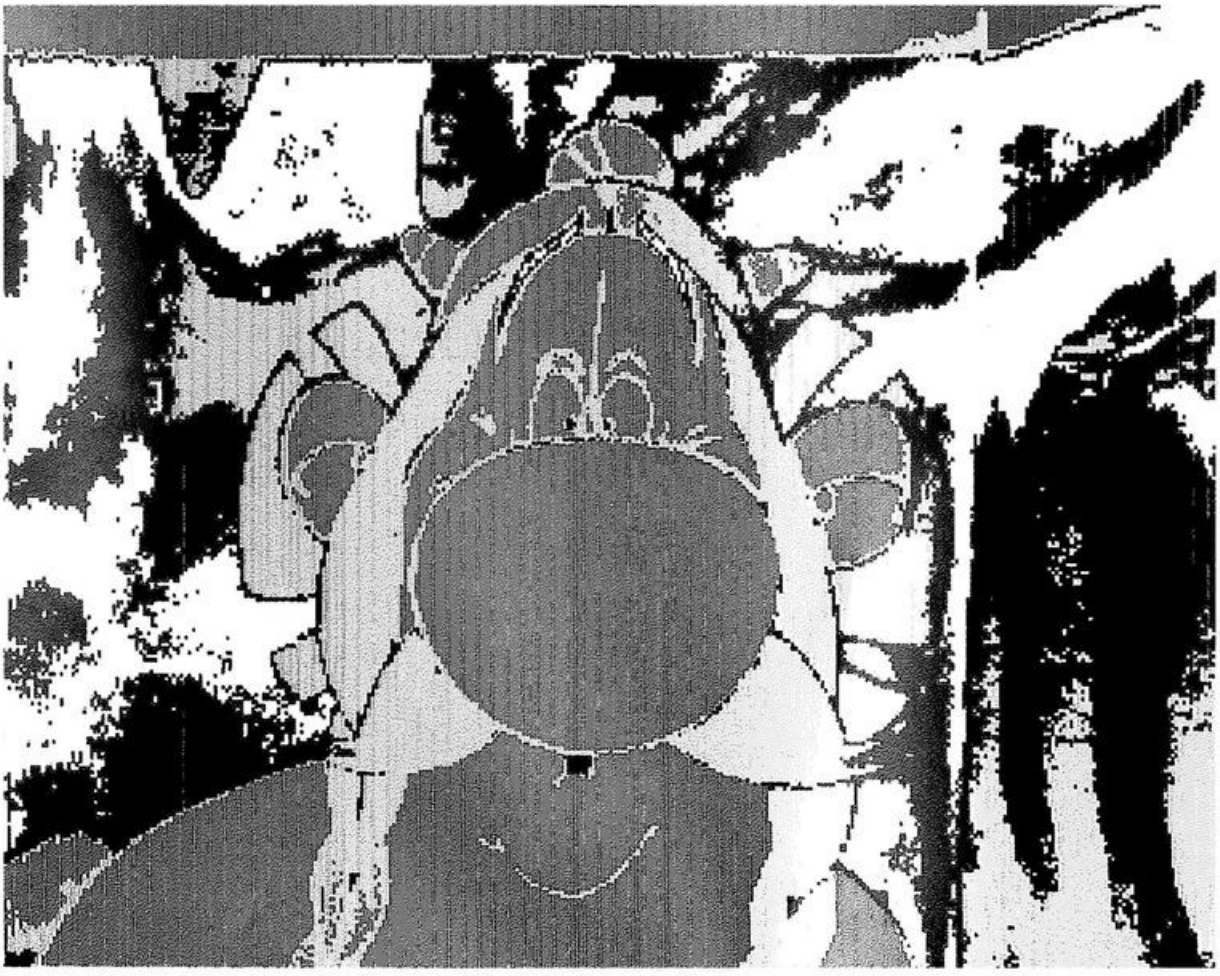
DAI NAMIC

33

DAI CLIC

6



tweemaandelijks tijdschrift maart-april 86

bimestriel mars-avril 86

een uitgave van DAInamic VZW en IDC ASBL
une publication de DAInamic VZM et 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 Liège-A :	041/79.66.66
MN2 Paris-A :	1/39.71.82.91

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 33 DAICLIC 6

INTERNATIONAL DAI CLUB

D A I C L I C 6

Vous voilà donc en possession du DAICLIC numéro 6 ! Nous vous souhaitons une lecture agréable, et restons attentifs à toute suggestion pour améliorer la revue, tant du point de vue présentation, que du point de vue contenu. Vous remarquerez que la partie DAICLIC se trouve cette fois en début de revue. Cette alternance est volontaire et se poursuivra dans les prochains numéros.

Quelques articles d'IDC BORDEAUX qui, suite à des petits problèmes d'ordre technique, n'avaient pu etre publiés dans les précédentes revues, paraissent dans ce numéro. Toutes nos excuses à Bordeaux !!! Vous trouverez notamment (p.7) les programmes MINITEL qui auraient du etre publiés avec l'article à ce sujet dans le numéro 4...

Un nouveau serveur Micro-Net2 va ouvrir à Paris !!! Bonne nouvelle donc pour les membres Français ! Le numéro est le 1/39718291. D'autre part, pour la Belgique, l'index IDC est maintenu sur les serveurs Bruxelles A (02/2427008) et Liège A (041/796666) uniquement. Les autres numéros communiqués précédemment restent cependant en activité.

Pour ce qui est des logiciels, ils sont maintenant tous disponibles sur VC1541 à l'exception de MAILING LLST,COM et MXDIR !

Nous envisageons très sérieusement d'étendre nos activités sur différentes autres machines... plus de renseignements dans un prochain numéro...

Au sommaire du prochain numéro: l'article tant attendu de Pascal JANIN, qui vous permettra assez facilement de donner la parole à votre DAI, et la suite de l'article sur le MINITEL.

La Rédaction.

O C T E T 1 9 8 4 - D A I C L I C 1 9 8 5

Tous les numéros de l'année passée seront prochainement réédités afin de permettre aux nouveaux membres de disposer de l'ensemble de la documentation éditée par le club jusqu'à ce jour. Seront aussi réédités, tous les numéros de 1984, qui s'appellaient alors 'OCTET'.

Les prix, port compris pour	Belgique:	Etranger:
OCTET 84:	600 FB	700 FB
DAICLIC 85:	700 FB	800 FB
OCTET 84 + DAICLIC 85:	1200 FB	1400 FB

Pour les paiements, utilisez la meme methode que pour les cotisations (V. p.0).

Les commandes doivent nous parvenir avant le 1er juillet 1986. Les revues seront envoyées dans le courrant de ce mois de juillet.

Supercastel - Daitona

Encore du neuf chez IDC software ! ! !

Deux nouveaux programmes vous sont présentés ce mois-ci par IDC software : **SUPERCASTEL et DAITONA.**

SUPERCASTEL :

En deux mots : vous vous déplacez dans un château à la recherche d'un trésor, toujours prêt à vous battre...et à vaillament sauvegarder votre peau.

