik zelf nog een vreemde gewoonte: Na het laden van een programma druk ik altijd meteen de diskette uit de drive, zodat de schijf ca. 2 cm naar voren komt. Mocht de stroom uitvallen, of je trekt aan je toetsenbord, dan gebeurt er tenminste niets met je schijf. Als het programma iets wil laden, kan je altijd nog de diskette erin duwen, ook als het LEDje al brand om aan te geven dat de motor draait. Dit doe ik heel vaak, en er gaat echt niets stuk. Als het niet hoeft dan moet je het natuurlijk ook niet express doen.

- Laat alsjeblieft NOOIT de diskette erin glijden met een luide POOIING, maar houdt met je duim het knopje vast en laat deze naar voren komen als je met je wijsvinger zachtjes de schijf

erin duwt. Dit is veel beter voor zowel de drive als de schijf. Ook het eruit halen kan beter door de knop in te drukken met de duim, en de diskette op te vangen met de wijsvinger. Dit voorkomt dat de diskette te snel omhoog komt, met het gevolg dat het inwendige mechanisme weer daalt en de diskette klem zet. Als dit toch gebeurt, druk dan gewoon de knop weer in.

- Het eruit halen van de schijf na het laden heeft nog een voordeel: Tijdens het kraken (overzetten van TAPE naar DISK) probeerde ik het zojuist gemuteerde programma met een RANDOMIZE USR-call. Het gevolg was dat het LEDje van de drive ging branden en de motor ging lopen. Mijn schijf zat er toen nog in. Wat leuk, dacht ik, een TAPE-BASED programma dat gaat laden van schijf. Maar dat laden was geen laden; mijn gezicht betrok toen ik een zacht tik-tik-tik hoorde. Hij was mijn schijf aan het FORMATteren! Door een fout in de machinetaal sprong de routine blijkbaar ergens in het DISCOVERY-ROM, alwaar het vrolijk (?) bezig was mijn zojuist gekonverteerde programma's (5 stuks) te vernaaien. Ik kon me wel voor mijn kop slaan. Dat deed ik dus ook. Het mocht echter niet baten. Later is het weer gebeurd dat het LEDje ging branden en zo. De schijf was er toen uit. Zoals je zult begrijpen, was ik niet zo'n heel klein beetje blij...

- Dan kan je nog iets doms doen. MICRDRIVE-bezitters weten het al: Nooit je systeem aan of uitzetten als de cartridge er nog in zit. Dat geldt in mindere mate voor de DISCOVERY. De handleiding rept met geen woord over deze situatie. Toch heb ik het geprobeerd met een schijf. Erin laten zitten, en 20x uit en aan zetten (arme schakelaar). Het gevolg was dat slechts 1 file onleesbaar was geworden, nl. die waar de leeskop stond. Volgens mij mag dit niet gebeuren, maar de feiten liegen er niet om. Dus eerst eruit halen en dan uitzetten.

- In mijn onwetendheid heb ik ook wel eens tijdens het laden, saven, compacteren (heet dat zo?) het licht in mijn kamer aan- of uitgedaan. Dat was dom, want net als bij het laden of saven van tape kan er dan een fout ontstaan, die alles of in ieder geval iets kan mollen. Ook als de wasmachine, koelkast, diepvriezer, oven, broodrooster, grill en toaster opeens plotseling en tegelijkertijd aanfloepen, en als die dan ook nog allemaal op dezelfde groep staan (verboden!!), en als je computer dan ook nog op die groep staat, zou het kunnen gebeuren dat tengevolge van de spanningsval van het net, je programma crasht, of op z'n minst een rij bits omklapt, met alle nare gevolgen van dien. Denk daar dus om! Dit lijkt zeer onwaarschijnlijk, maar bij mij thuis is het zo dat ik mijn slaapkamerlicht wel met het touwtje aan mag doen, maar niet met de schakelaar bij de deur. Een vreemde situatie, maar het is niet anders. Een netfilter zou misschien goede diensten bewijzen. Of de schakelaar bij de deur vervangen.

Deze dingen heb ik dus allemaal een keer (of vaker) meegemaakt, en diverse programma's zijn dan ook verloren gegaan. Het ergste is dat je niet eens weet WELKE programma's je kwijt bent! Ik had een schijf vol zelfgebrouwde MC-routines (o.a.). Nadat het fout

ging wist ik niet eens wat ik verloren had. Pas weken later ontdekte ik dat de RENUMBER-routine weg was, en allerlei andere frutseltjes, die toch heel leuk waren. Bedenk ook, dat al je programma's bij elkaar duurder en zeker waardevoller zijn dan de drive zelf. Ik ben liever mijn drive voor een week kwijt, dan al mijn programma's!! De drive zou je ooit nog eens nieuw kunnen kopen, maar je programma's niet. Sommige, zoals TASWORD zou je nog kunnen kopieëren (dat mag, want je eerste versie had je wel gekocht!), maar anderen had je misschien overgetypt uit een blad, of zelf gemaakt. Die ben je pleite, zoals dat heet...

Als andere gebruikers ook tips hebben, wil ik vragen of zij die ook willen publiceren, want daar heb je wat aan. Ik tenminste wel! Ook aanmerkingen op mijn tips zijn welkom, want ik ben van huis uit geen elektronika-figuur, en deze tips zijn dus alleen uit ervaring samengesteld. Mocht ik iets finaal fout hebben verteld, zeg dat dan! We worden er alleen maar beter van...

B.C.M Wesselius
Ruigeweg 37
1752 HB St. Maartensbrug
02246-1570 (19.00-21.00)

<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

DUBBELDIK

Tijdens het lezen in een bekend computertijdschrift, namelijk ZX COMPUTING, viel mijn oog op (gelukkig figuurlijk) een artikel/programma dat mij wel de moeite waard leek om in te tikken. De schrijver van het programma is Ben Stagnell, 13 jaar, uit Radway (UK). Het verandert alle 96 Spectrum karakters in dubbeldikke letters van zeer mooie kwaliteit. Zullen we maar beginnen met het programma in te tikken?

```
  5 REM
 10 REM * DUBBELDIKKE LETTERS *
 20 REM
 30 LET t=0
 40 FOR n=30000 TO 30029: READ a: LET t=t+a: POKE n,a: NEXT n
 50 DATA 33,0,61,17,0,118,1,0,3,237,176,33,0,118,17,0,3,126,79,
    203,63,177,119,35,27,122,179,200,24,243
 60 IF t<>2414 THEN LIST 50: PRINT '' FLASH 1;"KONTROLEER DATA
    IN REGEL 50 !!!!": RESTORE : STOP
 70 RANDOMIZE USR 30000
 80 INPUT "Programmanaam ? ";LINE n$: INPUT "Drivenr. ?";d
 90 CLS: PRINT "Saving :";n$
100 SAVE *d;n$ CODE 30208,768 : VERIFY *d;n$CODE
110 POKE 23606,0: POKE 23607,117
120 BEEP .1,10: PRINT n$;" staat op disk"
```

Als u nu met dubbeldikke letters wilt gaan werken, moet u de CODE laden (LOAD *1;" -naam- "CODE) en daarna twee POKEs uitvoeren: POKE 23606,0 en POKE 23607,117. Om de normale letters weer terug te krijgen, volstaan POKE 23606,0: POKE 23607,60.

Het is ook mogelijk om de CODE (768 bytes) op een andere plek in het geheugen te zetten. U moet dan als volgt handelen:

1) CLEAR adres-1, d.w.z. als u de CODE vanaf 60000 wilt hebben, moet u vooraf een CLEAR 59999 geven.
2) Laad CODE in vanaf adres 60000: LOAD *1;" -naam- "CODE 60000
3) De twee POKEjes worden een beetje anders:
   POKE 23606,adres-256*INT(adres/256):
   POKE 23607,INT(adres/256)-1
   ('adres' moet dus bv. veranderd worden in 60000)

Dick Kruithof

‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹››››››››››››››››››››››››››››››››››››››

EXTRA CORRECTIE VOOR DE ALGEMENE SCREEN$COPY

Degenen die een LSB-printer hebben zullen na het intypen van de correctie uit DUC#6 gemerkt hebben dat de grote COPY niet werkt. Na contact te hebben gehad met Rudie stuurde hij mij een routine om uit te proberen. Om de routine helemaal geschikt te maken moeten de volgende regels worden veranderd:

- In regels 7 en 17 moet de 'RANDOMIZE USR 41000' weg.
- In regels 10 en 19 moet achteraan worden toegevoegd: RANDOMIZE USR 41000
- In regel 9998 moet de dataregel worden veranderd in: 1,128,1, 33,212,92,126,197,6,8,203,39,203,30,16,250,193,35,11,120,177, 32,239,201
- In de code moeten twee POKE's worden ingevoerd: 40139,176 en 40143,68

Een andere verbetering in de screencopy is de screendisplay. Deze wordt nu niet meer vervormd. Dit programma van Rudie kan nu met recht 'Algemene Screen Copy' worden genoemd.

Marco Baayen

‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹››››››››››››››››››››››››››››››››››››››

DISCOVERIE

* Sinds kort heeft de DUC een eigen prikbord in een databank, naast 20 paginas nieuws. Deze databank heet MIAMITEL en is te bereiken door het volgende nummer te draaien: 023 - 355949. Eenmaal ingelogd, zoekt u pagina 10.000.000 op, waar gemiddeld om de twee weken nieuwtjes worden geplaatsd.

TE KOOP: IC 6116

Vindt u dat nou niet zonde; al die mooie programma's in ons blad en programmabank waar u niets aan heeft omdat u geen IC 6116 in uw Discovery heeft zitten ? Welnu, hier kan iets aan gedaan worden! Stort Fl 12,50 op gironr. 2866503 t.a.v. D.C.Kruithof te Zoetermeer (onder vermelding van '6116') en u krijgt zo'n IC met inbouwbeschrijving per post thuisgestuurd. Ook is er de mogelijkheid om het IC op een DUC-dag in te (laten) bouwen. De kosten zijn dan Fl 10,00, maar u moet dan wel uw Discovery meesjouwen naar Utrecht.

OPROEP

Al sinds jaar en dag wordt er in Amsterdam op de eerste zaterdag van de maand een bijeenkomst georganiseerd voor gebruikers van SINCLAIR computers. De opzet van deze bijeenkomsten is simpel maar doeltreffend getuige de stabiele populariteit door de maanden heen. Vanaf november'86 zijn er nu ook een aantal DISCOVERY-gebruikers op deze dagen te vinden die van de gelegenheid gebruik maken om ideeen en vooral schijfjes uit te wisselen.

Vanwege het pas uitkomen van het programma "THE ARTIST II" en de beschikking over de "VIDEOFACE" Videodigitiser en een video-camera hadden wij gedacht de populariteit van deze bijeenkomsten wat te verhogen door een 'Grafische dag' te organiseren met o.a. demonstraties van eerdergenoemde soft- en hard-ware. Ook zal misschien aandacht geschonken worden aan de eerste stappen op weg naar een geintegreerd pakket voor de DISCOVERY.

Wij willen iedereen dan ook oproepen om op Zaterdag 4 april naar De Koperen Knoop aan de van Limburg-Stirum straat 119 te Amsterdam te komen. De zaal is open vanaf 11.00 t/m 16.00 en de toegang bedraagt fl 2.50. Voor meer informatie kunt u bellen met Wiro van Schaik (tel 020-797440)

Met vriendelijke groeten

Wiro van Schaik
namens het comitee "THE ARTISTs 2"

Wederom: ALGEMENE MACHINE CODE LADER

Er zijn drie 'klachten' binnengekomen m.b.t. de algemene mc-lader. Ik zal ze behandelen, en daarna de definitieve vereenvoudigde versie geven.

1. Ik wil onderwijl stoppen (gezien de lange code-blokken wel eens wenselijk). Hoe doe ik dat?

Ik zal hiertoe een optie inbouwen in de vereenvoudigde versie. Door eenvoudigweg het keyword 'STOP' in te toetsen springt u uit de loop.

2. Waarom gebruikt het programma kleine letters?

Het maakt voor het programma zelf niets uit of u grote dan wel kleine letters gebruikt. De SPECTRUM functie 'VAL' herkent zowel de 'a' als de 'A' als zijnde een variabelenaam.
Ik stond voor de keuze kleine OF grote letters te gebruiken. Hoofdletters worden het meest gebruikt, em meestal voeg ik mij naar die gewoonte. Deze keer was ik echter dwars.
De reden hiervoor is technisch. Aangezien ons blad in A-5 formaat wordt gedrukt, worden twee A-4'tjes verkleind tot EEN. Bij dit verkleiningsproces hoeft er maar iets in de weg te liggen -een haartje, een vlek op het glas, een te lichte pagina- of dat kan al funest zijn. Een 8 gaat dan al gauw op een B lijken, of omgekeerd. Door kleine letters te gebruiken hoop ik dat te voorkomen, al sluipt er ook bij mij weleens een 'v' voor een 'c' binnen, of '15' waar er '16' hoort te staan.

3. We begrijpen geen sikkepit van die lader!

Daar kan ik in komen.

Nu de vereenvoudigde versie van de lader.
‹begin› en ‹lengte› moeten in regel 10 worden ingegeven.

```
5 CLEAR 27499
10 LET BEGIN=...: LET LEN=...
15 LET a=10:LET b=11: LET c=12: LET d=13: LET e=14: LET f=15
20 FOR z=0 TO LEN-1 STEP 16
25 LET TOT =0
30 PRINT z+BEGIN;" "
35 INPUT; LINE b$
40 IF LEN b$=1 THEN IF b$="STOP" THEN GOTO 80
45 IF LEN b$<>36 THEN BEEP .5,10: GOTO 35
50 FOR y=0 TO 15
55 LET p=16*VAL b$(1) + VAL b$(2):LET b$ =b$ (3 TO )
60 POKE (z+BEGIN+y),p:LET TOT = TOT+p
65 NEXT y: IF TOT = VAL b$ THEN GO TO 75
70 FOR y=20 TO -20 STEP 4:BEEP .25,y: NEXT y
71 PRINT "FOUT": GOTO 25
75 PRINT b$: NEXT z
80 SAVE *1;"naamppje"CODE BEGIN,LEN
```

Rudie

RANDOM ACCESS FILE

Met dit programma wil ik wat inzicht bieden over de werking van een Random Access File (RAF). Zoals ook in het handboek staat, een RAF wordt gecreëerd met OPEN# en RND. Bijvoorbeeld:

```
OPEN# 5;"m";2;"NAMEN"RND 20,10
```

Hiermee wordt de RAF-file "NAMEN" gecreëerd, die op drive 2 staat. Er wordt informatie van en naar de drive gestuurd via stream-nummer 5. Ieder record is 20 bytes (karakters) lang en er zijn 10 records.

Wanneer we niet weten -of niet willen aangeven- uit hoeveel records het bestand (de RAF-file) zal bestaan, dan dienen we dat aan te geven door de specificatie 20,-1. De lengte van de RAF-file is nu de helft van de grootst aanwezige vrije ruimte op de disk. Een RAF-file op een lege disk (178K) beslaat dus 89K.

De recordlengte is geDIMensioneerd op 20 karakters, DIM N$ (20); het systeem stuurt door ‹ENTER› een byte mee, vandaar dat het opzetten van de RAF-file "NAMEN" als volgt gaat:

```
OPEN#4;1;"NAMEN"RND21,-1
```

"m"; mag je dus laten vervallen.

Volgens het handboek zou USR 432 het einde van de file (EOF) bijhouden. De werking hiervan heb ik niet kunnen ontdekken. Daarom heb ik een eigen EOF-marker ingevoerd, en wel: "*".
Het programma vraagt ter afsluiting van een keuze steeds naar *.
"*" in het MENU verzorgt een RANDOMIZE USR 0.

