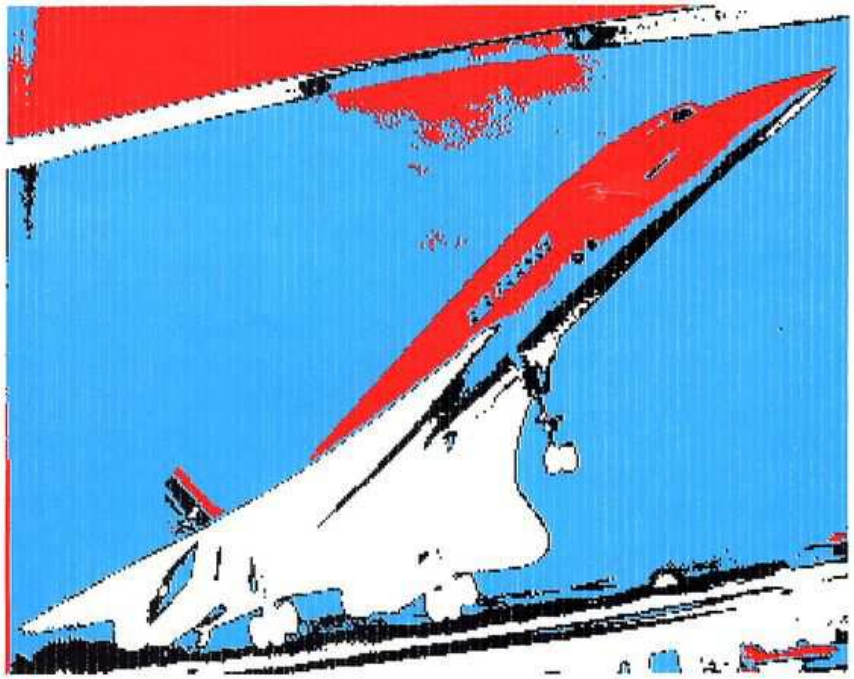
DAI NAMIC

DAI CLIC

32

5



tweemaandelijks tijdschrift januari-februari 86

bimestriel janvier - février 86

een uitgave van DAInamic VZW en IDC ASBL
verantw. uitgever : w. hermans, mottaart 20, 3170 herselt

International

COLOFON

DAInamic verschijnt tweemaandelijks.
Abonnementsprijs is inbegrepen in de jaarlijkse contributie.
Bij toetreding worden de verschenen nummers van de jaargang toegezonden.

DAInamic redactie :

Dirk Bonné	wdw
Freddy De Raedt	Herman Bellekens
Wilfried Hermans	Frans Couwberghs
René Rens	Guido Goyvaerts
Bruno Van Rompaey	Daniël Goyvaerts
Jef Verwimp	Frank Druijff
Cedric Dufour	Willy Coremans

Vormgeving : Ludo Van Mechelen.

U wordt lid door storting van de contributie op het rekeningnr. **230-0045353-74** van de **Generale Bankmaatschappij, Leuven,** via bankinstelling of postgiro.
Het abonnement loopt van januari tot december.

DAInamic verschijnt de pare maanden.
Bijdragen zijn steeds welkom.

CORRESPONDENTIE ADRESSEN.

Redactie en software bibliotheek

Wilfried Hermans
Mottaart 20
3170 Herselt
Tel. 014/54 59 74

Kredietbank Herselt
nr. 401-1009701-46
BTW : 420.840.834

Lidgelden / Subscriptions

Bruno Van Rompaey
Bovenbosstraat 4
B-3044 Haasrode
België
tel. : 016/46.10.85

Generale
Bankmaatschappij
Leuven
nr. 230-0045353-74

Voor Nederland :

GIRO : 4083817
t.n.v. J.F. van Dunne'
Hoflaan 70
3062 JJ ROTTERDAM
Tel. : (010) 144802

Inzendingen : Games & Strategy

Frank Druijff
's Gravendijkwal 5A
NL 3021 EA Rotterdam
Nederland
tel. : 010/25.42.75

DAICLIC INFOS :

DAICLIC paraît tous les deux mois.
L'abonnement est compris dans la cotisation annuelle à IDC (du 1/1 au 31/12). A l'inscription, les numéros déjà parus dans l'année sont envoyés.

Conseil d'administration de IDC :

Président : Christian POELS, rue des Bas-Sarts 10, B-4100 SERAING Tél. : 041/37.16.06
Secrétaire : Marc VANDEMEERSCH, av. Vert Bocage 17 B-1410 WATERLOO Tél. : 02/354.13.63
Trésorier : Fabrice DULUINS, allée Tour Renard 4, B-1400 NIVELLES Tél. : 067/21.82.10
Rédaction : Christian POELS
Soumissions logiciels : Marc VANDERMEERSCH
Inscriptions, vente logiciels : Fabrice DULUINS.
(mode de paiement : voir ci-dessous)

Cotisations :

Belgique : 1000 FB virement, chèque, cash,...
Compte BBL : 371-0356842-45.
F. DULUINS et CH. POELS
ALLEE DE LA
TOUR RENARD, 4,
1400 NIVELLES
Etranger : 1100 FB par mandat postal international uniquement.

Services télématiques IDC :

MN2 Bruxelles-A :	02/242.70.08
MN2 Bruxelles-B :	02/513.11.11
MN2 Liège-A :	041/79.66.66
MN2 Verviers-B :	087/88.31.31

CLUBS ASSOCIES :

IDC BORDEAUX : Bruno DELANNAY, Rés. ACACIAS B+ B3, Av. de Saige, F-33600 PESSAC
IDC LIEGE : Philippe RASQUIN, Rue Saivelette 89, B-4510 SAIVE
CAROLO-DAI : Etienne SZIGETVARI, R. Provinciale 7, B-1361 CLABECQ (Charleroi)
DAIC : Jacque MOENS, Clos Fontaine Ducs 6 B-1310 LA HULPE (Bruxelles)
DCA : Philippe CASIER, Rue de Paris 31ter, F-92190 MEUDON (Paris)

Inhoud - Sommaire

DAINAMIC 32 / DAICLIC 5

Remark

Beste leden,

Hier is hij dan : de eerste dubbeldekker, multi-club uitgave van 1986. Op de voorpagina vindt U de beide titels terug : DAInamic en DAICLIC. Concreet betekent dit dat van Groningen tot Marseille DAI-gebruikers nu hetzelfde blad kunnen ontvangen. We kunnen ons best indenken dat sommigen zich genoodzaakt zullen voelen om hun school-frans eens wat op te frissen. Had U trouwens voor de volgende vakantie geen reisje naar Zuid-Franrijk gepland ?

Als we wat betreft DAI-software enige mijlpalen in ons zesjarig bestaan willen plaatsen ,komen we al tot een aardig lijstje. (Vergeef me als ik een aantal titels over het hoofd zie):

FGT en DNA-assembler : F.De Raedt
SPL : gebroeders Rens
DAI-SCHEMA'S &
FIRMWARE MANUAL : J.BOERRIGTER & Co
SFGT : gebroeders Lambrechts
FWP : G.Gruiters
DRASIC & DISKBASIC : W.Coremans
SUPERBASE : U.Wienkop

In dit nummer hebben we het genoegen U een nieuwe mijlpaal aan te kondigen : SUPERFONT. Naar kwaliteit en kwantiteit een uniek pakket : We bieden U 300 K software, waarvoor we 2 audio-cassettes aan beide zijden moeten gebruiken, ofwel 4 DCR-cassettes. Hierbij horen 2 handboeken, waarvan een met meer dan 100 pagina's. Onze speciale dank aan de leden van het SUPERFONT-team die gedurende meer dan een half jaar hun vrije tijd aan SUPERFONT opofferden: Koen Vandeperre, Luc Maes,Paul Gobert en Willy Coremans. We zijn er van overtuigd dat U hun enthousiasme zal delen indien U SUPERFONT beter leert kennen...

This issue if the first result of the cooperation between DAInamic & DAICLIC. We hope that the French language doesn't cause too much troubles to you. Above you can find a list of the software-milestones of DAInamic. In thos issue, we are proud to announce another super software-package : SUPERFONT. 2 manuals, 300 K of data on 2 audio-cassettes (both sides) or 4 DCR-cassettes. We are sure you will appreciate the efforts of our SUPERFONT-team. (see description of this package on p. 21-22)

beste groeten,happy reading,

DAInamic redaction

Programmeertechnieken

Aan het eind van het vorige artikel kwam de faculteit nog eens ter sprake bij de reeksontwikkeling van de sinus. Ik dacht toen reeds aan een programmeerjuweeltje wat ik in iets andere vorm kort daarvoor was tegengekomen. Toen kwam het niet aan bod maar nu wil ik er mee gaan beginnen.

Het hieronderstaande programma berekent de faculteit van het ingevoerde getal. Loop echter niet te hard van stapel de DAI kan uw enthousiastme al snel niet meer bijhouden en stopt al bij 13 ! (dertien faculteit).

Faculteit

```
10  REM FACULTEIT / F.H.Druijff-10/85
20  INPUT N:PRINT :F=1:GOSUB 100
30  PRINT N;"! = ";F:GOTO 20

100 N=N-1:IF N < 0 THEN GOSUB 100
110 N=N+1:F=F*N:RETURN
```

Begrijpt U de werking van dit programma ? Ik zal het eens doornemen en hoop dat daarna iedere lezer het zal begrijpen.
De eerste drie regels zien er tamelijk normaal uit.
Het programma wordt geidentificeerd. Er wordt naar het getal N gevraagd waar de faculteit van moet worden uitgerekend.
Een variabele F wordt 1 gemaakt en de subroutine op 100 wordt aangeroepen. De resultaten worden afgedrukt (de variabele F bevat nu het antwoord) en we gaan weer opnieuw beginnen.
Het lijkt duidelijk dat F geinitialiseerd moet worden omdat het programma meerdere keren achter elkaar gebruikt kan worden. Het uitrekenen van de gewenste faculteit geschiedt in de subroutine op 100.

Tot zover geen grote problemen maar dan gaan we kijken naar de subroutine op 100.
Ons ingevoerde getal wordt met een verminderd en als het resultaat groter dan 0 is roepen we de subroutine op 100 aan !!!!!!!!
De DAI doet dit zonder problemen en na een aantal keren zal N niet meer groter dan nul zijn. We gaan dan dus niet de subroutine op 100 aanroepen maar vervolgen met regel 110.
Hier gaan we onze N weer met een verhogen (!?!) .

Daarna vermenigvuldigen we F met N. We begrijpen nu waarom F=1 werd gegeven; anders zou F=0 zijn en hier na elke vermenigvuldiging cok 0 blijven.

En dan komen we RETURN tegen. Deze RETURN 'hoort' bij de GOSUB van regel 100. Aangezien elk programma dat een GOSUB tegenkomt de desbetreffende subroutine uitvoert totdat hij in die subroutine RETURN tegenkomt om daarna te vervolgen na de GOSUB, die de subroutine aanriep, zal dat ook hier geschieden
We vervolgen het programma dus na de GOSUB van regel 100 ergo met regel 110. Hier wordt de N weer met een verhoogd en F met N vermenigvuldigd. F is dus nu 1*1*2 = 2 groot. Door de RETURN vervolgen we vaak weer na de GOSUB uit regel 100. Weer regel 110. N wordt 3 en F wordt 1*1*2*3 = 6. Dit gaat zo door totdat de RETURN niet meer bij de GOSUB uit regel 100 'hoort' maar bij de GOSUB uit regel 20. Na die GOSUB echter vervolgen we met regel 30 en daar wordt het resultaat afgedrukt.

Maar hoe kan de DAI nu weten dat hij op zeker moment naar regel 30 moet gaan en niet nogmaals regel 110 ?
En hoe kan het dat een RETURN bij twee GOSUBs kan 'horen' ?

Om met dat laatste te beginnen :
dit is eigenlijk nogal normaal, denk er maar eens over na. Maar ik merk bij mijn leerlingen steeds de verbazing over de werking van dit programma en dan komt deze vraag steevast.
Het geheim zit in de stack. Dit is om het zo te zeggen de la waarin de DAI zijn 'briefjes' bewaart waarop staat waar hij naar toe moet bij RETURN.
De 'briefjes' worden ingevuld en op de stapel (stack is engels voor stapel) gelegd.
We volgen het programma nog een keer. Er wordt bijvoorbeeld 4 ingetikt.
Bij de GOSUB van regel 20 schrijft de DAI 30 op een blaadje en stopt dat in de la. In regel 100 wordt N 3 gemaakt en we krijgen daarna de GOSUB van regel 100. De DAI schrijft braaf 110 op een blaadje en legt dat op (!!!) het blaadje met 30 in de la. We gaan daarna nog eens regel 100 uitvoeren; N wordt 2 en een volgend briefje met 110 komt op de stapel terecht. Dan wordt N 1 en weer een 110 komt op de

stapel. Tot slot wordt N 0 en nu komen we in regel 110 daar wordt N weer 1 gemaakt, we vemenigvuldigen F met N en komen bij RETURN. De DAI pakt nu het bovenste briefje van de stapel. Daar staat 110 op en daar gaat hij dus ook mee verder.
N wordt 2 en F wordt vermenigvuldigd met die 2. Weer vinden we een verwijzing naar 110 en zo wordt N 3 en F vervolgens 6. Met de greep in de la komt nu het laatste briefje (of het eerste als we denken aan de volgorde van inleggen) met 110 te voorschijn N wordt weer de oorspronkelijke 4 en F het antwoord 24.
Er zit nu nog maar een briefje in de la met 30erop. Als deze aanwijzing wordt opgevolgd komen we in regel 30 waar alles wordt afgedrukt.

Met grotere waarden voor N komen er navenant meer briefjes met 110 en wordt F dus ook vaker vermenigvuldigd.

Het programma kan overigens nog fractioneel worden versneld door niet N<0 te vragen maar N<1. Controleer zelf dat dit dezelfde resultaten geeft.

foutje

Jammer genoeg is het programma niet perfect. 0!=1 en dat komt niet uit dit programma al is die fout er simpel uit te halen.

De hier uitgelegde programmeermethode heet recursief programmeren.
Het lijkt mij voor diegenen die hier geen ervaring mee hebben best nuttig hier eens wat mee te oefenen.
Bereken bijvoorbeeld een twee tot de macht n door een algoritme als hiervoor gegeven werd voor de berekening van de faculteit. Dit programma kan ook nog eens zinvol zijn als we ons bedenken dat 212 op de DAI al niet meer nauwkeurig is. Het is namelijk 4096 en de DAI beweert 4095.99 als we de mathchip aan hebben staan. Zonder die chip gaat het nog goed maar bij exponent 13 komt dan de fout. Het is weliswaar een klein verschil maar als we iets exact willen hebben onjuist.

Deze fout is verklaarbaar door de berekeningswijze van de macht. Er wordt niet letterlijk 2x2x2x....x2x2 berekend maar de logaritme van 2 wordt bepaald en die wordt dan vermenigvuldigd met de exponent. Tot slot wordt deze waarde weer gebruikt als exponent van het grondtal van de gebruikte logatitme om zo tot het antwoord te komen. Vanzelfsprekend zijn bij deze methode alle waarden van het floating point type.
Door afrondingen kunnen dan kleine afwijkingen ontstaan.
De integerwaarden houdt de DAI echter vrij lang vol en kan een algoritme als hiervoor gevraagd best wel zinvol zijn in sommige gevallen.

Logaritme

Nu de logaritme toch al ter sprake is gekomen lijkt het mij best zinvol hiermee door te gaan.
De functies die we dan gaan bekijken zijn de volgende :

enerzijds	en	anderzijds
LOG en EXP		LOGT en ALOG

Ik geef deze functies in paartjes omdat die elkaars inverse zijn.
Theoretisch moet dus gelden :

X=LOG(EXP(X)) en X=EXP(LOG(X))
evenals
X=LOGT(ALOG(X)) en X=ALOG(LOGT(X))

Hierbij doet de werking van de ene functie om het zo te zeggen de werking van de andere weer teniet.
Behoudens afrondingsverschillen zal de DAI deze theorie ondersteunen.
Tussentijds ontstane te grote of te kleine waarden kunnen echter roet in het eten gooien.
De verkregen waarden van bijvoorbeeld de ALOG zijn al snel te groot voor de DAI. We halen de 19 als argument al net niet en krijgen dan overflow.
Bij de EXP kunnen de getallen wel wat groter zijn maar we zitten nog steeds in dezelfde orde van grootte.

Voor we verder gaan eerst een klein beetje theorie over de logaritme.
Deze theorie is niet alleen noodzakelijk voor diegenen die van plan zijn genoemde functies te gaan gebruiken, maar juist voor diegenen die er niets mee te maken willen hebben.
Zij zullen namelijk meermalen geconfronteerd worden met de werking van genoemde functies. Zij meenden echter die in het geheel niet te gebruiken !
Zoals reeds eerder betoogd werd blijken we bij de machtsverheffing gebruik te maken van de logaritme terwijl sommigen zich daar niet bewust van zijn.

Theorie

Als voorbeeld beginnen we met de zogenaamde 2-logaritme dit is de logaritme met basis (grondtal) 2.
We gaan nu elk getal omzetten in een macht van 2.

2 ^1 = 2	dus 2log 2 = 1
2 ^2 = 4	dus 2log 4 = 2
2 ^3 = 8	dus 2log 8 = 3
2 ^4 = 16	dus 2log 16 = 4
2 ^5 = 32	dus 2log 32 = 5
enz.	enz.

De eerste 2 van '2 log' dient eigenlijk wat hoger geschreven te worden net zo als een kwadraat.

De voordelen blijken al snel als we bijvoorbeeld 4 x 8 willen uitrekenen. In de tabel zien we 2log 4 = 2 en 2log 8 = 3. Nu tellen we die 2 en 3 op ! De vermenigvuldiging wordt dus een optelling en dat is -ook voor de DAI- gemakkelijker. Tot slot zoeken we de uitkomst van onze optelling (5) terug in de logaritmetabel en zien dat daar 32 staat. Ergo : 4 x 8 = 32

U hoeft zich er niet het hoofd over te breken hoe men berekend heeft dat 2log 3 = 1,58496... is, maar door 2 tot deze exponent te verheffen kunt U de uitspraak wel controleren.

Positief

De logaritme kan alleen met positieve getallen als basis en argument werken Vandaar dat we ook niet ?(-2)4 door de DAI kunnen laten uitrekenen.
In principe is elk positief (!) getal (muv 1) te nemen als grondtal voor de logaritme maar in de praktijk blijven er maar twee over.
Beide worden door de DAI gebruikt.

Grondtal tien

Tien is als basis van ons talstelsel een vrij voor de hand liggende keus.
Er zijn tabellenboekjes in de handel te verkrijgen waarin de tabellen op basis van grondtal 10 staan.
U kunt, mist U de nauwkeurigheid van de DAI in deze voldoende vindt, de tabel zelf op uw DAI maken.
De instructie die U hiervoor gebruikt is LOGT (LOGarithm op basis van Ten).

Het is leuk en leerzaam, als U de logaritme niet (meer) kende, er een poosje mee te spelen zodat U de bijzonderheden zelf ontdekt en daarmee meerdere voordelen. Maar tevens ook de beperkingen leert inzien.

Uitgespeeld ?
Dan zal ik een aantal zaken nogeens onder de loupe nemen.
Volgens de DAI :

LOGT(5) = 0.69897
LOGT(50) = 1.69897
LOGT(500) = 2.69897
LOGT(5000)= 3.69897

We zien in het tabelletje duidelijk dat als het argument tien maal zo groot wordt de logaritme slechts een toeneemt.
Dit is te begrijpen als we bedenken dat 50 = 5 x 10 en dat LOGT(50) dus de som is van LOGT(5) en LOGT(10).
LOGT(10) is natuurlijk 1 en zo is LOGT(50) ook 1 groter dan LOGT(5).
De logaritme tafel hoeft in dit geval dus slechts van 1 tot 10 te gaan.
Het verschuiven van de decimale komma heeft dus tot gevolg dat het getal voor de komma in de logatitme toe- of afneemt. Omdat dit bij terugzoeken lastig wordt bij getallen onder nul schrijven we die meestal anders:
LOGT(.5) = .069897 -1 die we snel terug kunnen vinden in de tabel.
Dit argument geldt niet voor de DAI en die geeft dan ook LOGT(.5)=-.30103

Inverse

De inverse functie van de LOGT (het terugzoeken in de tabel) geschiedt met de functie ALOG.
Deze naam zult U vrijwel nergens elders aantreffen en is ook vrij ongelukkig gekozen. Bedoeld werd een soort AntiLOGarithm te geven maar dan was omdat we het hier hebben over de inverse van de LOGT toch logischer geweest hem ALOGT te noemen ?
Ook namen als RELOGT of LOGTINV ware te prefereren.
Maar het ergste komt nog we hebben de inverse allang in de DAI zitten.
Tien tot de macht (10^.) doet hetzelfde als de ALOG. Hij zou het tenminste moeten doen maar na enkele

pogingen zult U zien dat er minimale verschillen zijn.
Niet dat die afrondings fouten ernstig zijn maar de DAI doet duidelijk iets anders.
Ik heb het vermoeden dat ALOG iets nauwkeuriger en sneller is.

Grondtal e

Deze e is de naam voor de constante die decimaal 2,718 281 824 593... is.

Voor niet wiskundigen is e een nogal vreemd getal en zeker om als basis voor een logaritme te nemen.
Hoe verder men zich echter in de wiskunde verdiept hoe meer men het belang van dit getal leert inzien.
De logatitme die op dit grontal is gebaseerd heet dan ook wel de natuurlijke logaritme en wordt in de wiskunde vaak met 'ln' aangeduid.
De DAI gebruikt er LOG voor.

Deze logatitme is verhoudingsgewijs gemakkelijk te bepalen en zeker het terugzoeken is simpel. Dit laatste komt door een van de wel zeer simpele reeksontwikkeling van e^X.

$$e^X = 1 + \frac{X^1}{1!} + \frac{X^2}{2!} + \frac{X^3}{3!} + \frac{X^4}{4!} + \ldots$$

Eventueel schrijven we de eerste term ook in dezelfde vorm als de andere.
De kenner ziet in deze reeksontwikkeling al snel dat de functie e^X zijn eigen afgeleide is. Daarin is deze functie uniek. Kent men het belang van de afgeleide(n) bij het functieonderzoek zal men begrijpen hoe belangrijk deze functie is.

De waarde van e zelf kan uit bovenstaande functie gehaald worden door voor X 1 te nemen.
De definitie van e wordt meestal niet gekoppeld aan deze reeksontwikkeling maar aan de volgende limiet :

$$e = \lim_{n} (1 + 1/n)^n \quad \text{of} \quad \lim_{n} ((n+1)/n)^n$$

voor n naar oneindig.
De beide definities zijn identiek.

Om e met deze definitie decimaal te bepalen (benaderen want e is net zoals pi niet een rationeel getal) is een hele klus.

De reeks convergeert slecht. Of minder wiskundig gezegd we moeten de reeks vrij ver ontwikkelen om in de buurt van de echte waarde te komen.

```
10  REM e-limiet / F.H. Druijff
20  I!=1.0
30  PRINT (1.0+1.0/I!)↑I!
40  I!=I!+1.0
50  GOTO 30
```

Voorgaand programma werkt volgens de definitie maar pas bij I!=164 is het tweede cijfer (achter komma) correct.
Beter is het de reeksontwikkeling te nemen als in de volgende programma's.