Le château comporte 3 étages (cave, rez-de-chaussée, 1er étage) avec dans chacun d'eux, 63 secteurs se recoupant par moitié, soit un total de 189 secteurs à visiter !!! Dans le chateau, vous croiserez 32 sortes de races de monstres, 32 sortes de pièges et 32 sortes de trésors !...mais aussi, des éléments tels que des plantes vénéneuses, de la nourriture, des portes secrètes, des potions magiques, etc...

SUPERCASTEL: Une simulation précise, en temps réel, tient compte d'une infinité de paramètres pour le déroulement du jeu. (poids de vos armes, rapidité, dextérité, intelligence, décision, blessures, etc...)

SUPERCASTEL, par R. Sip, uniquement sur DCR, livré avec mode d'emploi : 1250 Fb.

DAITONA :

Le sommum de la course automobile sur votre DAI ! 9 circuits différents, 3 type de voitures, utilisation du clavier ou des paddles pour le contrôle de la voiture.

Une vue en perspective du circuit... des autres voitures qui font la course avec vous et qu'il vous faudra éviter...telles sont les caractéristiques de ce nouveau jeu LM/basic réalisé par M.Billot.

DAITONA : AUDIO 500 Fb. DCR/KENDOS/VC1541 :supplément 250 FB

Toutes vos réalisations sont bien sur plus que jamais attendues chez **Marc Vandermeersch, 17 av du Vert Bocage, 1410 Waterloo, Belgique.**

Une petite précision supplémentaire : le programme MXDIR ne tourne que sur KENDOS 80 trk et son auteur est prêt à faire parvenir une copie du mode d'emploi du programme à toute personne intéressée...

Dailink Modem

IDC SOFTWARE PRESENTE: D a i L i n k V1.1

DaiLink est un nouveau programme de communication, idéal pour les communications par modems (serveurs télématiques, échanges de programmes,...) et les communications directes entre deux ordinateurs. Caractéristiques:

- Entièrement écrit en langage machine et compatible avec TOUS les supports: cassettes audio, MDCR, VC1541, Ken-DOS, drives Prodata et DAI-DOS 1541 !

- Paramétrage de la communication: Baud rate, nombre de stop-bits, parité, half/full duplex.

- Mode dialogue: communication directe sans utilisation du buffer.

- Mode réception: le fichier recu est sauvé sur support et peut être réutilisé à l'aide du traitement de textes TextEditor. Permet entre autres d'imprimer par la suite le fichier reçu.

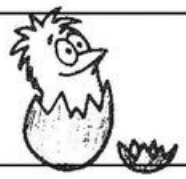
- Mode émission: envoi d'un fichier TextEditor se trouvant sur support. Permet par exemple de rédiger sa lettre à l'aise hors connexion !

- Pour les possesseurs de la carte XBASIC (de H. TEGETHOFF), il est aussi possible d'envoyer/ recevoir des programmes BASIC.

Certains caractères spéciaux sont correctement utilisés comme par exemple le CHR$(7) (Bell) qui produit un beep sonore, les caractères Xon/Xoff, etc.

Prix: 750 FB
(ajouter 250 FB pour la version DCR)

Pour la commande, suivez la meme procédure que pour la cotisation (V. page 0).



MODEMS - SERVEURS - MODEMS - SERVEURS - MODEMS - SERVEURS - MODEMS - SERVEURS -

Dans le Daiclic numéro 4, je vous présentais le modem BONDWELL 101. Rappel des caractéristiques: 300 bauds, full duplex, CCITT V21, modes Answer - Originate - Test1 - Test2, auto-answer. Livré avec cable RS232 et alimentation.

Ce modem peut désormais être commandé par le club:

Prix:	modem:	6600 FB
	(+ port Belgique:	350 FB)
	(+ port Etranger:	800 FB)

Paiement: voir paiement des cotisations.

```
n e w s n e w s n e w s n e w s
e                               n
w       I. D. C. Bordeaux.      e
s                               w
n e w s n e w s n e w s n e w s
```

A L'ATTENTION DE TOUS LES CLUBS ASSOCIES

Remarque :

En raison du manque de place dans les deux derniers numéros, des articles qui avaient été prévus d'etre publiés ensemble pour une meilleure compréhension , n'ont pu etre inclus dans DAICLIC 4 et 5, et nous vous prions de nous en excuser. Entre autres, vous remarquerez qu'à la fin de mon article sur le Minitel, je cite deux programmes: un basic et sa routine en assembleur , dont les listings auraient du etre mis après l'article .Mais en raison de leur longueur il n'a pas été possible de les y mettre ...

De très bonnes nouvelles !!

Le secteur bibliothèque du club se développe grace à la coopération de l'un de nos membres qui a traduit de l'anglais en français , de façon très professionelle, le manuel complet du KEN-DOS.

*En raison du don bénévol e de ce travail au club , ce livre vous est offert à prix coutant pour le prix de 40FF port compris. : MANUEL KEN-DOS en FRANCAIS:

*Vu le nombre des demandes , une réédition du 'TOS COMMENTE' a été faite.Que tous ceux qui n'avaient pu l'obtenir l'an dernier me réécrivent. Le prix lui n'a pas augmenté : 150FF port compris (= p.c.)

*Toujours disponibles : le livre indispensable au programmeur en assembleur ,du club FLAMMAND associé DAINAMIC : le FIRMWARE MANUEL écrit en Anglais: 250FF p.c. et les deux tomes de 'Apprendre à programmer sur le DAI' 50FF pièce p.c.

En raison de l'enthousiasme suscité par notre série d'articles sur le MINITEL, une interface améliorée est en cours de réalisation afin de permettre le branchement simultané de votre imprimante série avec le MINITEL et ce, sans devoir sans cesse débrancher l'un pour rebrancher l'autre. En outre cette connexion devrait autoriser la commutation de l'un des deux appareils non seulement par un interrupteur normal mais en plus par voie logicielle grace à une prise sur l'entrée K7 audio. Guettez le plan dans un prochain DAICLIC ...

Coté logiciels, c'est aujourd'hui plus d'une soixantaine de petits programmes didactiques en PASCAL qui vous sont offerts grace aux travaux d'un de nos adhérents : Mr. DEPRAZ

De moins bonnes nouvelles:

En raison de mon changement de lieu de travail, je ne peux me rendre chez moi qu'un week-end tous les 15 jours, et ceci retarde mes réponses. De meme Mr.LAFARGUE a déménagé et a donc lui aussi un peu de retard dans ses réponses.

-Pour faciliter vos demandes de programmes, il est désormais INDISPENSABLE d'indiquer pour toute demande de programme ou d'article de la BIBLIOTHEQUE :

*Le TITRE — par ex. 'Le LPS'
*La PARTIE de la liste concernée — par ex. 'Liste 9b'
*La CATEGORIE du prog. demandé — par ex. 'Langages'
*Le COMMENTAIRE pour ce titre
*Le No de FACE pour un pgm. ou le NOMBRE de pages pour un article.

Ces simples indications nous évitent une longue recherche à l'aide du seul titre parmi des centaines de programmes ou des kilogrammes d'articles ...