Wanneer je alle in het bestand aanwezige records wil zien, neem dan optie ZOEKEN en in plaats van een naam alleen ‹ENTER›. Bijgaande listing zal wat de BASIC betreft, naar ik hoop, geen problemen opleveren. Veel dezelfde handelingen zijn in subroutines ondergebracht. Wil je meer gegevens wegschrijven b.v. naam, adres, woonplaats, dan kan je beter e.e.a. in een string zetten. De POINTER (wijzer naar het betreffende record) houdt anders bij-elkaar-behorende gegevens niet bij elkaar.

Zoals je waarschijnlijk al zal zijn opgevallen is het een star programma geworden, met alleen "NAMEN" als toepassings-mogelijkheid. Er zal dus een algemeen programma moeten komen waarmee je een kloon kan maken. Hierin zullen het streamnummer, het drive-nummer, de filenaam en de recordlengte vrije keuzen moeten zijn.
Wanneer e.e.a. niet duidelijk is kun je me tussen 19.30 en 22.00h bellen.

Met vriendelijke groet
Joop van Brummelen
070-863577

DE LISTING

```
100 REM OPZETTEN BESTAND
120 CLOSE# 4 : OPEN# 4;1 ;"NAMEN"RND 21,- 1: LET z=1
140 PRINT "record nr. ;" ";z'"NAAM ; ";N$''
150 RESTORE 1710: GOSUB 900
160 IF w$="n" THEN GOTO  130
170 PRINT #4;M$
180 IF N$(1)="*" THEN RETURN
190 LET z=z+1: GO TO 130
200 REM ZOEKEN RECORD
210 GOSUB 1000: LET c$=0"
220 INPUT "TE ZOEKEN NAAM ? (* beeindigt) ";q$: IF q$="  " THEN
    GO TO 230: IF q$(1)="*" THEN RETURN
230 POINT #4;z: INPUT #4;N$
240 IF N$(1)="*" AND c$="0" THE PRINT q$;"NIET AANWEZIG":  PAUSE
    100: RETURN
250 IF N$(1)="*" AND c$="1" THEN PRINT #0;"DRUK EEN  TOETS  VOOR
    VERVOLG":PAUZE 10: RANDOMIZE USR 3438: RETURN
260 IF N$( TO q$) <>q$ THEN GOTO 290
270 IF N$( TO LEN q$)=q$ THEN LET c$="1"
280 PRINT "RECORDNR. ; ";z'"NAAM ; ";N$''
290 LE z=z+1: GOTO 230
300 REM WIJZIGEN RECORD
310 RESTORE 1720: GO SUB 600
320 RETURN
400 REM AFVOEREN RECORD
410 RESTORE 1730: GOSUB 600
420 RETURN
500 REM AANVULLEN BESTAND
510 GOSUB 1000
520 GOSUB 1100 530 IF N$(1)="*" THEN POINT #4;z: GO  SUB  130  :
    RETURN
540 LET z=z+1: GO TO 520
600 REM WIJZIG/AFVOER ROUTINE
610 GOSUB 200
620 INPUT  "RECORDNR.  VAN  DE  NAAM  (*  beeindigt)  ";q$:  IF
    q$(1)="*" THEN RETURN
630 LET z=VAL q$: GOSUB 1100
640 READ R$: INPUT VAL$ "R$";N$: IF N$(1)=" " THEN RETURN
645 PRINT "WIJZIGING",,"RECORDNR. ; ";z'"NAAM ; ";N$''
650 GO SUB 900
660 IF w$="n" THEN GOTO 610
670 POINT #4;VAL q$: PRINT #4;N$
680 RETURN
850 REM FOUTMELDINGS ROUTINE
860 RANDOMIZE USER 3438:FOR t=1 TO 3: BEEP .3,17: NEXT t
870 PRINT #0; FLASH 1"KEUZE NIET TOEGSTAAN"
880 FOR t=1 TO 6: BEEP .5,10 : NEXT t: FLASH 0: RETURN
900 REM AKKOORD ROUTINE
910 READ R$: POKE 23658,0
920 PAUSE 10: RSNDOMIZE USR 3438: PRINT #0;R$
930 LET w$=INKEY$ : IF w$="" THEN GO TO 930
940 IF w$<>"j" AND w$<>"n" THEN GO SUB 850: GO TO 920
950 RETURN
```

```
1000 REM OPEN BESTAND ROUTINE
1010 CLOSE# 4: OPEN#4;1;"NAMEN"RND 21
1020 LET z=1
1030 RETURN
1100 REM INLAAD ROUTINE
1110POINT #4;z: INPUT #4;N$
1120 PRINT "RECORDNR. ; ";z'"NAAM ;   ; "; N$''
1130 RETURN
1210 POINT #4; VAL Q$: PRINT #;N$
1500 REM MENU
1505 DIM N$(20)
1510 CLS: CLOSE #4: RESTORE 1700
1520 FOR t = 5 TO 17 STEP 2
1530 READ R$
1540 PRINT AT t,5;R$
1550 NEXT t
1560 RANDOMIZE USR 3438: PRINT #0;"TOETS KEUZE IN"
1570 LET K$=INKEY$
1580 IF K$="" THEN GO TO 1570
1590 IF K$="*" THEN CLOSE #4: RANDOMIZE USR 0
1600 IF K$<"1" OR K$>"5" THEN GOSUB 850:GOTO 1560
1610 CLS : GO SUB VAL K$*100: GO TO 1510
1700  DATA  "FILE;   NAMEN","1   OPZETTEN   BESTAND","2   ZOEKEN
RECORD","3 WIJZIGEN RECORD", "4 AFVOEREN RECORD",  "5  AANVULLEN
BESTAND", "* EIND PROGRAMMA"
1710 DATA "AKKOORD j/n"
1720 DATA "VOER WIJZIGING IN ","WIJZIGING AKKOORD j/n"
1730 DATA "DRUK OP <ENTER>", "AFVOER AKKOORD J/N"
2000 SAVE *1;"rnd access" LINE 1500
```

Joop v. Brummelen

<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

DISCOVERIES

* Wie heeft er al ervaring met een of meer modems die draaien op de nieuwe 128 in kombinatie met de DISCOVERY? Schrijf naar : SPECTRUM GEBRUIKERSGROEP TER AAR, Sperwerhoek 36, 2743 GC Waddinxveen.

* Wie kan mij helpen met het omzetten van de programmas VU-CALC en OMNICALC 1 naar schijf? J.T. Rommen, Lisgors 13, 2371 XA Mijnsheerenland.

* Te koop aangeboden: DK TRONICS printerinterface (software -gestuurd). Prijs f45,--. Jos Hens 040 - 817548 of 013 - 630872

* Te koop: Originele Editor/Assembler van Aakosoft incl. handleiding. Henk Kramer 020 - 120923 na 20.30 uur

CAT MOVER

Het maken van een back-up van een disk kan je uitvoeren met de opdracht:

```
MOVE"d";1 TO "d";3
```

Wanneer je slechts van enkele programma's een back-up wil maken komt er al wat meer type-werk bij kijken:

```
MOVE "m";1;"een naam" TO "m";2;"een naam"
```

Het hierbij afgedrukte programma maakt e.e.a. een stukje eenvoudiger. Laadt het in en RUN het. Allereerst wordt er gevraagd om in te toetsen van welke drive er een back-up gemaakt moet worden; vervolgens naar welke drive we gaan schrijven. Nu wordt de "CATalogue"-file geopend. Alle programma-namen verschijnen -in de volgorde zoals ze op de schijf staan- op het scherm. Door middel van het intoetsen van "j" of "n" wordt er van een programma wel of niet een back-up gemaakt. " " beeindigt het programma. Stop zonodig een nieuwe disk in de drive en RUN het programma opnieuw.

Het is wel belangrijk dat je weet welke programma-onderdelen bij elkaar horen. Met enige fantasie is het niet moeilijk om programma's te maken die vanuit de CATalogue file bijvoorbeeld LOADen of ERASEn. Veel succes toegewenst,

Joop van Brummelen

DE LISTING

```
 5 DIM A$(16)
10 INPUT "van drive ";vdr
15 PRINT ,"van drive nr.";vdr
20 INPUT "naar drive ";ndr
25 PRINT ,"naar drive ; ";ndr''
30 LET z=2
40 CLOSE #4: OPEN #4;"CAT"; vdrRND16
50 POINT #4;z
55 FOR t=1 TO 16
60 LET A$(t) =INKEY$#4
65 NEXT t
70 PRINT A$(7 TO )
80 POKE 23658 ,0
90 PRINT #0;"Programma verplaatsen j/n          * to escape"
100 LET q$=INKEY$: IF q$=" " THEN GOTO 100
105 RANDOMIZE USR 3438
110 IF q$="j" THEN GOTO 160
120 IF q$="n" THEN GOTO 140
130 IF q$="*" THEN GOTO 200
135 GOTO 90
140 LET z=z+1
150 GOTO 50
160 MOVE "m";vdr;A$(7 TO ) TO "m";ndr;A:(7 TO )
170 GOTO 140
200 CLOSE# 4
210 STOP
220 SAVE *1;" CAT mover" LINE 1
```

BASICAT 3BR

'CAT' in drie kolommen

Aangezien een filenaam normaliter slechts 10 posities op het beeldscherm in beslag neemt, kunnen er drie filenamen en twee spaties op een regel van 32 posities achter elkaar staan. Beide spaties dienen dan als afscheidingen tussen de drie filenamen. Het zou dus overzichtelijker zijn als we de listing van het CAT-commando op deze wijze op ons beeldscherm konden laten verschijnen.

Het is niet zo moeilijk om een routine te schrijven die dit doet. Je kunt namelijk een heel stuk van de ROM-routine kopieeren. Het ROM-programma zorgt ervoor dat na het afdrukken van een filenaam (van tien karakters, dus plus eventuele spaties achteraan) een RETURN-karakter (CHR$ 13) afgedrukt wordt. Indien we dit karakter door een spatie vervangen, schieten we al een heel eind op, alleen nemen de drie filenamen dan 33 posities in beslag. Als we elke derde spatie dus weer door een RETURN-tje vervangen, krijgen we het gewenste effect.

Er is echter nog ""n probleempje, namelijk dat niet alle karakters op het beeldscherm of de printer ""n positie in beslag nemen. Men denke bijvoorbeeld aan de karakters 'CODE', 'DATA' of 'SCREEN$', die velen waarschijnlijk zien als handige extensies (toevoegingen) aan een filenaam. Zoals INVADERS.SCREEN$ (9 karakters, 12 posities) of SPELLEN.DATA (10 karakters, 16 posities). Ook namen als IT'S A SIN TO COPY en DO NOT STOP, CONTINUE spreken denk ik wel aan.

Ik hoop dat nu wel duidelijk is wat ik bedoel. Voor karakters met een ASCii-waarde groter dan 127 moet iets speciaals geregeld worden. Als je zelf iets beters weet hou ik me aanbevolen, maar het lijkt mij het handigst om deze karakters ge-inverteerd, en 128 afgetrokken van de ASCii- waarde af te laten drukken, dus niet

```
PRINT CHR$ A;
```

maar

```
IF A<=127 THEN PRINT CHR$ A;
          ELSE PRINT INVERSE 1;CHR$ (A-128);INVERSE 0;
```

De machinetaal die hiervoor geschreven moet worden is volgens mij korter dan bv. die, om de hexadecimale waarde in smalle lettertjes af te laten drukken, en dat scheelt dus handenvol bytes, en dat voel je wel in je vingers.

Ook de karakters kleiner dan CHR$ 32 moeten een bepaalde behandeling ondergaan. Deze zal ik als '?' af gaan drukken.

Nu heb ik een tijdje geleden het tsjippie 6116 gekocht en nou wil ik dat ding wel gebruiken ook. Daarom zit er een joekel van een BASIC-programma bij om de 2K voor dit IC in elkaar te draaien. Als je het niet in wilt typen, kun je het bestellen bij de programma-bank, of het kopieeren op een van de komende DUC-dagen. Voor diegenen die het toch willen intypen, hier de listing met enige verklarende tekst.

We moeten de 'CAT'-routine van het 'm'-kanaal veranderen. Deze informatie staat in een subtabel van tabel #06. In tabel #06 staan wijzers voor de kanalen ' ' (spatie), 'm' en 'j' (tenminste bij versies 2.1, 2.2 en 2.22, dit kan altijd veranderen). Met de subtabellen kunnen 'CAT', 'ERASE', 'INQUIRE' en 'FORMAT' uitgevoerd worden. Het verschil tussen het ' ' en het 'm'-kanaal

is, dat ERASE *1;"xxx" een foutmelding geeft als "xxx" niet op de disk staat, en ERASE *"m";1;"xxx" niet. Dit in verband met microdrive compatibiliteit. We hebben dus de keus om de ' ' of de 'm'-entry van tabel #06 te wijzigen.

Ik heb gekozen voor de ' '-entry, omdat ik geen extra moeite wil doen om de drie-koloms listing te krijgen, in vakjargon: omdat ik dit als default wil. Dit brengt echter wat andere probleempjes met zich mee, maar daarover later meer. Eerst het programma.

De initialisaties:

```
10 LET ramstart=8*1024
20 LET tabel=ramstart+256
30 LET vrij=ramstart+512
40 DEF FN l(o)=o-256*FN h(o)
50 DEF FN h(i)=INT (i/256)
```

De machinetaal-routine maakt gebruik van wat subroutines in de DISCOVERY-ROM en deze staan niet allemaal in de tabellen vermeld. Daarom ben ik verplicht om voor de verschillende versies de absolute adressen in DATA-regels op te slaan. Deze adressen worden ingelezen in overeenkomstige variabelen.

```
60 LET versie=USR 8
70 IF versie=2.1 THEN RESTORE 200
80 IF versie=2.2 OR versie=2.22 THEN RESTORE 300
90 IF versie<>2.1 AND versie<>2.2 AND versie<>2.22 THEN
   PRINT "Other version.": STOP
100 READ opencat,closch,posnstart,stacksize,
   findnext
190 RESTORE 1000
200 DATA 3344,857,3721,3733,3623
300 DATA 3532,872,3909,3921,3811
```

Zoek het begin van tabel #06.

```
4000 LET HL=32768+PEEK (32768+ramstart)+
     256*PEEK (32768+ramstart+1)+7
4010 LET tabel06=32768+PEEK (HL-1)+256*PEEK HL
```

Nu moeten we tabel #06 kopieeren in RAM, en meteen de wijzer aanpassen in de hoofdtabel naar tabel #06.