I

```
10  REM E-ONTWIKKELING / F.H. Druijff
20  I!=1.0:E!=1.0:F!=1.0
30  PRINT I!,E!
40  E!=E!+1.0/F!:I!=I!+1.0
50  F!=F!*I!:GOTO 30
```

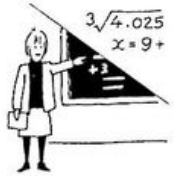
II

```
10  REM e-ONTWIKKELING / F.H. Druijff
20  I!=1.0:E!=1.0:F!=1.0
30  PRINT I!,E!
40  E!=E!+F!:I!=I!+1.0
50  F!=F!/I!:IF F!(<0.0 GOTO 30
60  END
```

Bij het limiet programma zien we bij grote waarden voor I! de afrondingsfouten op zulke ongelukkige plaatsen komen dat we haast denken met een randomgenerator te werken.

Kijk de twee reeksontwikkelingsprogramma's eens na op de betere werking in versie II. U ziet wat nadenkwerk vooraf kan best de moeite lonen.

Frank H. Druijff



16-colour textmode

Dear members,

Questionning myself how to use the 16-colour textmode, I found it was not that easy to use it adequately. Changing the colours without peeking and poking seemed only possible via the COLORT-instruction. As you can see, I made a routine that 'fills in' the colour-bytes of the characters on the screen according to the 3rd and 4th number in COLORT: the 3rd number gives the backgroundcolour of the printed characters, the 4th gives the colour of the character itself (see basic-program). Always initialisate the screen as shown in the program (but over more lines if you like), and change in UTILITY vector 5 into the start-address of the routine : type 'V5', 'V5 C6FD-' will appear, and type in the entry-address of your routine (here '3000';NOT in reversed order (NOT '0030')).The mlp-routine will change (every time ROM calls the screen interrupt) the colourbyte of the last printed character according to the actual COLORT.
Notice :

The ROM screen-scroll doesn't move up the colour-bytes (so after a printed screen, you will have to do a new screen-initialisation).

If you need fast screen initialisation, type in the following mlp-routine (only assembled mlp-codes are given). Type in these codes (given in hex) enywhere you like (be careful for graphics modes) and you only have to fill in the underlined numbers with the MSB of the entry-address of this second routine. Restriction : the routine has to begin at an address with LSB=00. (E.g.: you type it in, starting at #3700 : replace the underlined zeroes by '37')
The 'CLS'- routine :

```
E5 D5 C5 F5 2A 8A 00 EB 2A 8C 00 01 7A FF 09 0E FF 3A 7C 00
E6 0F 47 87 87 87 87 B0 47 3E 42 23 70 23 71 23 3D C2 20 00
36 48 23 36 FA CD 14 DE DA 1D 00 F1 C1 D1 E1 C9
```

This routine clears the screen and sets the screen backgroundcolour according to the first byte of the COLORT registers.

The screen background colour of a 16-colour mode cannot be changed by a COLORT-command, since each screen-location has its own bytes (one containing the ASCII-value of the character displayed, a second to give resp. its fore- and background colour). In the BASIC program the chosen background colour is 0 (black), but if you desire another colour, change (in line 30) 'POKE Y-1,0' with 'POKE Y-1,#NN' where N is the hex value of the colour (e.g.: red becomes #33, orange becomes #AA).

The routine changes printed or typed spaces into #FF. This is necessairy to stop the backgroundcolour of a printed character going on over a space following that character. (If you don't understand why, notice that the background colour only changes on a 'scan' (see article DAInamic 14, page 37-38) if on that scan at least one 'bit' is printed in

foregroundcolour. If you still have problems, try some 'poke'ing in a 16-colour graphics mode and discover it is possible to get 3 colours on 8 dots (2 bytes) on the screen (read articles about screen-RAM-setup.) This has as nasty consequence that in direct mode every direct command containing spaces will be mis-understood by the BASIC-interpreter (since it reads chr$(#FF) and not a space). Also for example running a BASIC program with the routine switched on, every space in an INPUT-statement will be converted into chr$(#FF).Thus, if you want to use direct mode in 16-colours, don't type in lines #3039 - #3044 and change in line #302A 'JZ SETCLR' into 'JZ SET2'or switch the routine temporary off with 'POKE #74,1' (see below). Now the spaces will have the back-groundcolour of the previous character.

Try the difference between the two routines : if you type in (line 110) 'COLORT 0 0 BGC X' (where X is doesn't matter and BGC is the screen background colour, chosen in line 30), the two routines will give the same result. If you run 'COLORT 0 0 NBGC X' (where NBGC<>X), the two routines will give other results. Of course, you can try all possibilities in direct mode, just using the BASIC program for screen initialisation (lines 5-50).

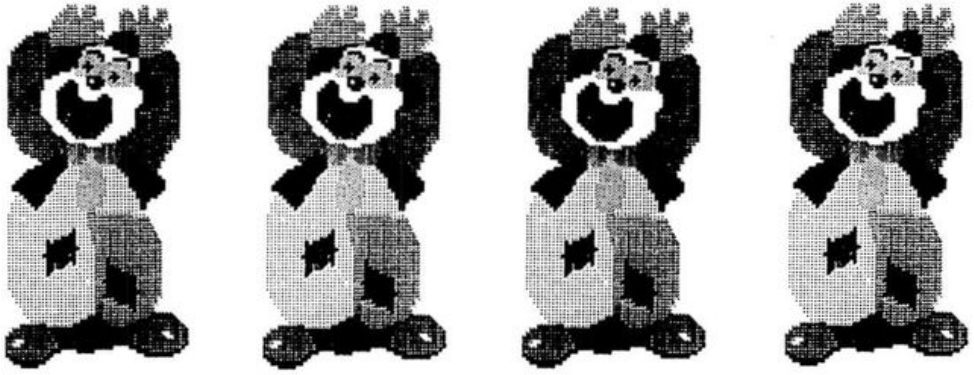
The routine is switched on/off with 'POKE #74,n' with n=0 or 1 (1 = normal cursor = non active).

Make sure using this routine in direct mode or in programs BASIC doesn't overwrite the routine (e.g. by EDIT, LOAD / MODE6A...), because that 'crashes' your Dai.

I hope this article is clear enough. If not, write to the DAInamic-redaction, and I will write some more about 16-colour modes, but read first the article 'Kleurconflicten' (in Dutch) of H. Bellekens (Dainamic 31)

Yours sincerely,

Dirk De Boeck
BELGIUM




```
SPL V1.1  PAGE 1

0000                    ORG       3000H
3000  E5      START     PUSH H
3001  D5                PUSH D
3002  C5                PUSH B
3003  F5                PUSH PSW
3004  3A7400            LDA       74H             ;get cursor typ
e
3007  B7                ORA A
3008  C26030            JNZ       END             ;skip routine i
f normal cursor
300B  2A7200            LHLD      72H             ;get address of
 cursorposition
300E  EB                XCHG                      ;DE=actual curs
orpos.
300F  2AC800            LHLD      0C8H            ;get address of
 cursorpos. during previous call
3012  CD1ADE            CALL      0DE1AH          ;HL=HL-DE
3015  7C                MOV A,H                   ;)if HL=0 or HL
>255 (cursor didn't move, or moved
3016  B7                ORA A                     ;)but didn't pr
int a character): skip routine
3017  C25C30            JNZ       ENDA            ;)then, if HL<>
2 : skip routine
301A  7D                MOV A,L                   ;)
301B  FE1E              CPI       1EH             ;check if curso
r moved 30 bytes
301D  C23030            JNZ       CHKCHR          ;jump if false
3020  3A7A00            LDA       7AH             ;end of actual
line
3023  C686              ADI       86H             ;add #bytes/lin
e
3025  47                MOV B,A
3026  3AC800            LDA       0C8H            ;get lobyte pre
v. cursorpos.
3029  B8                CMP B                     ;check if prev.
 curpos. was at end of prev.line
302A  CA3930            JZ        SETCLR          ;jump if true
302D  C35C30            JMP       ENDA            ;skip if false
3030  B7      CHKCHR    ORA A                     ;)only if the c
ursor points to the following
3031  CA6030            JZ        END             ;)character-byt
e on the screen, the colours of
3034  FE02              CPI       2H              ;)previous char
acter will be set
3036  C25C30            JNZ       ENDA            ;)
3039  2AC800 SETCLR     LHLD      0C8H            ;prev. curpos.
303C  3E20              MVI A     20H             ;A=ASC(" ")
303E  BE                CMP M                     ;check if just
printed char.is a space
303F  C24730            JNZ       SET2            ;continue if fa
lse
3042  36FF              MVI M     0FFH            ;set terminator
3044  C35C30            JMP       ENDA            ;skip colour-se
t
3047  EB      SET2      XCHG                      ;actual curpos.
```

```
3048  2B              DCX H                   ;clr.byte of pr
ev.char.
3049  3A7E00          LDA       7EH           ;)get cursor-ba
ckgroundclr.
304C  E60F            ANI       0FH           ;)
304E  47              MOV B,A
304F  3A7F00          LDA       7FH           ;)get colour of
 character on cursor
3052  E60F            ANI       0FH           ;)
3054  87              ADD A
3055  87              ADD A
3056  87              ADD A
3057  87              ADD A                   ;and shift it l
eft 4 places
3058  B0              ORA B                   ;or backgr.clr.
3059  77              MOV M,A                 ;store it in cl
r.byte
305A  23              INX H                   ;HL=new cursorp
os.
305B  EB              XCHG
305C  EB      ENDA    XCHG                    ;HL=new cursorp
os.
305D  22C800          SHLD      0C8H          ;save it
3060  F1      END     POP PSW
3061  C1              POP B
3062  D1              POP D
3063  E1              POP H
3064  C3FDC6          JMP       0C6FDH
3067                  END
```

```
10    FOR X=#BFEF TO #BFEF-134*10 STEP (-134)
20    POKE X,#FA:POKE X-1,#48
30    FOR Y=X-132 TO X-2 STEP 2:POKE Y,#FF:POKE Y-1,0:NEXT
40    NEXT
50    POKE #74,0:POKE #75,#FF:CURSOR 0,23
100   READ A$:IF A$="END" THEN CURSOR 0,8:POKE #74,1:COLORT 8
 0 0 8:END
110   FOR X=0 TO LEN(A$)-1:COLORT 0 0 0 Z+1:PRINT MID$(A$,X,1
);:Z=Z+1:Z=Z MOD 15:NEXT
120   PRINT :GOTO 100
200   DATA "This is a test"
210   DATA "Notice you can't change the screen-backgroundcolo
ur."
220   DATA "You can change the background of each character,"
230   DATA "but this is only (visually) good, if you change"
240   DATA "colours at the beginning of each word. (try it)"
250   DATA " "
260   DATA "This routine also works in direct mode."
270   DATA "Pay attention : if you typed in the whole routine
"
280   DATA "direct commands containing spaces won't work."
300   DATA "END"
```


Programming contest

Hello programmer,

anyone, who has a computer and who wants something more than just to play some games on it, is for most of the time surprised by the speed of such a machine. Therefore, we have here a little contest which is not difficult to understand (we hope), but which will give your machine, and yourself perhaps, a lot of work. To make this work a bit more interesting, we could persuade the DAInamic redaction to give away some software for the winners.

What is this all about now ? Well, a little time ago, we saw a man busy trying to fill a matrix of 10 elements by 10 with ascending numbers, going from 1 till 100. Those will of course fit right in that matrix and fill it completely. But there were some limits in filling the matrix, and these were the following :

(assuming that the matrix is represented as a square in which you proceed with your ascending numbers)

1) start (number 1)is always in the upperleft corner of the matrix

2) you have to leave some space between 2 following (ascending) numbers, in the following way:

- when going horizontally or vertically, you should skip 2 and only 2 places between 2 following numbers
- when going obliquely, you should skip 1 and only 1 place between 2 following numbers.

These are the limitations you'll have to work with. As we know it is rather ununderstandble without a litlle drawing, we will try to represent it here.

In the next drawing (fig. 1) , the possible places to put 65 (after 64) are represented by a '**'. This is of course only the case if they are not yet occupied by another number (36 or 22 for example,see fig.2). In fig. 2 you can also see that the border of the matrix also forms a limitation for the putting of numbers.

The final purpose is to fill the whole square with the numbers from 1 till 100, paying attention to the limitations as they are described above and as they now are clear to you after seeing the figures. We did it for the numbers 1 till 10 and you will work together with your computer to fill the whole square. For those who want to quit early, we can assure them that there are very many possible resolutions to the posed problem.

FIG. 1

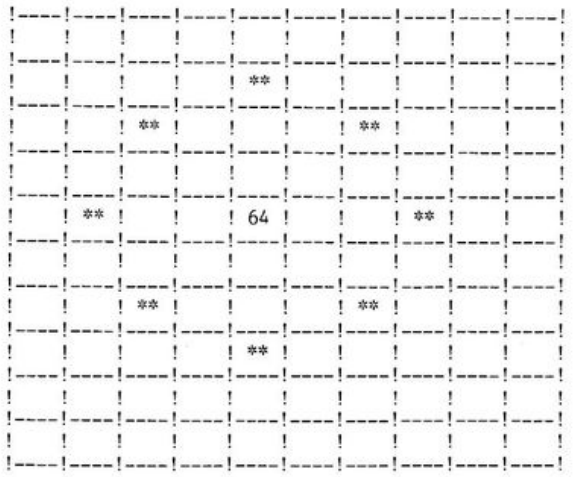
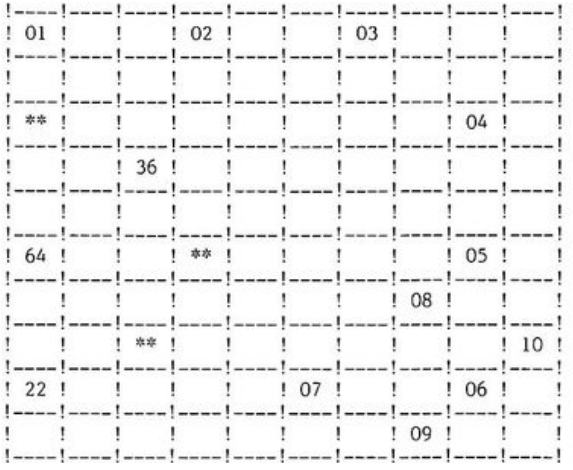


FIG. 2



After describing the problem, let us describe the practic elements of the contest. First of all, we wish to say that we hope this will be a contest more for fun than for prizes, and we hope there will be no discussion about who earns which prizes or so. We will form a little jury with some members of DAInamic redaction and associes if needed for special languages such as PASCAL or D-BASIC. That jury will try to select the best programs and will try to satisfy each competitor. To give each programmer a fair chance, there will be a division in classes of program language,and there will be at least one prize for each language class.

What are the best programs ? Of course, speed will be one of the major criteria. But also the length of the program, the structure and the understandability will make some difference between programs. It is also recommended that the program doesn't stop after finding the first possibility, but keeps on searching for others.

Now some more practical things:

- please comment your program. A BASIC program is hard to understand without comments, a machine language program is not understandable without it. And if we then cannot run it or don't find the solution, it would be a pity for all the work.
- please comment the programs in ENGLISH Since we have readers in many different languages, we would have to translate all comments for publication. And we think we'll have work enough studying and comparing all programs.
- if possible, send the programs on a DCR-cassette. This causes the least problems for reading. We can however also read them from audio cassette or KEN-DOS (200K) , if you cannot send it in DCR form.
- if possible,accompany the program with a listing (with eventually more remarks), or a second version with remarks in it if the REM's should slow it down.
- We will return your programs if you wish (please indicate it in that case) after the publication of the contest
- closing date of the contest is:
  *** 31/07/1986 24:00 hr ***

If you have any more questions about the contest or if you wish to send in programs, you can do this at one of the following addresses:

Paul Gobert
De Bergen, 49
B-2241 Zoersel (Belgium)

or

Luc Maes
Collegestraat, 60 E
B-2300 Turnhout (Belgium)

We will answer your questions as soon as possible.

Hoping for a mass of reactions and some very smart programs,
With DAInamic greetings,

Paul Gobert Luc Maes.

Zoeken in een array

Wie op de DAI een bestand heeft van namen en adressen zal ook zo nu en dan een naam moeten zoeken die op dat moment niet volledig bekend is.
Met een BASIC zoek-programma kan men wel een trefwoord opzoeken (met behulp van de MID$-functie), maar dat duurt nogal (erg) lang.
Vandaar dat ik gezocht heb naar een assembler-routine die snel een trefwoord in een stringarray kan signaleren en die me vertelt, welke index de gevonden string heeft. Ik toets nu 'Jans' en vind Janse tot en met Janssens-de Boer.

De eerste taak die je hebt als je de strings in een array wilt afzoeken is: hoeveel strings moet je controleren en waar staan ze opgeborgen.

Een string-array is twee stappen in de 'HEAP' geborgen:
1 een reeks adressen (pointers) die wijzen naar
2 de reeks strings

---

1 een reeks adressen.

Deze begint met het vaststellen van het aantal dimensies van het array en de grootte ervan:
Voorbeeld: Om het na te kunnen pluizen kunt u het volgende BASIC programma intoetsen:

```
IMPINT
10  DIM A$(2,3)
20  FOR I=0 TO 2:FOR J=0 TO 3:READ A$(I,J):NEXT:NEXT
30  DATA AAP,NOOT,MIES,WIM,ZUS,JET,TEUN,VUUR,GIJS,LAM,KEES,BOK
RUN
```

Met UT en D2EC 28F vind je in de heap vanaf #2EE staan
02 02 03 dus 2 dimensies nl 2 en 3

Dan begint een serie adressen (pointers). In ons voorbeeld zijn dat er 12. Immers: DIM A$(2) geeft drie variabelen omdat A$(0) ook meetelt, en A$(3) geeft er vier; drie maal vier strings dus.

2 de reeks strings

Het eerste adres is bijvoorbeeld 0F 03, dat betekent 30F.
Zoeken op 30F geeft 03 41 41 50; de 03 is de lengte van de string, er komen dus drie letters nl A, A en P want 41 is ASCII-code van de letter A (enz).

Op deze manier ziet u ook dat de pointers in deze volgorde staan:
A$(0,0) ,A$(0,1), A$(0,2); dan A$(1,0), A$(1,1), A$(1,2) enz.

Met deze kennis is het niet zo moeilijk om het aantal te doorzoeken strings te bepalen,ook als er meer dims zijn. Een probleem is nog: WAAR begint dat DIM gedeelte in de heap. Dat adres is nl niet in alle omstandigheden hetzelfde.

Dat is bijvoorbeeld mogelijk langs de volgende weg: Na het uitvoeren van de machine-taalaanroep CALLM wijst het BC register in het programma ("textbuffer") voorbij deze aanroep. Je zoekt dan terug tot je de CALLM-opdracht tegenkomt, dat is de code ("TOKEN"): B3. 7 plaatsen verder in de textbuffer staat een offset, een getal, waarmee BASIC de plaats van A$ bepaalt. Dit gebeurt in een routine die op 0E95A staat. De uitkomst is een adresgetal (vector) dat naar het begin-adres van A$ wijst.

Nog een opmerking: het rekenwerk, dus 3*4 of bij meer dimensies nog meer, gebeurt in een ROM-routine op DE8F.

Nu volgt de source van de het eerste deel "RESET". Dat bepaalt dus van een array het aantal pointers en zet die in MAX en het zoekt de eerste pointer op en zet die in STRING.
De routine wordt aangeroepen met: CALLM #RESET,A$(0,0) of een ander element van A$.

```
                TITL    'ZOEKEN IN EEN STRING-ARRAY'
TL              EQU     12H         ;type/lengte van TT%
EFEPT           EQU     132H
EFSW            EQU     135H
ZKNAAM          EQU     0CA57H
COMPDE          EQU     0DE14H
MAALHL          EQU     0DE8FH
MKVCTR          EQU     0E95AH
;
                ORG     300H
BRESET          JMP     RESET
BZOEK           JMP     ZOEKEN
VARNAM          DB      'TX'
STRING          DW      0H          ;stringvector
AANT            DW      0H          ;aantal vectoren gehad
NUL             DW      0H
INDEX           DW      0H          ;deze bytes blijven 0
                DW      0H          ;wordt de waarde van TX%
FFFF            DW      0FFFFH      ;wordt TX% bij
                DW      0FFFFH      ;  'niet gevonden'
MAX             DW      0H          ;maximum aantal in array
FND             DB      0FFH        ;vind-vlag
;********** R E S E T **************
;zet het aantal string-pointers in MAX
;zet de eerste stringpointer in STRING
RESET           PUSH B              ;BC bewaren voor BASIC
;zoek begin
CALLM           LDAX B
                DCX B
                CPI     0B3H        ;CALLM-token
                JNZ     CALLM
                LXI H   7H          ;7 verder geeft offset
                DAD B
                PUSH H
                POP B               ;via stack in BC
                MVI D   0H          ;MACC niet leegmaken
                CALL    MKVCTR      ;maak vector
                CALL    VCPTR       ;maak pointer in DE
aant.vectoren:  LDAX D
                MOV C,A             ;aantal DIMS in A
                INX D
                LDAX D
                LXI H   0H
                SHLD    AANTAL      ;rangnr van laatste gevonden string
                MOV L,A             ;eerste DIM in L
                INR L               ;rekenen vanaf 1
                INX D
KEER            DCR C
                JZ      K1
                LDAX D
                INX D
                INR A               ;idem
```

```
            CALL    MAALHL    ;HL=HL*A
            JMP     KEER
K1          SHLD    MAX       ;aantal vectoren
            XCHG
            SHLD    STRING    ;eerste vector
            LXI H   FND       ;reset 'gevonden'-vlag
            MVI M   OH
            POP B
            RET
VCPTR       MOV E,M           ;maak pointer in DE van vector in HL
            INH H             ;HL=HL+2
            MOV D,M
            INX H
            RET
```

De volgende delen gaan over (2) het zoeken en (3) hoe kun je bij de DAI een USR-functie verwezenlijken, zodat een CALLM-aanroep terugkomt met een waarde.

ZOEKEN IN EEN STRING-ARRAY (2)

Nu volgt de eigenlijke zoek-routine. Deze wordt aangeroepen met CALLM #ZOEKEN,ZOEK$. ZOEK$ is de naam of het gedeelte van de naam die ik moet opzoeken. Na deze aanroep in de vector - dat is het adres van de pointer- in HL.

De plaats waar je begint met zoeken is door STRING aangegeven in de RESET-routine.

Het zoeken gaat als volgt: Je neemt de eerste letter van ZOEK$ en je loopt daarmee langs een string van het array tot je daar eenzelfde letter gevonden hebt.

Zit die er niet in, dan ga je terug naar VOORT, daar kijk je of je het hele array al afgezocht hebt - KLAAR -, zonee dan het AANTAL gepasseerde strings aanpassen, zodat je straks de index van het array krijgt, en de volgende string kunt afzoeken.