Les nouvelles bonnes et mauvaises:

Des évenements peu agréables nous ont obligés à prendre des décisions somme toute plutot positives !!! : La personne qui fabriquait les KEN-DOS ,en Hollande, vient d'abandonner la fabrication de ce produit. Cela voulait dire deux choses : plus de possibilité pour les personnes qui voulaient acheter ce qui ,à mon sens ,représente le meilleur outil pour le DAI ,plus de Ken-Dos !! Et pour ceux qui l'avaient déjà acheté , pas de facilités de réparations !!.

Les actions alors entreprises ont permis d'obtenir l'autorisation du constructeur de continuer la fabrication de son système et parallèlement, développer les possibilités de réparation à BORDEAUX. Des contacts ont été pris avec des réparateurs professionels qui ont accepté d'effectuer les réparations non seulement du KEN-DOS, mais aussi du DAI et du MEMOCOM !!! Ces memes réparateurs offrent aux personnes intéressées la fabrication de leur Ken-Dos à un prix légèrement inférieur à celui obtenu de Mr. Gooswit.(pour tout renseignement, joindre 5 timbres pour les frais de photocopies et port).

I.D.C.Bordeaux.
s/c DELANNAY Bruno.
Res. les acacias Bt.B3
Avenue de SAIGE
33600 PESSAC .

NOUVEAU JEU-CONCOURS

Les nombres de 2 chiffres 25 et 76 possèdent une particularité intéressante. Toutes leurs puissances se terminent par 25 et 76 respectivement.

Par exemple : $25 \times 25 = 625$ (--)

$76 \times 76 \times 76 = 438976$ (--)

Quels sont les nombres de 3 chiffres qui possèdent la même propriété ?

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 !

```
SPL V1.1  PAGE 1

0000                    TITL      'UTILITAIRE MINITEL par Sebastien DUBOURG'
0000            ;
0000            ;15 juillet 1985
0000            ;
0000            ;
0000            ;*************
0000            ;* ADRESSES *
0000            ;*************
0000            ;
0000            ;
0000  à=FF05 BAUD    EQU       0FF05H          ;Baud register (code de la vitesse de
                                                transmission)
0000  à=FFF0 INPUT   EQU       0FFF0H          ;Contient le dernier caractere recu sur la
0000  à=FF03 STATUT  EQU       0FF03H          ;Registre de statut.                [ RS232.
0000  à=0131 OUTDIR  EQU       131H            ;Flag indiquant la direction de l'output.
0000            ;
0000            ;
0000            ;*******************
0000            ;* TABLE D'ENTREES *
0000            ;*******************
0000            ;
0000                    ORG       900H
0900  00      BUBBLE    DB        0H              ;Flag si <>0 l'envoie sur le minitel est
0901  C31109  ENTREE    JMP       AGAIN           ;Vers la routine d'affichage.  [autorise.
0904  C37409  MARCHE    JMP       ON              ;Vers la routine de mise en parallele
0907            ;                                 ;d'une sortie vers le minitel.
0907  C37D09              JMP       SCROLL          ;Place le minitel dans le mode SCROLL
090A            ;                                 ;et dans le mode CURSEUR ACTIF.
090A  00      TERM      DB        0H              ;Flag si =0:mode emmulation terminal
090B            ;                                 ;(le chardel le return sont comme sur le
090B            ;                                  DAI)
090B            ;*************************************************************
090B            ;* CALCUL DE PARITE ET MISE EN FORME DE L'OCTET A ENVOYER  *
090B            ;*************************************************************
090B            ;
090B            ;  A=OCTET A PARITER
090B            ;
090B  E67F    PARITE    ANI       7FH             ;Met le bit 7 de (A) a 0.
090D  E8                RPE                       ;Si (A) est pair,laisser le bit 7 a zero..
090E  F680              ORI       80H             ;Sinon mettre le bit 7 a 1.
0910  C9                RET
0911            ;*************************************
0911            ;* AFFICHAGE SIMULTANNE DAI-MINITEL *
0911            ;*************************************
0911            ;
0911            ;est execute apres un RST5 et avant l'execution d'un
0911            ;RST5 DB 3 (affichage d'un caractere sur l'ecran.),
0911            ;pour envoyer ce caracters au minitel via la RS232.
0911            ;
0911            ;Mise en marche: sous utilitary: G MARCHE
0911            ;       (mettre la valeur de MARCHE lue par S).
0911            ;Arret : sous utilitary, Z3
0911            ;
0911  F5      AGAIN     PUSH PSW
0912  3A3101            LDA       OUTDIR          ;Si la sortie est  sur  la RS232,
0915  B7                ORA A                     ;mettre la sortie sur ecran seulement,
0916  C21E09            JNZ       LIAGA           ;afin de ne pas envoyer des caracteres
0919  3E01              MVI A     1H              ;parasites au minitel.
091B  323101            STA       OUTDIR
```

```
SPL V1.1  PAGE 2     UTILITAIRE MINITEL par Sebastien DUBOURG

091E  3A0009 LIAGA   LDA       BUBBLE          ;Si la sortie vers le minitel
0921  B7             ORA A                     ;n'est pas permise effectuer
0922  C27009         JNZ       NOTHIN          ;un RST5 normal
0925  3E88           MVI A     88H             ;la vitesse 1200 bauds est
0927  3205FF         STA       BAUD            ;selectionnee.
092A  F1             POP PSW
092B  E1             POP H
092C  E3             XTHL                      ;HL contient a ce niveau l'adresse
092D        ;                                  ;qui suit l'instruction RST 5.
092D  E5             PUSH H                    ;HL pointe donc le DATA.
092E  D5             PUSH D
092F  C5             PUSH B
0930  F5             PUSH PSW
0931  7E             MOV A,M                   ;A=DATA.
0932  FE03           CPI       3H              ;Si ce data est 3 (ordre d'affichage
0934  CA4009         JZ        CAR             ;sur l'ecran) envoyer le car. sur la RS232.
0937  F1      OUTO   POP PSW                   ;Sinon effectuer un RST5 normal.
0938  C1             POP B
0939  D1             POP D
093A  E1             POP H
093B  E3             XTHL
093C  E5             PUSH H
093D  C3FDC6         JMP       0C6FDH
0940  F1      CAR    POP PSW
0941  F5             PUSH PSW                  ;A=caractere a afficher.
0942  CD0B09         CALL      PARITE          ;Ajouter au caractere la bonne parite.
0945  CD94DD         CALL      0DD94H          ;Envoyer le tout sur la RS 232.
0948  F5             PUSH PSW
0949  3A0A09         LDA       TERM            ;Si le mode termnal n'est
094C  B7             ORA A                     ;pas actif,continuer le
094D  C29609         JNZ       PROG            ;RST5 DB 3H.
0950  F1             POP PSW
0951  FE8D           CPI       8DH             ;Si le caractere est return
0953  C25E09         JNZ       CONTO           ;(revient au debut de la ligne
0956  3E0A           MVI A     0AH             ;courrante sur minitel),aller aussi
0958  CD94DD         CALL      0DD94H          ;sur la ligne suivante.
095B  C33709         JMP       OUTO
095E  FE88    CONTO  CPI       88H             ;Si le caractere est chardel,qui sur
0960  C23709         JNZ       OUTO            ;le minitel revient simplement en
0963  3EA0           MVI A     0A0H            ;arriere sur la ligne,aficher un space
0965  CD94DD         CALL      0DD94H          ;en plus,puis renemir une position en
0968  3E88           MVI A     88H             ;arriere sur la ligne courante du minitel.
096A  CD94DD         CALL      0DD94H
096D  C33709         JMP       OUTO
0970        ;
0970  F1      NOTHIN POP PSW
0971  C3FDC6         JMP       0C6FDH
0974        ;
0974        ;*********************************************
0974        ;* MISE EN PLACE DE L'AFFICHAGE SIMULTANE *
0974        ;*********************************************
0974        ;
0974  F3      ON     DI
0975  210109         LXI H     ENTREE          ;Poke le point d'entree
0978  226C00         SHLD      6CH             ;de AGAIN dans le vecteur 5.
097B  FB             EI
097C  C9             RET
097D        ;
```

```
SPL V1.1  PAGE 3     UTILITAIRE MINITEL par Sebastien DUBOURG

097D          ;*****************************************************************
097D          ;* MET LE MINITEL EN MODE SCROLL ET EN MODE CURSEUR ACTIF *
097D          ;*****************************************************************
097D          ;
097D          ;protovole pour 'scroll'
097D          ;
097D  3E1B    SCROLL  MVI A     1BH
097F  CD94DD          CALL      0DD94H
0982  3E3A            MVI A     3AH
0984  CD94DD          CALL      0DD94H
0987  3E69            MVI A     69H
0989  CD94DD          CALL      0DD94H
098C  3EC3            MVI A     0C3H
098E  CD94DD          CALL      0DD94H
0991          ;
0991          ;mot de controle pour 'curseur actif'
0991          ;
0991  3E11            MVI A     11H
0993  C394DD          JMP       0DD94H
0996          ;
0996  F1      PROG    POP PSW
0997  C33709          JMP       OUTO
099A                  END
```

```
1     CALLM #904:POKE #900,1:PRINT CHR$(12);TAB(20);"UTILITAIRE MINITEL 2"
2     PRINT TAB(20);"(C) Sebatien DUBOURG"
3     PRINT TAB(20);"   & le fameux IDC BORDEAUX"
5     PRINT TAB(40);"27 juillet 1985"
6     PRINT
7     REM UTILISATION : taper directement les caracteres sur le clavier.
8     REM Si tab , on rentre les caracteres en valeur ASCII decimales.
9     REM Pour sortir du mode tab,entrer le code 0.
10    POKE #900,0:PRINT CHR$(0);:REM ?CHR (0); pour le pokage de la bonne vitess
      e de transmition.
11    REM CALLM#904:MISE EN MARCHE DE LA LIAISON RS232
12    REM CALLM#907:MODE SCROLLING ET CURSEUR ACTIF POUR LE MINITEL
13    REM POKE#90A,0:MODE EMULATION TERMINAL
20    CALLM #907:REM 900=0:SORTIE RS232 AVEC PARITE
100   POKE #90A,1:REM MODE DIRECT POUR LE MINITEL
110   GET%=GETC:IF GET%=0 THEN 110
120   IF GET%=16 THEN GET%=#B:GOSUB 1000:GOTO 110
130   IF GET%=17 THEN GET%=#A:GOSUB 1000:GOTO 110
140   IF GET%=18 THEN GET%=8:GOSUB 1000:GOTO 110
145   IF GET%=19 THEN GET%=9:GOSUB 1000:GOTO 110
150   IF GET%<>9 THEN GOSUB 1000:GOTO 110
160   POKE #900,1:INPUT GET%:PRINT
170   IF GET%=0 THEN POKE #900,0:GOTO 110
180   POKE #900,0:GOSUB 1000:GOTO 160
1000  PRINT CHR$(GET% IOR #80);
1010  POKE #900,1:PRINT " (#";HEX$(GET%);" )."
1020  POKE #900,0:RETURN
```