Eerst de wijzer aanpassen.

```
4020 POKE HL-1,FN l(tabel)
4030 POKE HL,FN h(tabel)
```

Dan verplaatsen we tabel #06 naar RAM. Dit gaat in porties van drie bytes (een zoekbyte en twee adresbytes), en houdt op als de zoekbyte (a) 0 is. Als de zoekbyte 32 is (de CODE van een spatie), wordt het adres van deze entry opgeslagen in HL, omdat deze entry veranderd moet worden. Als je dit wilt, kun je dit veranderen in CODE "m".

```
4040 LET f=0
4050 LET a=PEEK (tabel06+3*f)
4060 POKE (32768+tabel+3*f),a
4070 POKE (32768+tabel+3*f+1),PEEK (tabel06+3*f+1)
4080 POKE (32768+tabel+3*f+2),PEEK (tabel06+3*f+2)
4090 LET f=f+1
4100 IF a<>32 THEN GOTO 4050
```

De ' '-entry is gevonden, sla het adres op in HL.

```
4110 LET HL=32768+tabel+3*f-1
```

Nu de rest van de tabel.

```
   4120 LET a=PEEK (tabel06+3*f)
   4130 POKE (32768+tabel+3*f),a
   4140 POKE (32768+tabel+3*f+1),PEEK (tabel06+3*f+1)
   4150 POKE (32768+tabel+3*f+2),PEEK (tabel06+3*f+2)
   4160 LET f=f+1
   4170 IF a<>0 THEN GOTO 4120
```

Laat 'tabel' naar de eerste vrije plaats wijzen.

```
   4180 LET tabel=tabel+3*f
```

Bepaal het start-adres van de subtabel voor ' '.

```
   4190 LET subtabel06=32768+PEEK (HL-1)+256*PEEK HL
```

Kopieer nu de subtabel (8 bytes).

```
   4200 FOR f=0 TO 7
   4210 POKE (32768+tabel+f),PEEK (subtabel06+f)
   4220 NEXT f
```

Pas de wijzer in tabel #06 aan.

```
   4230 POKE HL-1,FN l(tabel)
   4240 POKE HL,FN h(tabel)
```

Verander de entry in de subtabel naar de 'CAT'-routine en laat deze wijzen naar 'vrij'. Hier komt onze routine:

```
   4250 LET HL=32768+tabel+5
   4260 POKE HL-1,FN l(vrij)
   4270 POKE HL,FN h(vrij)
```

Laat 'tabel' wijzen naar de eerste vrije plaats.

```
   4280 LET tabel=tabel+8
```

Nu kunnen we op adres 'vrij' de 'CAT'-routine plaatsen. Eerst geef ik de assembly listing.

```
                CALL OPENCAT     ; open het catalog-file
                CALL POSNSTART   ; POINT naar het begin
                XOR  A           ; begin in kolom 0
                LD   (END),A     ; sla deze waarde op
                LD   HL,0        ; HL = - blokken gebruikt
                CALL PRINT       ; print de disknaam
                CALL CR          ; met een CHR$ 13
                CALL CR          ; en een lege regel
                XOR  A           ; begin weer in kolom 0
                LD   (END),A     ; en onthoudt dit
        LUS1    CALL PRINT       ; print de volgende filenaam
                JR   NC,LUS1     ; tot het laatste file
                LD   B,H         ; BC := HL = het aantal
                LD   C,L         ;    gebruikte blokken
                RST  #10         ; stack BC op de floating-
                DEFW #2B2D       ; point calculator
                CALL STACKSIZE   ; stack de blokgrootte
                RST  #28         ; roep de fp-calculator aan
                DEFB #08         ; er zijn 8 bytes
                DEFB #04         ; vermenigvuldig blocks & size
                DEFB #34         ; stack het getal
                DEFB #3B,#00     ; 2↑10 = 1024 = 1 K
                DEFB #05         ; en deel dit
                DEFB #38         ; ende van de calculatie
                LD   A,(END)     ; A := kolomnummer
                AND  A           ; is het 0 ? dan staan we al op
                JR   Z,SKIP1     ; een nieuwe regel. Anders druk
                CALL CR          ; een CHR$ 13 af.
```

```
SKIP1   CALL CR          ; en sla een regel over
        PUSH IX          ; bewaar IX
        RST  #10         ; print een floating point
        DEFW #2DE3       ; getal, het aantal K's
        POP  IX          ; haal IX weer terug
        CALL CR          ; naar begin volgende regel
        JP   CLOSCH      ; en close het catalog-file
```

Nu krijgen we de subroutines. Merk op dat HL de negatieve waarde van het aantal gebruikte blokken tot het nieuwe file bevat. In het laatste record in de catalog is het aantal blokken op de disk opgeslagen. Dus als alle records gelezen zijn, bevat HL het aantal blokken op de disk - het aantal blokken dat gebruikt is, dus het aantal vrije blokken.

```
PRINT   PUSH HL          ; bewaar HL
        CALL FINDNEXT    ; zoek volgend file
        POP  HL          ; haal HL weer terug
        LD   DE,(#5C9E)  ; DE := beginblok van dit file
        ADD  HL,DE       ; tel dit bij HL op
```

Merk op dat HL nu het aantal vrije blokken tot dit file bevat.

```
        LD   DE,(#5CA0)  ; DE := eindblok van dit file
        LD   A,D         ; Is dit record de end-marker ?
        AND  E           ; dan bevat DE #FFFF, en is A
        INC  A           ; nu 0
        SCF              ; sein 'laatste blok'
        RET  Z           ; en keer terug als A = 0
        SBC  HL,DE       ; trek DE van HL af
```

Merk op dat HL nu weer het aantal gebruikte blokken bevat tot en met dit file.

```
        LD   DE,#5CA2    ; DE := start van de filenaam
        LD   A,(DE)      ; pak het eerste karakter
        AND  A           ; is het CHR$ 0 ?
        RET  Z           ; druk de filenaam dan niet af
        LD   B,10        ; B := aantal letters in naam
LUS2    LD   A,(DE)      ; pak de letter
        BIT  7,A         ; is het groter dan 127 ?
        PUSH AF
        CALL NZ,GROTER   ; roep dan GROTER aan
        POP  AF
        CALL Z,KLEINER   ; anders roep KLEINER aan
        INC  DE          ; wijs naar volgende letter
        DJNZ LUS2        ; en herhaal tot het einde
        LD   A,(END)     ; pak het kolomnummer
        INC  A           ; hoog het op
        LD   (END),A     ; en sla het weer op
        CP   3           ; is het nu 3 ?
        JR   NZ,SPACE    ; nee, druk dan een spatie af
        XOR  A           ; anders maak het kolom-nummer
        LD   (END),A     ; gelijk aan 0 en
CR      LD   A,13        ; print een CHR$ 13
        JR   PRI
SPACE   LD   A,32        ; print een spatie
        JR   PRI
KLEINER CP   31          ; groter dan 31 ?
        JR   NC,SKIP2    ; print dan de letter
        LD   A,"?"       ; print anders een vraagteken
```

```
SKIP2   JR   PRI
GROTER  AND  %01111111  ; maak de letter tussen 0 & 127
        PUSH AF         ; bewaar de letter
        LD   A,20       ; print inverse 1;
        CALL PRI
        LD   A,1
        CALL PRI
        POP  AF         ; haal de letter terug
        CALL KLEINER    ; en druk hem af
        LD   A,20       ; print inverse 0;
        CALL PRI
        XOR  A
PRI     PUSH IX         ; bewaar een stel registers.
        PUSH DE         ; IX = start van kanaal,DE = start
        PUSH HL         ; van filenaam,HL = aantal
        PUSH BC         ; blokken, BC is tellertje
        RST  #10        ; print de letter
        DEFW #15F2
        POP  BC         ; en haal ze weer terug
        POP  HL
        POP  DE
        POP  IX
        AND  A          ; sein 'nog niet laatste file'
        RET             ; en keer terug
END     DEFB #FF        ; opslag van het kolom-nummer
```

Duidelijk? Dacht ik al.

Deze routine bevat enkele (interne) subroutines die aangeroepen moeten worden, dus moeten we de absolute adressen van deze subroutines te weten komen. We weten het start-adres, dus de absolute adressen zijn eenvoudig te bepalen. De regels die deze routine op zijn plaats zetten volgen nu:

```
4290 LET print=vrij+72
4300 LET cr=vrij+130
4310 LET kleiner=vrij+138
4320 LET groter=vrij+146
4330 LET pri=vrij+169
4340 LET end=vrij+184
```

Vlak voor het insturen van dit artikel heb ik de volgende regel nog aan dit programma toegevoegd. Het stelt je in staat om een eigen versie aan te maken met een ander aantal kolommen (voor bijvoorbeeld een 64 koloms beeld met BetaBasic).

```
4345 INPUT "number of columns [3] >";LINE a$:
     LET a$=a$+("3" AND (a$="")): LET columns=VAL a$
4350 READ a
4360 POKE 32768+vrij,a
4370 LET vrij=vrij+1
4380 READ a
4390 IF a<>999 THEN GOTO 4360
4500 DATA 205,FN l(opencat),FN h(opencat),205,
     FN l(posnstart),FN h(posnstart),175,50,
     FN l(end),FN h(end),33,0,0,205,FN l(print),
     FN h(print)
4510 DATA 205,FN l(cr),FN h(cr),205,FN l(cr),FN h(cr),
     175,50,FN l(end),FN h(end),205,FN l(print),
     FN h(print),48,251,68,77,215,43,45,205,
```

cat3code 8192, 1048
cat.2code 62653, 2829 → OPUSCAT.2.

```
        FN l(stacksize),FN h(stacksize)
 4520 DATA 239,8,4,52,59,0,5,56,58,FN l(end),FN h(end),
        167,40,3,205,FN l(cr),FN h(cr),205,FN l(cr),FN h(cr)
 4530 DATA 221,229,215,227,45,221,225,205,FN l(cr),
        FN h(cr), 195,FN l(closch),FN h(closch)
 4540 DATA 229,205,FN l(findnext),FN h(findnext),225,
        237,91,158,92,25,237,91,160,92,122,163,60,55,200,
        237,82,17,162,92,26,167,200
 4550 DATA 6,10,26,203,127,245,196,FN l(groter),
        FN h(groter),241,204,FN l(kleiner), FN h(kleiner),
        19,16,242,58,FN l(end),FN h(end),60,50,FN l(end),
        FN h(end),254,columns,32,8,175,50,FN l(end),FN h(end)
 4560 DATA 62,13,24,35,62,32,24,31,254,31,48,2,62,
        CODE "?",24,23
 4570 DATA 230,127,245,62,20,205,FN l(pri),FN h(pri),
        62,1,205,FN l(pri),FN h(pri),241,205,FN l(kleiner),
        FN h(kleiner),62,20,205,FN l(pri),FN h(pri),175
 4580 DATA 221,229,213,229,197,215,242,21,193,225,209,221,
        225,167,201,255
 4590 DATA 999
```

RUN het programma op de bekende (?, zie DUC 3) manier:

```
        RANDOMIZE USR 14070
        SAVE *1;"10K"CODE 0,10240
```

indien je dit nog niet eerder gedaan hebt. Vervolgens:

```
        CLEAR 32768
        LOAD *1;"10K"CODE 32768
        RUN                          (en wacht een tijdje)
        SAVE *1;"ram"CODE 32768+8*1024,2048
        LOAD *1;"ram"CODE 8*1024
```

En probeer nu eens CAT 1. Als het werkt mag je blij zijn, anders kun je het programma (vooral de DATA-regels) eens nakijken en het opnieuw proberen (na het her-laden van het file "10K").

Merk op dat je het vraagtekentje in regel 4560 (in CODE "?") zelf kan veranderen in iets anders, bijvoorbeeld een punt ('.') of een sterretje ('*'). Dit karakter wordt afgedrukt als er een karakter met een ASCii-waarde kleiner dan 32 afgedrukt moet worden (ook bij een waarde van 128 tot 159 natuurlijk, alleen dan inverse).

Let op de 'pri'-routine, het eigenlijke print-gedeelte. Er worden een stel registers op de stapel bewaard voor de aanroep en erna weer teruggehaald. Dit is nodig omdat de SPECTRUM dit zelf niet doet, en er gekke dingen kunnen gebeuren als wij dit niet doen. De DISCOVERY-ROM houdt er vreemd genoeg ook geen rekening mee, hoewel het veel schade aan kan brengen aan je disk. Een ongevaarlijk voorbeeld: neem het programma

```
    10 LIST
```

en RUN dit programma. Als je 'scroll ?' krijgt, typ dan CS-9 (oftewel GRAPH) of CS-SS (oftewel EXTEND MODE) in. Nu krijg je de laatste regel die je hebt ingetypt in het beeld. Druk nu op nog een toets, en met een beetje geluk verschijnen op je scherm nu een paar vieze karakters (nee, geen naakte UDG's!) en gaat het listen met datzelfde geluk gewoon door (het kan namelijk ook met 'Invalid Colour' stoppen). Dat deze tekens afgedrukt worden ligt aan een 'bugje' in de SPECTRUM-ROM. Dit zorgt er echter ook

voor dat bij een scroll-bericht tijdens het opvragen van een catalogus enige informatie op de disk overschreven kan worden na het indrukken van rare toets-combinaties, en ook het aantal vrije blokken op de disk niet meer klopt. Probeer dit maar niet!

Als je een 'run'-programmaatje hebt dat de catalog naar een file stuurt (met bv. CAT #4;1) en dit file dan uitleest (met bv. INPUT #stream;a$), kan het zijn dat dit programma niet meer werkt. CAT 1 geeft namelijk niet meer een listing die gemakkelijk gesplitst kan worden in filenamen (soms is dit zelfs onmogelijk, AND is een onomkeerbare functie). Vervang in dat programmaatje dan CAT #4;1 door CAT #4;"m";1 en alles werkt dan weer als tevoren.

Ik hoop dat je veel plezier (en nut) hebt van deze routine.

Marcel van Dongen.

<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

BREAK-OUT

In dit artikel wil ik het niet hebben over het ping-pong-achtige spelletje waarbij je blokjes uit een muur moet stoten, maar over een routinetje waarmee je uit de meeste BASIC-programma's kunt breken, zelfs als deze iets als ON ERROR hebben ingebouwd. Om te snappen hoe dit routinetje werkt, moet je eerst weten wat 'ON ERROR' doet.

Als een programma loopt, staat bovenop de machinestapel het adres #1303 (1303 hex = 4867 dec). Dit adres ligt in de hoofdlus van het SPECTRUM ROM-programma. Als er een fout optreedt, wordt er naar dit adres gesprongen. Het zorgt er voor dat de foutmelding op de onderste regel wordt afgedrukt, en het programma stopt. Ook het beeindigen van een programma genereert een melding, namelijk: '0 O.K.'. Ook in dit geval wordt er naar adres #1303 gesprongen. Er is een systeem-variabele die naar het adres in de stack wijst waar deze #1303 staat. Deze sysvar. heet ERRSP, oftewel error-adres in de stack.

Door ERRSP of de waarde in de stack te veranderen, kun je ervoor zorgen dat bij een fout in het programma (zoals 'BREAK into program' of 'STOP in INPUT') naar een ander adres dan #1303 wordt gesprongen. Dit kan het adres 0 zijn, waardoor een 'reset' veroorzaakt wordt bij een fout, of het adres van een fout-onderscheppende routine, zoals 'ON ERROR'. Deze routine kijkt dan wat de foutmelding was, en als het een BREAK betrof, kan het een berichtje afdrukken als: 'dit programma is beveiligd' of zo, en dan doorgaan met het programma, of naar een bepaalde regel springen.

Voorbeelden zijn: SUPERCODE (aparte routine), ANT ATTACK, ZOMBIE ZOMBIE, een stelletje van ARGUS PRESS en waarschijnlijk vele andere programma's die een stuk(je) BASIC gebruiken voor bijvoorbeeld hun menu's. Ook ART STUDIO gebruikt dit bij het laden en saven.