Vind je wel een overeenkomstige letter, dan ga je kijken of de tweede, derde enz. tot en met de laatste letter van ZOEK$ kloppen. Is dat het geval, dan passen we de VIND-VLAG aan en springen we uit de zoekroutine naar VIND, zoniet dan gaan we gewoon met die eerste letter weer verder.

```
ZOEKEN      PUSH B              ;BASIC pointer
            CALL    VECPTR      ;ZOEK$-pointer in DE
            PUSH D              ; bewaren
VOORT       LHLD    MAX
            XCHG
            LHLD    AANT
            INX H
            CALL    COMPDE      ;vergelijk AANT-MAX
            JZ      KLAAR       ;heel array gehad
            SHLD    AANT
            POP D               ;begin weer voor in ZOEK$
            PUSH D              ;bewaren
;maak van STRING een stringpointer; pas STRING aan (+2)
            LHLD    STRING      ;stringvector
            PUSH D
            CALL    VECPTR      ;maak pointer in DE
            MOV A,D
            ORA E               ;lege pointer?
            SHLD    STRING
            XCHG
            POP D
            JZ      VOORT       ;dan volgende.
```



```
            MOV C,M             ;lengte string in C
            MOV A,C
            ORA A
            JZ      VOORT       ;lege string
            INR C               ;straks aftellen tot 0
            INX H
            XCHG                ;stringpointer in DE
            MOV B,M             ;lengte ZOEK$ in B
            INX H
;
;loop hele string door met eerste zoekkarakter:
ZKKAR       LDAX D              ;karakter in A
            INX D               ;volgende karakter staat klaar
            DCR C               ;aantal -1
            JZ      VOORT       ;volgende string
            CMP M
            JNZ     ZKKAR       ;verder in string
;is er wel een karakter gevonden, -zoek verder:
            PUSH B              ;onthoud pointers en tellers
            PUSH D
            PUSH H
ZKVRDR      DCR B               ;nog meer zoekkarakters?
            JZ      VIND        ;zoekstring gevonden
            DCR C               ;hele string gehad?
            JNZ     ZK2         nee
            POP H
            POP D
            POP B
            JMP     VOORT       ;volgende string
;
ZK2         INX H               ;verder in zelfde string
            LDAX D
            INX D
            CMP M
            JZ      ZKVRDR
            POP H               ;pointers en tellers herstellen
            POP D
            POP B
            JMP     ZKKAR
            RET
```

ZOEKEN IN EEN STRING-ARRAY (3)

Dit deel gaat erover, hoe je een getal uit de zoekroutine (nl AANTAL) in BASIC overbrengt zonder allerlei PEEKs en POKEs te moeten gebruiken.

De waarden van een CALLM-'functie'

De uitkomst AANT gaan we in het 4-bytes waardenveld zetten van de variabele TT%. We laten DE naar de gewenste inhoud wijzen:

FFFF (=-1) voor 'niet gevonden', 0 voor 'klaar' en INDEX voor het rangnummer van de gevonden string. Maar eerst moet de uitkomst AANT in de gewenste 4-bytes vorm gezet worden, in de volgorde msb-lsb, hetgeen wil zeggen: eerst het belangrijkste cijfer, aan het eind het minst waardige (net als bij onze decimale getallen; zoals U weet wil de 8080-microprocessor zelf die waarden in omgekeerde volgorde).

```
VIND        POP H
            POP D
            POP B
;nodig om stackniveau te herstellen
```

```
                LXI H     AANT
                LXI D     INDEX+3H  ;begin achteraan met lsb
                MOV A,M
                STAX D
                INX H
                DCX D
                MOV A,M
                STAX D
                DCX D               ;beide msb blijven 0
                DCX D               ;DE wijst begin INDEX aan
                LXI H     FND       ;al eerder iets gevonden?
                MVI A     0H
                CMP M
                JNZ       UIT       ;ja
                DCR M               ;nee zet vlag
                JMP       UIT

KLAAR           LDA       FND       ;wel eens iets gevonden?
                ORA A
                JZ        VINDNT
                LXI D     NUL
                JMP       UIT

VINDNT          LXI D     FFFF
UIT             POP H
                CALL      VAR
                POP B
                RET
```

USR op de DAI

Met CALLM kan men op de DAI wel een machine-taal routine aanroepen en EEN argument meegeven, maar een FUNCTIE is dit niet: men krijgt het bewerkte argument niet naar BASIC terug.

We kunnen dit wel imiteren door een vooraf bepaalde variabele - in dit verhaal de integer TX%- in de variabelenlijst (symboltable) te vullen met de gewenste waarde.

Het belangrijkste probleem is dan: waar is het waardenveld (4 bytes) van die variabele te vinden. We maken daarbij gebruik van een ROM-routine op CA57 (ZKNAAM), die in de variabelenlijst een naam zoekt, de plaats ervoor in HL aangeeft en bij 'aanwezig' met CY=1 terugkomt.

```
VAR             PUSH B
                LXI H     EFSW      ;EFSW=1 dan
                MVI M     1H        ; input from string
                PUSH H
                LXI H     VARNAM
                SHLD      EFEPT     ;EFEPT wijst naar gezochte naam
                PUSH D
                MVI D     TL        ;TYPE/LENGTE
                MVI B     0H
                CALL      ZKNAAM
                POP D
                CC        VUL       ;bij succes vul waardenveld
                POP H
                MVI M     0H        ;zet EFSW terug
                POP B
                RET
```

De inhoud van TX% wordt met vier bytes gevuld die door DE zijn aangewezen:

```
VUL             PUSH B
                MVI B     4H        ;tellertje
```



```
V1              INX H
                LDAX D
                MOV M,A
                INX D
                DCR B
                JNZ       V1        ;klaar? nee
                POP B               ;ja
                RET
EIND            END
```

Hoe ZOEKEN in BASIC gebruikt wordt

* assembleer het programma of toets het in UT in met S300 - enz. en zet het op band of schijf
* bescherm het met S29B -1F -4 -return/B/NEW
* laad het bestandsprogramma en installeer de zoek-functie.

Sjiek is natuurlijk om het programma achter in de symbol-table te assembleren; dan moet het BASIC gedeelte wel eerst voltooid zijn.

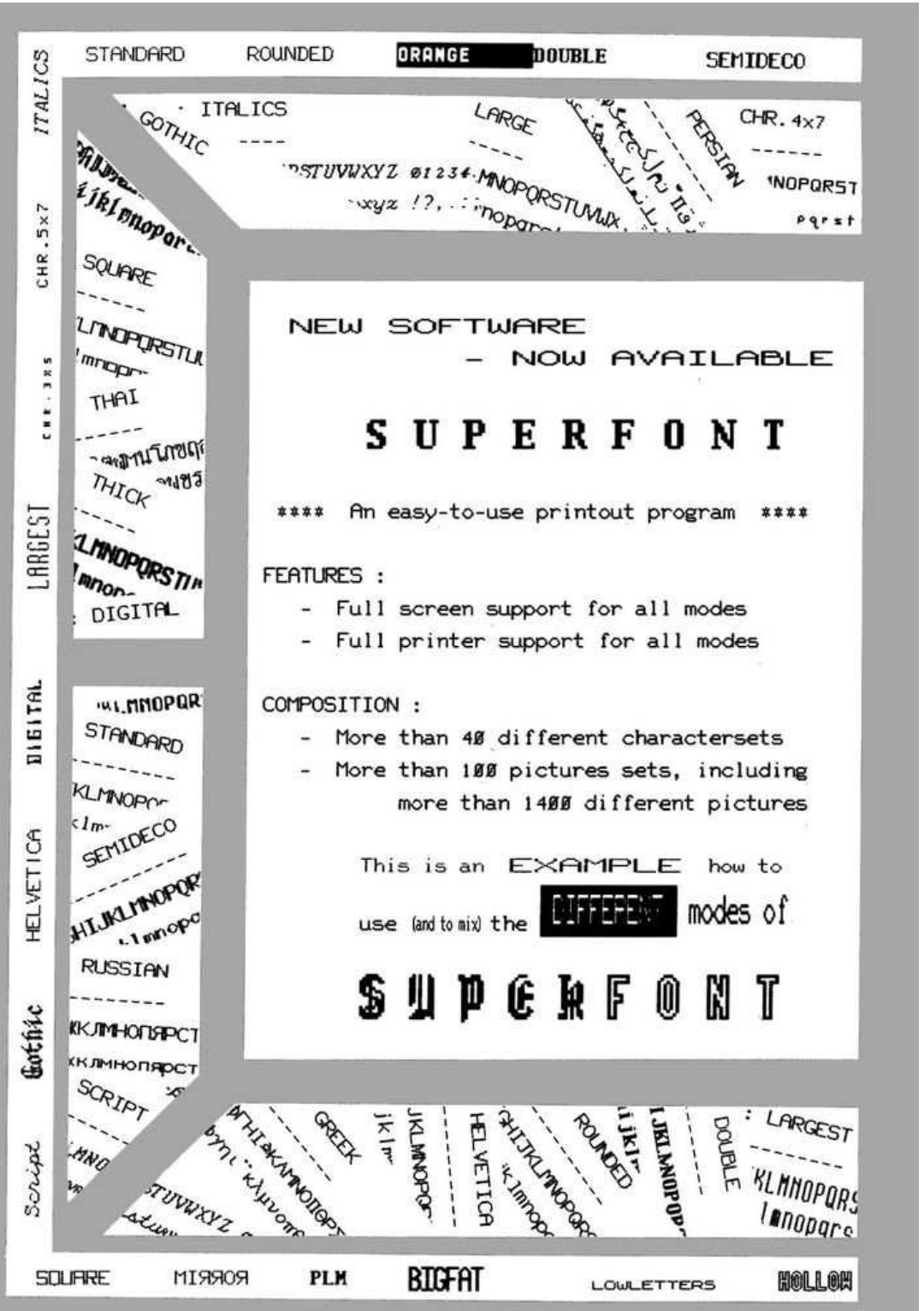
= vooraf moet de variabele TX% gebruikt zijn
= start het zoeken met CALLM #300,xx; hierin is xx het nul-elemnt van het string-array
= zoek met CALLM#303,ZOEK$. Dit kun je herhalen; TX% komt telkens met de volgende vondst terug totdat het hele array is afgezocht. Daarna is TX%=0.
= Als ZOEK$ niet aanwezig is, komt TX% terug met -1
= Maak een index van TX%:
als het array EEn dimensie heeft, dan index=TX%-1
bij TWEE dimensies a en b worden de indices (TX%-1)/a en (TX%-1)/MOD a.

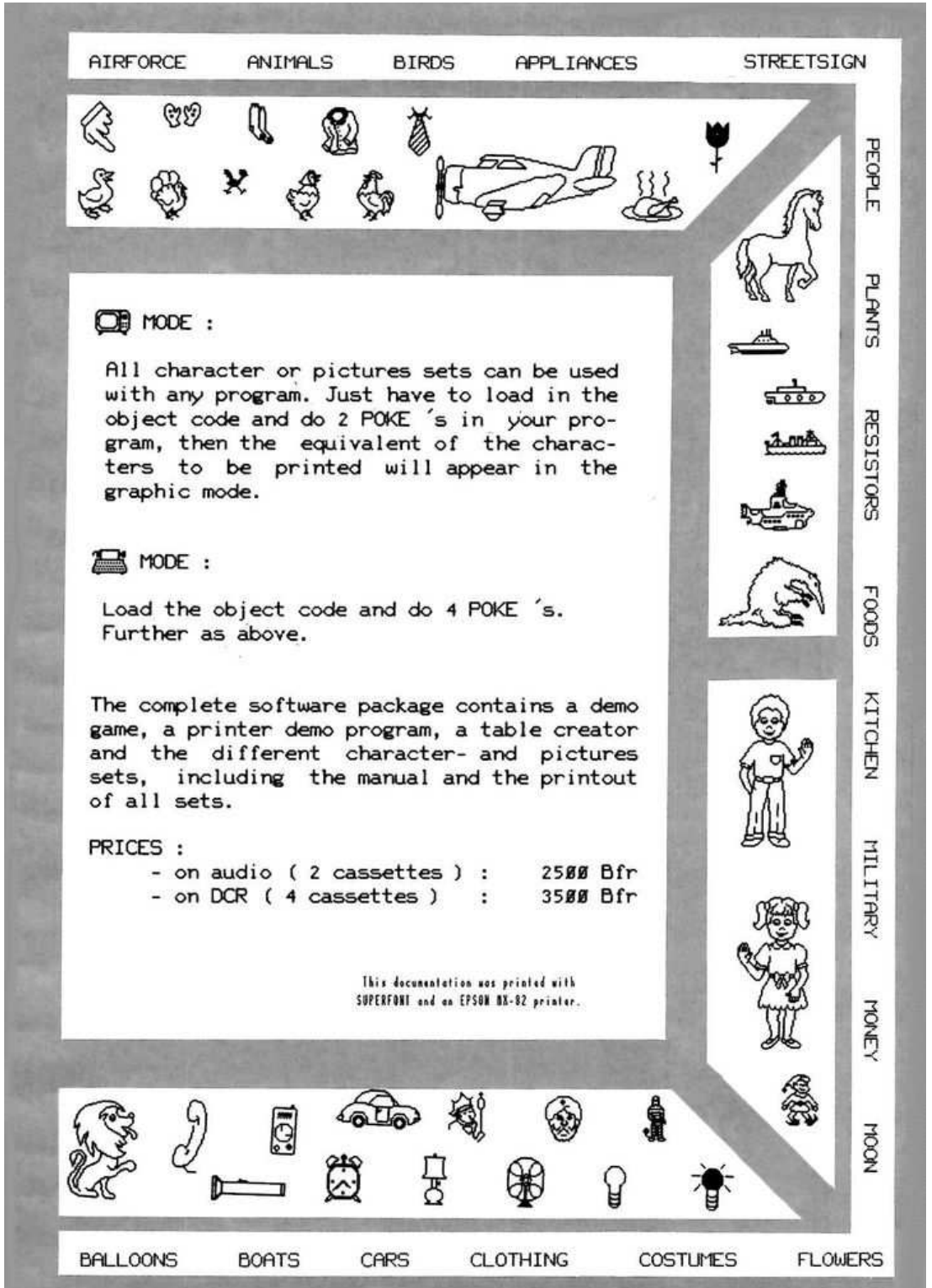
voorbeeld

```
xxxx DIM A$(100,6)
xx00 TX%=0                          zet TX% in de symboltable
xx10 CALLM #300,A$(0,0)             reset
xx20 INPUT ZOEK$
xx30 CALLM #303,ZOEK$               -1=niet gev.0=klaar
xx40 ON SGN(TX)+2 GOTO 80,90,50
xx50 TX=TX-1:I=TX/7:J=TX MOD 7      bereken vanaf index=0
xx60 ?A$(I,J)
xx70 GOTO 30
xx80 ?"niets gevonden"
xx90 GOTO 10
```

```
300 C3 19 03 C3 5C 03 54 58 00 00 00 00 00 00 00 00
310 00 00 FF FF FF FF 00 00 FF C5 0A 0B FE B3 C2 1A
320 03 21 07 00 09 E5 C1 16 00 CD 5A E9 CD 57 03 1A
330 4F 13 1A 21 00 00 22 0A 03 6F 2C 13 0D CA 49 03
340 1A 13 3C CD 8F DE C3 3C 03 22 16 03 EB 22 08 03
350 21 18 03 36 00 C1 C9 5E 23 56 23 C9 C5 CD 57 03
360 D5 2A 16 03 EB 2A 0A 03 23 CD 14 DE CA D7 03 22
370 0A 03 D1 D5 2A 08 03 D5 CD 57 03 7A B3 22 08 03
380 EB D1 CA 61 03 4E 79 B7 CA 61 03 0C 23 EB 46 23
390 1A 13 0D CA 61 03 BE C2 90 03 C5 D5 E5 05 CA B9
3A0 03 0D C2 AB 03 E1 D1 C1 C3 61 03 23 1A 13 BE CA
3B0 9D 03 E1 D1 C1 C3 90 03 C9 E1 D1 C1 21 0A 03 11
3C0 11 03 7E 12 23 1B 7E 12 1B 1B 21 18 03 3E 00 BE
3D0 C2 E7 03 35 C3 E7 03 3A 18 03 B7 CA E4 03 11 0C
3E0 03 C3 E7 03 11 12 03 E1 CD ED 03 C1 C9 C5 21 35
3F0 01 36 01 E5 21 06 03 22 32 01 D5 16 12 06 00 CD
400 57 CA D1 DC 0B 04 E1 36 00 C1 C9 C5 06 04 23 1A
410 77 13 05 C2 0E 04 C1 C9 00 00 00 00 00 00 00 00
```

C.W.A.van Dijk
Watermunt 5
8265 EL Kampen.


STANDARD
ROUNDED
ORANGE
DOUBLE
SEMIDECO
ITALICS
CHR.5x7
LARGEST
DIGITAL
HELVETICA
Gothic
Script
ITALICS
GOTHIC
LARGE
PERSIAN
CHR. 4x7
SQUARE
THAI
THICK
DIGITAL
STANDARD
SEMIDECO
RUSSIAN
SCRIPT
GREEK
HELVETICA
ROUNDED
DOUBLE
LARGEST
NEW SOFTWARE
- NOW AVAILABLE
S U P E R F O N T
**** An easy-to-use printout program ****
FEATURES :
- Full screen support for all modes
- Full printer support for all modes
COMPOSITION :
- More than 40 different charactersets
- More than 100 pictures sets, including
more than 1400 different pictures
This is an EXAMPLE how to
use (and to mix) the DIFFERENT modes of
S U P E R F O N T
SQUARE
MIRROR
PLM
BIGFAT
LOWLETTERS
HOLLOW


AIRFORCE
ANIMALS
BIRDS
APPLIANCES
STREETSIGN
PEOPLE
PLANTS
RESISTORS
FOODS
KITCHEN
MILITARY
MONEY
MOON
MODE :
All character or pictures sets can be used with any program. Just have to load in the object code and do 2 POKE 's in your program, then the equivalent of the characters to be printed will appear in the graphic mode.
MODE :
Load the object code and do 4 POKE 's.
Further as above.
The complete software package contains a demo game, a printer demo program, a table creator and the different character- and pictures sets, including the manual and the printout of all sets.
PRICES :
- on audio (2 cassettes) : 2500 Bfr
- on DCR (4 cassettes) : 3500 Bfr
This documentation was printed with SUPERFONT and an EPSON RX-82 printer.
BALLOONS
BOATS
CARS
CLOTHING
COSTUMES
FLOWERS

W. Herrmann contributions

Sehr geehrter Herr Hermans.

Hiermit sende ich Ihnen einige Beiträge zum Abdruck in Ihrer Club-Zeitschrift DAInamic zu.

Es handelt sich um :

a) Zwei Modifikationen des DNA-Assemblers :

Der Transfer zwischen Text-Source und Editor ist sehr stark beschleunigt worden. (jetzt ca. 60 mal so schnell !)

Mein Basic-Programm DNA-Space-Compression habe ich jetzt als Maschinen-Programm umgeschrieben und direkt in den DNA-Assembler integriert. (Aufruf des Compressors durch 'C.')

b) Modifikation von Fast-Word-Prozessors :

Jetzt sind alle Befehle direkt im Editor aufrufbar, man drückt einfach CTRL und die entsprechende Befehlstaste (T,W,K,L,S,D,P,....).
Außerdem werden die Cursor-Funktionen nicht mehr mit SHIFT + RETURN + CURSOR, sondern nur noch mit CTRL CURSOR aufgerufen.
Es ist jetzt auch moeglich, ein Textfile direkt in BUFFER2 zu laden .....

c) Grafik-Demo-Mode 7 :

Dieses Bild entstand durch ein selbstgebautes Bild-Scanner-Interface und Hilfe meines Druckers.
Im Interface habe ich eine Infrarot-LED-Diode und Infrarot-Transistor eingebaut und an den PDL-Eingang vom DAI angeschlossen. Dann habe ein Orginal-Bild in den Drucker eingespannt und mit Hilfe eines Maschinen-Programms und des Scanners dieses Bild in den DAI übertragen.

Ich wünsche Ihnen ein frohes Weihnachtsfest
und ein gutes neues Jahr 1986.

Willi Herrmann

FWP modifications

Mit diesen Modifikationen laesst sich das Programm FWP besser und einfacher bedienen.

1. Die Kommandos A,L,S,P,M,K,R,E,T,W,D,C,H,U,X,G,I,F,B und Z lassen sich nun auch im EDITOR benutzen.
   Dazu druecht man <CTRL> und die entsprechende Taste.

CTRL x	==> Befehl ausfuehren
CTRL CHARDEL	==> ASCII-Code-Eingabe
CTRL 1-7	==> Ausgabe CHR$(1)-CHR$(7) fuer MARKER

2. Die Cursor-Steuerungen wird vereinfacht:

CTRL + CURRIGHT	==> ein Wort nach rechts
CTRL + CURLEFT	==> ein Wort nach links
CTRL + CURUP	==> ein STEP hoeher
CTRL + CURDOWN	==> ein STEP tiefer
CTRL + SHFT + CURRIGH	==> Zeilenende
CTRL + SHFT + CURLEFT	==> Zeilenanfang
CTRL + SHFT + CURUP	==> 1.Zeile des STEP's
CTRL + SHFT + CURDOWN	==> Letzte Textzeile
CTRL + SPACE	==> 1.Textzeile
CTRL + 'C'	==> Zeile loeschen
CTRL + RETURN	==> Umschaltung Gross/Kleinschrift
CTRL + TAB	==> a) TAB : Tabulator setzen b) DEL : Tabulator loeschen

3. Modifikation SAVE :

   Der Text aus BUFFER1 oder BUFFER2 wird vom Anfang des STEP's bis zum Textende oder bis zum MARKER1 abgespeichert.

4. Modifikation LOAD :

   Nur fuer BUFFER1. Der geladene Text wird an das Textende oder an MARKER1 geladen.