Extensions des commandes DCR

Dans le manuel 'TOS COMMENTE', lors de la routine 'FIND AN INSTRUCTION IN THE TOS TABLE & EXECUTE IT' (adresse F0A7), je parle de la possibilité de definir sa propre table d'instructions. En voici donc la démonstration par l'exemple...

Un autre exemple avec d'autres fonctions peut etre envoyé aux membres d' I.D.C.Bordeaux. grace au source du programme 'DIRECTORY I.D.C.Bordeaux' pour DCR dont le descriptif est donné dans les pages !! I.D.C.Bordeaux NEWS !! .

Avant de programmer l'application, il est bon de voir comment le TOS gère la table des commandes. Comme on le voit a l'adresse F0A7 du 'TOS COMMENTE', la routine de recherche d'une instruction fait appel au sous-programme de la ROM du BASIC implanté a l'adresse CA34 et intitulé dans le 'FIRMWARE MANUEL' LOOKC. Ce sous-programme est utilisé pour la précompilation du BASIC mais peut très bien servir a toutes vos applications. En voici le listing...

```
*********************************************
*  TROUVE UNE INSTRUCTION  DANS UNE TABLE   *
*********************************************
```

Cherche si dans la ligne en cours à la position pointée par C se trouve une instruction présente dans une table d'instructions , et si oui, donne l'adresse ou se trouve l'instruction dans la table . La table est terminée par 0H.

entrée : HL : début de la table d'instructions.
C : position dans la ligne testée.
E : nombre d'octets d'information se trouvant après le nom de l'instruction dans la table, décrémenté de 1.

sortie : - si l'instruction existe dans la table : CY=1.
HL : pointe l'adresse ou se trouvent les informations relatives a l'instruction trouvée.
C : position sur la ligne en cours après l'instruction détectée.

- si l'instruction n'existe pas dans la table : CY=0
C : pointe le caractère suivant.
HL : contient l'adresse située juste après la table .

```
CA34    LOOKC    CALL    0DDD2H    ;Charge A avec le caractère
                                   ;pointe par C sur la ligne
                                   ;( en sautant les 'SPACE' et
                                   ;les 'TAB' ) , C pointe le
                                   ;caractère suivant.
CA37    LKC10    MOV D,M           ;D = longueur du nom de la
                                   ;chaine pointée par HL dans
                                   ;la table.
CA38             INX H             ;HL pointe le premier
                                   ;caractère de la chaine.
```

```
CA39                    MOV A,D             ;A = longueur de la chaine.
CA3A                    ORA A               ;Si la longueur est 0 , la fin
CA3B                    RZ                  ;de la table est atteinte exit
CA3C                    PUSH B              ;Sauve la pstn sur la ligne.
CA3D        LKC20       CALL        0DDE0H  ;A = car. actuel sur la ligne.
CA40                    INR C               ;C pointe le caractère suivant
CA41                    CMP M               ;Si les caractères actuels de
CA42                    INX H               ;la ligne et de la table
CA43                    JNZ         0CA4EH  ;sont <>,instruction suivante.
CA46                    DCR D               ;Dcr. le compteur de chaine.
CA47                    JNZ         0CA3DH  ;Teste le caractére suivant.
CA4A                    XTHL                ;Annule le PUSH B de la
CA4B                    POP H               ;ligne CA3C
CA4C                    STC                 ;CY = 1 : instruction trouvée.
CA4D                    RET
                        Si les chaines sont <>
CA4E        LKC30       MOV A,D             ;A = octets de la chaine
                                            ;n'ayant pas été testés.
CA4F                    ADD E               ;Calcule l'adresse de la
CA50                    CALL        0DE30H  ;chaine suivante dans la table
CA53                    POP B               ;Redonne le C d'origine.
CA54                    JMP         0CA37H  ;Essaie la nouvelle chaine.
```