Het is heel moeilijk of haast onmogelijk om uit zo'n programma te breken, omdat alle foutmeldingen onderschept worden. Aan de hand van wat voorbeeldjes kan ik laten zien hoe deze beveiliging werkt. Type in:

```
10 CLEAR 30000
20 POKE 30000-2,0
30 POKE 30000-3,0
40 PRINT INK RND*8;AT 10,10;"PRESS BREAK"
50 GO TO 40
```

RUN dit programmaatje en druk dan op BREAK. Het systeem reset. Regel 10 zet de stack op adres 30000 en lager, regels 20 en 30 zorgen ervoor dat de waarde #1303 in de stack overschreven wordt met de waarde #0000. Bij een fout wordt naar adres #0000 gesprongen: het systeem reset (vgl. RANDOMIZE USR 0).
Iets dergelijks valt uit te halen met ERRSP.

```
10 POKE 23613,0
20 POKE 23614,0
30 PRINT INK RND*8;AT 10,10;"PRESS BREAK"
40 GO TO 30
```

Op adres 0 staat 243 en op adres 1 staat 175. Bij een fout wordt dus naar het adres 175*256+243 gesprongen en -tenzij daar een machinetaal-routine staat- het systeem zal hoogstwaarschijnlijk CRASHen.
ERRSP is ook nuttig te gebruiken. Als we deze sysvar namelijk niet laten wijzen naar #1303 in de stack, maar naar het adres in de stack eronder, kunnen we een 'ON ERROR CONTINUE' simuleren.
Kijk maar bij het volgende programmaatje.

```
10 CLEAR 32767
20 LET a=PEEK 23613
30 LET x=8*INT (RND*25)
40 LET y=8*INT (RND*15)
50 LET s=7+8*INT (RND*10)
60 LET w=7+8*INT (1+RND*15)
70 LET c=INT (RND*8)
80 POKE 23613,a
90 PLOT INK c;x,y+s
100 DRAW INK c;w,0
110 POKE 23613,a
120 IF s<1 THEN GO TO 30
130 LET s=s-1
140 GO TO 80
```

Er hoeft niet gecontroleerd te worden of de lijn buiten het beeldscherm komt, want als dit gebeurt, wordt er naar het adres gesprongen dat vlak onder #1303 in de stack staat. Dit adres is STM-RET, oftewel 'statement return', en zorgt ervoor dat met de uitvoering van het volgende statement wordt begonnen.
Nu wil het geluk dat, bij een fout, de DISCOVERY eerst de foutmelding onderschept, (het kan immers een extended-commando zijn!) en, als het ook een DISCOVERY-fout blijkt te zijn, de

SPECTRUM ROM weer inschakelt en de foutmelding laat genereren.

Dit onderscheppen gaat als volgt: bij een fout springt het SPECTRUM ROM-programma -altijd!- naar adres #0008 (de error-handler), waardoor de DISCOVERY ROM ingeschakeld wordt. De DISCOVERY kijkt dan of het een extended-commando kan zijn (door in de hookcode-tabel te zoeken), en als dit niet zo is, wordt het adres in de DISCOVERY ROM gevonden waar een routine staat die weer naar de SPECTRUM ROM laat springen. Tenslotte wordt de foutmelding dan gegenereerd. Dit error-adres in de DISCOVERY ROM ligt ook in de hookcode-tabel opgeslagen, en wel als het laatste adres, behorende bij de laatste zoek-byte. We kunnen dus zelf foutmeldingen onderscheppen door dit adres te veranderen in het adres van een routine die de sysvar ERRSP en de stack zelf weer goed zet, en dan weer naar de SPECTRUM ROM terugspringt.

Een andere beveiliging kunnen we zo ook ontkrachten, namelijk 'POKE 23659,0', die ervoor zorgt dat het systeem vastloopt als er een foutmelding op de onderste regel afgedrukt moet worden.

23659 is een sysvar en heet DEFSZ, oftewel 'de gedefinieerde grootte (size) van het onderste deel van het beeldscherm'. Normaal is dit twee, maar door veel tekst in een regel (in het onderste deel van het scherm) in te typen, kan dit oplopen tot meer dan 20.

Door op dit adres de waarde 0 te zetten, kan de SPECTRUM op 24 regels tekst kwijt. Probeer eens:

```
10 CLS #
20 POKE 23659,0
30 FOR f=0 TO 23
40 PRINT "regel ";f
50 NEXT f
60 POKE 23659,2
70 PAUSE 0
```

De eerste POKE zorgt ervoor dat we op regel 23 kunnen schrijven, de tweede herstelt de oude toestand weer. PAUSE 0 zorgt ervoor dat de onderste twee regels niet meteen worden gewist en overschreven met de O.K.-melding.

Als we op adres 23659 0 poken, hebben we 23 regels op het bovenste deel van het beeldscherm, maar hebben we geen regels voor het onderste deel meer over, dus kan de SPECTRUM geen foutmeldingen meer op het onderste deel kwijt, waardoor het systeem crasht bij een fout (of het beeindigen van een programma of een direct commando). Dit is een handige beveiliging, want het mes snijdt hier aan twee kanten: je hebt meer ruimte op het beeldscherm, en tevens een (bijna) waterdichte beveiliging. Het routinetje moet er dus voor zorgen dat op het adres 23659 de waarde 2 gezet wordt.

Merk op dat je ook handig gebruik kunt maken van dit adres; door daar namelijk 1 te poken, krijg je een leuk patroontje. Zie:

```
10 POKE 23659,1
20 STOP
```

Aangezien we voor onze routine het IC 6116 nodig hebben (we moeten immers de hookcode-tabel aanpassen), kunnen we het routinetje er net zo goed ''k inzetten. Zoals met alle programma's die ik voor het IC maak, is het ook hierbij weer nodig om het file '10K' aangemaakt te hebben (zie een handvol andere artikelen van mijn vingertoppen). Hier volgt het programma.

De initialisaties:

```
10 LET ramstart=8*1024
20 LET tabel=ramstart+256
30 LET vrij=ramstart+512
40 DEF FN l(o)=o-256*FN h(o)
50 DEF FN h(i)=INT (i/256)
```

Zoek het begin van de hookcode-tabel, tabel #0.

```
5000 LET HL=32768+PEEK (32768+ramstart)+
     256*PEEK (32768+ramstart+1)+1
5010 LET tabel00=32768+PEEK (HL-1)+256*PEEK HL
```

Kopieer tabel #00 naar RAM, maar pas eerst de wijzer in de hoofdtabel aan.

```
5020 POKE HL-1,FN l(tabel)
5030 POKE HL,FN h(tabel)
5040 LET f=0
5050 LET a=PEEK (tabel00+3*f)
5060 POKE (32768+tabel+3*f),a
5070 POKE (32768+tabel+3*f+1),PEEK (tabel00+3*f+1)
5080 POKE (32768+tabel+3*f+2),PEEK (tabel00+3*f+2)
5090 LET f=f+1
5100 IF f<>0 THEN GO TO 5050
```

Pas 'tabel' aan.

```
5110 LET tabel=tabel+3*f
```

De 3 laatst gekopieerde bytes bevatten het error-adres waar naartoe gesprongen wordt bij een foute hookcode. Dit willen we graag weten, dus we slaan het adres ervan op in HL, en bepalen het error-adres.

```
5120 LET HL=32768+tabel+3*f-1
5130 LET error=PEEK (HL-1)+256*PEEK HL
```

Bij een foute hookcode moet naar de nieuwe routine gesprongen worden, dus veranderen we de laatste entry van de tabel.

```
5140 POKE HL-1,FN l(vrij)
5150 POKE HL,FN h(vrij)
```

Nu kunnen we op adres 'vrij' onze nieuwe error-routine zetten. De assembly listing volgt nu:

```
START  PUSH AF          ; bewaar de fout-code
       LD   A,2         ; zet 0 op adres 23659
       LD   (23659),A
       LD   HL,(23613)  ; pak ERRSP
       LD   A,H         ; staat de stack in ROM ? dan
       CP   64          ; is ERRSP verandert, zo niet,
       JR   C,SKIP      ; ga dan door naar SKIP.
       LD   HL,(23730)  ; pak RAMTOP
       DEC  HL          ; laag dit 3 maal af zodat het
       DEC  HL          ; wijst naar de plek waar
       DEC  HL          ; #1303 had moeten staan,
       LD   (23613),HL  ; en sla dit adres op
SKIP   LD   HL,#03      ; plaats nu #1303 weer in de
       INC  HL          ; stack op de oude plaats.
       LD   HL,#13
       POP  AF          ; haal de fout-code terug
       JP   ERROR       ; en ga normaal verder
```

Als we deze machinetaal routine in DATA-regels zetten en in laten lezen, krijgen we het volgende:

```
5160 READ a
5170 POKE 32768+vrij,a
5180 LET vrij=vrij+1
5190 READ a
5200 IF a<>999 THEN GO TO 5170
5300 DATA 245,62,2,50,107,92,42,61,92,124,254,64,48,8,
     42,178,92,43,43,43,61,92,54,3,35,54,19,241,195,
     FN l(error),FN h(error)
5310 DATA 999
```

Type nu:

```
SAVE *1;"break.bas"
CLEAR 32768
LOAD *1;"10K"CODE 32768
```

en RUN het programma. Als het klaar is, save dan de 2K voor het IC 6116.

```
SAVE *1;"ram"CODE 32768+8*1024,2048
```

Laad nu de 2K RAM weer in over het IC door

```
LOAD *1;"ram"CODE 8*1024
```

en kijk of het werkt. Probeer hiervoor

```
POKE 23569,0: STOP
```

Het STOP-statement is erg belanrijk! Als het beeld nu toch zwart slaat, of er iets anders vreemds gebeurt (behalve het verschijnen van de STOP-melding onderaan), dan heb je een typfout gemaakt, opnieuw dus (controleer het programma, RANDOMIZE USR 14070:LOAD *1;"ram"CODE 32768: RUN). Als het goed gaat kun je ook eens proberen

```
10 POKE 23613,0
20 POKE 23614,0
30 STOP
```

Nog steeds goed? Wow! Nu kun je proberen in wat BASIC-programmaatjes in te breken. Probeer ANT ATTACK eens. Misschien kun je met deze routine wel je favoriete programma op disk zetten. Kom je dan toch nog wat problemen tegen, dan kun je de hulplijn bellen als je het zelf niet redt, anders kun je misschien wel eens een artikeltje schrijven wat je precies gedaan hebt om het programma op disk over te zetten.
Trouwens, als je zo'n programma goed over wilt zetten, zoek dan de regel waar de LOAD-commando's staan, met daarachter waarschijnlijk een 'RANDOMIZE USR adres' of wat POKEs. Deze laatste zorgen er meestal voor dat ERRSP of de stack gewijzigd wordt. Verander de 'LOAD' in 'LOAD *1;', voeg eventueel een naam toe en SAVE het programma dan met 'LINE dat-regelnummer'. Dan gaat het meestal wel goed. Vergeet ook niet eventuele bytes mee te saven, al zijn het alleen maar de UDG's.
Merk op dat de beveiligingen ongedaan gemaakt worden als er een foute hookcode gegenereerd wordt. 'Ø O.K.' is ook een hookcode, (namelijk hookcode 255, of #FF). Deze wordt gebruikt bij RUN. Als je dit namelijk ingetypt hebt, en er zit geen BASIC-programma in het geheugen, dan krijg je normaal de Ø O.K.'-melding, maar de DISCOVERY onderschept dit en als aan de voorwaarden voldaan is (geen BASIC en extra kanalen) dan wordt het programma 'run' van drive 1 ingeladen. Het is dus erg belangrijk dat een beveiligd programma (zoals de voorbeelden) niet met Ø O.K. stoppen.
Ik hoop dat je veel plezier hebt van deze routine, en als je nog op- of aanmerkingen hebt, of voorstellen voor wijzigingen, dan weet je me wel te vinden denk ik.

Marcel van Dongen

‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹››››››››››››››››››››››››››››››››››››

DISCOVERIES

* Wie kan mij een aamgepaste versie leveren van HI-SOFT's PASCAL dat volledig op de DISCOVERY werkt? Voor school-doeleinden. Alle kosten worden vergoed. Marco Baaijen, Jagermeester 11, 6713 KE, Ede.

* Wie kan mij helpen aan inlichtingen en software van de VTX 5000? G. Tjallinks, tel 03408 - 81126.

* GEVRAAGD: oude nummers van ZX COMPUTING, van voor AUG. '85. Kopen tegen redelijke vergoeding, danwel lenen om te foto -kopiëren van artikelen. W. Spigt, Woetduinstraat 144-2, 1059 SZ Amsterdam.

* Te koop: LO-PROFILE toetsenbord f75,--. Tevens SHEIKOSHA GP50 printer f125,--. G. v. Avermaete, tel. 043 - 620836.

PROGRAMEREN MET PASCAL

Zoals u wellicht wel weet is het (althans voorlopig nog) onmogelijk te werken met files in HISOFT-PASCAL. Uitgaande van dit gebrek heb ik mij aan het werk gezet en ziehier het resultaat: een reeks procedures die het werken met files en streams (zoals in BASIC) simuleert.

Een eerste vaststelling: er wordt enkel gewerkt met reeds bestaande files d.w.z. files die in de catalogus van de disc aanwezig zijn (ze mogen dus wel nog volledig leeg zijn).

De procedures vormen in hun geheel een simulatie van het stream-systeem in basic. Simulatie betekent dat er in werkelijkheid geen (basic-)streams aan te pas komen. Wel definiëren we in Pascal onze eigen streams.

```
TYPE STREAM=RECORD
               VECT,BLK,BEG,EIND:INTEGER;
               UITPAG:BOOLEAN;
               BUF:BUFFER
             END;
```

Waarin BUFFER een ARRAY[0..255]OF CHAR vertegenwoordigt. Iedere stream zal dus 4*2+1+256=265 bytes in het geheugen opslorpen.

Het openen van een stream.

Vooraleer we een file kunnen gebruiken moeten we net zoals in basic een stream openen. Dit gebeurt door de procedure OPEN. Deze procedure vereist 2 parameters, nl. de stream die we willen gebruiken en de naam van de file die we willen openen. De eerste parameter is een variabele parameter, de tweede een waarde-parameter en is van het type 'WOORD:PACKED ARRAY[1..10]OF CHAR'. Daar het een waarde-parameter is kunnen we in onze parameter-lijst bij de procedure-oproep zowel een variabele van het type woord specificeren als een string van 10 letters direct in de parameter-lijst plaatsen.

De procedure OPEN zoekt nu in de catalogus (de 8 eerste blokken op de disc) naar de gespecificeerde file-name. Indien de file niet op de disc aanwezig is wordt de boodschap 'File not found' afgedrukt. Indien de file wel gevonden wordt, wordt de file-informatie in de 'stream' geplaatst, nl.

```
st.beg:=beginblok van de file op de disc
st.eind:=eindblok van de file op de disc
st.blk:=het huidige blok in de buffer
st.buf:= de inhoud van het huidige blok
st.vect:=het volgende karakter in de buffer dat moet
         geinterpreteerd worden.
st.uitpag:=false
```

Bemerk: st.beg<=st.blk<=st.eind

Nu we de file geopend hebben beschikken we over een direct toegankelijk Read/Write bestand.

Hoe lezen we nu uit een bestand?

Hiervoor gebruiken we de procedure INPUT. Deze procedure staat ons toe een variabele van een willekeurig type in te lezen. In

de parameter-lijst moeten volgende gegevens doorgegeven worden:

st: de gebruikte stream (variabele parameter)
len: het aantal bytes dat we willen lezen
adr: het adres vanaf waar we de gelezen bytes stockeren.

Op het eerste zicht lijken dit tamelijk onhandelbare parameters. Niets is echter minder waar. Indien we bijvoorbeeld de variabele var1 van het type type1 via STREAM1 zouden willen lezen dan doen we de oproep:

```
INPUT(STREAM1,SIZE(var1),ADDR(var1))
```

De procedure gaat nu de bytes (die reeds in de buffer van de stream aanwezig zijn) copieren in de variabele var1. Indien de procedure aan het einde van de buffer komt wordt er automatisch een nieuwe buffer ingelezen en of End of File gecontroleerd.