5. Printer Steuerzeichen im Text :

   Steuerzeichen fuer den Printer lassen sich jetzt problemlos in eine Textzeile einfuegen. Diese Zeile wird bei TRIM oder WIDTH richtig behandelt.
   Der Steuercode muss zwischen zwei MARKER3 stehen, als zweistellige Hexadezimal, durch Komma getrennt,

   Beispiel : ▮1B,57,01▮

   entspricht : PRINT CHR$(#1B);CHR$(#57),CHR$(#1)

Modifikationen :

1. UT / R FAST WORD PROCESSOR
2. Aendern der Speicherzellen nach Aufstellung (zB durch SUBSTITUTE in der UTILITY)
3. W400 2AFF FAST WORD PROCESSOR 2

```
#522:   #CD,#29,#05,#C3,#1F,#05,#00,#FE,#0D,#CA,#02,#06,#FE,#20,#CA,#10
        #0D,#FE,#31,#CA,#EF,#05,#FE,#32,#CA,#91,#0B,#CD,#D0,#05,#FE,#43
        #CA,#5E,#06,#FE,#4C,#CA,#E5,#15,#FE,#53,#CA,#1B,#17,#FE,#50,#CA
        #13,#07,#FE,#49,#CA,#C2,#06,#FE,#4D,#CA,#9C,#09,#FE,#48,#CA,#74
        #09,#FE,#52,#CA,#46,#14,#FE,#45,#CA,#98,#18,#FE,#54,#CA,#51,#19
        #FE,#57,#CA,#80,#1A,#FE,#44,#CA,#2E,#1C,#FE,#42,#CA,#D7,#05,#FE
        #41,#CA,#57,#16,#FE,#58,#CA,#7F,#0A,#FE,#47,#CA,#13,#09,#FE,#48
        #CA,#B7,#0B,#FE,#55,#CA,#DE,#0B,#FE,#46,#CA,#1D,#14,#FE,#5A,#CA
        #D7,#05,#C9
#81B:   #C0,#17
#82D:   #C0,#17
#D1D:   #00,#00,#CD,#40,#0F
#E55:   #4F
#ECA:   #91
#EDD:   #00,#00,#00,#00,#00,#00,#3A,#AE,#02,#E6,#40,#CA,#71,#12,#E5,#21
        #00,#00,#22,#3F,#03,#E1,#AF,#32,#1B,#03,#7A,#FE,#61,#DA,#05,#0F
        #FE,#7B,#D2,#05,#0F,#D6,#20,#57,#FE,#09,#CA,#78,#13,#FE,#08,#C2
        #1E,#0F,#3E,#23,#CD,#4F,#0F,#3E,#01,#32,#16,#03,#0E,#02,#C3,#71
        #12,#FE,#0D,#C2,#2F,#0F,#3A,#49,#0F,#2F,#32,#49,#0F,#AF,#57,#C3
        #71,#12,#FE,#31,#DA,#56,#0F,#FE,#38,#D2,#56,#0F,#D6,#30,#57,#C3
        #71,#12,#00,#3A,#49,#0F,#32,#C3,#02,#C9,#00,#00,#FF,#00,#00,#00
        #00,#00,#F5,#21,#03,#B7,#77,#E1,#C9,#7A,#CD,#40,#05
#FA1:   #00,#00,#00
#1288:  #00,#00
#1370:  #49,#0F
#171B:  #3A,#21,#03,#B7,#CA,#38,#17,#21,#8A,#27,#CD,#D4,#DA,#CD,#E8,#05
        #CD,#D0,#05,#FE,#4E,#CA,#7C,#04,#FE,#59,#C2,#28,#17,#2A,#A2,#00
        #E5,#7E,#23,#FE,#02,#D2,#3C,#17,#2B,#E5,#EB,#2A,#A2,#00,#CD,#14
        #DE,#CA,#7C,#04,#21,#62,#27,#CD,#D4,#DA,#CD,#1D,#18,#0E,#01,#3E
        #31,#CD,#C5,#02,#21,#00,#30,#CD,#63,#EF,#D1,#E1,#0E,#FF,#CD,#01
        #EF,#C3,#7C,#04,#00,#CA,#7B,#17,#FE,#0D,#C2,#14,#1A,#C3,#9A,#19
        #23,#7E,#FE,#0D,#CA,#9A,#19,#B7,#CA,#9A,#19,#FE,#03,#C2,#7B,#17
        #C3,#14,#1A,#00,#00,#23,#7E,#FE,#0D,#CA,#C6,#1A,#B7,#CA,#DE,#1A
        #FE
#179D:  #C2,#90,#17,#23,#7E,#C3,#C6,#1A,#00,#00,#23,#7E,#FE,#0D,#CA,#F8
        #1A,#B7,#CA,#1D,#0D,#FE,#03,#C2,#A7,#17,#23,#7E,#C3,#F8,#1A,#00
        #00,#00,#00,#FE,#03,#C2,#44,#08,#23,#7E,#CD,#15,#EB,#DA,#44,#08
        #87,#87,#87,#87,#57,#23,#7E,#CD,#15,#EB,#DA,#44,#08,#B2,#CD,#B6
        #03,#23,#7E,#FE,#2C,#CA,#C5,#17,#FE,#03,#C2,#44,#08,#23,#7E,#C3
        #44,#08,#00,#00,#00,#00,#00,#00,#00,#00,#00,#00,#00,#00,#00,#00
        #00,#00,#00,#00,#00,#00,#00,#00,#00,#00,#00,#00,#00,#00,#00,#00
        #00,#00,#00,#00,#00,#00,#00,#00,#00,#00,#00,#00,#00,#00,#00,#00
#1996:  #03,#C3,#70,#17
#1A5E:  #00,#00,#00,#00,#00,#00
#1AC1:  #F8,#19
#1ACC:  #03,#CA,#90,#17,#FE,#20,#C2,#DE,#1A,#23,#0D,#C3,#BA,#1A,#00,#00
        #00,#00
#1AE8:  #03,#CA,#A7,#17,#CD,#F8,#19,#C2,#F8,#1A,#00,#00,#00,#00
#1B10:  #F8,#19
```


DNA space compression

```
DNA - SPACE - COMPRESSION 2
===========================

Dieses Maschinen-Programm beseitigt alle ueberfluessigen Leerzeichen
in einem DNA-Assembler-Source.

Dieses MLP wird aber im Gegensatz zu dem in DAInamic (85-28 S.151)
abgedruckten Basic-Programm direkt in das Assembler-Programm integriert
(Adr. #1080-#10FF). Daduch braucht es nicht jedesmal getrennt ein-
gelesen werden.

Im Hauptprogramm wurde deshalb eine zusaetzliche Tastenabfrage ein-
gebaut (Adr. #2BD8), damit der Assembler den Befehl 'C.' (Compress)
kennt. Vor dem Abspeichern eines Source-Files sollte man JC. ein-
geben, damit die ueberflüssigen SPACE's beseitigt werden.

Modifikation :

1.  UT / R DNA ASSEMBLER
2.  Eingabe der MLP-Daten mit Substitute (#1080-#10FF)
3.  Eingabe des Sprungvektors (#2BD8-#2BDC)
4.  W1080 2FFF DNA ASSEMBLER 2
```

```
1080  CD 01 2C 2A 3B 12 44 4D 2A 45 12 54 5D AF 32 FC
1090  10 2B 7E 32 F9 10 23 7E 32 FA 10 23 7E 32 FB 10
10A0  2B 3A F9 10 FE 8D C2 B6 10 3A FA 10 FE AA C2 B6
10B0  10 3E FF 32 FC 10 3A FA 10 FE A7 C2 C5 10 3A FC
10C0  10 2F 32 FC 10 3A FC 10 FE 00 C2 E1 10 3A FA 10
10D0  FE A0 C2 E1 10 3A FB 10 FE A0 C2 E1 10 23 C3 91
10E0  10 7E EB 77 EB 23 13 FE 8D C2 F1 10 AF 32 FC 10
10F0  0B 78 B1 C2 91 10 C3 19 11 00 00 00 00 00 00 00

2BD8  FE C3 CA 80 10
```

Fast up DNA-editor

```
10    REM
11    REM FAST UP DNA-EDITOR                    01.12.85
12    REM
13    REM WILLI HERRMANN
14    REM RICHARD-DEHMEL-STR.4
15    REM D-4320 HATTINGEN
16    REM
17    REM
18    REM Mit dieser Aenderung der DNA-EDITOR-Routine
19    REM wird die Zeit beim EDIT-Kommando ( JE1 XX. )
20    REM auf einen Bruchteil gesenkt.
21    REM
22    REM z9  200 Source-Zeilen :
23    REM .                          vorher ca. 120 sek.
24    REM .                          jetzt  ca.   5 sek.
25    REM
26    REM Modifikation von DNA :
27    REM
28    REM UT / R DNA-ASSEMBLER / B
29    REM LOAD "FAST DNA-EDITOR":RUN
30    REM UT / W1100 2FFF DNA-ASSEMBLER
31    REM
32    REM
1000  CLEAR 256
1010  FOR ADR=#2D04 TO #2D68
1020  READ BYTE
1030  POKE ADR,BYTE
1040  NEXT ADR
1050  REM
60000 DATA #CD,#DD,#11,#2A,#53,#12,#22,#73
60010 DATA #2E,#2A,#3D,#12,#22,#75,#2E,#21
60020 DATA #00,#00,#22,#B4,#00,#21,#00,#30
60030 DATA #22,#A2,#00,#22,#A4,#00,#2A,#47
60040 DATA #12,#2B,#22,#A6,#00,#2A,#75,#2E
60050 DATA #CD,#21,#16,#21,#9D,#17,#3E,#8D
60060 DATA #32,#D1,#17,#7E,#E6,#7F,#CD,#75
60070 DATA #DD,#23,#FE,#0D,#C2,#37,#2D,#2A
60080 DATA #75,#2E,#7E,#23,#FE,#8D,#C2,#46
60090 DATA #2D,#22,#75,#2E,#2A,#73,#2E,#23
60100 DATA #22,#73,#2E,#7C,#B5,#FA,#29,#2D
60110 DATA #AF,#CD,#75,#DD,#EF,#2A,#3E,#5F
60120 DATA #32,#75,#00,#00,#00
```

Drucker-bild-scanner

Text highlighting

```
59010 REM by Emil Zahner CH 8910 Affoltern 16.March 1984
59020 REM use it to highlight menu sections or selected instructions
59030 REM note 5 lines are scrolling
59040 REM large letter can be had with poke #bfef but an extra
59050 REM cleaning operation is needed because line 0 becomes invisible
59060 REM screen color line numbers are counted from top
59070 REM this operates with 48 kbyte only
59080 REM this is a demo program. Type in horizontal position
59090 REM of text and select coloring J / no coloring N / end of program E
59100 REM have fun              IMPINT !!
59120 GOSUB 59450:REM Bildschirm Aufbau *** uebergehe mit REM screen
59130 GOSUB 59180:REM Position fuer Demotext eingeben
59140 GOSUB 59260:REM Feldfaerbung Variable festlegen set
59150 GOSUB 59330:REM faerben (reverse video) oder loeschen
59160 GOSUB 59380:REM Feldfarbe verarbeiten do
59170 GOTO 59130
59180 INPUT "HOR POSITION 0 40 ";HOR%:FOR I%=0 TO 5:PRINT TAB(I%);I%:NEXT
59190 LANGBLOCK%=10:POSHO%=10:ZEIL0%=15:ZEILZAHL%=4
59200 CURSOR HOR%,15:CHO%=CURX:CVO%=CURY:PRINT "AAAAA";
59210 CURSOR HOR%,14:PRINT "BBBBB";
59220 CURSOR HOR%,13:PRINT "CCCCC";
59230 CURSOR HOR%,12:PRINT "DDDDD";:CHU%=CURX:CVU%=CURY
59240 RETURN
59250 REM FELD FAERBEN VARIABLE FESTLEGEN
59260 SCRBYTE%=0:REM SCRBYTE muss 0 oder 2 sein
59270 POSHO%=CHO%:ZEIL0%=CVU%:REM 0= unterste Zeile des zu faerbenden Feldes
59280 ZEILZAL%=CVO%-CVU%+1:LANGBLOCK%=CHU%-CHO%
59290 NULPO%=#BFEF-#D11:POSHO1%=(59-POSHO%)*2
59300 ZEIL1%=ZEIL0%+1:VERPO%=ZEIL1%*#86+NULPO%
59310 LANGBLO%=LANGBLOCK%*2+SCRBYTE%/2-1
59320 RETURN
59330 COLT%=0:PRINT :PRINT "FAERBEN highlight  J / N   E=END ";
59340 G%=GETC:IF G%=0 THEN 59340
59350 IF G%=ASC("J") THEN COLT%=#FF
59360 IF G%=ASC("E") THEN PRINT CHR$(12):MODE 0:END
59370 RETURN
59380 FOR HORCOL%=0 TO LANGBLO% STEP 2
59390 FOR VERCOL%=0 TO ZEILZAL%-1
59400 POKE VERPO%+(POSHO1%)-HORCOL%+VERCOL%*#86+SCRBYTE%,COLT%
59410 NEXT:NEXT
59420 CURSOR 5,1
59430 RETURN
59440 REM - ---------------------------------------
59450 SCROLLZ%=5:FARPOS%=0:FARPOS0%=1:FARPOS1%=3:FARPOS2%=17
59460 MODE 0:PRINT CHR$(12);:REM SCHIRM AUFT
59470 COLORT 6 0 11 0
59480 SCROLLZ%=#B3E5+SCROLLZ%*#86:POKE #8A,SCROLLZ% MOD 256
59490 POKE #8B,SCROLLZ% SHR 8
59500 POKE #BFEE-FARPOS%*#86,198:REM neu
59510 POKE #BFEE-FARPOS0%*#86,202:REM orig 202
59520 POKE #BFEE-FARPOS1%*#86,204:REM orig 204
59530 POKE #BFEE-FARPOS2%*#86,203:REM orig 203
59540 REM POKE #BFEF-#86,#6A:REM breit
59550 CURSOR 20,23:PRINT "prior vorherige Funktion"
59560 CURSOR 10,21:PRINT "********** Now Zur Zeit im Betrieb **********"
59570 CURSOR 0,5
59580 RETURN
59590 REM lines on poke farpos see DAINAMIC 82/50
59600 REM line poke #bfef see DAINAMIC 10/82
59610 REM line scroll see DAINAMIC 11/179
59620 REM line scrbyte to langblo, ask Belgium club for program
59630 REM 'screenreading' by E. ZAHNER
```


DAI-typewriter

Met dit programma kan je de computer gebruiken als schrijfmachine met enkele extra-mogelijkheden.

Het toetsenbord wordt automatisch in "LOWER CASE" gezet. De ingevoerde tekst gaat via een GETC-routine, zodat alle leestekens, inklusief komma en aanhalingstekens mogelijk zijn.

Daarenboven kan je de TABULATOR op een vooraf gekozen positie instellen. Deze instelling, evenals de printer-bevelen (hier voor EPSON), verschijnen in een vast schermhoofd, waaronder de tekst scrollt.

De printer-commando's blijven gelden tot wanneer met de toets <CURSOR UP> de printer geinitialiseerd wordt. Deze commando's gelden voor een volle lijn en zijn niet per woord instelbaar. Met de toets <~> wordt het scherm schoongemaakt.

Alvorens afgedrukt te worden verschijnt de ingetypte tekst, per lijn, op het scherm en kan alsdan nog verbeterd worden.

ZIEHIER ENKELE MOGELIJKHEDEN:

<CURSOR LEFT>:
CONDENSED PRINT MODE
<CURSOR RIGHT>:
ENLARGED
<CURSOR DOWN>:
SUBSCRIPT MODE
<SHIFT>+<CURSOR UP>:
EMPHASISED
<SHIFT>+<CURSOR LEFT>:
LINE SPACING 7/72;;
LINE SPACING 7/72;;
<SHIFT>+<CURSOR RIGHT>:
DOUBLE PRINT
<SHIFT>+<CURSOR DOWN>:
UNDERLINED

KOMBINATIES VAN DEZE PRINT-MODES ZIJN TOEGELATEN VOLGENS DE SPECIFIKATIES VAN DE PRINTER-HANDLEIDING.

PROGRAMMA LADEN EN KIJKEN WAT ER GEBEURT!

Met vriendelijke DAI-groeten,
Luc Beyens
Vinkendal,32

9810 GENT/Dr

Tel.: 091/26.52.05

DAI-typewriter

```
PAGE 01  --  DAI-TYPEWRITER

10      MODE 0:COLORT 8 0 15 0:POKE #131,1:POKE #75,#FF:
        PRINT CHR$(12);
20      GOSUB 1000
30      PRINT :PRINT "KEYBOARD AUTOMATICALLY SET IN 'LOWER
        CASE'!"
31      PRINT "USE CURSOR-KEYS (+ SHIFT) FOR PRINT-COMMANDS"
32      PRINT "<~>= CLEAR SCREEN"
33      PRINT "PRINTER ON PLEASE !"
34      INPUT "SET TABULATOR ...";T:CURSOR 56,21:PRINT T:T=
        T-1
50      VLAG1=0:VLAG2=0:VLAG3=0:VLAG4=0:VLAG5=0:VLAG6=0:
        VLAG7=0:VLAG8=0:VLAG9=0
60      PRINT CHR$(12);:POKE #2C3,#FF:REM LOWER CASE INPUT
        IF GETC
70      GOSUB 100
80      POKE #131,0:CURSOR 0,CURY:GOSUB 400:CURSOR 0,CURY
81      IF VLAG2=1 THEN PRINT TAB(T);
82      PRINT TEXT$;SPC(SP)
83      SP=0
90      POKE #131,1:GOTO 70

100     REM * INPUT TEXT AND PRINT-COMMANDS *
110     TEXT$=""
120     J=GETC:IF J=0 GOTO 120
121     IF J=13 THEN RETURN
122     IF J=8 AND CURX>0 THEN GOSUB 200
123     IF J<32 THEN GOSUB 300:IF J<32 GOTO 120
124     IF J=126 THEN PRINT CHR$(12);:GOTO 100
130     J$=CHR$(J):PRINT J$;
140     TEXT$=TEXT$+J$
141     IF LEN(TEXT$)>55 THEN GOSUB 520
142     IF VLAG2=1 AND LEN(TEXT$)>55-T THEN GOSUB 500
143     IF VLAG5=1 AND LEN(TEXT$)>34 THEN GOSUB 500
144     IF VLAG2=1 AND VLAG5=1 AND LEN(TEXT$)>34-T THEN
        GOSUB 500
145     IF CURX=60 GOTO 600
190     GOTO 120

200     REM * SUBROUTINE 'BACK SPACE' *
210     CURSOR CURX-1,CURY:PRINT " ";:CURSOR CURX-1,CURY
220     TEXT$=LEFT$(TEXT$,LEN(TEXT$)-1)
299     RETURN

300     REM * CURSOR-KEYS *
310     IF J=9 THEN GOSUB 2020:REM [TAB]=TABULATOR
311     IF J=16 THEN GOSUB 2010:REM [CURSOR UP]=INIT PRINTER
312     IF J=17 THEN GOSUB 2030:REM [CURSOR DOWN]=SUBSCRIPT
313     IF J=18 THEN GOSUB 2040:REM [CURSOR LEFT]=CONDENSED
314     IF J=19 THEN GOSUB 2050:REM [CURSOR RIGHT]=ENLARGED
315     IF J=20 THEN GOSUB 2060:REM [SHIFT+CURSOR UP]=
        EMPHASISED
316     IF J=21 THEN GOSUB 2070:REM [SHIFT+CURSOR DOWN]=
        UNDERLINED
317     IF J=22 THEN GOSUB 2080:REM [SHIFT+CURSOR LEFT]=
        LINE SPACING 7/72''
318     IF J=23 THEN GOSUB 2090:REM [SHIFT+CURSOR RIGHT]=
        DOUBLE PRINT
399     RETURN

400     REM * PRINT-COMMANDS *
```



```
PAGE 02  --  DAI-TYPEWRITER

410     IF VLAG1=1 THEN PRINT CHR$(27);CHR$(64);:VLAG1=0:
        REM INIT PRINTER
411     IF VLAG3=1 THEN PRINT CHR$(27);"S";CHR$(1);:REM
        SUBSCRIPT
412     IF VLAG4=1 THEN PRINT CHR$(15);:REM CONDENSED
413     IF VLAG5=1 THEN PRINT CHR$(27);"W";CHR$(1);:REM
        ENLARGED
414     IF VLAG6=1 THEN PRINT CHR$(27);"E";:REM EMPHASISED
415     IF VLAG7=1 THEN PRINT CHR$(27);"-";CHR$(1);:REM
        UNDERLINED
416     IF VLAG8=1 THEN PRINT CHR$(27);"1";:REM LINE
        SPACING 7/72''
417     IF VLAG9=1 THEN PRINT CHR$(27);"G";:REM DOUBLE PRINT

418     REM
499     RETURN

500     REM * END LINE SIGNAL *
510     FOR X=1 TO 3:GOSUB 520
511     NEXT
520     SOUND 0 0 15 0 FREQ(1200):WAIT TIME 5:SOUND OFF :
        RETURN
599     RETURN

600     REM * NEW LINE *
610     SP=0
620     IF TEXT$="" GOTO 699
630     IF RIGHT$(TEXT$,1)=" " OR RIGHT$(TEXT$,1)="-" GOTO
        699
640     SP=SP+1:IF SP=60 THEN SP=0
650     TEXT$=LEFT$(TEXT$,LEN(TEXT$)-1):GOTO 620
699     RETURN

1000    REM * SCREEN-LAY-OUT *
1010    POKE #BFEE,#C7:POKE #BDD6,#C8:REM SCREEN COLOR
1020    POKE #8A,#51:POKE #8B,#BD:REM NON-SCROLL ZONE

1100    REM * TITLE *
1110    CURSOR 0,23
1120    FOR X=0 TO 14:PRINT CHR$(#FF);:NEXT
1121    PRINT " D A I - T Y P E W R I T E R ";
1122    FOR X=0 TO 3:PRINT CHR$(#FF);:NEXT:PRINT "By
        L.BEYENS";:PRINT CHR$(#FF)
1130    FOR X=0 TO 59:PRINT CHR$(#FF);:NEXT:PRINT
1140    PRINT CHR$(#FF);:PRINT "INIT";
1141    PRINT CHR$(#FF);:PRINT "SUBSCR";
1142    PRINT CHR$(#FF);:PRINT "CONDEN";
1143    PRINT CHR$(#FF);:PRINT "ENLARG";
1144    PRINT CHR$(#FF);:PRINT "EMPHAS";
1145    PRINT CHR$(#FF);:PRINT "UNDERL";
1146    PRINT CHR$(#FF);:PRINT "7/72";
1147    PRINT CHR$(#FF);:PRINT "DOUBLE";
1148    PRINT CHR$(#FF);:PRINT "TAB    ";CHR$(#FF)
1150    FOR X=0 TO 59:PRINT CHR$(#FF);:NEXT:PRINT
1160    PRINT "1.......1";CHR$(#A);"0.......2";CHR$(#A);
        "0.......3";CHR$(#A);
1161    PRINT "0.......4";CHR$(#A);"0.......5";CHR$(#A);
        "0.......6";CHR$(#A)
1199    RETURN
```

```
PAGE 03  --  DAI-TYPEWRITER

2000    REM * CHOISE-INDICATION AND PRINTER-FLAGS *
2010    VLAG1=1:FOR X=#BED6 TO #BECF STEP -2:POKE X,#FF:NEXT
2011    VLAG2=0:VLAG3=0:VLAG4=0:VLAG5=0:VLAG6=0:VLAG7=0:
        VLAG8=0:VLAG9=0
2012    FOR X=#BE6E TO #BE63 STEP -2:POKE X,0:NEXT:FOR X=
        #BECC TO #BEC1 STEP -2:POKE X,0:NEXT
2013    FOR X=#BEBE TO #BEB3 STEP -2:POKE X,0:NEXT:FOR X=
        #BEB0 TO #BEA5 STEP -2:POKE X,0:NEXT
2014    FOR X=#BEA2 TO #BE97 STEP -2:POKE X,0:NEXT:FOR X=
      #BE94 TO #BE89 STEP -2:POKE X,0:NEXT
2015    FOR X=#BE86 TO #BE7F STEP -2:POKE X,0:NEXT:FOR X
        #BE7C TO #BE71 STEP -2:POKE X,0:NEXT
2019    WAIT TIME 20:GOSUB 500:FOR X=#BED6 TO #BECF STEP -2:
        POKE X,0:NEXT:RETURN
2020    VLAG2=1:FOR X=#BE6E TO #BE63 STEP -2:POKE X,#FF:
        NEXT:RETURN
2030    VLAG3=1:FOR X=#BECC TO #BEC1 STEP -2:POKE X,#FF:
        NEXT:RETURN
2040    VLAG4=1:FOR X=#BEBE TO #BEB3 STEP -2:POKE X,#FF:
        NEXT:RETURN
2050    VLAG5=1:FOR X=#BEB0 TO #BEA5 STEP -2:POKE X,#FF:
        NEXT:RETURN
2060    VLAG6=1:FOR X=#BEA2 TO #BE97 STEP -2:POKE X,#FF:
      NEXT:RETURN
2070    VLAG7=1:FOR X=#BE94 TO #BE89 STEP -2:POKE X,#FF:
        NEXT:RETURN
2080    VLAG8=1:FOR X=#BE86 TO #BE7F STEP -2:POKE X,#FF:
        NEXT:RETURN
2090    VLAG9=1:FOR X=#BE7C TO #BE71 STEP -2:POKE X,#FF:
        NEXT:RETURN
```

Stringcomparator

```
PAGE 01  --  STRINGCOMPARATOR #1

1000    REM +++ DEMO 0017-05-00 / V1.0
1010    REM +++ string comparator #1
1020    REM +++ DAI
1100    PRINT :INPUT "Wat is de hoofdstring  : ";X$
1200    PRINT :INPUT "Hoe luidt de substring : ";XS$
1300    GOSUB 31102:REM string comparator #1
1400    PRINT :PRINT "De waarde van X is :";X
17000   GOTO 1100
17500   END

21025   REM %%%%% string comparator #1 %%%%%
21031   XL=LEN(XS$):XR=LEN(X$)+1
21032   IF XL>=XR THEN XL=XR-1
21033   FOR X=1 TO XR-XL
21034 1  IF MID$(X$,X-1,XL)=XS$ GOTO 21036
21035   NEXT X:X=0
21036   RETURN
31102   GOTO 21031:REM string comparator
```


EDITO

Nous souhaitons la bienvenue à tous nos nouveaux lecteurs, ainsi qu'à nos amis de DAInamic. Pour ceux qui ne le savent pas encore, DAICLIC est une publication du club IDC (International DAI Club ASBL). IDC a été crée fin 1984 sur l'initiative de quelques clubs DAI francophones qui souhaitaient regrouper une partie de leurs activités. Les originalités d'IDC sont:
- ses clubs associés constituant des points de rencontre entre les membres;
- ses logiciels dont 90% des bénéfices retournent aux auteurs. C'est le prix payé par IDC pour motiver la création de software sur DAI;
- son magazine entièrement en français;
- son service télématique pour les utilisateurs de modems;
- etc.