Pour le TOS,la table des instructions est située en F786 (voir son listing complet dans le 'TOS COMMENTE'). Les octets d'information sont constitués par l'adresse d'exécution de l'instruction. Le système d'interprétation du TOS va plus loin, dans le cas ou il n'a pas trouvé l'instruction demandée dans la table, il va lire l'adresse située juste après l'octet 0 qui marque la fin de la table. Si cette adresse est 0000H,le TOS arrête sa recherche et donne la main au système d'interprétation BASIC (en mode direct uniquement. C'est ce qui explique la syntaxe du DCR : REW1:LOAD est valable mais LOAD:REW1 provoque une erreur de syntaxe car le TOS n'as pas LOAD dans sa table et donne donc la main au BASIC qui ne comprend pas REW1 !).Si l'adresse est au contraire non nulle, celle-ci est interpretée comme l'adresse d'une autre table de recherche et le TOS va continuer la recherche de l'instruction dans cette autre table...etc...Ceci permet de mettre en série autant de tables que l'on veut, et de pouvoir écrire sa propre table sans avoir a recopier la table du TOS (ouf! merci monsieur MEMOCOM!).

Pour l'explication de la mise en service de la routine d'interprétation, se reporter aux chapitres suivants du 'TOS COMMENTE':

-INITIALISATION (LHF4D4) : lignes F4D8,F4FA
-DCR (LHF566)
-DATA FOR DCR (LHF4B6) : ligne F4D1
-NEW DINC (LHF054) : pour le mode direct
-INTERPRETATION & EXECUTION IN DIRECT MODE (LHF08B)
-ENTRYPOINT'S TABLE (LHF000) : ligne F000 pour mode pgm
-REM INTERPRETOR (LHF021) : pour le mode pgm
-FIND A INSTRUCTION... (LHF0A7)

A la suite de tout ceci, l'exemple que vous attendez tous...

```
SPL V1.1  PAGE 1

0000                    TITL      'Extension des commandes DCR'
0000            ;
0000            ;====================================
0000            ;=                                  =
0000            ;=    Auteur : Sebastien DUBOURG    =
0000            ;=               & IDC BORDEAUX     =
0000            ;=   Res. les ACACIAS bt. B3        =
0000            ;=        Av. de Saige              =
0000            ;=                      33600 PESSAC=
0000            ;=                                  =
0000            ;=     Reference : TOS COMMENTE     =
0000            ;=  livre edite et distribue par    =
0000            ;=                   IDC BORDEAUX   =
0000            ;=                                  =
0000            ;====================================
0000            ;
0000            ;
0000  @=F786 ANCTAB  EQU       0F786H          ;Table DCR d'origine.
0000  @=0297 TABADR  EQU       297H            ;Pointeur de la table de commandes.
0000  @=01B0 FILE    EQU       1B0H            ;Argument de la commande.
0000  @=0040 POROM   EQU       40H             ;copie de FD06 (commande cassette).
0000            ;
0000            ;
0000                    ORG       900H
0900            ;
0900            ;
0900            ;*****************************
0900            ;* Table des points d'entree *
0900            ;*****************************
0900            ;
0900  C30609            JMP       MARCHE          ;Activation des nouvelles commandes.
0903  C31309            JMP       ARRET           ;Desactivation.
0906            ;
0906            ;
0906            ;*************************************
0906            ;* Activation des nouvelles commandes *
0906            ;*************************************
0906            ;
0906  2A9702 MARCHE  LHLD      TABADR          ;HL=adresse de l'ancienne table.
0909  227109            SHLD      OLDTAB          ;elle est stoquee a la fin
090C            ;                                 ;de la nouvelle table pour que
090C            ;                                 ;les anciennes commande DCR
090C            ;                                 ;restent valides.
090C  215809            LXI H     TABLE           ;Met l'adresse de la nouvelle
090F  229702            SHLD      TABADR          ;table dans le pointeur de table.
0912  C9                RET
0913            ;
0913            ;
0913            ;*******************************************
0913            ;* Desactivation des nouvelles commandes *
0913            ;*******************************************
0913            ;
0913  2A7109 ARRET   LHLD      OLDTAB          ;Met l'adresse de l'ancienne
0916  229702            SHLD      TABADR          ;table dans le pointeur de table.
0919  C9                RET
091A            ;
091A            ;
091A            ;*************************
```

```
SPL V1.1  PAGE 2     Extension des commandes DCR

091A            ;* Effacement de l'ecran *
091A            ;************************
091A            ;
091A  3EFF      HOME    MVI A     0FFH          ;MODE 0.
091C  EF                RST 5
091D  18                DB        18H
091E  3E0C              MVI A     0CH           ;PRINT CHR$(12);.
0920  EF                RST 5
0921  03                DB        3H
0922  C9                RET
0923            ;
0923            ;
0923            ;*****************************************************
0923            ;* Manipulation des moteurs des cassettes audios *
0923            ;*****************************************************
0923            ;
0923  3AB001  MOTOR   LDA       FILE          ;A=argument de l'instruction.
0926  FEFF            CPI       0FFH          ;Si il n'y a pas d'argument,
0928  CA4609          JZ        MOTOR1        ;allumer le moteur 1,eteindre le 2.
092B  FE00            CPI       0H            ;Si l'argument est 0,
092D  CA4009          JZ        MOTOR0        ;etteindre les 2 moteurs.
0930  FE01            CPI       1H            ;Si l'argument est 1,
0932  CA4609          JZ        MOTOR1        ;allumer le moteur 1,eteindre le 2.
0935  FE02            CPI       2H            ;Si l'argument est 2,
0937  CA4C09          JZ        MOTOR2        ;allumer le moteur 2,eteindre le 1.
093A  FE03            CPI       3H            ;Si l'argument est 3,
093C  CA5209          JZ        MOTOR3        ;allumer les 2 moteurs.
093F  C9              RET                     ;Si l'argument est autre,ne rien faire.
0940            ;
0940  3E30    MOTOR0  MVI A     30H
0942  324000          STA       POROM
0945  C9              RET
0946            ;
0946  3E20    MOTOR1  MVI A     20H
0948  324000          STA       POROM
094B  C9              RET
094C            ;
094C  3E10    MOTOR2  MVI A     10H
094E  324000          STA       POROM
0951  C9              RET
0952            ;
0952  3E00    MOTOR3  MVI A     0H
0954  324000          STA       POROM
0957  C9              RET
0958            ;
0958            ;
0958            ;************************
0958            ;* Table des commandes *
0958            ;************************
0958            ;
0958  04      TABLE   DB        4H
0959  484F4D          DB        'HOME'
095D  1A09              DW        HOME
095F  05                DB        5H
0960  415252          DB        'ARRET'
0965  1309              DW        ARRET
0967  06                DB        6H
0968  4D4F54          DB        'MOTEUR'
```

```
SPL V1.1  PAGE 3     Extension des commandes DCR

096E  2309           DW        MOTOR
0970          ;
0970  00             DB        0H
0971  86F7   OLDTAB  DW        ANCTAB
0973          ;
0973          ;
0973                 END
```