Hoe schrijven we nu naar een bestand?

Hiervoor gebruiken we de procedure OUTPUT. De procedure heeft dezelfde parameters als de procedure INPUT en wordt dus op dezelfde wijze opgeroepen.
Bij het schrijven naar een bestand is echter de grootste waakzaamheid geboden. De procedure schrijft na het recentst gelezene of het recentst geschrevene en overschrijft alles wat daar reeds aanwezig zou zijn!!!
Hier komt eindelijk het tot nu toe nogal mysterieuze record-veld 'uitpag' om de hoek kijken. Daar alle in- en uitvoer gebufferd wordt, zal niet iedere output direct naar de disc geschreven worden. Dus zullen we, voor we een nieuw blok van de disc willen lezen, moeten nagaan of we aan het vorige gelezen blok iets veranderd hebben, m.a.w. of we de procedure OUTPUT gebruikt hebben. Indien dit niet zo is (UITPAG=FALSE) kunnen we onmiddellijk overgaan tot het inlezen van een nieuw blok; indien dit wel zo is (UITPAG=TRUE) moeten we het huidige blok eerst wegschrijven en pas daarna het nieuwe blok inlezen.
Deze techniek wordt in vaktermen wel eens UITPAGineren genoemd.

Het uitpagineren is volledig transparant voor de gebruiker, d.w.z. hij hoeft er zich niets van aan te trekken. Alles wordt door de procedures zelf geregeld.

Direct toegankelijke bestanden?

Iedere file is inderdaad op elk moment direct toegankelijk en wel met een willekeurige recordlengte. Hiervoor dient de procedure (hoe kan het ook anders) POINT. Deze procedure is voorzien van drie parameters:

st: de stream
rnd: de recordlengte
pt: het nummer van het gewenste record

Zoals u ziet zullen we niet bij het openen van de file de recordlengte opgeven maar bij het gebruik van POINT zelf. Een oproep van deze procedure zal er dus als volgt uitzien:

```
POINT(STREAM1,SIZE(VAR1),12)
```

In het voorbeeld willen we dus het twaalfde element lezen in een file met elementen die allen van het zelfde type zijn van 'var1' Indien we bijvoorbeeld de 1302de byte willen lezen uit een file dan specifieren we een recordlengte van 1, dus de opdracht:

```
POINT(STREAM1,1,1302)
```

Indien u de stream weer op het begin van een file wilt positioneren geeft u de opdracht:

```
POINT(STREAM1,1,1)
```

nb. POINT kan eveneens gebruikt worden bij het schrijven.

Het CLOSEN van een stream.

Een stream moet enkel 'geclosed' worden als er zich in de buffer nog informatie bevindt die naar de disc geschreven moet worden. Indien u dus enkel van een bestand gelezen heeft, hoeft u de stream, alvorens hem opnieuw te openen of het programma te verlaten, niet te closen. De opdracht close voor bv. STREAM1 gaat als volgt:

```
CLOSE(STREAM1.BUF,STREAM1.BLK)
```

Dit is een directe oproep van de procedure die een buffer uit het geheugen naar een blok op de disc schrijft.
Voor het praktisch gebruik zijn dus slechts vijf procedures van belang:

-OPEN(stream,file-name) waarin file-name 10 karakters lang moet zijn;

-INPUT(stream,lengte,adres) waarin lengte de lengte is van de gevraagde variabele en adres het adres van de variabele in het geheugen;

-OUTPUT(stream,lengte,adres);

-POINT(stream,lengte,nummer) waarin lengte de recordlengte van de file is en nummer het volgnummer van de gewenste variabele;

-ClOSE(stream1.buf,stream1.blk) waarin stream1 de stream is die we willen closen;

Daarnaast hebben we nog de procedures 'LEES' en 'LEES1'.
LEES staat in voor het uitpagineren en roept daarna LEES1 op die dan instaat voor het lezen van een blok van de disc.

Bemerk dus tot slot dat de gebruikte streams in feite recordvariabelen zijn die naar behoeven van de gebruiker eventueel kunnen worden aangepast.

Rik Simoens
Belgie

Programmeren in Basic (2)

In het vorige artikel hebben we ons bezig gehouden met gestruktureerd programmeren door verfijningen. We gebruikten hiervoor procedures. In het voorbeeld dat toen behandeld is ,hadden we nog geen gebruik gemaakt van parameters.
Daar eerst nog maar een voorbeeld van:

```
100 DEF PROC vierkant zijde,beginx,beginy
     test.invoer zijde,beginx,beginy
105  PLOT beginx,beginy
     DRAW zijde,0
     DRAW 0,zijde
     DRAW -zijde,0
     DRAW 0,-zijde
110 END PROC

115 DEF PROC test.invoer zijde,beginx,beginy
120 IF beginx<0 OR beginx+zijde>255 OR beginy<0 OR beginy+zijde>
    175 THEN PRINT #0;"Invoerfout bij PROC vierkant"
    PRINT #1;"Druk op een toets"
    PAUSE 0
    STOP
125 END PROC
```

Deze beide proc's spreken voor zich.
Probeer maar eens : vierkant 100,80,50
en : vierkant 80,100,100

II.Dan nu iets recursiefs:

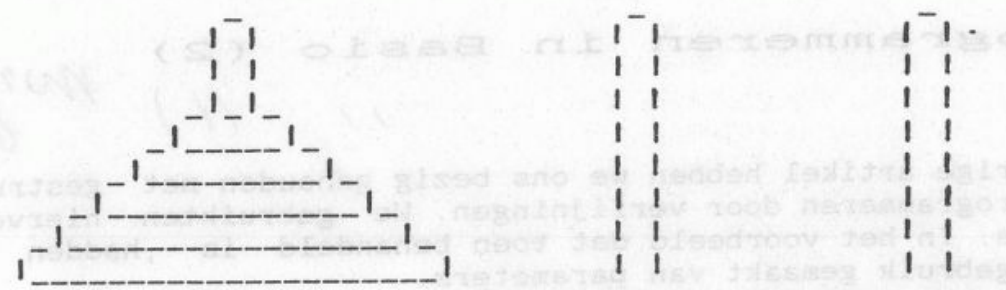
```
130 DEF PROC tunnel a,b,c
     vierkant a,b,c
132  Let a=.9*a,b=b+5,c=c+5
135  IF a<3 THEN STOP
140  tunnel a,b,c
145 END PROC
```

Je ziet ,dat in de defenitie van de procedure (de body) de procedure zelf wordt aangeroepen (regel 140).
Verder is in een recursieve proc. een afbreekvoorwaarde van essentieel belang, omdat anders de proc. zich tot in het oneindige blijft aanroepen.

Probeer eens : tunnel 150,10,10
en : verander regel 132 een paar keer.

Dan nu een klassiek recursief probleem.

De TOREN van HANOI.



ZUIL 1 ZUIL 2 ZUIL 3

Hierboven zijn drie zuilen afgebeeld. Zuil 1 bevat een willekeurig aantal schijven (in bovenstaande tekening 5). De andere 2 zuilen zijn leeg.
Het probleem is nu:
Zorg dat de schijven naar een andere zuil verhuizen , met de volgende spelregels:

a. Verplaats steeds maar een schijf tegelijk.
b. Een grotere schijf mag nooit op een kleinere liggen.

Bij een klein aantal schijven is het probleem niet zo lastig op te lossen ,maar het aantal "zetten" neemt bij toenemend aantal schijven snel toe.(1 schijf 1 zet
2 schijven 3 zetten
3 schijven 7 zetten
.
10 schijven 1023 zetten
n schijven 2↑n -1 zetten.)

We bekijken het probleem met 10 schijven en redeneren als volgt:
- Verplaats de bovenste 9 schijven naar bv zuil 2 (mbv zuil 3)
- Verplaats de 10-de schijf naar zuil 3
- Verplaats de 9 schijven van zuil 2 naar 3 (mbv zuil1).

Het bizondere van deze gedachtengang is, dat we het probleem van de 10 schijven hebben teruggebracht tot hetzelfde probleem , maar nu voor 9 schijven. Dit terugbrengen van een probleem tot zichzelf , maar dan van lagere orde is een typisch kenmerk van recursie.
Wat we kunnen van 10 naar 9 schijven, kunnen we natuurlijk ook van 9 naar 8 etc.

Dan nu het programma:

```
10 REM Toren van Hanoi.
   REM ---------------
20 CSIZE 3,7:OVER 1
   INPUT "hoogte van de zuil (max. 10) ";k
30 IF k>10 THEN GOTO 20
40 REM de schijven worden in strings gezet en geprint.
   REM t(i)=aantal schijven op zuil i.
50 DIM a$(k,k+1)
   FOR a=1 TO k
    LET a$(a)=STRING$(k-a," ")+STRING$(2*a+1,"*")
    LET a$(a)=a$(a)+STRING$(k-a," ")
    PRINT AT 20-k+a,0;a$(a)
   NEXT a
60 DIM t(3):LET t(1)=k,t(2)=0,t(3)=0
```

Dan nu twee procedures. De eerste is simpel; deze zorgt dat op het scherm de schijven verplaatst worden.
De tweede is de recursieve procedure...

```
100 DEF PROC beweeg.schijf n,van,naar
     PRINT AT 21-t(van),(van-1)*28;INK 7;a$(n)
     LET t(van)=t(van)-1
     PRINT AT 20-t(naar),(naar-1)*28;a$(n)
     LET t(naar)=t(naar)+1
110 END PROC

200 DEF PROC toren h,van,naar,via
     IF h>0 THEN  toren h-1,van,via,naar
                  beweeg.schijf h,van,naar
                  toren h-1,via,naar,van
210 END PROC

300 toren k,1,2,3
```

Analyse voor het geval k=3:

```
                          Zuil 1      Zuil 2    Zuil 3
                            *
toren 3,1,2,3              ***
   toren 2,1,3,2           *****
     toren 1,1,2,3      ----------------------------------
       h=0                  ***
     beweeg 1,1,2           *****        *
       h=0              ----------------------------------
   beweeg 2,1,3             *****        *         ***
   toren 1,2,3,1        ----------------------------------
     h=0                                             *
   beweeg 1,2,3             *****                   ***
     h=0                ----------------------------------
beweeg 3,1,2                                         *
toren 2,3,2,1                        *****           ***
   toren 1,3,1,2        ----------------------------------
     h=0                     *       *****           ***
   beweeg 1,3,1         ----------------------------------
     h=0                             ***
beweeg 2,3,2                 *       *****
toren 1,1,2,3           ----------------------------------
  h=0                              *
beweeg 1,1,2                      ***
  h=0                            *****
                        ----------------------------------
```

Dit was moeilijk genoeg.
Als je het bovenstaande schema bekijkt dan zie ,dat op een bepaald moment 3 versies van de procedure gelijktijdig actief zijn (toren 3,toren 2 en toren 1).

Tot slot, voor de liefhebber nog een listing van een ander recursief programma.

Programma voor de kromme van Peano.

```
100 DEF PROC noord n
     IF n=0 THEN
      ELSE oost n-1
           lijn 1,0
           noord n-1
           lijn 0,1
           noord n-1
           lijn -1,0
           west n-1
105 END PROC
110 DEF PROC oost n
     IF n=0 THEN
      ELSE noord n-1
           lijn 0,1
           oost n-1
           lijn 1,0
           oost n-1
           lijn 0,-1
           zuid n-1
115 END PROC
```

```
120 DEF PROC zuid n
     IF n=0 THEN
      ELSE west n-1
           lijn -1,0
           zuid n-1
           lijn 0,-1
           zuid n-1
           lijn 1,0
           oost n-1
125 END PROC
130 DEF PROC west n
     IF n=0 THEN
      ELSE zuid n-1
           lijn 0,-1
           west n-1
           lijn -1,0
           west n-1
           lijn 0,1
           noord n-1
135 END PROC
```

```
140 DEF PROC lijn ow,nz
     PLOT x,y
     DRAW ow*d,nz*d
     LET x=x+ow*d,y=y+nz*d
145 END PROC

150 REM uitvoering progr.
        ******************
160 LET n=6,d=INT(170*2↑(-n))
    LET x=40+d/2,y=d/2
    noord n
```

In regel 160 kun je de waarde n nog wat wijzigen , daardoor ontstaat een groffere of fijnere tekening op het scherm.
Een analyse van dit programma is een hoofdstuk apart.
Iets voor de puzzelaar ??

N.L.A.F. Simons.

ULTI-MATE (which game?)

Enkele wisten dat er aan gewerkt werd, maar het was nog maar de vraag f het lukken zou. En het was helemààl afwachten wanneer het klaar zou zijn.

Vlak nadat D.U.C. #8 naar de drukker gebracht was, verscheen er (plotseling) een test-uitgave. Het programma liep, maar niet vlekkeloos. Daar was dan ook alles mee gezegd. De schoonheidsfoutjes werden eruit gehaald, er werden nog wat zaken aangepast, het programma werd nog eens binnenstebuiten gekeerd. Een definitief test-exemplaar werd gemaakt.

Nog voordat D.U.C. #8 bij U in de bus gevallen was waren de testers het al eens over de nieuwste DISCOVERY-telg. Ze hadden aan de wieg gestaan van het ultieme RUN-programma. Daar komt ook de naam vandaan:

=== ULTI-MATE ===.

De ULTI-MATE zal gaan fungeren als standaard voor een (toekomstige) familie van nieuwe RUN-programma's. Het zal nog tijden duren voordat ULTI-MATE door BASIC-RUN-programma's geëvenaard wordt.

Ook als je tevreden bent met je eigen RUN-programma's, moet je toch eens naar de ULTI-MATE kijken. Alle andere RUNners verbleken bij dit programma. En deze loopt veel sneller!

De eerste kennismaking met de ULTI-MATE was op de D.U.C.-dag van 17 januari in Utrecht. Op die dag werd de ULTI-MATE geintroduceerd.

De ULTI-MATE is in machinecode geschreven. Een eigenschap van kleine MC-routines is dat uiterlijk-schoon geheel genegeerd is om de routine zo klein mogelijk te houden.

Toegegeven, van uiterlijke schoonheid moet de ULTI-MATE het niet hebben. Handhaven van die schoonheid zou de ULTI-MATE vertragen, en dat wou de schrijver niet. De (zichtbare) schoonheid is meer toegepast vanwege praktische doeleinden.

Het CODE-blok van de ULTI-MATE is zeer klein. Iets minder dan 1900 bytes zijn er gebruikt voor het uiteindelijke resultaat. En door de machinecode is de ULTI-MATE ontzettend snel.

Omdat bepaalde zaken in machinecode moeilijk te implementeren zijn worden deze vaak eenvoudig gehouden. Ook de ULTI-MATE had een implementatie-fase (veel ideeën, maar nu nog uitvoeren!) en daardoor is de bediening van de ULTI-MATE erg simpel gemaakt.

De opties van de ULTI-MATE zijn:

- LISTEN.
  + list alleen BASIC-laders
  + Op het scherm worden ze vertoond in drie kolommen
  + CHR$-display-mogelijkheid van TRINIDAD is gebruikt.
  + Per pagina 63 BASIC-laders vertoonbaar op scherm.
  + Meerdere pagina's mogelijk.
  + Melding indien geen BASIC-programma's aanwezig.

- DISCETTES.
  + Maximum van een CATalog is 6.25 Kbyte (normaal: 1.75Kb).
  + Alle sector-groottes worden geaccepteerd (128, 256, 512 en 1024 byte per sector).

+ abnormale omvang discettes geen probleem (203Kb en 718Kb)

- PRINTER.
  + kan alle gevonden BASIC-namen afdrukken op de printer.
  + aantal namen per printer-regel instelbaar.
  + vreemde CHR$'s worden -door een zelf te kiezen tekentje- vervangen (instelbaar).