En même temps que l'année 1986, est apparue une transformation radicale de DAICLIC ! En effet, l'accord qui unit les publications des deux clubs DAInamic et IDC, nous permet de vous proposer un magazine double réalisé d'une façon nettement plus professionelle ! Nous espérons que ce nouveau look vous plaira et que vous serez intéressé par les articles de la partie "DAInamic".

Cette collaboration correspond tout à fait à un des objectifs d'IDC: la possibilité pour les membres de tous les clubs DAI de disposer de l'ensemble des informations émises par ces clubs. En effet, à partir de ce numéro, les membres de IDC recevront automatiquement "DAInamic" et réciproquement. D'autre part, pour rassurer ceux qui ne comprennent pas les langues germaniques (néerlandais, allemand, anglais), DAICLIC continuera d'être entièrement rédigé en français, ils n'y perdent donc rien.

Un autre changement non négligeable est la parution bimestrielle de la revue. Cela implique donc moins de pages à la fois, mais plus souvent ! Finalement, la quantité d'informations reste sensiblement la même.

Concernant le software, IDC continue d'éditer de nouveaux logiciels et donne l'accès à ses membres à toute la programmathèque de DAInamic. Pour obtenir les logiciels de DAInamic, vous pouvez soit utiliser la procédure normale via IDC, soit passer commande directement chez DAInamic.

Côté télématique, les activités se développent de plus en plus et une partie croissante de nos membres s'équipent de modems. Nous rappellons qu'il existe un service télématique IDC disponible 24 heures sur 24. Plus de renseignements dans ce numéro.

Nous restons aussi toujours ouverts à la création de clubs régionaux. C'est grâce à ces clubs que les membres peuvent se rencontrer lors de réunions et échanger leurs idées et réalisations. Si vous êtes intéressé par la création d'un tel club, n'hésitez pas à nous contacter.

Nous remercions ceux qui ont participé à cette revue. Le nombre de pages de ce premier numéro de l'année ayant du être réduit pour des raisons techniques, nous demandons à toutes les personnes dont les articles n'ont pu être publiés par manque de place, de nous excuser et de ne pas s'inquiéter, leurs contributions paraîtront ultérieurement.

I.D.C. ASBL.

DAI soft

New programs

IDC software : MXDIR

MXDIR est un utilitaire qui permet de sauvegarder par transferts des DIRECTORY. MXDIR a une capacité de 64 "DIRECTORY", soit 64 disques 'simple face' ou 32 disques 'double face'. Il est nécessaire de posséder deux lecteurs. Il existe deux version de MXDIR : MXDIR v2 pour les lecteurs simple face et MXDIR v3 pour les lecteurs double face...système KENDOS uniquement.

MXDIR, maxi-directory, offre 8 fonctions principales et 53 fonctions secondaires.

En quelques mots, MXDIR garde en mémoire le directory de 64 disquettes. Le nom de chaque disquette (NAME) est placé dans une liste de toutes les disquettes disponibles. Vous pouvez choisir une des disquettes à visualiser (par ex: DISK JEUX 1...) et vous verrez alors s'afficher à l'écran le directory de cette disquette. Dans ce directory principal, vous pouvez encore appeler séparément tous les programmes de la disquette pour avoir de plus amples renseignements à leur sujet... par exemple, emplacement sur le disque, longueur,... Vous pouvez également rechercher un programme dans votre catalogue... Par exemple, vous cherchez le programme "ZZTOP", mais vous ne savez pas où il se trouve... et hop ton appel à MXDIR et vous le retrouvez directement grace à une fonction de recherche... Vous pouvez également imprimer des listings de vos directory selon différentes caractéristiques.

MXDIR reconnait également plusieurs nouveaux types de fichier... 23 nouveaux types pour etre précis... par ex: UTP (utilitaires de programmation...

En bref, ce programme est très pratique pour gérer sa logithèque... et pourra aussi vous sauver si vous effacez par erreur le directory d'une de vos disquettes ! ce directory étant toujours enregistré dans MXDIR, vous pourrez récupérer vos programmes à l'aide de la commande MANUAL...

MXDIR est donc disponible uniquement sur KENDOS, livré avec un mode d'emploi de 6 pages...

Prix sur disquette KENDOS : 1000 FB. 150 FF.

(c) IDC software et D. Boiteau '86

IDC software : CDM

Encore un programme de plus dans la série des programmes d'IDC software. Ce programme séduira très certainement tous les amateurs de modems... il se nomme CDM...

Ce programme est distribué sous la forme d'une EPROM, à mettre à la place de l'EPROM DCR qui se trouve sur la carte TOSDCR ou à la place de l'EPROM DCR qui se trouve sur la carte KENDOS à l'intérieur du DAI. Vous l'aurez compris, ce programme n'est donc disponible que sur KENDOS ou DCR... et se charge très rapidement en mémoire. En effet, en mode commande, il suffit de taper "CDM" et le programme se charge et se lance en moins de temps qu'il n'en faut pour le dire...

Ce programme vous permet donc de connecter 2 DAI entre-eux ou deux machines différentes par l'intermédiaire d'un modem ou d'un cable null-modem...

CDM possède un nombre impressionnant de commandes... Il possède également deux tampons : un tampon de réception et un tampon d'édition dont les grandeurs sont paramétrables... très pratique donc pour préparer son courrier à l'avance en mode EDITEUR et pour l'envoyer ensuite sur le réseau, tout ceci pendant que le programme continue encore à recevoir des caractères. Un annuaire est également disponible dans ce programme, ainsi que différentes procédures de connections. Les tampons précités peuvent recevoir ou émettre des caractères en HEX à tout comme en ASCII. On peut lister, restaurer, échanger le contenu des deux tampons. BREAK fait toujours revenir au menu principal. Un mode dialogue est également disponible pour une communication immédiate. En mode dialogue, on peut envoyer différents codes après une pression sur TAB. Le format de transmission peut également être modifié... (parité, vitesse....). Une commande de filtrage est également existante pour supprimer automatiquement certains caractères à l'émission ou à la réception... Plusieurs autres commandes sont également disponibles...mais nous n'avons pas la place pour les citer toutes ici...

Sachez enfin que ce programme est livré avec un mode d'emploi très complet. (14 pages. format A4).

A ne pas manquer pour tous les 'dinques' de la communication... et pour les autres aussi...

CDM V1.6. par Eric Choupinet et Claude Picard, distribué par IDC software. Prix EPROM conjoint : 1500 FB/ 210 FF.

Disponible uniquement sur KENDOS ou DCR. version à préciser lors de la commande.

IDC software '86.

IDC software

problème d'un menu à l'autre en présent simplement une touche. Ce programme permet de tirer soit des étiquettes, soit des listes de votre fichier. Efficace et rapide...! (DCR/AUDIO) 1500 FB.

DAIBASE II : programme de création de gestion de fichiers indexés ou non. Deux originalités par rapport aux autres gestionnaires de fichiers DAI : les fichiers sont stockés et utilisés sur disquette. Une fois le programme chargé, vous pouvez taper directement des commandes sans devoir charger le fichier en RAM. Dès que vous avez fini d'utiliser un fichier, vous ne devez pas le ré-enregistrer, cela se fait tout seul et cette opération est assurée des modifications... ensuite, ce programme ne fonctionne pas selon une suite de menu mais selon des COMMANDES. C'est en fait une sorte d'interpréteur... Plusieurs commandes sont à votre disposition... (KENDOS/VC 1541) 1800 FB.

Jeux :

OTHELLO MICROTEX : Tous nos amis Français connaissent le concours d'OTHELLO organisé chaque année lors du SICOB par la revue l'ordinateur Individuel. Ce qu'ils ne savent peut-être pas, c'est que la branche belge de l'OI organise également cela à Bruxelles lors du salon Microtex. Cet OTHELLO a participé à ce concours et s'est classé 4ième.... Cela vous en dit beaucoup sur sa façon de jouer... Outre cette qualité primordiale pour un bon jeu d'OTHELLO, ce programme possède également une présentation TRES soignée... à ne pas manquer... (CAS/DCR) 800 FB.

DAI RUNNER : qui ne connait pas le célèbre LODE RUNNER tournant sur ordinateur Apple ??? Et bien ici, vous prenez un DAI et vous faites la même chose... Différentes vitesses, 50 terrains de jeu (niveaux) différents...superbe quoi ! (DCR/KENDOS/VC 1541 : 500 FB. disquette comprise. (ne tourne pas sur cassette audio).

QUIX : une adaptation valable du jeu STYX tournant sur Commodore 64, Apple, IBM pc... etc. (DCR/KENDOS/VC 1541) : 500 FB. audio : 400 FB.

et les auteurs ?

DAITEXT v.21	: Alain Marlette	OTHELLO : Chris. Pools
EDITEUR DE DESSINS	: Walter Costa	DAI RUNNER : Mr Stransky
MAILING LIST	: Christian Pools	QUIX : Pascal Janin
DAIBASE II	: Christian Pools	

Les prix cités plus haut sont ceux des versions cassette audio (seuls pour DAI RUNNER et QUIX). il faudra encore ajouter à cela 250 FB pour une version disquette ou cassette digitale. (DCR).

N'hesitez pas à envoyer vos réalisations à Marc Vandermeersch, 17 av du Vert Bocage, B-1410 Waterloo. (Tel.:02/354.13.63)

MAILING LIST : un questionnaire de fichiers d'adresses classique. Retient environ 240 fiches en mémoire en 1 fois, mais possède aussi deux autres caractéristiques alléchantes : le fichier est sauvé sur un seul fichier cassette de longueur toujours égale. Vous pouvez donc ajouter des fiches à votre fichier sans prendre plus de place sur la cassette... et de ce fait sans effacer le programme suivant. La deuxième caractéristique alléchante de ce programme est son fonctionnement par menus...vous passez sans

IDC software

Bonjour d'outre vous découvrant IDC pour la première fois sur DAI tout...

Il nous semble donc bon de vous rappeller la liste des softwares à votre disposition dans notre boutique ainsi que notre principe de "Commercialisation" de logiciel...

IDC a tenu, dès ses débuts, à proposer à ses membres une formule originale de distribution du software. Les consommateurs de logiciels commerciaux subissent automatiquement 50% des bénéfices des ventes réalisés sur leur programme... ce qui n'est pas négligeable... Avec cette nouvelle année 1986 nous nous sommes mis encore une fois en quatre (!), le nombre de lecteurs de la revue à augmenté, vu notre association avec Dalnamic. Vos logiciels seront donc proposés à bien plus de personnes... et les bénéfices devront s'en suivre...

Tout au long de notre première année d'activité, nous avons déjà eu l'occasion de vous proposer plusieurs logiciels différents réalisés par certains d'entre vous...

En voici une liste...

Utilitaires :

DAITEXT V.21 : traitement de texte puissant avec partie télématique qui permet le branchement sur réseau et la mise en mémoire des communications...(compatible CAS/DCR/KENDOS) 1800 FB.

EDITEUR DE DESSINS : Un superbe programme de traitement d'images en mode 6 qui fonctionne selon le principe du célèbre MACPAINT d'Apple. Vous avez entre autres accès à plusieurs fonctions (cercle, point, ravon.... ainsi qu'à une table de symboles entièrement redéfinissable... Ce programme fonctionne également à l'aide de sprites... par exemple, vous pouvez déplacer partout à l'écran avant de le placer à un endroit précis sans modifier l'image de départ. Ce programme possède également un puissant mode d'édition qui vous permet de corriger les différentes fautes que vous aurez pu faire... Le programme retient en fait une table d'ordre (c-à-d toutes les commandes que vous avez faites pour faire votre dessin) et cette table d'ordre est modifiable... (insertion/suppression.). Il est également possible d'enregistrer cette table d'ordre sur cassette, ce qui fait que le fichier cassette est nettement plus court qu'une copie d'écran normale...) (CAS/DCR/KENDOS) 1300 FB.

MAILING LIST : un questionnaire de fichiers d'adresses classique. Retient environ 240 fiches en mémoire en 1 fois, mais possède aussi deux autres caractéristiques alléchantes : le fichier est sauvé sur un seul fichier cassette de longueur toujours égale. Vous pouvez donc ajouter des fiches à votre fichier sans prendre plus de place sur la cassette... et de ce fait sans effacer le programme suivant. La deuxième caractéristique alléchante de ce programme est son fonctionnement par menus...vous passez sans

PRESENTATION DU CLUB D.A.I.C.

Le club D.A.I.C. (Digital Applications International Club) est une association d'utilisateurs de micro-ordinateurs personnels. Ces micros étaient uniquement des DAI au départ. Aujourd'hui des utilisateurs de PCs - MS-DOS - sont venus se joindre au groupe existant et forment d'ailleurs la majorité des membres.

Le club est avant tout un lieu de rencontres. Les réunions se déroulent dans les vastes locaux du Cercle Eratosthène au 39, rue Montoyer à 1040 Bruxelles (513.98.89) tous les 1ers dimanches du mois de 14 heures à 18 heures et tous les 3es jeudis du mois de 18 heures à 22 heures.

Divers micros sont mis à la disposition des membres ainsi que des imprimantes. Diverses aides peuvent également être obtenues, aussi bien en matière de hardware que de software. La meilleure façon de se rendre compte de son fonctionnement est finalement d'y venir. Alors, si vous en avez la possibilité, n'hésitez pas. Venez nous voir!

Il y a encore de la place !

DAIC

REPONSE au JEU-CONCOURS du DAICLIC No 4

Le jeu-concours est déjà devenu une tradition dans la revue DAICLIC. Nul doute que l'augmentation du nombre des lecteurs amènera une nouvelle augmentation du nombre des participants.

Mais tout d'abord, voici la réponse au jeu-concours paru dans le DAICLIC no 4. Le tas de billes contenait respectivement 16 billes vertes, 49 rouges, 100 bleues, 324 blanches et 576 noires. Le programme le plus court a été envoyé par Michel BILLOT de Vitry/Seine; le voici :

```
INP INT
1 FOR D=3 TO 32:FOR E=D TO 32:IF E*E-D*D<>232 THEN NEXT:NEXT
2 FOR A=2 TO D:FOR B=A TO D:FOR C=B TO D:IF C*C+B*B+A*A=E*E
<>1065 THEN NEXT:NEXT:NEXT
```

Après exécution du programme, il suffit de faire :
PRINT A*A;B*B;C*C;D*D;E*E pour obtenir la réponse.

Ce programme occupe 119 octets en mémoire centrale.

NOUVEAU JEU-CONCOURS

Voici maintenant le petit problème d'aujourd'hui.

Les cryptarithmes, vous connaissez ? Chaque lettre représente chaque fois le même chiffre; et réciproquement, chaque chiffre donné est toujours représenté par la même lettre. Aucun nombre ne commence par 0.

Sachant ceci, voici l'énigme : TACLEF divise LEFTAC, ou, si vous préférez, LEFTAC est un multiple de TACLEF.

Combien vaut ELLE ?

Le programme n'est pas difficile, mais il vous faudra sans doute de la patience avant de découvrir la réponse sur votre écran. Vous voudrez donc bien ajouter votre temps obtenu afin de m'éviter de devoir exécuter tous les programmes reçus.

Vos réponses doivent être envoyées à Jacques MOENS, Clos Fontaine des Ducs, 6 à 1310 La Hulpe (Belgique). Le programme trouvant le plus rapidement la bonne réponse vaudra à son auteur de recevoir une cassette bourrée de programmes. Bon courage !

La folie du modem : act 2

Chers amis.

Je vous avais promis dans le dernier numéro de DAIclic de vous parler aujourd'hui de la communication par modem par l'intermédiaire du réseau **DCS**. C'est donc ce que je vais faire, après vous avoir donné les **NOUVEAUX** numéros d'appel des serveurs du club **MICRO-GDV** en Belgique. Je vous rappelle que ces serveurs sont à accès **GRATUIT... :**

MN2 = Micronet 2, serveur du club Micro-GDV.

MN2 Bruxelles A :	**02/242.70.08**	MN2 Bruxelles B :	**02/513.11.11**
MN2 Liège A :	**041/79.66.66**	MN2 Liège B :	**041/31.21.31**
MN2 Verviers A :	**087/88.34.34**	MN2 Verviers B :	**087/88.31.31**

Ces serveurs sont donc a votre disposition **24h/24** !!!

Vous trouverez l'index **IDC** dans "l'arbre" des serveurs MN2 Bruxelles A & B, Liège A et Verviers B...

N'hésitez pas non plus à nous contacter sur ces machines qui maintenant sont interconnectées... c-à-d que les différents systèmes s'appellent entre-eux pendant la nuit pour échanger leur courrier... Si vous décidez de ne vous brancher que sur Bruxelles A, par exemple, votre courrier vous sera automatiquement transmis sur ce serveur...sur simple demande à l'opérateur du système de Verviers A... (son pseudonyme est JPU). (dispatching).

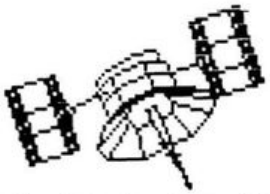
Nous vous rappelons également que vous pouvez contacter l'équipe d'**IDC** sur ce réseau...nos pseudonymes sont toujours les suivants:

CHRIS (Christian Poels)
STARSKY (Fabrice Duluins)
HUTCH (Marc Vandermeersch)

Bonne nouvelle également pour nos amis français : le **M-GDV** parle également d'ouvrir un serveur dans la zone de **Paris**... Nous ne savons pas encore si ce serveur va etre interconnecté par ce qui est appelé le "soft'nocturne" avec les serveurs belges, mais cela serait très pratique pour les relations dans le club !

Après cette longue introduction, passons dans le vif du sujet, voici...

DATA
COMMUNICATION
SERVICE



En plus du réseau téléphonique et du télex, le **Data Communication Service (DCS)** constitue une réseau spécialisé pour la transmission de données entre équipements de traitement de l'information. Le **DCS**, réseau public belge de transmission de données avec commutation par paquets, est relié à **Euronet** et à des réseaux similaires d'autres administrations des P.T.T, notamment la France, le Royaume-Uni, la République Fédérale d'Allemagne, l'Espagne, le Portugal, la Suède, les Etats-unis d'Amérique. le Canada, le Japon et Singapour. D'autres liaisons internationales seront mises progressivement en service.

La commutation par paquets : un grand pas en avant !...mais que signifie la "commutation par paquet ?"

Le réseau de paquets se compose de centraux (commutateurs) auxquels sont reliés des terminaux et des ordinateurs. Ces centraux sont reliés entre-eux par des circuits de transmission à vitesse élevée.

Les données sont introduites dans le réseau sous forme de paquets. Chaque paquet se compose au maximum de 128 caractères et est complété par une information permettant son acheminement entre l'expéditeur et le destinataire.

Dès l'envoi d'un paquet, la liaison est de nouveau disponible pour un autre paquet qui peut provenir d'autres utilisateurs. De cette facon, il est possible d'établir simultanément plusieurs liaisons sur une meme ligne.

Le système belge répond aux normes internationales (avis du X.25 du C.C.I.T.T). de sorte que les échanges de données entre ordinateurs et terminaux soient possibles sur le plan international.

L'application de la technique de commutation par paquets augmente et améliore le rendement de ligne.

La conversion de vitesses lors de communications entre systèmes a vitesse de traitement différentes s'effectue par le réseau.

Les liaisons sont de qualité supérieure. La correction automatique des fautes est assurée sur chaque liaison.

Il existe un haut degré de disponibilité du réseau grace au dédoublement de l'appareillage de commutation et des voies de transmission.

A qui ce réseau est-il destiné ?

Ce réseau est destiné au départ pour les entreprises et institutions qui ont besoin de communiquer avec différentes banques de données internationales. avec en meme temps une haute sécurité de liaison...

Ce réseau me semble aussi destiné aux particuliers car, bien qu'assez couteux, il est tout de meme agréable pour nous, belges, de pouvoir profiter, par exemple, du réseau français TRANSPAC et de toutes les machines qui s'y connectent.

Moins drole maintenant...les tarifications : elles sont de trois types différents. mais nous ne nous occuperons ici que d'un seul. Il y a 3 manières différentes de se brancher sur **DCS**... Vous avez soit une ligne a accès direct... ce qui est fort cher mais qui permet de travailler à toute haute vitesse... inutile de vous dire que ce genre de liaison est réservée aux sociétés..., soit un accès au réseau via le réseau du télex... mais de nouveau réservé quasiment aux sociétés...car qui possède un télex... et enfin une troisième facon de se connecter : via une ligne téléphonique normale à des vitesses de 300 ou 1200 bauds... ce qui sera le plus souvent le cas pour nous... Voyons les prix de ce troisième type de liaison :

Tout d'abord, il faut se faire enregistrer sur le réseau...(et de ce fait recevoir un mot de passe...). Les frais de dossier s'élèvent a 1000 Fb. Ensuite, il faut également payer la somme de 450 FB par bimestre (donc a chaque facture téléphonique). Ca, ce sont les sommes fixes... Après cela, si vous etes possesseur d'un modem à couplage acoustique, le prix de la communication zonale ou interzonale est également à payer. Si vous etes l'heureux possesseur d'un modem à couplage direct et que ce dernier est **AGREE** (donc. un modem 'cher'...), vous ne devrez pas payer ce supplément... Une taxe de 0,15 fb sera également percue lors de l'établissement de l'appel (et meme si l'appel n'a pour finir pas lieu...). 0.10 fb toutes les 30 secondes pour la Belgique (1.20 fb pour TRANSPAC) et enfin, 0.80 fb tout les 10 segments pour la Belgique et 1.20 fb pour TRANSPAC... Un paquet est compté pour 1 ou 2 segment suivant qu'il contient 64 octets ou plus...(ces prix varient selon le pays que vous appellez...comme vous pouvez le constater...)