MODE D'EMPLOI : faire CALLM#900 pour activer l'extension. HOME passe en MODE0 et efface l'écran, 'MOTOR n' allume le moteur de la cassette audio numéro n.Si n n'est pas tapé le moteur 1 est allumé,si n=0 tous les moteurs sont éteints, si n=3 les 2 moteurs sont allumés.

Il faut signaler pour la fonction HOME. qu'elle diffère de l'appui sur TAB par le passage en MODE 0 lorsqu'on est en graphisme .
La fonction MOTEUR utilise la possibilité d'argument allant de 0 à 9 offerte par le TOS, lorque l'on tape MOTEUR n,1B0 est chargé avec la valeur n si il n'y a pas d'argument 1B0 prend la valeur FFh .
Par cette méthode simple à mettre en oeuvre, on peut rendre l'utilisation des programmes en language machine plus facile qu'avec de simples CALLM .

NOTES A L'INTENTION DES POSSESSEURS DE KEN-DOS:

-La mémoire morte livrée en option sur KENDOS et contenant le TOS est la meme que celle livrée avec le DCR sauf qu'elle contient l'instruction DISK dans la table d'instructions en F7DF.

-Le programme HOME ci-desus fonctionne aussi avec la banque 0 du DOS co mmutée; mais il faut alors refaire un G900 à chaque commutation de banque car les adresses des tables DCR et DOS ne coincident pas forcément (à vérifier). Les possesseurs de KENDOS peuvent donc utiliser cette méthode en testant quelle banque est commutée et en pokant la valeur correspondant à la banque à la fin de leur table en RAM. Je n'ai pas testé sur KEN-DOS le passage des arguments pour MOTEUR.

Sébastien Dubourg, le 15 aout 1985
pour IDC Bordeaux.

Les maux du DAI : le clavier 3

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

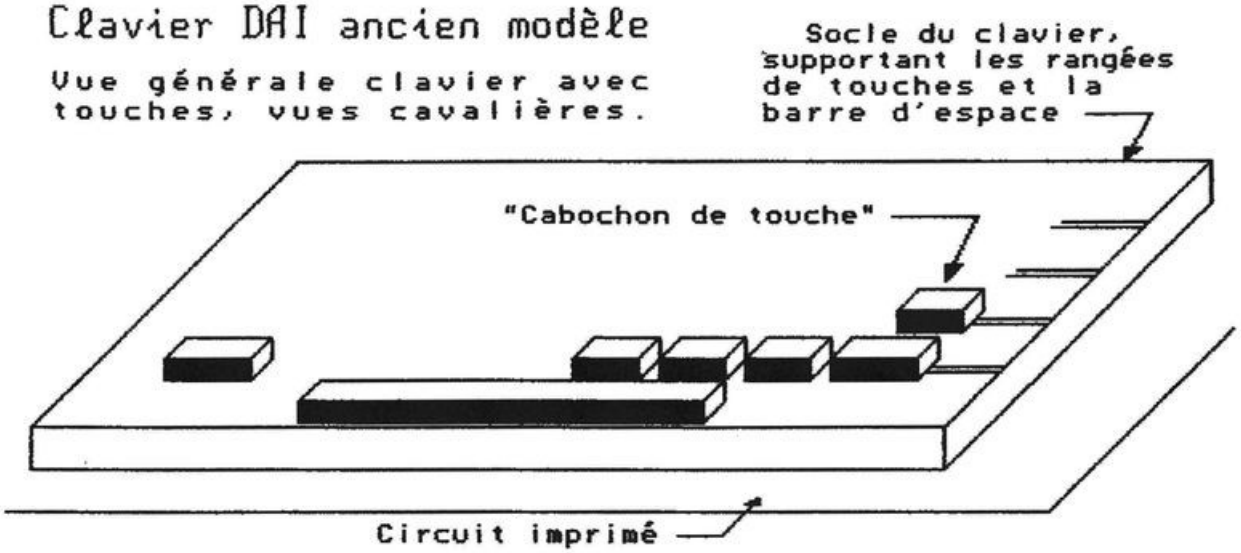
Série : le clavier 3

1ère Partie : l' Ancien clavier
ou 'A TOUT SEIGNEUR TOUT HONNEUR'

Dans le texte, les mots entre '' font référence aux annotations des schémas . Avant toute opération, TOUJOURS débrancher tous les fils et prises, puis enlever le capot blanc du DAI. Pour commencer le démontage, il convient tout d'abord d'enlever la partie portant l'inscription de la lettre, que j'appellerai par la suite 'le cabochon'. Sa couleur varie selon la fonction de la touche : grises ou noires pour les anciens et memes bleues, vertes et rouges pour les nouveaux .

Clavier DAI ancien modèle

Vue générale clavier avec touches, vues cavalières.



ATTENTION : c'est la première opération délicate ou vous risquez de casser du bois.

Mettez le DAI face à vous sur un plan horizontal bien dégagé comme si vous alliez programmer, puis munis de deux lames suffisament rigides, vous allez enlever les cabochons. Vous choisirez des lames dont l'épaisseur puisse passer entre le cabochon et le socle lorsque la largeur de la lame et sa longueur sont dans un plan horizontal (donc épaisseur dans un plan vertical) et dont la largeur soit supérieure d'environ 5mm à la distance séparant la partie inférieure du cabochon de la partie supérieure du 'socle'.

Le plus facile est de commencer par un coin dégagé (éviter les touches du milieu) par exemple le 'shift' en bas à droite (sans doute celui à gauche serait plus facile pour les gauchers et le coin supérieur pour nos lecteurs des Antipodes ... (it's a joke !!) .. bonjour NOUMEA ...) .