- JOYSTICK.
  + Je kunt kiezen of je de joystick-poort aan/uit wilt hebben.
  + Kan standaard ingesteld worden.
  + De toekostige status van de j.s.-poort is continu op het beeldscherm weergegeven.

- DISC-DRIVES.
  + standaard-drive instelbaar (1 of 2).
  + wisselen van drive mogelijk tijdens het RUNnen.
  + wisselen van drive veroorzaakt het uitlezen van een disc in de nieuwe drive.
  + Als je slechts één drive bezit is het mogelijk de wissel-optie te negeren (d.m.v. een POKEje).
  + Welke drive gebruikt wordt is continu op het beeldscherm te zien.

- CURSORS.
  + 'Current'-BASIC-lader wordt door een duidelijke cursor-balk aangegeven.
  + Cursor-balk 'vliegt' a.h.w. over het scherm heen.
  + Bewegingen van de cursor-balk d.m.v. de cursor-toetsen.

- HANDLEIDING.
  + Handleiding aanwezig.

De testers van de ULTI-MATE zijn enorm enthousiast. Het is het RUN-programma waar we zolang op hebben zitten wachten. Je zou kunnen zeggen dat het een professioneel programma is. Een van de testers meende zelfs dat mensen met een IBM-PC jaloers zouden zijn op zo'n opstart-programma. Dit alles wil toch wel iets zeggen over de ZX SPECTRUM/DISCOVERY-combinatie.

Er is lang nagedacht over de distributie van de ULTI-MATE. Uiteindelijk is er gekozen voor de programmabank (tegen betaling), vergezeld door een behoorlijke promotie voor het programma (en de club) in buiten- en binnenland.

De ULTI-MATE is verkrijgbaar voor fl. 12,50 (inclusief discette en bijkomende kosten) via de programmabank. Lees in de colofon hoe je bij de programmabank kan bestellen.

Voor die fl. 12,50 krijg je twee keer ULTI-MATE (één geschikt voor drive 1, de andere voor drive 2), en een handleiding als TASWORD III en TASWORD II file.

Wil je de handleiding uitgeprint erbij hebben dan kost dat fl. 1,50 extra.

Mike

CALPHY (2)

(of ..... gestructureerd programmeren in machinetaal.)

Om een aantal redenen heb ik een vervolg geschreven op het vorige CALPHY-artikel.

De eerste reden is dat het artikel een enorm succes was. Vele reacties heb ik gekregen van mensen die nu eindelijk (zij het wat schoorvoetend) in machinetaal met de DISCOVERY zijn gaan werken. Het was het artikel waar ze al zo lang op hadden gewacht

De tweede reden is dat ik iets wil hebben tegenover het artikel van de heer Simons (over het programmeren in BASIC). In het rijtje van opsommingen betreffende talen waarin men gestructureerd kan programmeren ontbrak de machinetaal. Toegegeven, als je zo kijkt naar een assembly-listing dan zie je niet zo snel wat het doet. Maar juist de machinetaal dwingt je tot het leggen van bepaalde structuren in routines. Dit geldt nog niet zo bij kleine routines, maar bij grotere scheppingen (vanaf 500 bytes) is de structuur een noodzakelijk bestanddeel van je programma.

De derde reden is dat ik hoop met deze artikelen-serie een groep mensen te krijgen die de machinetaal-werkgroep gestalte gaat geven. Want laten we wel wezen, alle aanpassingen en programma's die ook maar enigszins voor de DISCOVERY geschikt gemaakt moeten worden, moeten allemaal vanuit die werkgroep komen. En niemand heeft noch aan de machinetaal-werkgroep gedacht.

Iets rechtzetten ...

Ik moet twee opmerkingen plaatsen bij het vorige artikel. Ergens staat 'DJNZ BC', maar dat moet 'DJNZ loop' zijn (zoals uit de begeleidende tekst wel blijkt).

Ik kreeg ook nog twee reacties van mensen die geprobeerd hadden met het 5e routine de RAMDISC te formatteren, maar eindigden met een foutmelding. Zij waren vergeten de RAMTOP te verlagen met een CLEAR-opdracht zodat een RAMDISC mogelijk was. U over- komt toch niet hetzelfde?!

STACK-gebruik

(Red: een STACK is een lijst die op zodanige manier wordt opgesteld en bijgehouden, dat het eerstvolgende gegeven dat zal worden opgezocht of verwijderd, het gegeven is, dat het laatst aan die lijst wordt toegevoegd.)

We hebben bij de Z80 een stack. Er zijn massa's instructies die met de STACK werken. Ik heb bijv. in het vorige artikel veel gebruik gemaakt van de STACK.

Als ervaren machinetaal-programmeur moet ik beginners echter waarschuwen voor het gebruik van de STACK. De STACK is meestal de bron van alle ellende. Zelfs nu nog gaan er bij mij routines de mist in door ongeoorloofd gebruik van de STACK.

Omdat ik in dit artikel de STACK ga manipuleren (d.w.z. echt het uiterste van de STACK ga vergen) ben ik genoodzaakt de registers niet meer op de normale manier te bewaren. Ik bewaar ze in instructies!

In plaats van een veld te creëren waar de registers netjes geparkeerd worden en na verloop van tijd weer vandaan worden op-

gehaald, schep ik die parkeerplaats op een andere manier. Ik berg de waarde op in een LOAD-instructie. Dat gaat als volgt:

```
        LD    (pr+1),HL
        ...
        ...
        ...
pr      LD    HL,0
```

Op de plaats waar de '0' staat wordt de waarde van HL 'geparkeerd'. Even later komt de routine langs 'pr' en de waarde van HL wordt weer hersteld.

PRINTER

De print-routine van de vorige keer werkt goed. Alleen, als je 25 verschillende zinnetjes hebt die je naar de printer kunt sturen heb je er weinig aan. Je kan toch niet 25 verschillende routines maken die in feite toch hetzelfde doen.

Ik wil een procedure definiëren -we noemen haar PRINTER- die een tekst naar de printer stuurt. Welke tekst weet ze niet! De definitie ziet er als volgt uit:

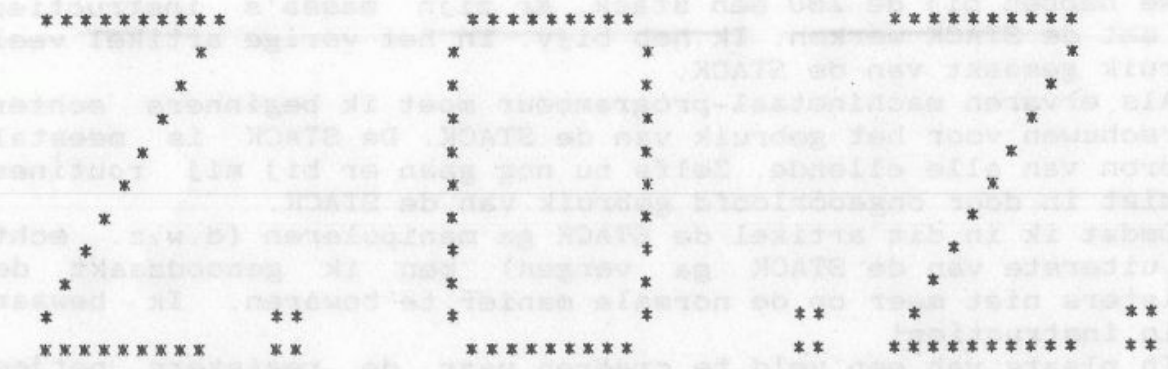
```
DEF PROC PRINTER text
    ; PRINTER stuurt de tekst aangewezen
    ; door 'text' naar de printer.
END PROC
```

Wat we dus nodig hebben is een stukje routine dat in staat is 'text' op te halen. Maar geven we eerst wat andere regels:

```
prop_1   DEFB 3,13,10,15                ; proportioneel schr. aan
prop_0   DEFB 3,13,10,18                ; prop. schr. uit
onds_1   DEFB 3,27,45,1                 ; onderstrepen aan
onds_0   DEFB 3,27,45,0                 ; onderstrepen uit
text_1   DEFB 20 ; leuke tekst
         DEFM "(c)1987 R.O. Aalders"
```

De waarden zoals hier gegeven gelden voor de S.C. FASTEXT 80. Voor de waarden van Uw eigen printer moet U in de handleiding van Uw printer kijken.

PRINTER gaat er als volgt uitzien:



```
PRINTER  EX   (SP),IX
         PUSH HL
         LD   L,(IX+0)
         LD   H,(IX+1)
         INC  IX
         INC  IX

         POP  HL
         EX   (SP),IX
         LD   (PR3+1),HL
         LD   (PR4+1),DE
         LD   (PR5+1),BC
         LD   (PR6+1),A
         LD   (PR7+2),IX
         CALL #1708
PR1      LD   HL,0
         LD   B,(HL)
PR2      PUSH BC
         INC  HL
         PUSH HL
         LD   B,0
         LD   A,#81
         LD   H,(HL)
         CALL calphy
         POP  HL
         POP  BC
         DJNZ PR2
         CALL #1748
PR3      LD   HL,0
PR4      LD   DE,0
PR5      LD   BC,0
PR6      LD   A,0
PR7      LD   IX,0
         RET
```

Alle registers behouden hun waarde. De 2 bytes achter de CALL worden opgehaald en in de instructie bij 'PR1' geparkeerd. Dit is eerder al uitgelegd.

Het bewaren van alle belangrijke registers. Dit is eerder al uitgelegd.

Dit is de routine van de vorige keer, een beetje aangepast. Let erop dat de '0' in de instructie bij 'PR1' vervangen wordt door het adres waar de tekst staat.

Het herstellen van de registers

Klaar.

Hier staat nu de routine PRINTER. De kracht van de routine blijkt uit het volgende voorbeeldje:

```
         CALL PRINTER
         DEFW prop_1
         CALL PRINTER
         DEFW onds_1
         CALL PRINTER
         DEFW text_1
         CALL PRINTER
         DEFW onds_0
         CALL PRINTER
         DEFW prop_0
```

Dit drukt proportioneel

(c)1987 R.O. Aalders

af op een nieuwe regel papier, waarna naar een nieuwe regel papier gesprongen wordt.

Dit gedeelte is in feite nog gesneden koek voor allen die het vorige CALPHY-artikel aandachtig gelezen hebben. Zij zien in feite niet meer dan een methode van programmeren.

Ikzelf gebruik PRINTER in deze vorm niet. Ik heb in bepaalde routines een verbeterde versie van PRINTER zitten. Het voorbeeld kan ik met die routine als volgt doen:

```
        CALL PRINTER*
        DEFW prop_1,onds_1,tekst,onds_0,prop_0
```

De puzzelaars onder ons (de Hofnar?) mogen zoeken hoe ik dat voor elkaar gekregen heb.

Catalog-lader CATALOG

Na het eenvoudige werk komt nu het moeilijke. De routine die een CATALOG ophaalde (uit het vorige artikel) werkte alleen met normale discettes. Dat heet: 'met sectoren van 256 bytes en een catalog-grootte van 7 sectoren'. Andere discettes waren (nog) niet haalbaar voor onze leden. Echter het artikel van Marcel van Dongen gooide roet in het eten. Daar ging mijn UTOPIA!

De CAT-LOADER die ik in het tweede gedeelte van dit artikel ga ontwikkelen is geschikt voor alle discettes! Beginnen we met het schrijven van een routine die in staat is een hele sector binnen te halen:

```
SECTOR   LD   (SEC2+1),HL
         LD   (SEC3+1),DE
         LD   (SEC4+1),BC     \ Bewaren van de registers
         LD   (SEC5+1),A      /
         LD   (SEC6+2),IX
         CALL #1708
         LD   B,2
SEC1     LD   C,0             \
         LD   A,(DRIVE)       / De routine die een volle sector
         CALL calphy          / ophaalt (zie vorig artikel).
         CALL #1748
SEC2     LD   HL,0
SEC3     LD   DE,0
SEC4     LD   BC,0            \ Herstellen van de registers
SEC5     LD   A,0             /
SEC6     LD   IX,0
         RET
```

So far so good. De aanwezigheid van het label (adreskaartje) van 'SEC1' leg ik in het derde gedeelte wel uit. Nu moeten we een procedure schrijven die in staat is de CAT- alog van de discette te lichten. De procedure moet dus de disc- ette onderzoeken naar de samenstelling, en daarna sector na sector op de juiste plaats in het geheugen transporteren. Schem- atisch (BASIC-geschreven) ziet het er dus als volgt uit:

```
DEF PROC CATALOG
    ; onderzoek de discette
    FOR B=0 TO (lengte CAT -1)
        pak de sector ‹B› van discette
        plaats deze op de aangewezen plaats
    NEXT B
END PROC
```

Het onderzoeken van de discette gebeurt met een onderdeel van CALPHY, nl. de 'enquiry'-optie bij drives. Na het onderzoek van CALPHY draagt het BC-registerpaar de sectorgrootte, het DE regis -terpaar de CATalog-grootte (in sectoren uitgedrukt) en het HL-

registerpaar het totaal aantal bruikbare sectoren op de discette Aan de BC- en DE-registerparen hebben we wat!

We moeten het opbergen van de sector- en catalog-grootte zien als het opbergen van twee maal twee bytes achter de CALL. Voor het gemak echter (en in zijn algemeenheid is dat ook zo) ga ik er vanuit dat een CATALOG met meer dan 255 sectoren niet voorkomt. Van het DE-registerpaar hebben we dus alleen het E-register nodig. Dit allemaal wetende gaan we beginnen aan de routine:

```
CATALOG  LD   (CAT4+1),HL
         LD   (CAT5+1),DE
         LD   (CAT6+1),BC
         LD   (CAT7+1),A
         LD   (CAT8+2),IX
         CALL #1708
         LD   A,(DRIVE)
         LD   B,4
         CALL calphy
         LD   A,E
         LD   (CAT1+1),A
         LD   (CAT3+1),BC
         CALL #1748
         LD   DE,RAMCAT
         LD   HL,0
CAT1     LD   B,0
INCB
CAT2     PUSH BC
         PUSH HL
         PUSH DE
         CALL SECTOR
         POP  DE
CAT3     LD   HL,0
         ADD  HL,DE
         EX   DE,HL
         POP  HL
         INC  HL
         POP  BC
         DJNZ CAT2
CAT4     LD   HL,0
CAT5     LD   DE,0
CAT6     LD   BC,0
CAT7     LD   A,0
CAT8     LD   IX,0
         RET
```

Opbergen van de registers

Onderzoek van de discette op dat moment aanwezig in ‹DRIVE›.

Het informeren van de registers over de plaats, de lengte en begin van de CATalog.

Verhogen van de wijzer waar de volgende sector geplaatst moet worden in het geheugen.

Volgende sector.

De SECTOR-LOOP. Zie het vorige artikel voor de bijzonderheden.

Afsluiten van de routine

Zo moeilijk was het ook weer niet. Zoals je nu kan zien is het werken met sectoren heel eenvoudig. Misschien zie je nu ook wel in dat routines zoals TRINIDAD (D.U.C. #6) en MOVER (D.U.C. #7) uit vrij eenvoudige bouwsteentjes zijn samengesteld. En echt, je moet durven om zulke grapjes uit te halen. Je moet natuurlijk niet direct je beste discettes aan de grilligheden van het machinetaal-programmeren bloot stellen, maar zoals je ziet wordt het allemaal d.m.v. eenvoudige routines gedaan.