Comme vous le voyez, ce n'est tout de meme pas tellement donné ! De plus, il ne faut pas oublier que la plupart des serveurs que vous appellerez avec ce réseau sont payants !!! Certaines banques de données sont cependant gratuites...par exemple celle de la CEE qui se trouve à Luxembourg... Un autre reproche que je fais au système, mais qui est tout-a-fait normal, est l'impossibilité pour un particulier d'appeller directement un autre particulier. Cela serait très agréable, par exemple, pour les contacts inter-clubs... Il est à noter que toutes les communications via **DCS** apparaissent, parait-il, vraiment très clairement dans votre note téléphonique... descriptions précises des temps,...

D'un autre cote évidemment, la porte de serveurs fous des USA nous est grande ouverte pour pas trop cher... Vous pouvez grace à DCS vous brancher sur des serveurs tels que **COMPUSERVE** et **THE SOURCE** aux Etats-Unis pour une somme qui reste tout de meme acceptable...

Encore une fois à vous de savoir ce que vous désirez faire... pour des renseignements plus précis au sujet de **DCS**, vous pouvez toujours écrire à :

Régie des Télégraphes et des Téléphones.
Département Transmission de données.
Boulevard de l'Imperatrice 17-19
B - 1000 Bruxelles.

Vous recevrez pas retour du courrier une brochure explicative des possibilités de **DCS**, type de connections, type de commandes...

Cette meme brochure existe également en Néerlandais, nos amis de DAInamic peuvent donc aussi écrire à l'adresse précitée...

528 points par ligne

```
PAGE 01  --

1       REM RESOLUTION 528 POINTS PAR LIGNE
2       REM METTRE LIGNE EN MODE GRAPHIQUE (16 COULEURS)
3       REM LIGNE 2
4       POKE #B4F1,#BA
5       REM
6       REM METTRE A CHAQUE POSITION DE LA LIGNE LA COULEUR (F)
7       REM ET UN DOT (00)
8       FOR X!=#B4F0-132 TO #B4F0-2 STEP 2
9       POKE X!,F!
10      POKE X!+1,0
11      REM INITIALISER LE BIT DE LA COULEUR DE TEXTE POUT LA BOUCLE
12      A!=1
13      REM DECALER LE DOT 8# A GAUCHE PAR OCTET
14      FOR Z!=1 TO 8
15      REM METTRE NOUVEAU POINT
16      POKE X!+1,A!
17      AALT!=A!:A!=A! SHL 1:A!=AALT! IOR A!
18      REM METTRE LA POSITION ACTUELLE DU POINT
19      L!=L!+1
20      CURSOR 1,5:PRINT L!
21      NEXT Z!
22      REM CHANGEMENT DE LA COULEUR
23      F!=F!+16
24      IF F!>#F0 THEN F!=0
25      REM NOUVELLE POSITION
26      NEXT X!
```


Nouveautés Xbasic

NOUVEAUTES XBASIC

(Article original : Heinrich Tegetoff - DAInamic Germany)
(Traduction : Frédéric BACQUET)

L'intérêt porté à XBASIC est beaucoup plus important qu'escompté. Ceux qui ont jusqu'à maintenant utilisé la version maximale de XBASIC, soit 32Ko de langage machine, sont totalement satisfaits. Dans l'article du dernier bulletin du club (allemand), certains ont éprouvé - semble-t-il - quelques difficultés de compréhension. Voici donc quelques précisions :

- L'interface Commodore fait partie intégrante de la version de base. elle est normalement conforme à l'interface qui est vendue avec ComDOS v2.3 de Mr Koldehoff. Le DOS, ComDOS 3.0, est compatible : c'est une version améliorée. L'interface v3.0 se distingue par le fait qu'elle est plus ou moins un sous-produit de la carte X-Bus qui permet la connection avec le drive Commodore ; le DCE-Bus n'est donc pas utilisé. Dans ComDOS v3.0, il y a en plus du Quickload un Quicksave, qui permet une sauvegarde très rapide de programmes ou autres fichier, à env. 14000 bauds. Pour comparaison : cassette 600, MDCR 6000, Indata 80Ko 6000, KenDOS 50000. Avec le ComDOS, sont fournis sur la disquette "DC Disk Copy","DA Disk Analyse", "SE Sector Edit" et "Lock/Unlock". Le connecteur est bien évidemment inclus.

-Comme il y a également de très nombreux possesseurs de MDCR qui désirent tout de même utiliser XBASIC - la carte du TOS n'étant plus compatible avec le X-Bus -, il exite maintenant un XBASIC-TOS ! Bien sûr, tous les ordres du TOS s'emploient comme des commandes basic normales :
exemple : 10 DCR :REWIND NF+1:PRINT "Terminé"
Les autres améliorations sont : VERIFY avec un nombre de fichiers (VERIFY multiple), LOOK en marche arrière depuis le debut de la cassette, DELETE en marche arrière, avertissement lorsqu'on essaie d'utiliser SAVE, DELETE, LAST ou DELREW sur une cassette protégée, et "EXEC" : appel d'un MLP. Avec l'ordre "FIND+E C CALLM", les intructions du TOS peuvent etre facilement modifiées. Le TOS est gratuit dans la mesure ou il tient dans 32Ko, dans le cas où, en plus de la version de base, des programmes comme SPU/SGT sont inclus dans l'eprom, mais pas l'Eprommer (gratuit). Sinon, il faudrait 34Ko et payer un supplément de 13 DM. Pour pouvoir utiliser XBASIC-TOS, il faut connaître le TOS normal, ce qui ne doit pas etre un gros problème.

Il n'est pas exclu qu'il y ait eu d'autres versions de XBASIC, entre la rédaction et la publication de cet article, qui ne correspondent plus à ce descriptif. Je pense par exemple à un disque virtuel de 64Ko sur EPROM. Toute modification ou amélioration démandée sera étudiée. Encore une remarque, qui s'adresse au désarroi de certains : BASIC v1.2 n'est pas nécessaire à l'utilisation de XBASIC. XBASIC est également compatible avec les version v1.0 et v1.1.

version de base	: 205 DM	Eprommer	: gratuit
SPU	: +36 DM	TED-tools	: + 8 DM
XBASIC-SGT	: +58 DM	MDCR-TOS v2.1	: gratuit
avec Basic v1.2	: +88 DM		

Version 32Ko sans Basic v1.2 : 296 DM ; avec Basic v1.2 : 278 DM

Renseignements : écrire à :
Heinrich Tegethoff
In der Provitze 87
4630 Bochum
Allemagne

Dossier assembleur: partie 4

MICROPROCESSEUR PART. 4

(HP. LEGRY, DOUAI (F))

ASSEMBLAGE ET PSEUDO-INSTRUCTIONS

OUVRAGES UTILISES : SPL FRANCAIS DOCUMENTATION DAInamic.
; PROGRAMMATION EN ASSEMBLEUR Lance A Leventhal.

SIGNIFICATION DES INSTRUCTIONS

Le jeu d'instructions d'un microprocesseur est le jeu des entrées binaires qui produisent des actions définies pendant un cycle d'instruction.

Instructions binaires

Une instruction est une configuration binaire. Elle doit etre disponible à l'entrée des données du microprocesseur à un temps précis afin d'être interpretée comme une instruction. Par exemple, lorsque le microprocesseur reçoit le mot binaire 1000 0000 en entrée, au cours d'une opération de recherche mémoire, cela signifiera: additionner le contenu du registre B au contenu de l'accumulateur.

De même la configuration 0011 1110 siginifie : Charger l'accumulateur avec le contenu du mot suivant, de la mémoire de programme.

Le microprocesseur ne reconnait dans les configurations binaires que des instructions ou des données; il ne distingue ni les mots, ni les nombres, que ce soit en octal, décimal ou hexadécimal.

LE PROGRAMME INFORMATIQUE

PROGRAMME INFORMATIQUE

En fait, un programme inclu, outre des instructions, des données et des adresses en mémoire utilisées par le microprocesseur pour accomplir les tâches définies par les instructions. Exemple : le microprocesseur doit faire une addition, il doit disposer de deux nombres et d'un endroit ou il placera le résultat. Le programme informatique doit déterminer les sources des données et la destination du résultat, qui s'ajoute au type d'opération à exécuter.

La plupart des microprocesseurs exécutent les instructions séquentiellement, à moins qu'une des instructions ne modifie la séquence ou n'arrête le microprocesseur. Ainsi le microprocesseur acquiert toujours l'instruction suivante sauf si l'instruction en cours en décide autrement.

Exemple: un programme qui commandera au 8080 d'additionner le contenu des positions mémoires #60 et #61 et ranger le résultat en #62

```
00111010
01100000
00000000
01000111
00111010
01100001
00000000
10000000
00110010
01100010
00000000
```

Cette liste est un PROGRAMME OBJET ou PROGRAMME EN LANGAGE MACHINE. Si un tel programme était entré dans la mémoire d'un ordinateur construit avec un 8080, le microprocesseur serait capable de l'exécuter directement.

Problèmes de programmation

Le programme est difficile à comprendre,à mettre au point.
L'introduction du programme est longue.
Le programme ne décrit pas la tache à accomplir.
Programme fastidieux et long à écrire.
Les erreurs sont difficiles à détecter.

Utilisation de l'hexadécimal

La situation s'améliore sensiblement si, au lieu d'écrire en binaire, on rédige les instructions en hexadécimal. Le programme d'addition du 8080 devient alors:

3A 60 00 47 3A 61 00 80 32 62 00

Mnémoniques des codes d'instructions

Un progrès évident consiste à affecter un nom à chaque code d'instruction. Ce nom est appelé mnémonique. Il doit évoquer ce que l'instruction est censée faire.

Ils sont standards pour un microprocesseur donné et de ce fait compris par tous les utilisateurs. Lors de la sélection des codes mnémoniques, le problème consiste à trouver des noms évidents. C'est le cas pour quelques instructions (ADD, AND, OR). Si nous utilisons les instructions standards du 8080 et les mnémoniques des registres tels qu'ils ont été définis par INTEL, notre programme d'addition devient :

MNEMONIQUES	HEXADECIMAL	BINAIRE
LDA	3A	00111010
60	60	01100000
00	00	00000000
MOV B,A	47	01000111
LDA	3A	00111010
61	61	01100001
00	00	00000000
ADD B	80	10000000
STA	32	00110010
62	62	01100010
00	00	00000000

Ce programme est loin d'être évident mais il est compréhensible . ADD B est un progrès considérable sur #80. Un tel programme est un programme en langage ASSEMBLEUR.

Le programme assembleur

L'assemblage traduit les mnémoniques du programme assembleur en nombres binaires. Le programme qui exécute cette fonction est L'ASSEMBLEUR. Ce programme ASSEMBLEUR traduit le programme utilisateur, ou programme SOURCE, rédigé à l'aide de mnémoniques, en un programme en langage machine ou programme OBJET que le micro ordinateur peut exécuter.

Les assembleurs disposent de leurs propres règles. Elles comprennent l'utilisation de certains marqueurs (espaces, virgules, point-virgule, deux points ...) placés à bon escient.

STRUCTURE D'UNE LIGNE

Une ligne comporte 4 colonnes. Les colonnes LABEL, INSTRUCTIONS, OPERANDES et COMMENTAIRES.

1 - LABEL

Si vous utilisez cette colonne, en y plaçant un nom (label), le SPL attribuera à ce nom la valeur correspondante à l'adresse ou il est placé. Très utilisé pour les JMP et les CALL.

2 - INSTRUCTION

C'est la colonne dans laquelle vous placez vos instructions (STA, LDA, MOV A, M...).

3 - OPERANDE

Dans cette colonne, sont placés les nombres utilisés par l'instruction qui les précède. ex STA 5000H : 5000H sera placé dans cette colonne. Les labels correspondent à un nombre ou une adresse. Ils seront donc considérés comme des nombres.

4 - COMMENTAIRE

Un programme assembleur bien fait doit comporter le maximum de commentaires nécessaires à la compréhension du programme par un utilisateur autre que l'auteur.

LABEL	INSTRUCTIONS	OPERANDES	COMMENTAIRES
DEBUT	LDA	VAL1	CHARGE 1er NOMBRE DANS A
	MOV B,A		TRANSFERE A DANS B
	LDA	VAL2	CHARGE 2e NOMBRE DANS A
	ADD B		AJOUTE B a A
	STA	SOMME	RANGE LE RESULTAT
SUITE	.	.	
.	.	.	
.	.	.	
VAL1	EQU	60H	VAL1=#60
VAL2	EQU	61H	VAL2=#61
SOMME	EQU	62H	SOMME=#62

Attention ! (ADD B) est une instruction et ici B n'est pas un opérande.

LES PSEUDO-INSTRUCTIONS

Certaines instructions de l'assembleur ne sont pas directement converties en instructions en langage machine. Il s'agit de directives pour l'assembleur. Elles attribuent certaines zones en mémoire au programme, définissant des symbôles, désignant les portions de la RAM qui recevront des données transitoires, des tableaux ou d'autres informations et exécutent des fonctions d'intendance.

Pour utiliser ces directives,ou PSEUDO-OPERATIONS, le programmeur place leurs mnémoniques dans le champ du code opération et une adresse ou une donnée dans le champ opérande si la pseudo-opération le requiert.

Pointeur objet: Ce pointeur contient l'adresse mémoire à laquelle le logiciel assemble l'instruction en assembleur.

Les pseudo-opérations courantes (SPL) sont :

* DB (donnée)
* DS
* DW
* EQU (equate,équivalence)
* SET
* ORG (origine)
* END (fin)
* LST (list)
* UNL (unlist)
* ELSE
* MEND
* MACRO
* IF
* TITLE

1 - Origine

La pseudo opération origine permet au programmeur d'assembler ses programmes, sous-programmes ou données n'importe ou en mémoire.

Les programmes et les données peuvent etre rangés dans diverses zones de la mémoire.

Exemple : ORG 300H

Signifie que le programme source (assembleur) sera assemblé à partir de l'adresse 300H.

Plusieurs ORG peuvent etre utilisés dans un programme source. Chaque fois que l'assembleur rencontrera une telle instuction, il continuera à assembler à partir de l'adresse suivant le ORG.

```
Exemple : ORG   300H
          ORG   400H
```

Le pointeur objet est placé à la valeur qui suit le ORG.

2 - Les réservations

Instructions DB,DW

Format : LABEL (DB,DW,DS) OPERANDE

Ces instructions placeront le nombre (OPERANDE) dans la prochaine cellule mémoire et attribuera à cette position le label "LABEL".

DB réserve 8 bits
DW réserve 16 bits

Exemples :

```
      assembleur                    machine
      ----------                    -------
                                  adresse   contenu
        ORG   400H
NOM     DB     15H                  400     15

        ORG   400H
NOM     DW    1234H                 400     34 23

        ORG   400H
VAL     DB    12,34,56              400     12 34 56
```

Si une chaine de caractères suit le DB, elle sera stockée sous forme de caractère ASCII.

```
CHAI    DB    'ARTHUR'              400     41 52 54 48
                                    404     55 52
```

DS data augmente la valeur du pointeur objet de la valeur data .


```
        ORG   300H
NOM     DB    10H                   300  10
BUF     DW    1122H                 301  22 11
        DW    3344H                 303  44 33
RES     DB    0F4H,0D5H             305  F4 D5
CHAIN   DB    'ARTHUR'              307  41 52 54 48 55 52
        DS    5                     30D  00  00  00  00  00  00
ADR     DW    6887H                 313  87 68
```

VOUS AVEZ SANS DOUTE REMARQUE QUE L'ASSEMBLEUR STOCKE LES 16 BITS SUIVANT L'INSTRUCTION DW, LES 8 BITS DE POIDS FAIBLES D'ABORD, PUIS LES 8 BITS DE POIDS FORT.

3 - Définition

La pseudo opération EQUATE permet au programmeur d'attribuer des noms à des adresses ou à des données.

EQU attribue au label qui la précède l'opérande qui la suit

```
Exemple :  FIN    EQU   1000H
           DEPA   EQU   400H
           SUI    EQU   DEPA+500H
```

Chaque fois que le SPL rencontrera le label FIN lors de la phase d'assemblage, il attribuera à ce label la valeur 1000H. Cette définition est définitive. En aucun cas FIN ne devra être défini par un autre EQU.

SUI est défini en fonction de DEPA. Le label DEPA doit être défini auparavant. Le SPL attribuera à SUI la valeur DEPA+500H soit 400H + 500H = 900H.

La pseudo opération SET travaille de la même manière que EQU à la différence que le label peut être défini plusieurs fois dans un programme. Le label prendra la dernière valeur qui lui est attibuée, au cours de l'assemblage.

```
Exemple : SUI   SET   400H      SUI=400H
          SUI   SET   500H      SUI=500H
```

4 - FIN

La directive END indique au SPL la fin du programme en assembleur. Elle est obligatoire lors de l'assemblage.

```
      ORG     400H
      STA     2000H
      END
```

5 - TITLE

Dans un hardcopy, le SPL imprimera en début de chaque page à partir de la colonne label, la chaine de caractère se trouvant dans la colonne opérande du dernier title.

6 - UNL-LST

UNL suspend le listing objet jusqu'au moment ou le SPL rencontre la commande LST.

7 - ASSEMBLAGE CONDITIONNEL

Les instructions IF, ENDIF, ELSE permettent un assemblage plus souple.

```
    IF   COND        Si la condition CONd est vraie, alors
      .              lors de l'assemblage, les instructions
      .              entre IF et ENDIF seront incluses dans
    ENDIF            le programmme, sinon elles seront
                     ignorées.
```

ELSE détermine la séparation entre ce qui doit être assemblé si la condition est vraie et ce qui doit être assemblé si la condition est fausse.

```
Exemple        IF     VAL=0
                 .        1ERE PARTIE DU PROGRAMME
                 .
                 .
               ELSE
                 .        2EME PARTIE
                 .
                 .
               ENDIF
```

Si VAL=0 alors la première partie sera assemblée, sinon ce sera la deuxième partie.

8 - les MACRO-INSTRUCTIONS

On remarquera souvent que certaines séquences d'instructions reviennent plusieurs fois dans un programme source. Ces séquences peuvent refléter les besoins logiques de votre programme, ou encore viennent compenser les défficiences du jeu d'instructions de votre microprocesseur. Vous pouvez éviter d'avoir à les réecrire à chaque fois en utilisant une macro.

Les macros vous permettent d'affecter un nom à une séquence d'instructions. L'assembleur remplacera ce nom,si vous l'utilisez, par ladite séquence d'instructions.

```
                                        résultat

SEQU    MACRO
        instruction M1
        instruction M2
        instruction M3
        MEND

        instruction P1          instruction   P1
        instruction P2                        P2
        instruction P3                        P3
```



```
        SEQU (: appel de la macro)      M1
                                        M2
                                        M3

        instruction P4                  P4
        instruction P5                  P5
        instruction P6                  P6

        SEQU (: appel de la macro)      M1
                                        M2
                                        M3

        instruction P7                  P7
        instruction P8                  P8
        instruction P9                  P9

        END                             END
```

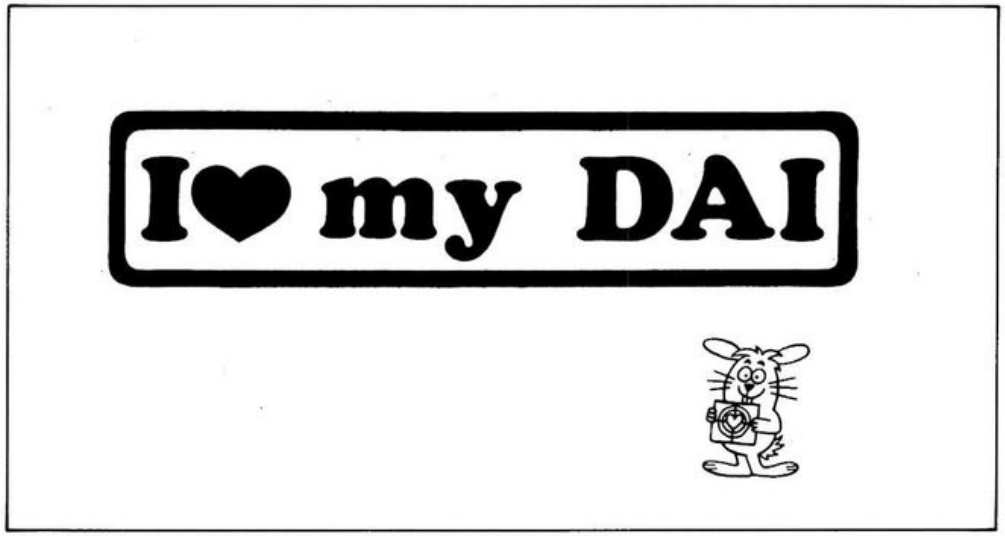
Une macro peut nécessiter l'emploi de variables. Il faut alors faire suivre MACRO par le nom des variables utilisées

```
SEQU    MACRO     VALEUR
        MVI A     VALEUR
        STA       5000H
        MEND
```

Appel de la macro : SEQU 15H

VALEUR prendra la valeur 15H lors de l'appel de SEQU

Ceci conclut le cinquième chapitre concernant le SPL et les pseudo-instructions. Dans l'article suivant, nous examinerons plus en détail l'utilisation du SPL et ses commandes.



Les maux du DAI : le clavier 2

Les MAUX du DAI par I.D.C. Bordeaux.

Série : Le clavier 2

Plusieurs personnes se plaignent, à juste raison , de la mauvaise qualité des anciens claviers du DAI. Au bout de quelques temps, ce ne sont que rebonds ou touches 'dures d'oreille'. Ayant eu à réparer plusieurs claviers, dont certains par correspondance je dois vous prévenir que cette opération demande patience et douceur. Elle n'est pas sans risques car tout faux mouvement lors du démontage mécanique peut vous faire casser une partie de la touche. Si cela vous arrivait sur une ou deux touches, c'est loin d'etre dramatique comme nous le verrons plus loin .

De ceci il découle que cette intervention est loin d'etre bénigne et ne sera indiquée que pour les claviers qui en ont vraiment besoin. Aussi le club décline toute responsabilité quant aux accidents qui pourraient résulter du mauvais emploi de ces indications effectuées hors de son atelier

D'abord mon expérience personnelle se doit de vous mettre en garde contre les deux plus grands ennemis (à ma connaissance) du clavier :

* Les produits 'améliorant' les contacts, qui agissent en enlevant l'oxydation des parties conductrices. Malheureusement notre problème ne vient pas en général d'une oxydation des contacts, mais d'un encrassement par la poussière et ces produits viennent coller la poussière et faire empirer le mal...

* Le bol de chocolat au lait, dont l'épaisse texture vient s'immiscer dans tous les interstices du clavier, meme à l'intérieur des superbes touches des nouveaux claviers dont je vous décrirai le démontage plus tard. Je ne ferai aucun commentaire sur l'ignoble procédé de dopage de sa machine par le chocolat au lait... N'est ce pas Sébastien ?