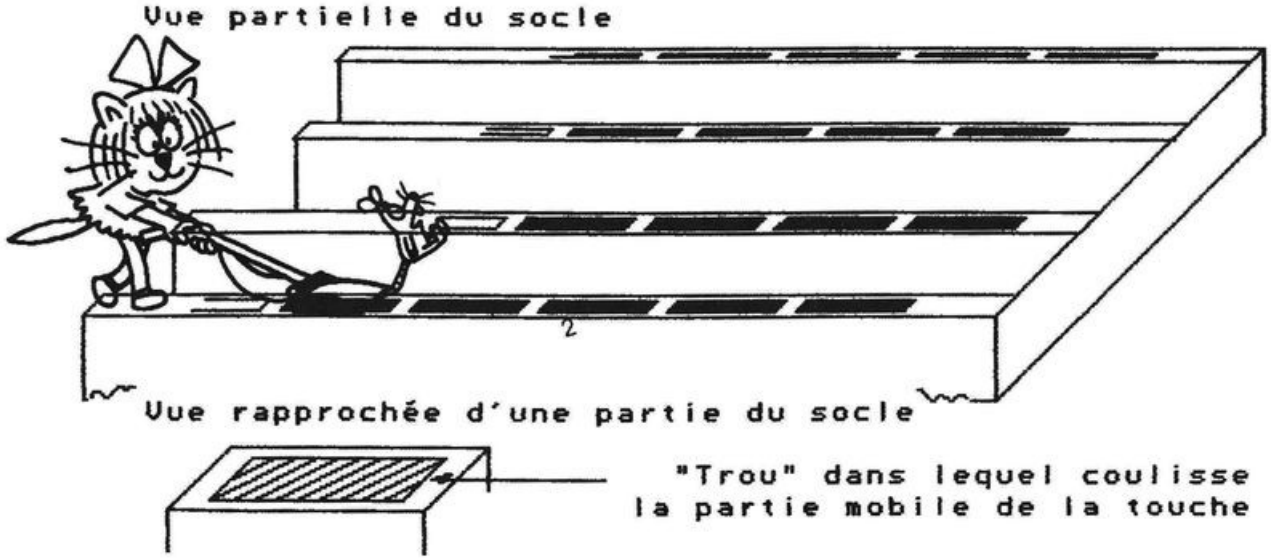
Vous immiscez donc chacune de vos deux lames (un tournevis fin faisant très bien l'affaire) à la base du cabochon, horizontalement , à angle droit par rapport au grand axe du clavier, donc droit devant vous de part et d'autre de la tige qui relie le 'socle' du clavier (en général de couleur marron) , au cabochon .

Vous exercez alors une pression STRICTEMENT verticale, de bas en haut, en faisant tourner le manche des tournevis, maintenu dans un plan horizontal le tournevis de la main droite tournant dans un sens horaire, celui de la main gauche dans un sens antihoraire. Ainsi la largeur de votre lame va venir s'appuyer d'un bord sur le bas du cabochon et de l'autre sur le haut du socle, poussant le cabochon vers le haut et le désemboitant de la tige qui le pénètre .

ATTENTION : tout manque de synchronisation entre vos deux mains va immanquablement faire dévier la pression qui DOIT rester verticale sous peine de casser la petite tige fort fragile à sa base .

Si un tel malheur vous arrivait, ce n'est pas dramatique, si vous ne cassez qu'une ou deux tiges. En effet les grosses touches (return et shift) sont fixées par deux de ces tiges et peuvent se contenter d'une seule. Vous pouvez donc casser au maximum trois tiges !!! grosses brutes s'abstenir ...

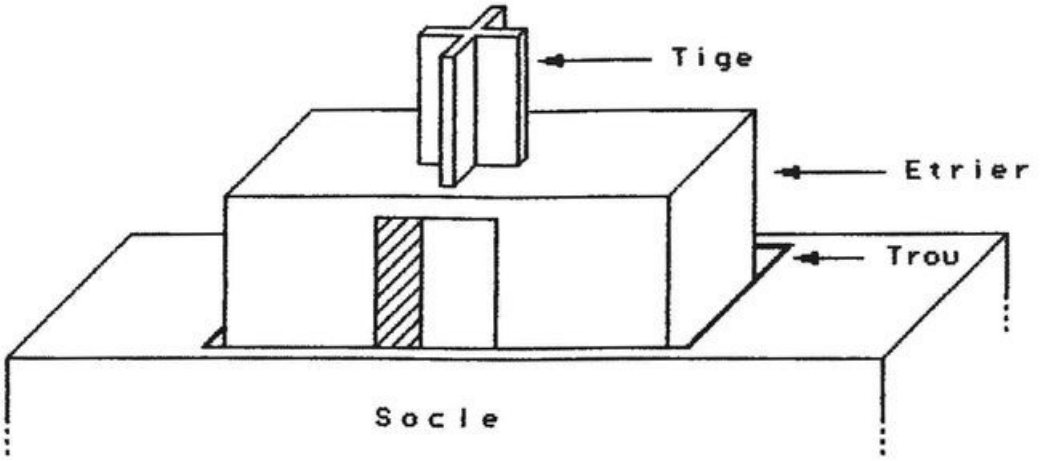
D' ailleurs pour ces trois 'touches doubles', une précaution supplémentaire est à prendre : il faut dégager progressivement un peu une des deux tiges puis un peu l'autre car si l'on extrait d'une seule traite une des deux tiges, meme en restant bien vertical, on fait subir à la tige voisine qui est solidaire du meme cabochon une contrainte oblique qui risque de la casser .



Donc de proche en proche, vous avez dégagé tous les cabochons, sans casser une seule tige et en ayant repéré, j'espère, l'emplacement des lettres !! Vous voila à pied d'oeuvre pour attaquer la partie vraiment délicate des opérations !!!