PROJECT

Het derde deel van dit artikel gaat over SMALL- en SUPER-CAT, de handige routinetjes van Sander Plomp. Ik wil jullie allemaal de gelegenheid geven een aanval te doen op die routines. We kunnen ze sneller maken, en het terugspringen van de kop naar de CATALOG is niet meer nodig.

Zoals een ieder weet bestaan de eerste 7 bytes van iedere file op disc uit informatie betreffende de file. Als we dus naast de gegevens van de catalog-file ook nog die eerste 7 bytes van iedere file erbij hebben, dan spuien we zo alle gegevens (en kenmerken) van de files op.

Dit is eerder gezegd dan gedaan. Maar laten we beginnen:

```
BUFFER   DEFS 23
```

In de buffer komen straks eerst de gegevens uit de CATalog-file te staan, gevolgd door de 7 bytes file-gegevens van de file zelf.

```
MOVE_CAT LD   DE,BUFFER
         LD   BC,16
         LDIR
```

'MOVE-CAT' zorgt ervoor dat de CATALOG-gegevens van de file die door het HL-registerpaar worden aangegeven naar de BUFFER worden overgeseind.

```
GET_INFO LD   (GET1+1),HL
         LD   (GET2+1),DE     \
         LD   (GET3+1),BC      > Opbergen van de registers (IX
         LD   (GET4+1),A      /  dit keer niet).
         LD   A,7
         LD   HL,(BUFFER+2)   \
         LD   DE,BUFFER+16     > Zetten van de gegevens voor de
         LD   (SEC1+1),A      /  subroutine 'SECTOR'.
         CALL SECTOR
         XOR  A               \
         LD   (SEC1+1),A       > Herstellen van de subroutine
                              /  'SECTOR'.
GET1     LD   HL,0            \
GET2     LD   DE,0             > Herstellen van de registers
GET3     LD   BC,0            /
GET4     LD   A,0
         RET
```

'GET_INFO' pakt de file-informatie van de file aangewezen door het HL-register in de RAM-CATALOG netjes van disc af en plaatst deze achterin de BUFFER. Van de file zijn nu alle beschikbare gegevens in de BUFFER opgeslagen, geheel gereed om door de gebruiker geplukt te worden.

Natuurlijk moet je zelf ook wat sleutelen. Je moet zelf inbouwen hoe je routine weet dat er geen files meer aanwezig zijn, en andere grappen. In ieder geval heb je nu voldoende stof tot nadenken.

Ook zie je in dit project het nut van de label 'SEC1'. Ik kan met dat label aardige dingen doen met het laden van een specifiek aantal bytes van een discette-sector.

Opmerkinen en andere zaken svp. via de redactie.
Rudie Aalders.

Programmapakket bespreking

Disassembler/Monitor DISMON 16
DISMON 48
Assembler MICASS 2.0
Macro-uitbreiding voor MICASS 2.0
DISASSEMBLER OPTIE VOOR MICASS 2.0

Als een Monitorprogramma eenmaal in het geheugen van de SPECTRUM zit, staat er een veelheid aan kommando's ter beschikking van de gebruiker. Om er een van uit te voeren, volstaat een druk op een toets. Wat er dan gebeurt is afhankelijk van het programma. Wanneer Decimaal en Hex genoemd worden, wordt de laatste meestal voorafgegaan door een 'h'. Naast de standaard mogelijkheden en kommando's van een MONITOR/ DISASSEMBLER, biedt deze Disassembler/Monitor de volgende extra's.

1) Willekeurige groottes van blokken kunnen naar cassette worden geSAVEd, geVERIFieerd en terug geladen. Dit gebeurt in een bijzonder tempo.

2) De MONITORfunctie kan ook achteruit door het geheugenbereik bewegen, dwz. bij een hogere adres beginnen en naar nul toe werken. Een verdere bijzonderheid is de zoekfunctie. Met behulp hiervan kan het geheugen worden afgezocht naar een door de gebruiker geplaatste STRING van max. 32 tekens lengte. Deze STRING mag naar keuze uit letters of cijfers bestaan.

In sommige gevallen kan zelfs vanuit de DISASSEMBLER de ASSEMBLY-optie worden opgeroepen. Hierover later meer. Een stapsgewijs uitvoeren van machinecode (TRACE) is ook mogelijk. Printerbesturing loopt vlekkeloos; de juiste tekens worden via de bekende st. 3 verzonden. Alles wat met #3 werkt, moet dus ook met dit progamma werken. Op naar MICASS 2.0.

Gelukkig werkt dit programma niet alleen met een cassette-recorder, maar ook met Microdrive, Beta-Disc en OPUS Discovery. Het programma werkt echter alleen op de 48k SPECTRUM, maar wie heeft tegenwoordig nog een 16k? Na het laden en initiëren van het programma verschijnt een copyright-boodschap en wordt het geheugen aangegeven dat ter beschikking staat van de scourcecode. Vrij geheugen is ongeveer 3K (machinecode). Men kan natuurlijk ook langere programma's schrijven; deze worden dan stapsgewijs ge-assembleerd, hoog in geheugen gestopt en later als een geheel weggeSAVEd. De gebruikte labels blijven daarbij natuurlijk behouden; slechts de text- en objectcodes worden gedumpt.

Alle voor de assembler 'gereserveerde' woorden (kommando's) moeten voluit ingetypt worden (keywords bestaan niet in het programma); afkortingen worden echter wel toegestaan. Behalve de kommando's voor het SAVEn naar cassette, zijn er ook de benodigde kommando's voor bovengenoemde massa-opslag-apparaten; u hoeft slechts een '*' voor het kommando in te toetsen. Bij het werken met microdrive en discettes dient u op het volgende te letten: de file-headers geven niet het startadres van de machinecode aan, maar het gebied waar de assembler ze heeft gestopt. Deze moeten natuurlijk worden geladen met gebruikmaking van de startadres.

De Editor van het programma is redeljk te noemen, behalve wat betreft de cursor die na een SPACE naar de volgende regel springt. Hierdoor loopt men het gevaar een Operator in een labelfield in te voeren.
Assembleren van text gaat vlot van het handje, labelwaarden en adressen worden goed getoond, en de lengte van de objectcode wordt aangegeven. Normaal gesproken wordt 'in de achtergrond' geassembleerd, maar bij dit programma kan dat naar believen op het beeldscherm of via de printer geschieden.

Absoluut uitblinker wordt deze assembler zonder twijfel door zijn Macro-kapaciteit. Het programmeren krijgt een nieuwe dimensie door de intrede van macro's. De gevorderde, inmiddels beruchte machinecode-tovenaars onder ons, wordt nu de gelegenheid geboden hun kreaties met nog meer raffinement te doorspekken.
Voor de 'zielige beginnelingen' onder ons: 'Macro's zijn kommando's die de processor eigenlijk niet kent, maar die hij eigenlijk ook niet te verwerken krijgt. De programmeur stelt als het ware deze kommando's samen uit een aantal reeds bekende kommando's, en die worden dan als een geheel voorgelegd aan de processor. Wanneer de processor deze 'nieuwe' kommando's in de listng tegenkomt, vervangt hij deze door de individuele kommando's. Da's niets bijzonders zullen verwende lieden onder ons brommen. Maar het geniep zit hierin, dat de parameters worden overgeheveld en dat assembly onder bepaalde kondities kan plaatsvinden. De assembler vertaalt alle overgehevelde registers, adressen, waarden en wat er dan ook met vlaggen gemerkt wordt in de macrotext. Ter verduidelijking het volgende.

```
0010 ;
0030 LDINDEX     .MAC HREG,LREG,OFFSET
0040             IF   "HREG ="S
0050             THEN EENVOUDIG
0060             LD   HREG,(IX+OFFSET+1)
0070 EENVOUDIG   DL   $
0080             LD   LREG,(IX+OFFSET)
0090             ENDM
0100 ;
```

Deze macro zorgt ervoor dat er een in het IX-register genoemde eenvoudige of meervoudige waarde tegelijk wordt ingeladen.
De oproep zelf is betrekkelijk eenvoudig:

```
0150             LDINDEX * S,A,100
```

Bij het assembleren wordt dit vervangen door:

```
0150             EINZEL  LD A,(IX+100)
```

Een meervoudig register wordt als volgt ingeladen:

```
0160             LDINDEX * H,L,45
```

De code bij het assembleren:

```
0160             LD H,(IX+46)
0170  EENVOUDIG LD L,(IX+45)
```

Dit eenvoudige voorbeeld toont duidelijk waartoe een macro-assembler in staat is. Het is echter aan te raden eerst wat met de mogelijkheden te experimenteren, daar niet alle kombinaties automatisch worden geaccepteerd. De inleiding toont wel een paar voorbeelden, maar men mag natuurlijk niet verwachten dat alle mogelijkheden en oplossingen hier worden getoond!
Een andere bijzondere kapaciteit is het retro-definiëren van labels. In kombinatie met de IF-functie kan men konstructies opbouwen.

De Reassemble-optie vereist dat assembler en monitor in het geheugen aanwezig zijn. Voor ongeletterden: een Reassembler maakt uit een lopende machinecode een assembler-textfile. Deze kan dan weer door de Editor bewerkt worden. (Vooropgesteld natuurlijk, dat men enigzins vertrouwd is met de functies van de codes die gere-assembleerd moeten worden. Men moet tenslotte de Assembler enigzins voor de gek houden. Hierover staat in de handleiding het een en ander.
Zwakke plek van het geheel vind ik persoonlijk de inleiding van het installatie-gedeelte. Om alle programma-onderdelen juist in de computer onder te brengen moet men dit gedeelte zeer oplettend lezen!!!

Naar believen kan de gebruiker ook een konversieprogramma ontvangen dat uit de scource-file van de EDITAS Assembler (van PICTURESQUE), een MICASS 2.0 - Sourcefile kan maken. Oude EDITAS-rotten zoals ik zullen dit zeer waarderen; onze oude programma's krijgen weer een vernisje! De allereerste versie liep niet helemaal goed, maar de jongste versie is perfect!

In Duitsland verkrijgbaar bij: MSB-Software,
Michael Stramm - Ratscher Str. 155/1513 - 5100 Aachen.
Prijs: DM 40,- Assembler, Dismon 16/48
DM 10,- Macro- en Reassembleroption

Het hier besproken programmapakket is een onontbeerlijk hulpmiddel voor het programmeren in Assemblytaal. Het wordt geleverd voor de volgende opslagsystemen:

- Microdrive de tape-option wordt meegeleverd
- Discovery
- Beta Disc

Het pakket wordt op cassette geleverd, met Duitstalige handleiding. Ik heb de DISCOVERY-versie getest. Dit is het eerste pakket dat zowel de drive- als de printeruitgang benut. Ook bij de Beta-Disc-versie is er sprake van een van de eerste Assembler/Monitors die het besturingssysteem rechtstreeks aanspreken. Dit maakt het programmapakket zeer sympathiek; zelfs in het programma hoeft de gebruiker niets meer te veranderen!

R. Frank & K. Mùller

HET SPEELHOEK

Een van onze jeugdige lezers, Marco Snijders uit Heerhugowaard, vindt dat er in ons blad te weinig aandacht word besteed aan spelletjes. Daarom heeft hij de stoute schoenen aangetrokken en zelf wat geschreven.

**********ROCK 'N WRESTLE**********
Uitgever: Melbourne House Waardering: 3

Als u weleens naar SKY CHANNEL heef gekeken, dan heeft u vast wel het worstelen gezien. Dit spel is op dezelfde sport geënt. Zelf bent u Georgeous Greg en u moet het opnemen tegen:

* Redneck McCoy
  Hij is makkelijk te verslaan. Kijk wel uit dat hij u niet in de lucht krijgt, want dan draait hij u heen en weer in de lucht (de Aeroplane spin)!!

* Missouri Breaker.
  Deze tegenstander is gelijk wat moeilijker, en kan trucs toepassen die u als speler niet kan. Zo gooit hij u de lucht in als u hem probeert te pakken.

* Vicious Vivian.
  Deze tegenstander is bijna niet te verslaan; hij heeft op iedere greep een antwoord.

* Lord Toff.
  De kampioen van de worstelaars. Hij kent alle bewegingen. Niet te verslaan.

Het spel biedt de volgende mogelijkheden:
* scoreverloop tijdens het spel (onthoudt de hoogste score niet)
* demonstratie mode
* extra energie
* meerdere toets-mogelijkheden, afhankelijk van de situatie.

Marco is duidelijk een spelletjesliefhebber, getuige het vervolg van zijn brief: tips om POKEs in je spelletjes te verwerken.

Hoe voeg je POKEs in je spel?
*1 Laad je programma door MERGE "".
*2 Als er OK. 0.1 verschijnt, stop dan de band en toets LIST.
*3 Haal d.m.v. EDIT de regel omlaag waarin het laatste RANDOMIZE USR statement staat. Als dit een regel 0 is, POKE dan eerst 23765,a (a=gewenste regelnummer; 1/2/3/.....).
*4 Voeg de POKE in vr het laatste RANDOMIZE USR statement en RUN dan het programma.

Hier volgen enkele POKE's voor oneindig leven.
Sai Combat: POKE 65354,201: POKE 32421,1
Saboteur: POKE 29894,0
Sweevo's world: POKE 33219,0
Pentagram: POKE 49917,0

Twee POKE's for TRANZ AM: 28610,0 maakt het spel moeilijker, en 98573,0 maakt het ng erger!

Manic miner: 65132,0
Jetset Willy: 35899,0

Een programma om JACK THE NIPPER slim af te zijn:

```
  5 RESTORE
 10 LET w=1
 15 LET TOT=0
 20 FOR A=49998 TO 50070
 25 READ B: LET TOT=TOT+B*W
 30 LET W=W+1
 35 POKE A,B
 40 NEXT A
 45 IF TOT <> 319540 THEN PRINT "FOUTJE!": BEEP 1,1: STOP
 50 PRINT AT 14,1;"START DE BAND"
 55 RANDOMIZE USR 49998
 60 DATA 0,221,33,203,92
 61 DATA 17,116,1,62,255
 62 DATA 55,205,86,5,210
 63 DATA 78,195,33,135,195
 64 DATA 17,213,93,1,17
 65 DATA 0,237,176,33,35
 66 DATA 191,27,99,26,93
 67 DATA 33,209,255,237,99
 68 DATA 33,93,175,61,50
 69 DATA 167,93,33,159,209
 70 DATA 237,99,165,93,195
 75 DATA 0,93,175,50,0
 80 DATA 170,33,0,0,237
 85 DATA 99,1,170,175,211
 90 DATA 254,55,201
100 DATA 0
101 STOP
```

‹‹‹‹‹‹‹‹‹‹‹‹››››››››››

Spelers van FRANK BRUNO'S BOXING opgelet! Gebruik de naam AND voor onverwachte resultaten.

BOXER 2 MM710F49B
BOXER 3 B7XI00L05
BOXER 4 FK5IN0A07
BOXER 5 CE9IN9817
BOXER 6 IHCIN96A8
BOXER 7 ML6ION4B6
BOXER 8 BFAINN2L5
Voor het finale effect: CGAINA5CA

Marco Snijders
Heerhugowaard

‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹‹››››››››››››››››››››››››››››››››››››

PROGRAMMABANKNIEUWS

Om misverstanden te voorkomen heet de BIBLIOTHEEK voortaan BANK, zoals oplettende lezers in de kop reeds zullen hebben gezien. Het is tenslotte zo dat u de programma's niet leent, maar aanschaft.