Vont suivre donc, deux articles sur le démontage des deux types de claviers que je connais. Si quelqu'un a eu affaire à un autre type de clavier que ceux indiqués il serait très intéressant pour tous qu'il vienne grossir la documentation sur ce sujet :

Envoyer vos réflexions au président du Club I.D.C.Bordeaux.

Mr Delannay Bruno
Res. Les Acacias Bt. B3
Av. de Saige
33 600 Pessac (France)

Comme pour le Testament vous aurez droit d'abord à l' Ancien puis au Nouveau

A suivre.

Delannay Bruno (IDC BORDEAUX)

Principe de l'interface centronics

PRINCIPE DE L'INTERFACE CENTRONICS

SEPTEMPRE 85

D.A.I.C.L.I.C

HP & L LEGRY
628 Bd LAHURE
59500 DOUAI, FRANCE

Centronics est la norme généralement utilisée par les imprimantes . Nous n'allons pas étudier cette norme dans le détail. Seuls les signaux utiles à la carte vous seront présentés. Les manuels d'utilisation fournis avec les imprimantes vous donnent en général le brochage complet du connecteur.

A LA NORME CENTRONICS

1 - LES SIGNAUX

La carte utilise les signaux suivants : les datas, le strobe et le busy.

a - Les datas

Les données (caractères ASCII) transitent, de l'ordinateur à l'imprimante, sur ce bus (8 bits).

b - Le busy

Ce signal provient de l'imprimante. Il indique si celle-ci est en mesure de recevoir un caractère.

c - Le strobe

L'ordinateur signale, par une impulsion sur cette ligne, qu'une donnée valide est présente sur les datas.

2 - FONCTIONNNEMENT D'UNE TRANSMISSION

a - L'imprimante est prête à recevoir les données. Le busy est à 0. Le strobe est à 1.

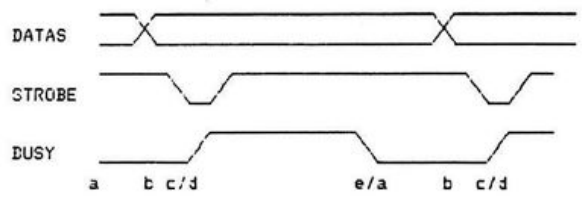
b - L'ordinateur place le caractère (8 bits) sur les datas.

c - Un signal, le strobe, est transmis à l'imprimante. Il signale à celle-ci qu'un caractère valide est placé sur les datas.

d - Le busy passe aussitôt à 1. L'imprimante est occupée à traiter le caractère. Le DAI ne doit alors plus en envoyer. L'imprimante ne pourrait pas le traiter immédiatement.
Le busy, coté imprimante, est connecté au DTR, coté DAI. Le DTR existe déjà sur la prise RS232. La routine d'affichage attendra, pour envoyer un nouveau caractère sur les datas, que l'imprimante soit prête (busy à 0).

e - Dès que l'imprimante a traité le caractère, le busy repasse à 0. Elle est de nouveau prête à recevoir.

3 - CHRONOGRAMMES



B L'INTERFACE CENTRONICS

Comme vous le savez votre DAI est equipé d'une sortie série de type RS 232. Elle est gérée par un circuit :le 5501. Il n'est pas utile d'en connaitre le fontionnement. Il faut simplement savoir que l'envoi d'un caractère sur la RS 232 s'effectue en chargeant le buffer d'émission du 5501 avec le code ASCII du caractère à transmettre. Cependant, certaines précautions doivent être prises :

- il faut vérifier que le recepteur est prêt: état haut sur la borne DTR de la RS 232.
- Et aussi que le caractère précédent a été émis par le 5501 : test d'un des registres.

L'interface fonctionne en parallèle avec le 5501. L'avantage de cette formule est évident : TOUS les logiciels conçus pour le DAI et utilisant une imprimante pourront fontionner avec cette interface sans AUCUNE modification.

Le fontionnement de la carte est assez simple à comprendre lorsqu'on connait la norme CENTRONICS :

Le caractères à transmettre est écrit à l'adresse du buffer d'émission du 5501, c'est à dire en FFX6 (le X signifiant que les bits A4 à A7 de l'adresse peuvent prendre n'importe quelle valeur). Les DATAS de la CENTRONICS seront obtenus en recopiant le bus de données sur la sortie d'un latch (qui le mémorise) lorsque FFX6 est présent sur le bus d'adresse.

L'interface fonctionnant en parallèle sur la RS 232, le BUSY de l'imprimante sera tout simplement inversé et envoyé sur le DTR de la sortie série. Ainsi, lorsque l'imprimante ne sera pas prête à recevoir, le BUSY sera à l'état haut donc le DTR à l'état bas. Aucun caractère ne sera envoyé par l'ordinateur.

Le dernier signal à gérer est le STROBE. Une impulsion de quelques microsecondes est envoyé à l'aide d'un monostable dès qu'un caractère a été recopié sur la sortie du latch.

Le schéma de la carte (figure 2) ne comporte que 4 circuits intégrés :

a - Le decodeur 74LS138

Un état bas sera présent sur la sortie O6 lorsque :

- FF=0 (décodage de FF)
- A0=0,A1=1,A2=1,A3=0 (décodage de X6)
- MW=1 (memory write)

REMARQUES - FF est présent en sur la patte No 4 de IC 45 (voir figure 3 ou PC SCHEMATICS)
- A0,A1,A2,A3 sont les bits de poids faible du bus d'adresse. On peut les obtenir sur le XBUS de même que MW.

Pour plus de détails reportez vous a la figure 3

b - le latch 74LS373

La sortie O6 du 74LS138 (inversée) commande la recopie de l'entrée du latch sur sa sortie.
Output Enable (OE) sera maintenue à 0 pour valider la sortie (lorsque OE=1, la sortie est en haute impédance).

c - le monostable 74LS123

Sur le front descendant de O6 le monostable envoie une impulsion (le STROBE) de quelques microsecondes : le minimum imposé par la norme CENTRONICS ést 1 microseconde.

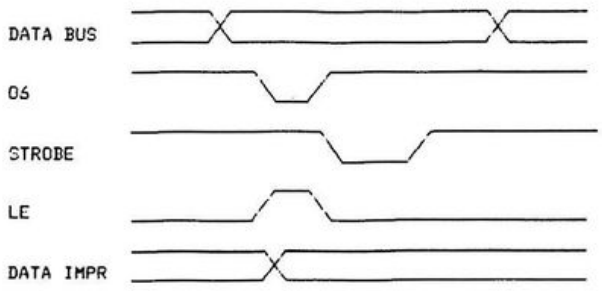
d - le transistor 2N 3904

La gestion du BUSY réclame quelques explications. Le signal DTR arrive sur la patte No 14 de IC 31 (voir figure 3).
Juste avant cette patte il faut couper la piste. Ceci réalisé, DTR1 correspond à la piste venant du connecteur RS 232 et DTR2 à celle reliée à IC 31. Comme vous l'avez sans doute compris, cette modification permet une utilisation simultanée d'imprimantes sur la RS 232 et la CENTRONICS). Si cela ne vous est pas utile, vous pouvez envoyer le BUSY inversé sur la broche DTR du connecteur SUB-D (CANNON,RS 232).
Le transistor (2N 3904 ou équivalent) monté en émetteur suiveur permet à l'ordinateur de fonctionner lorsque l'imprimante n'est pas sous tension. En effet, le DAI envoie les caractères simultanément sur l'écran et la RS 232. Si BUSY=1, aucun caractère ne peut etre envoyé à l'écran, l'ordinateur est bloqué. Lorsque l'imprimante est éteinte l'état de BUSY est indéterminé. Il peut, selon imprimante, se comporter comme un état haut. Le dispositif proposé permet d'éliminer ce défaut :
- Lorsque l'imprimante est connectée, le transistor fonctionne en émetteur suiveur. Le BUSY est transmis moyennant une légérement chute de tension.
- Eteinte, le transistor est bloqué. Tout se passe comme si l'entrée du 74LS02 était à la masse par une résistance de 180 ohms.

Le circuit imprimé que nous proposont est représenté figure 4. Il ne représente pas un investissement, en temps et en coût, important. Nous avons intégré le DOS du DCR sur cette carte. Ceci demande quelques explications :
- Le dos est une EPROM 2716 implantée entre les adresses F000 et F7FF.
- Le signal qui permet de sélectionner ce boitier peut être pris sur la patte No 9 de IC 44 (voir PC SCHEMATICS). Sur le circuit imprimé cette sélection est appelée F0.

Voici maintenant les chonogrammes des signaux que vous pouvez trouver sur la carte :

Chronogrammes



J'espère que cette interface, peu couteuse (50 à 100 FF) vous permettra de résoudre le problème que nous avons rencontré lorsque nous avons voulu connecter une imprimante sur le DAI. Cette carte fonctionne sur notre DAI. Elle est fiable.

L LEGRY

Le jour de la semaine

LE JOUR DE LA SEMAINE

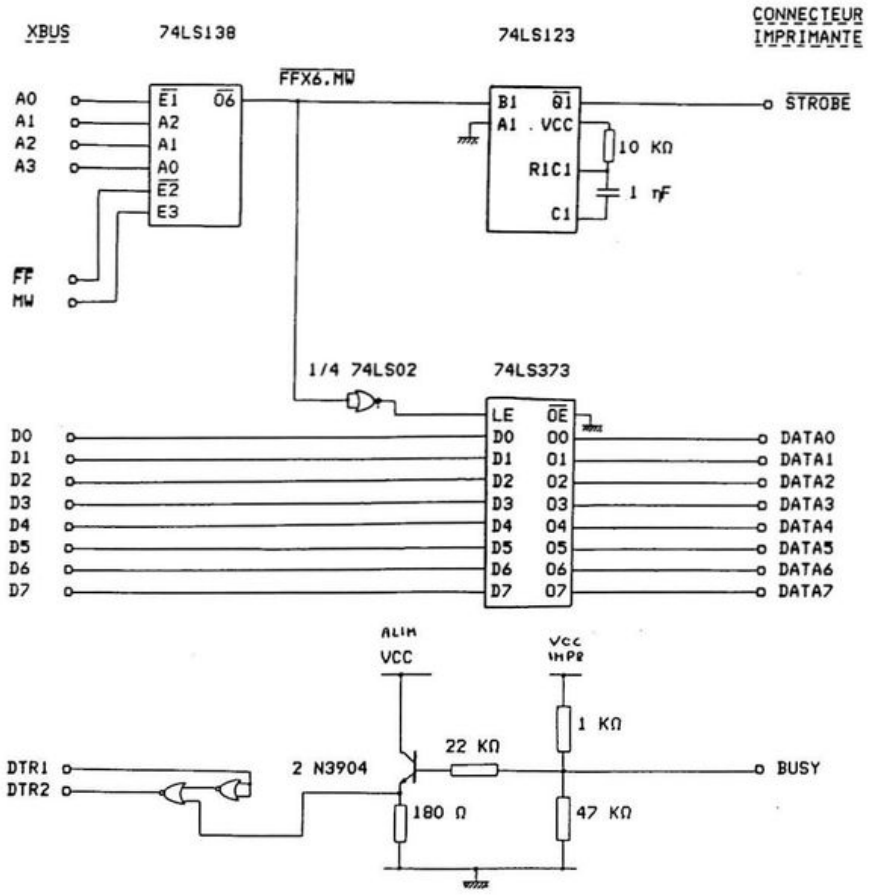
(Fabrice DULUINS)

Voici un petit programme tout simple qui peut s'avérer très utile pour savoir le jour de la semaine correspondant à n'importe quelle date postérieure à l'année 1585, date d'un changement dans le calendrier.

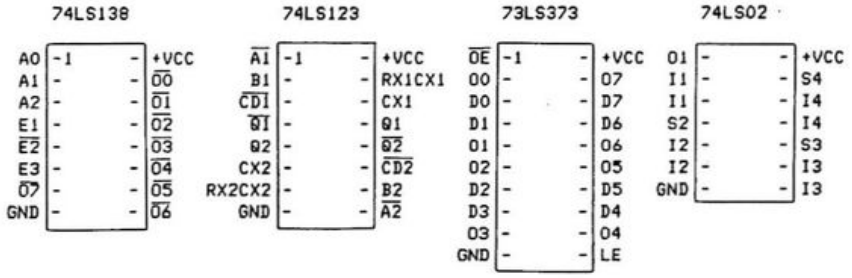
```
10      REM ************************************************************************
11      REM ** LE JOUR DE LA SEMAINE          F.DULUINS      19/12/1984    **
12      REM ************************************************************************
15      PRINT CHR$(12) : DIM JOUR$(7)
20      JOUR$(0) = "Dimanche"
21      JOUR$(1) = "Lundi"
22      JOUR$(2) = "Mardi"
23      JOUR$(3) = "Mercerdi"
24      JOUR$(4) = "Jeudi"
25      JOUR$(5) = "Vendredi"
26      JOUR$(6) = "Samedi"
30      PRINT "Entrez le jour,le mois,l'année et le siècle"
31      INPUT J,M,A,S
32      S = S-1
33      IF M<2 THEN M = M+12 : A = A-1
34      X = INT((M-2)*2.59) + INT(A*1.25) + INT(S*5.25) + J
35      X = X - INT(X/7) * 7
40      PRINT "Le ";J;".";M;".";A;" était un ";JOUR$(X)
45      END
```

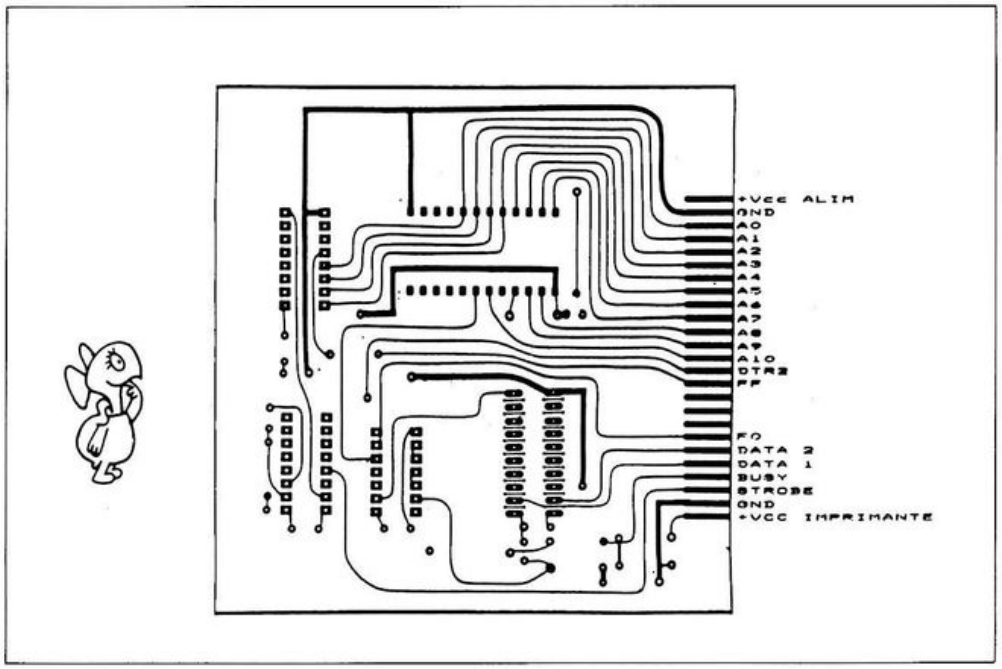
Ce petit programme peut très facilement etre transformé en sous-routine dans un autre programme, par exemple pour effectuer une vérification en comparant si à la date introduite par l'utilisateur correspond bien le jour introduit par ce dernier pour éviter de rentrer des données erronées.

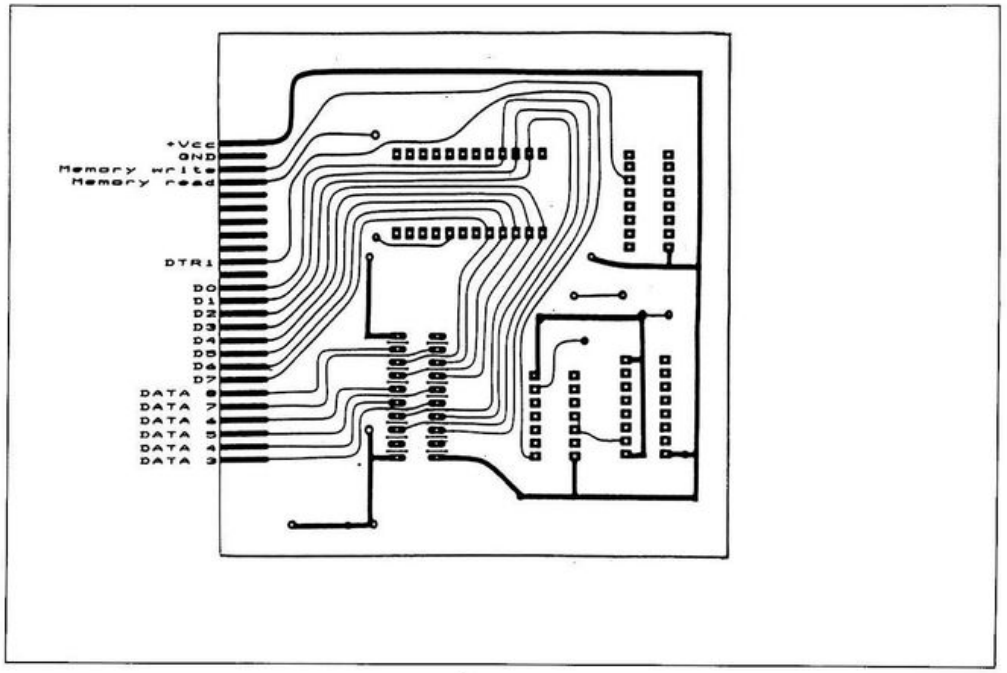
SCHEMA DE PRINCIPE DE L'INTERFACE CENTRONICS

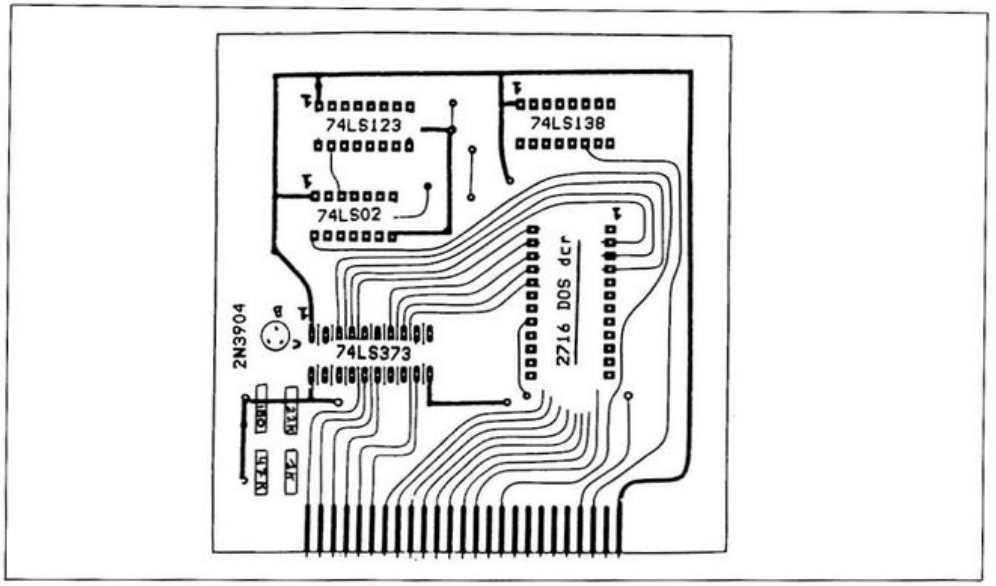


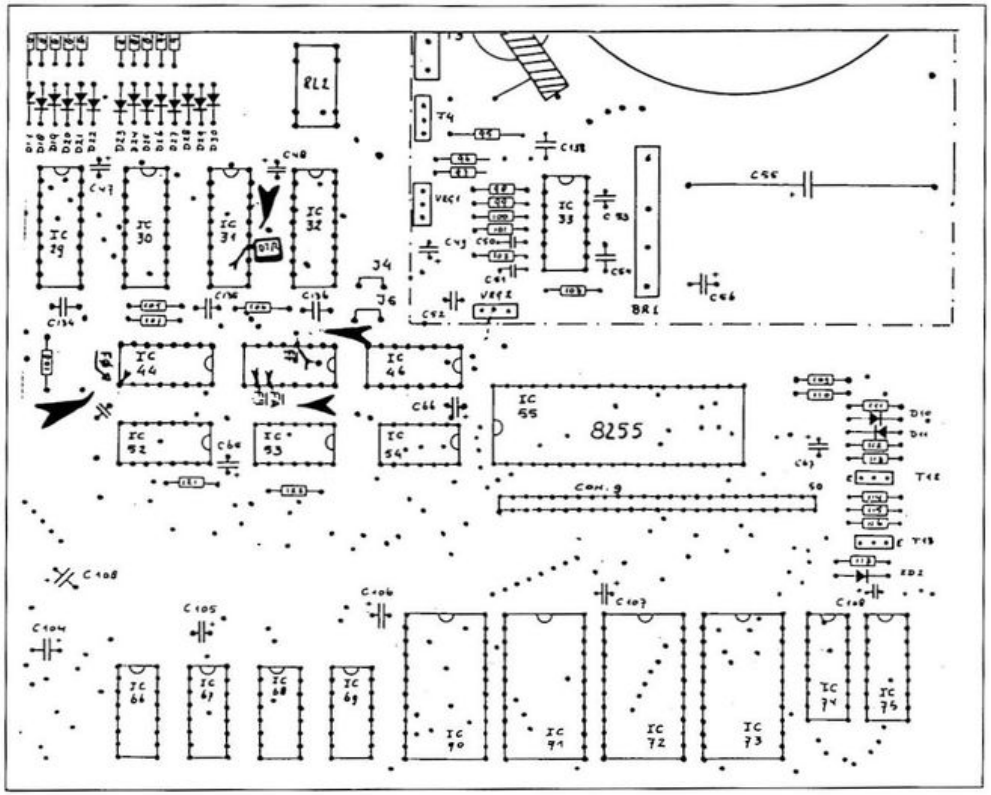
BROCHAGE DES COMPOSANTS




+VCC ALIM
GND
FO
DATA 2
DATA 1
BUSY
STROBE
GND
+VCC IMPRIMANTE


DTR1


74LS123
74LS138
74LS02
74LS373
2N3904
2716 DOS der


8255
BRL

Le fameux poke #29B

Le Fameux POKE 29B...

... ou comment déplacer son BASIC sans PLANTER la bete ...

Pour se convaincre du problème posé par #29B,il suffit d'éteindre puis de rallumer votre DAI, puis de taper le programme basic suivant :

```
10   HL%=#1000
20   POKE #29B,HL% IAND #FF:POKE #29C,HL%/256:CLEAR #100
30   PRINT "IDC BORDEAUX = BEST CLUB IN THE WORLD"
40   REM Il faut ce qu'il faut !
```

Vous constaterez que, malgré la véracite du message que vous désirez écrire, celui-ci ne s'affiche pas et que le DAI se plante.Cela est du à la gestion des pointeurs du basic par le CLEAR :

(1) #29B/#29C = pointeur du début de la 'heap'.
(2) #29D/#29E = longueur de la pile.
(3) #29F/#2A0 = pointeur de la fin de la 'heap'.