Verder staat er in de kop "nieuws" en dat is er dan ook: DUCDISK-2 is uit! Na 21 maart a.s. kunt u hem bestellen voor de all-in prijs van f. 12,50. All-in wil zeggen dat daar niets meer bij komt, geen extra kosten voor de schijf en geen extra kosten voor de porto. Tegelijkertijd geldt voor DUCDISK-1 een all-in prijs van f. 10,00. Het prijsverschil wordt veroorzaakt door het op DD-2 gebruikte RUN-programma dat geheel in machinetaal is geschreven en waar een vergoeding van f. 2,50 voor wordt afgedragen aan de maker. Op 21 maart a.s., in de Bron in Utrecht zal DUCDISK-2 worden geïntroduceerd. We zullen proberen een behoorlijk aantal kant-en-klare DD-2's in voorraad te hebben.

De volgende programma's staan op DD-2:

CHARSTW2/3	Hiermee kunt u in TASWORD-2 en TASWORD-3 de karakterset zoals die op het beeldscherm verschijnt aanpassen.
TKST/BOEKH	Een in BASIC geschreven tekstverwerkings / boekhoudprogramma.
SCRABBLE	Geschreven in BetaBasic 3.0
CHARMASTER	Uitgebreid programma om karaktersets, zowel op papier als op het beeldscherm, te ontwerpen.
COMLIN	Een grafisch, bewegend grapje.
FOTORUN	Voor de fotografeerenthousiasten onder ons.
NUTTINGCAT	Weer een CATprogramma
COPYDRIVE2	Kopieert programma's van drive 2 naar drive 1.
RENAME	Geeft uw diskettes een andere naam.
PROGRENAME	Geeft uw programma's een andere naam.
OPUSCAT.2	De opvolger van OPUSCAT op DD-1.
FINPROG	Om o.a. samengestelde interesten te berekenen.
REPCIJFERS	Hiermee kunnen de in het onderwijs werkzamen onder u de "repetitie"-cijfers bijhouden.
AGENDA	Hoort bij REPCIJFERS.
CATEXPEND	Hiermee kan de standardgrootte van de catalogus, 110, worden uitgebreid.
USERTOUSER	Hiermee kunnen TW-3 files via het VTX 5000 modem worden verstuurd.
BASICAT3BR	Nieuwste versie van de 6116-routine waarmee de CAT 3 namen breed op het beeldscherm verschijnt. Past zich automatisch aan de div. ROM-versies aan.
MESSAGES	6116-routine om de Discovery foutmeldingen naar wens aan te passen.
KLAVERJAS	Houdt voor een klaverjasclub de puntentelling bij. Werkt alleen met BetaBasic 3.0.
RUNPROGRAM	Dit is het reeds genoemde runprogramma.
FORMATS	6116-routine om diskettes op verschillende manieren te formatteren.
6116FLASH	Ook een 6116-routine.
6116BREAK	Idem.

MOVER Verwijdert de "gaten" op uw diskette op een efficiente manier.

Verder staan er een aantal textfiles op die uitleg geven over de programma's. De programma's waarbij dat niet het geval is worden summier besproken in een algemene textfile. Ook bij deze DUCDISK is soms enige zelfwerkzaamheid gevraagd. Hopelijk vindt u dat, gezien de prijs, niet erg.

Er is op deze diskette nog enige ruimte over en die is nodig omdat de 6116-routines een CODEblok van 10K saven.

De snelste manier om aan DUCDISK-2 te komen is dus door op 21 maart a.s naar Utrecht te komen. Door het ontbreken van portokosten zal het dan ook iets goedkoper zijn.

Ander nieuws is dat ART-STUDIO nu ook zijn tekeningen op diskette kan zetten. Maar voor u te vroeg juicht: dat geldt alleen voor de Extended Version. Dat is de versie die reeds naar Microdrive kon saven. Marcel van Dongen heeft een konversie-programma geschreven dat het door het masterprogramma aange-maakte, aan uw printer en andere wensen aangepaste, codeblok geschikt maakt voor de OPUS Discovery. Prijs f. 10,00.

Ook van zijn hand zijn 2 programma's waarmee THE QUILL en THE ILLUSTRATOR op diskette kunnen worden gezet. Samen voor f.10.00.

Voor LINK II is er nu een aanvulling: LINK ED. ED is een Editor. In LINK II zit reeds een editor, maar deze aanvulling geeft wat meer mogelijkheden, o.a. een 64 koloms gedeelte. LINK ED is te koop voor -het wordt een beetje saai- f.10,00. LINK II blijft f. 12,50. Koopt u echter LINK II en LINK ED samen dan betaalt u slechts f. 20,00.

In het volgende nr. van dit blad hopen wij een vergelijkende test te publiceren van LINK ED en een editor van BRADWAY Software.

Voor alle duidelijkheid: Alleen voor DUCDISK-1, DUCDISK-2 en MOUSE-UTILITIES gelden all-in prijzen. Alle andere bedragen dienen te worden verhoogd met f. 10,00, zijnde de kosten voor de diskette en porto.

Op prijzengebied is er nog wel iets meer te melden, maar spectaculair is dat niet. Ik streef er naar om in blad nr. 10 onder andere de lengte van de diverse programma's te noemen, zodat u eventueel gecombineerde bestellingen kunt doen. In verband met tijd- en ruimtegebrek laat ik het voor deze keer hierbij.

P.S: Bestellen gaat als volgt: maak het verschuldigde bedrag over op ons girorekening, met vermelding van artikel-naam, en uw eigen lidmaatschapsnummer. U krijgt het bestelde na storting.

Ton Al.

INHOUDSOPGAVE DUC# . Pag...

Algemene Machinecode Lader	7.19; 9.11
AMX-mouse	4.17
Art-studio	4.17;
Back-up card	4.10
Basicode	3.12; 7.27
Beta-Basic	1.6; 2.1,3,12; 3.17; 4.16; 5.21; 8.5; 9.31
Blast	2.3
BREAK uit programmas	9.22
Cat- en Code-channels	4.18; 6.35
CAT in 3 kolommen	9.16
CAT mover	9.15
CAT small-, super-	3.5; 6.31
CAT via de printer	8.43;
CALPHY	8.22; 9.37
channels (CAT, LOAD-)	6.7
Control Codes Printers	2.13
DATA-banken	2.16
Discovery Rom (2.2)	6.29,37;
Discovery 2	1.7
Discovery-utilities	3.5; 4.22; 9.15
Disk-disk	3.5; 9.15
DISC I/O error	9.5
Dubbeldikke letters	9.8
ERASEd, file's terughalen	6.4
Ever-draw	4.11
file's plaatsingsstrategie	6.7
file's terughalen op disk	6.4; 9.4
Geheugen tester	7.44
Graphica 1	4.11
Graphica 2	5.15
Hide	4.11
IC 6116	2.19; 3.19
Kraakinterface	2.3
Jack The Nipper	9.47
Last Word, The	5.14; 6.20,27;
line 0 wordt 1 vv	1.8
Link II	7.30
Lords of Midnight, The	2.5
MCN copy	2.9
MCODER II	9.4

MICASS (bespreking)	9.43
Modems	1.7; 2.16; 3.14; 5.19; 7.39
Mover	7.20
Multiface	2.3
Omnicalc	1.4; 2.9
PASCAL programeren	9.28
PCW Londen	7.6
Poke's, div.	1.8; 2.4; 9.46-47
Printer besturing	2.13; 3.3; 5.20
Prism VTX 5000	1.7; 3.14
Probe 1/2	1.9; 3.5; 4.22; 6.31
Programeren in BASIC	8.28; 9.31
Programeren in Pascal	9.28
Psi Chess	7.9
Qualitas	8.9
Ramdisk	1.8; 2.2
Ramdisk wissen	6.8
Random Access Files	4.16; 4.18; 5.20; 9.12
rename disk (Trinidad)	6.13
Restore programma	6.31
Rock 'N Wrestle	9.46
Romantic Robot	2.3
ROM 128k	8.8
Run-programma	2.17; 9.35
Save automatisch-TAPE	1.8
SAVE naar disk...	2.5
Screen/copy/dump	2.9; 2.15; 3.9; 3.12; 4.9 + 5.3; 5.5 + 6.16; 5.11; 6.8; 8.45; 9.9
Section 4	1.7; 5.21; 5.23
Shadow-ROM	2.2
SmallCAT	6.31
Smith-Corona-Fastext-80	3.3; 4.5; 5.7
Spec-mate	2.3; 4.15
Spectrum 128 (test)	4.23
SuperCAT	1.9; 6.31
Supercode	1.5
TAB via printerpoort	8.38
Talstelsels	7.35; 8.12
Tapeheadreader	1.9
Tasprint	5.8; 8.33
Tasword toolkit (aankondiging)	6.24
Tasword	2.13; 3.3; 3.4; 3.6; 3.13; 5.7
Tasword-paginanummering	4.4
Tasword 2	1.6; 2.2; 2.12; 3.3; 3.6; 5.1; 5.20
Tasword 2.4	4.6
Tasword 2.5	6.19
Tasword 3	3.2; 3.12; 4.6; 5.23; 6.38; 7.15 7.26
Trans-Express	1.5; 3.7
trinidad - toolkit	6.13
ULTI-MATE (RUN programma)	9.35
Update SPECTRUM	7.43
User 14070	8.27
Video-face digitiser	6.32
Video-versterker	8.10
24-koloms	6.10
Voetbal-manager	2.10
VTX modem	2.16; 3.14
5.1/4" Disk drive	2.20
Werkgroepen	7.18; 8.4
werkgroep communicatie	6.25; 7.39
Wijzigen disc- en file naam	8.48
Zakboekje vd Spectrum	5.2
Zakboekje Z80	5.2
718K formateren	5.1; 6.4; 8.39
ZLXprint-III	5.10; 5.20

****

BETA BASIC 3.OD en 2.2/2.22 ROM

Als er mensen zijn die problemen hebben met hun Beta Basic 3.0 D in combinatie met een 2.2 of 2.22 ROM, dan worden zij verzocht kontakt op te nemen met Intermediary Int. Trade

Postbus 5599
1007 AN Amsterdam

Let op! Dit geldt alleen voor de mensen met de -originele- NEDERLANDSE handleiding met daarop het adres van Intermediary. Uw ROM-versie kunt u te weten komen door PRINT USR 8 in te tikken.

****

MISCOVERIES DUCnr. 8

Tot onze spijt zijn er enkele foutjes in nr. 8 geslopen.

Blz. 5 : 10e regel van onder: CLOSE 5 moet zijn CLOSE #5

Blz. 8 : Cartoon: 'Rudie' veranderen in 'Mike'

Blz. 12 : Op de helft van de pagina staat -(2(n-1)-1). Dit moet worden: -(2^(n-1)-1)

Blz. 24 : 18e regel. Staat DJNZ BC, moet zijn DJNZ loop.

Blz. 39 : Onderste artikel in het midden. Regel 10 CLEAR 32768 moet worden: 10 CLEAR 32767.

Blz. 40 : In listing opnemen regel 155: 155 LET b=b/2

Blz. 43 : Tabel 3, in kolom 128 en rij 128 moet het getal 124 veranderd worden in 100.

Blz. 46 : In de listing zijn twee foute regels, nl. regels 160 en 230. Ze moeten als volgt luiden:

```
160 DATA 125,128,111,126,179,95,225,36,16,231
230 DATA 35,193,16,241,225,201,10,27,42,0
```

excuses voor het ongemak

****

DISCOVERY ROM-DISASSEMBLY (2.1)

Reeds velen hebben de ROM-Disassembly van Marcel van Dongen besteld. Heeft u ook interesse in een exemplaar? Maak dan Fl 17,50 over op gironummer 3588565

t.a.v. M. van Dongen
te Den Haag

U krijgt dan zo spoedig mogelijk de disassembly (87 pagina's!) toegestuurd.

****

DATA-SKIP
UW SINCLAIR-GIDS

Software, games

Xevious*	fl 39,-
Gauntlet*	39,-
1942	32,-
Ligthforce	32,-
Academy	39,-
Space Harrier*	32,-
Terra Cresta*	32,-
Sam. Fox strippoker	36,-
Bom 6 Jack	29,-
Strike Force Harrier	39,-
PSI Chess, 3 D*	45,-
Thamatos*	39,-
Graphic Adr. Creator	95,-
Konami's Coin-op Hits*	45,-
Elite Hit-pak	45,-
They-sold-a-milion	45,-
Explorer*	32,-
Starglider*	59,-

* is nieuw

Aanbieding 1

fl 399,-

Midi interface 48/128/+Z ..... fl 125,-

Software, serieus

Tasword III, cart.	fl 69,-
Tascopy, screendump	45,-
Tasprint, 6 fonts	45,-
Tas-diary, dagboek	45,-
Mini-Office	29,-
Omnicalc	69,-
Skip-64, 64 kolom	19,-
Art Studio	65,-
Masterfile	69,-
Beta Basic	69,-
Laser Genius	65,-
Laser Basic	65,-
Laser Compiler	65,-

Aanbieding 2

GROOTBOEK
VOOR
OPUS DISCOVERY (only)

F1 89,00

Aanbieding 3

ZX-Spectrum Plus Twee

fl 649,-

Software, 128 K.

Art Studio 128 k	89,-
Star Glider	59,-
Winter Games	39,-
Knight Time	19,-
3-weeks in paradise	29,-
Music Box	45,-
Tasword 128	65,-
Zub	19,-
Glider Rider	39,-
Samanta Fox Strip	39,-
Rasputin	36,-
Fairlight	36,-
Thanatos	39,-
Gladiator	36,-
Laser Genius (Assembl.)	65,-
Anals of Rome	39,-

Stofhoezen

Spectrum plus	19,-
Spectrum 128 k	19,-
Sinclair QL	19,-
Saga Elite	24,-
Saga 2+	24,-
Saga Emperor	24,-
Lo Profile	24,-

Aanbieding 4

Compleet modern pakket voor Sinclair Q.L. Incl. auto-dial en auto-answer.
Elke QL-bezitter start nu z'n eigen databank voor maar

Tandata

fl 325,-

Printers

Centronics G.L.P. een 80-koloms printer voor normaal papier. Met ser. en par. interface. Div. lettertypen o.a. N.L.Q... 499,-
Citizen 120 D, schitternde par. printer met vele mogelijkheden, o.a. proportioneel, inverse, N.L.Q.... 795,-

Monitors

Groen vanaf	299,-
Philips 7542, wit	369,-
Philips, kleur	695,-

Diversen

Seiko RC-1000, polshorloge met o.a. opslag voor bijv. 30 stuks adressen, telefoonnrs, formules e.d. wereldtijden, alarm, 1 jaar vooruit te programmeren
Voor gebruik let Interface I 109,-
Seiko QL software 29,-

Sinclair Q.L. hardware

Sinclair Q.L. compleet	495,-
CST Disk-interface	345,-
Modem-pakket, compleet	349,-
NLQ-printers vanaf	695,-
Monitors vanaf	299,-

Aanbieding 5

AMX-muis, incl. software

fl 249,-

ZX-Spectrum Hardware

Multiface One	175,-
Multiface One 128	199,-
Videoface Digitiser	249,-
Cartridge tox	19,-
VTX-5000 modem	199,-
ZXL-printinterface	199,-
3 1/2" diskette	5,-
Konix Speedking, joystick	39,-
Joystick Interface	49,-
Saga Elite, toetsenbord	275,-
Saga TNO-plus	225,-

POSTORDER: BEL 01820-20581
porto software fl 2,50
hardware fl 5,-
rembours fl 10,-

Prijswijzigingen voorbehouden

CST Thor ..... bel voor inlichtingen en prijzen

TEL 01820-20581
Data-Skip,
L.Willemsteeg 10
2801 WC Gouda

Bestellingen dmv ingesloten betaalkaart of vooruitbetaling op giro 4727958 tnv Data-Skip Gouda.