Quand on effectue le POKE #29B/#29C,on change le pointeur du début de la 'heap' sans changer les autres pointeurs. On se trouve alors dans une situation extraordinaire : en effet, en situation normale le pointeur (3) s'obtient en additionnant la longueur (2) au pointeur (1), symboliquement : (3)=(1)+(2). Or dans le programme basic ci-dessus , (3) est différent de (1)+(2) , après le poke, (2) ne représente plus la longueur de la 'heap' qui est en fait (3)-(1).Voici le programme du CLEAR qui explique le problème : j'appelle fausse longueur le contenu de (2) qui ne représente rien.

EFFECTUE CLEAR POUR DU BASIC

```
0E6B5   CALL    0E6F8H    ;Charge l'argument de CLEAR en HL.
0E6B8   PUSH H            ;Stack:=argument.
0E6B9   MOV A,H           ;Si cet argument
0E6BA   DCX H             ;est superieur
0E6BB   DCX H             ;a 32 kilo-octets,
0E6BC   DCX H             ;effectue un
0E6BD   DCX H             ;'NUMBER OUT OF RANGE'
0E6BE   ORA H
0E6BF   JM      0DA15
0E6C2   POP D             ;DE=nouvelle longueur.
0E6C3   LHLD    029DH     ;HL=ancienne fausse longueur.
0E6C6   XCHG              ;Echange HL et DE.
0E6C7   SHLD    029DH     ;Charge la nouvelle longueur en (2).
0E6CA   JMP     0D214H    ;Continue l'execution du CLEAR.
```

A ce niveau DE contient l'ancienne fausse longueur,HL la nouvelle longueur.


```
0D214   CALL    0DE1AH    ;Calcule HL-DE soit la différence des 2
;                         ;longueurs en utilisation normale.
;                         ;(cela n'a plus de sens avec
;                         ;notre pgm basic).HL=HL-DE
0D217   PUSH H            ;Sauve le résultat de ce calcul.
0D218   CALL    0CB23H    ;Vide la table des symboles et la HEAP,
;                         ;& Restore les pointeur basic.
0D21B   LHLD    100H      ;HL=flag <>0 si un programme est en
;                         ;cours.(HL pointe la ligne basic en
;                         ;cours).
0D21E   MOV A,H           ;Exit si ce flag est nul (dans le cas
0D21F   ORA L             ;d'un CLEAR en mode direct).
0D220   POP D             ;DE=différence enre les 2 longueurs.
0D221   RZ
0D222   DAD D             ;Sinon HL=POINTEUR BASIC+DE.
0D223   JMP     0CEBBH    ;Suite du clear.
```

A ce niveau,dans le cas d'une utilisation standard du CLEAR,HL contient le nouveau pointeur basic.

RAPPEL : lors de l'exécution d'un programme basic, BC est utilisé comme pointeur des instructions dans le programme basic.

```
0CEBB   SHLD    100H      ;Charge la nouvelle valeur dans le
;                         ;pointeur 100H
0CEBE   CALL    0E401     ;Effectue un RESTORE pour les DATAs
0CEC1   PUSH D            ;Stack:=<>ente la nouvelle longueur et
;                         ;la fausse longeur.
0CEC2   JMP     0D87F     ;Suite de CLEAR.

0D87F   POP H     ;HL=différence entre les longueur
0D880   DAD B     ;Calcule le nouveau pointeur basic a partir de
;                 ;cette différence.
0D881   MOV B,H   ;Stoque le nouveau pointeur dans BC.
0D882   MOV C,L
0D883   ORA A     ;CY=0 signifie pas d'action spéciale.Inutile pour
;                 ;nous.
0D884   RET
```

On voit donc que la nouvelle valeur du pointeur basic est calculée à partir de la fausse longueur. Par contre le programme basic est déplace, lui en fonction de la vraie ancienne longueur de la 'HEAP',en effet, examinons ce qui se passe a la fin de la routine de vidage de la 'heap' et de la table des symboles (adresse CB23).

```
0CB53   CALL    0DECAH    ;Organise la Heap + le buffer.
0CB56   POP H             ;Rappelle les registres sauvés au debut
0CB57   POP D             ;de la routine.
0CB58   POP B
0CB59   POP PSW
0CB5A   RET
```

```
0DECA   LHLD    29DH    ;HL=nouvelle taille de la 'heap'.
0DECD   XCHG            ;DE=nouvelle taille de la 'heap'
0DECE   CALL    0D195H  ;Initialise la 'heap' et la remet a zéro
0DED1   MVI A   01H     ;Codage pour l'instruction basic'NEW',
0DED3   STC             ;inutile ici.
0DED4   RET
0D195   LHLD    029FH   ;HL=ancienne fin de la 'heap'.
0D198   PUSH D
0D199   PUSH H
0D19A   PUSH D
0D19B   XCHG            ;DE=HL.
0D19C   LHLD    029BH   ;HL=ancien début de la 'heap'.
0D19F   XCHG            ;échange DE et HL.
0D1A0   CALL    0DE1AH  ;HL=HL-DE :HL contient la VRAI ancienne
                        ;longueur qui est utilisé pour la suite.
```

Donc BC n'a pas subi le meme 'offset' que le programme,il ne pointe donc plus la ligne courante du programme basic.Pour plus de clarté, reprenons notre petit programme basic:

-Le début de la heap a l'initialisation, donc le programme est déplacé de #1000H-#2ECH soit D14H.

-La taille de la heap est a l'origine #100, BC est donc augmenté de #100-#100 soit 0 !!!. BC pointe la meme mémoire, las!, l'ancienne position du programme correspond maintenant a la 'heap',donc BC ne pointe plus aucune instruction basic, d'ou le plantage.

Mais non ne pleure pas sur ton sort, heureux lecteur de DAICLIC, il existe une solution qui te redonnera le sourire, la voici: éteint et rallume ton DAI et tape :

```
10 HL%=#1000:DE%=(#10000+PEEK(#29F)+#100*PEEK(#2A0)-HL%) MOD #10000
20   POKE#29B,HL% IAND #FF:POKE#29C,HL%/256
30   POKE#29D,DE% IAND #FF:POKE#29E,DE%/256
40   CLEAR #100:PRINT "IDC BORDEAUX A ENCORE TERRASSE L'HIDEUSE BUGG"
```

Et voila.Il suffisait de calculer la vraie longueur et de la poker en #29D/#29E...

le 20.08.85
par Sebastien Dubourg,
pour IDC BORDEAUX.

Les instructions d'entrée/sortie

MICROPROCESSEUR

LES INSTRUCTIONS D'ENTREES/SORTIES

(Francis AUBERT, B-7860 LESSINES)

Cet article est un complément à l'article de HP.Legry sur le jeu d'instructions du 8080.

Format: IN port
OUT port

port: adresse (8 bits) du périphérique

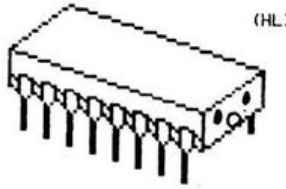
IN: l'accumulateur est chargé avec la donnée se trouvant sur le "databus" lors de l'accès au registre (port) du périphérique adressé.
OUT: le contenu de l'accumulateur est placé sur le "databus" et est transféré au registre (port) du périphérique adressé.

Le nombre de registres adressables par ce type d'instruction est au maximum égal à 256 (0 à 255--8 bits).
Les instructions IN et OUT de l'assembleur n'ont aucune similitude avec les instructions IN et OUT du BASIC.

Les instructions IN et OUT (de l'assembleur) ne sont pas utilisées par le DAI car les registres des périphériques sont adressés de la même manière qu'une case mémoire. Ce type d'adressage des périphériques est appellé "MEMORY MAPPED I/O". Le principal avantage de ce système est que les registres d'entrée/sortie ont le même type d'accès qu'une mémoire et bénéficie donc des instructions de lecture/écriture (MOV R,M - LDA adr - STAX rp - etc).
L'inconvénient de ce type d'adressage est qu'une partie de la mémoire doit être réservée pour les périphériques. Dans le cas du DAI, la zone mémoire de FB00 à FFFF est utilisée pour les différents registres d'entrée/sortie. L'autre type d'adressage des périphériques est appellé "I/O-MAPPED I/O". Il permet un adressage mémoire complet (64K RAM) mais deux instructions seulement sont disponibles pour accéder aux registres.
Dans le cas "MEMORY MAPPED I/O", les registres sont accessibles sous BASIC par PEEK et POKE. Tandis qu'en "I/O-MAPPED I/O", les registres restent inaccessibles sous BASIC.

NB: Dans l'article MICROPROCESSEUR part 3 de HP.LEGRY, une instruction de lecture/écriture de la pile a été omise.

XTHL: échange les deux derniers bytes de la pile avec le contenu des registres H et L.



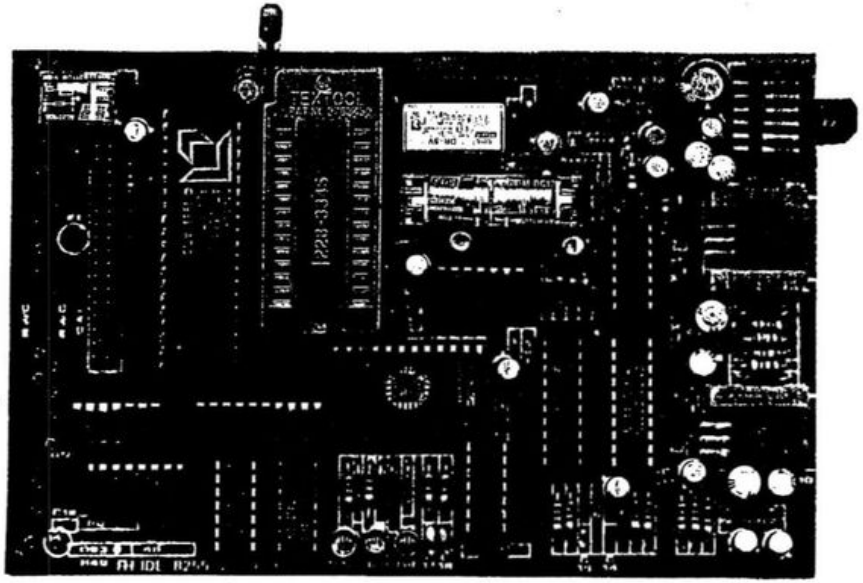
(HL) -> ((SP)) et ((SP)) -> HL

Programmeur d'eprom

PROGRAMMEUR D'EPROM UNIVERSEL POUR
TMS 2708, 27L08, 2716, 2732, 2732A, 2764, 27128, 27256
TMS 2508, 2516, 2564
INTEL 2716, 2732, P2732, 2732A, 2764, P2764, 27128, 27256
INTEL 27512, 27513
AMD 27512
et toutes les EPROM compatibles

En option, vous pouvez programmer des EPROM telles que les INTEL 2816A, les HITACHI HN 48016P et les NVRAM (INTEL 2004). De même, il est possible de lire des PROM comme les HN 62301.

Photo du nouveau programmeur d'EPROM pour DAI PC,
tous droits réservés par Adelbert Hoff:



Le programmeur peut être utilisé sur le DAI PC via le Bus DCE (connecté par un cable du type du DCR) ou via le format "Real-Word-Card" (Bus-Connector).

PRIX:
Kit programmeur d'EPROM avec
Alimentation

Software - programme machine
- programme Basic
- en EPROM pour contrôles

Documentation - carte princ.
- layout général

Exemples - installation
- programmation

445.00 DM tva incl.
sans frais d'envoi.

Programmeur d'EPROM assemblé et testé
595.00 DM tva incl.
sans frais d'envoi.

Ralph Hahn, Franz-Schneider-Strasse 15, D-5353 Mechernich-Vussem (R.F.A.)
(traduction: Christian POELS)

Courrier des lecteurs

Daniel MOULES, St GERMAIN LEMBRON (F):

(...) Il y a à peu près quinze jours, je recevais le numéro 4 de DAICLIC (que j'attendais depuis bien longtemps !) et je le dévorais page après page. Quand tout à coup, fut ma surprise de voir enfin le test de la carte X-Bus et de son interface VC1541, de plus, il y avait aussi le DAI STAR !

Je me plongeais alors dans ma collection de DAICLIC pour en ressortir le test du DAI DOS 1541...

Mes outils en mains, et dans la ferme conviction de pouvoir départager ces deux "bêtes" utilisant le lecteur Commodore, je me mis à dépecer les articles correspondants (ne vous tracassez pas, mes DAICLIC se portent bien !)...

Mais alors là, ce n'était pas la même chose, les concepteurs de l'un vantant les mérites de leur pouliche, les autres accusant les défaut du rival, c'était la petite guerre (entre DAIistes quand même !). Alors, et je pense que beaucoup d'autres DAIistes sont eux aussi restés sur leur faim, la conclusion était: "à vous de voir, dém... -vous !"
HSL+DAIDOS <-> X-BUS + VC1541, le match du siècle !
Ne voulant surtout pas créer un climat d'hostilité entre les deux parties (on se croirait en politique !), j'aimerais un jugement objectif (et totalement impartial !) sur ces deux systèmes et surtout UN RESULTAT final !

(...) Alors, je fais appel à vous. J'ai totalement confiance en toute la rédaction de DAICLIC ! et je pense que le sujet est suffisamment important et qu'il peut intéresser d'autres membres (vous pouvez d'ailleurs publier cette lettre si vous le jugez bon mais prenez garde à corriger les fautes d'orthographe !)
Je pense notament à des tests de rapidité de lecture, de compatibilité avec d'autres interfaces, des comparaisons du HSL et de la banque d'EPROM du X-Bus, etc.

Voilà, pour terminer, je tiens absolument à féliciter toute l'équipe pour ses efforts et les résultats dont elle fait preuve. On n'a pas l'habitude de lire dans vos colonnes, des éloges de IDC, alors pour une fois qu'on a trouvé un club de DAIistes digne de ce nom et qui de plus édite une super revue, on ne va pas se gêner ! Bravo à tous !!

PS: Vite, vite, beaucoup de DAIistes d'IDC et moi même voulont changer de mémoire de masse, un petit lecteur Commodore serait le bienvenu... (vous savez le 1541...)

Encore bravo et bonne chance à tous.

DAICLIC:

Nous vous remercions vivement pour vos encouragements, cela fait toujours plaisir à lire...
Vous comprendrez facilement qu'en tant que rédaction de la revue, nous ne pouvons vous conseiller un système plutôt qu'un autre. Cependant, si quelqu'un juge bon de donner son avis personel, il peut le faire via la revue (nous attendons vos articles...) et nous ne verrons pas d'objection à le publier !
Nous jugeons nos lecteurs suffisamment intelligents que pour leur laisser libre arbitre dans leur décision d'achat. Nous faisons tout pour que cette décision se fasse en connaissance de cause en publiant un maximum d'informations sur les systèmes proposés.
Nous pensons à aussi, comme vous le proposez, à publier (sans doute dans le prochain numéro) un test comparatif entre les deux systèmes ainsi qu'une présentation des caractéristiques de chacun de telle façon qu'il soit très facile de se forger une opinion personelle...

Jean DEPRAZ, VILLEURBANNE (F):

1. Je me suis équipé avec le drive Commodore 1541, et je voulais vous faire savoir qu'il est possible d'acquérir ce drive à un prix raisonable 1990 FF TTC avec cable + disquette Commodore. La maison qui fait ce prix est "Microdiffusion" installée largement dans la région parisienne (99 Rue Balard, 75015 Paris, tél 45541890) et en province (Tours, Bordeaux, Toulouse, Lyon et Maubeuge (Pringault, 39ter route de Férignies, 59600 Maubeuge, tél 27648526)).

2. Dans le DAICLIC no.2 p.79, dans LIBMAT librairie: calcul matriciel, il y a une petite erreur:
à la 7ème ligne à partir du bas:
au lieu de: C(I,J)=C(I,J)+A(I,J)*B(K,J)
il faut lire: C(I,J)=C(I,J)+A(I,K)*B(K,J)

3. Dans ce même article MATTRN A(),B() a été omise. Il faut ajouter:

```
700  PROCEDURE MATTRN ARR A
705  FOR I=1 TO DIM (A,1)
707  FOR J=1 TO DIM (A,2)
708  B(J,I)=A(I,J)
709  NEXT:NEXT
710  END PROC
```

Le DAI à 4 MHZ

LE DAI A 4 MHZ: DESILLUSIONS !

(Pascal JANIN, F-73 LA MOTTE SERVOLEX)

Eh oui, désillusions !
Pourtant, le dernier DAICLIC (4) le laissait sous-entendre en page 1... mais cela ne DOIT pas marcher, et d'ailleurs 4 MHz, c'est voir trop grand... Mais commençons par le commencement.

Assez "récemment" ont été distribués deux nouvelles versions du 8080A (2 MHz): le 8080A1 (3.125 MHz) et le 8080A2 (2.6 MHz). Plus rapides mais plus chers ! Le 8080A1 vaut 3 fois plus cher que le 8080A tout simple (140 FF au lieu de 50 FF), le 8080A2 deux fois plus (95 FF), et tout cela pour un gain en vitesse assez faible ! De plus, sachez messieurs les DAIistes, que notre "chère" fréquence de 2 MHz coordine non seulement le CPU en lui-même, mais aussi les interruptions, le son, la lecture de la mémoire, l'interface série, et SURTOUT le chargement et la lecture de données sur nos supports habituels (K7 audio, DCR, disk) !

Autrement dit, si vous tentez la greffe EN CHANGEANT DE QUARTZ (c'est le quartz qui détermine la fréquence d'horloge du CPU), votre DAI marchera sans doute un peu mieux mais deviendra COMPLETEMENT INCOMPATIBLE avec tous les autres DAI, et pire, ne lira même plus les programmes qui marchaient si bien avant (snif !)...

Par contre, comme les nouvelles versions du 8080 sont ENTIEREMENT COMPATIBLES BROCHE A BROCHE avec l'ancienne, une simple substitution SANS CHANGER LE QUARTZ est tout à fait possible, laissant votre DAI entièrement compatible avec tous les autres, mais sans grand intérêt (si ce n'est de dépenser inutilement son argent...).

Alors, comme je le disais précédemment, désillusions...

Petites annonces

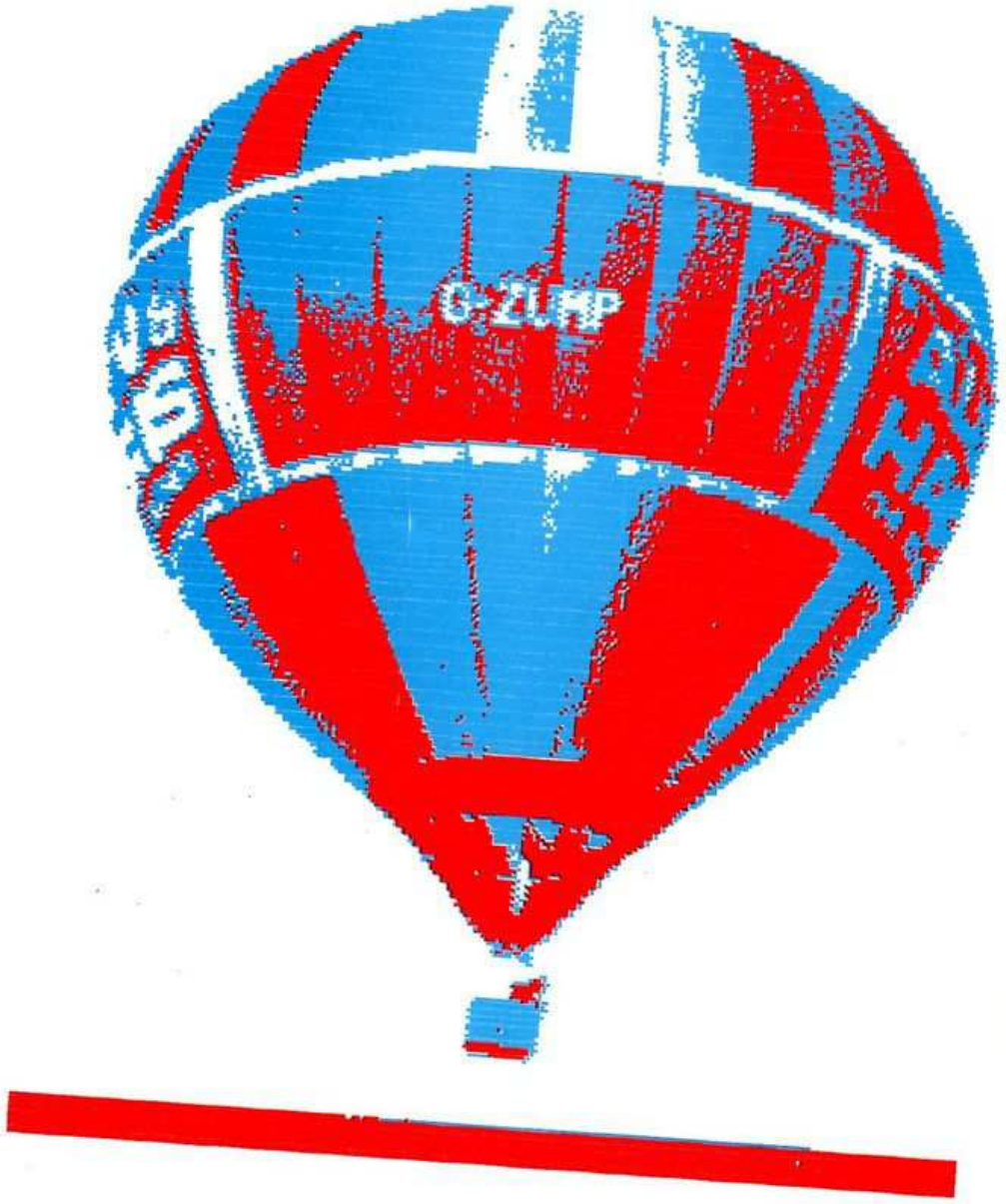
- Vends memocom mdcr-d avec TOS, equipe de témoins REW, FWD, cassette présente, cassette protégée 10000 FB. Cassettes SYSTEM 800: 720 FB les 6. Cassettes MDCR certifiées: 1600 FB les 6. Imprimante Microline 84, 200 cps, 131 col., interfaces // et série 50000 FB (achetée 90000 FB). D'Haene Filip, Bld H. Spaak 6, B-7900 Leuze-en-Hainaut...

- Floppy DAI 2 x 80 K cherche correspondant pour résoudre problèmes d'adaptation du DOS V1.0 au DBASIC V2.2: AUBERT Francis - ch. du Foubertsart, 104, 7860 LESSINES - 068/33.30.81.

- Vends pour cause utilisation disquettes, cassettes DCR SYSTEM 800: 120 FB/pce. AMD9511: 4900 FB. Christian POELS (rédaction).

- Vends DAI + DCR + documentation + programmes: 26000 FB. Bernard André, rue de Plainevaux 81/34, B-4100 SERAING.

REMARQUE: Ces petites annonces gratuites pour les abonnés sont exclusivement réservées à des propositions entre particuliers sans objectifs commercial et relatives à l'informatique. DAICLIC se réserve le droit de refuser une annonce sans avoir à fournir de justification.


G-ZLHP