Vue de la partie mobile de la touche, dans le socle

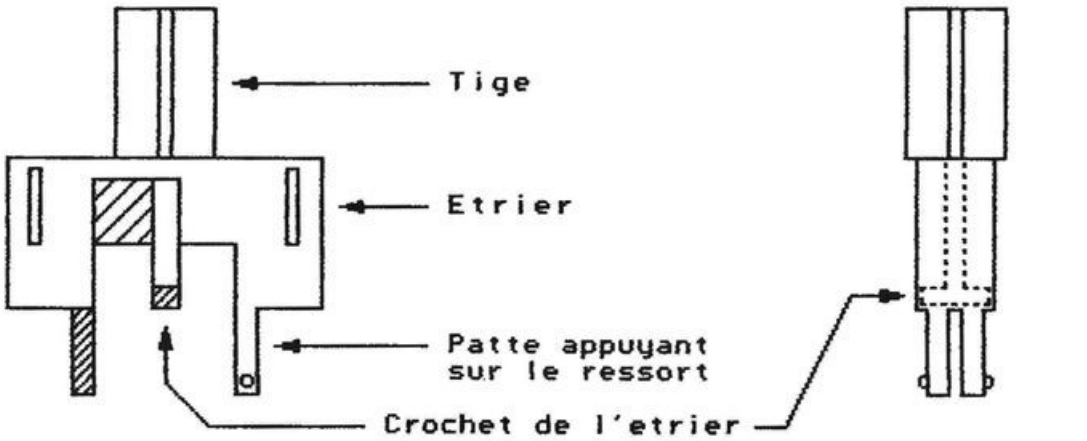
Vue cavalière



Partie mobile de la touche

Vue de face

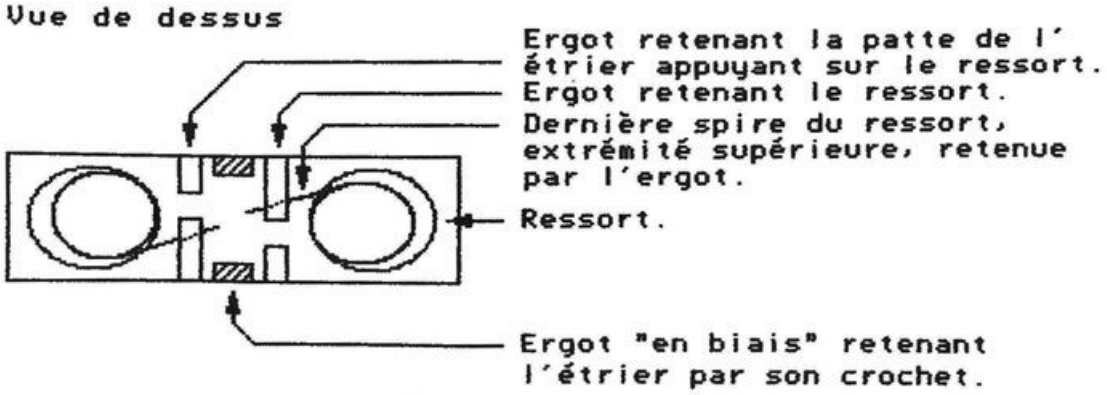
Vue de profil



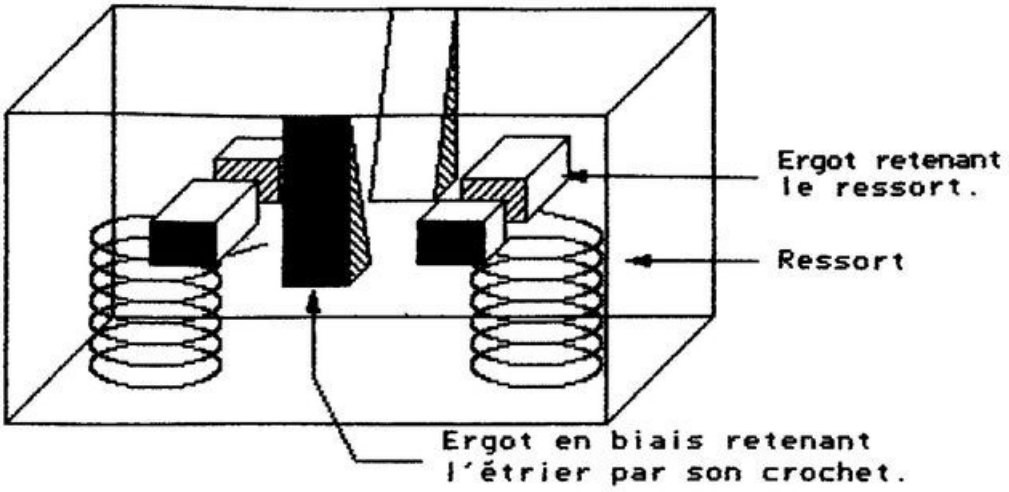
Son but est d'enlever du 'socle' la pièce mobile de la touche, constituée (cf.schémas) de 'la tige' (partie sup.) et de 'l'étrier' (partie inf.). Cet étrier vient appuyer sur deux ressorts verticaux contenus dans le socle grace à la partie nommée 'patte' sur le schéma qui font les contacts et dont l'encrassement provoque rebonds et mauvais contacts. Cet étrier comporte aussi des aspérités en forme de 'crochet' qui font butée et empêchent l'étrier de sortir du socle.

Vues du trou dans le socle, contenant les ressorts

Vue de dessus



Vue cavalière, en transparence



Le but de la manoeuvre va être d'écarter les parties du socle qui comportent les reliefs nommés ergots afin de dégager l'étrier sans casser la partie fragile (crochet), qui casse facilement si on tire brusquement sur l'étrier. Les différents reliefs étant symétriques, aussi bien sur la pièce mobile que pour le socle, il est inutile de repérer l'orientation de la pièce mobile quand vous l'enlevez car vous pouvez la remettre dans le sens que vous voulez .

Sur la 'vue en transparence' du trou du socle contenant l'étrier, vous verrez les différentes formes des ergots du trou :

- L'ergot central est le plus important car c'est surtout lui qui retient l'étrier. Sa forme est triangulaire afin de présenter une pente douce lors de l'introduction de l'étrier et une pente abrupte qui forme un cran d'arret avec la partie en 'crochet' de l'étrier .

- Les autres ergots sont surtout là pour guider les 'pattes' de l'étrier et bloquer la partie supérieure des ressorts qui est libre .

Pour cela, le mieux est de prendre deux lames (tournevis) plus fines que celles précédemment utilisées. Chaque lame sera insérée entre l'étrier et le socle, à hauteur de la tige, une lame au dessus , l'autre au dessous, de façon à écarter le socle en regard de l'ergot central pendant que, avec votre troisième main, vous tirez doucement sur la tige afin de dégager la partie mobile de la touche. Il vous faudra peut etre batailler un peu pour trouver le bon angle de levier à trouver pour dégager l'étrier sans peine sans casser la tige ni fendre le socle ... Mais une fois comprise, cette opération va assez vite .

Vous voila à présent devant les ressorts mis à nu .

Avec le plus grand soin et des pincettes, vous enlevez de ce trou béant, les conglomérats de poussière, les cheveux et autres miettes de pain. Vous pouvez meme étirer certains ressorts, s'il vous paraissent écrasés, à condition de ne réserver ce traitement qu'à ceux qui sont vraiment bien plus ramassés que les autres. Si vous tirez trop sur ces ressorts cela pourrait les faire se toucher en permanence comme si la touche était enfoncée (vous vous en apperce-vriez en appuyant sur 'repeat').

Jamais au cours de ces opérations, vous n'avez besoin d'un fer à souder et il est meme à proscrire. Une fois le trou propre vous pouvez alors si cela est vraiment nécessaire utiliser une bombe anti oxydation pour les contacts. Usez-en modéremment et assurez vous que les contacts sont secs avant le remontage !!

Pour remonter le clavier, rien de plus simple :

- enfoncer la partie mobile dans le trou du socle jusqu'à ce que les ergots emplissent leur office en encliquetant l'étrier .

- vérifier que les ergots retenant les ressorts contiennent bien l'extrémité libre du ressort .

- remettre le cabochon de touche .

ET C'EST FINI !!!

Vous pouvez alors profiter du plaisir dûment mérité de tapotter sur un clavier 'kazy 9'.

Bonne frappe .

Le président d'IDC Bordeaux:
Delannay Bruno

Petites annonces

Cherche correspondants pour échanges de programmes et documentation. Ecrire à Victor NIJS, Rue MARCHAND 5/7, B-4530 HERMALLE S/ARGENTEAU.

Cherche drives PRODATA 1 Mb avec contrôleur. Vends modem acoustique DATAPHON S21D. Paul CREMER, Rue E. MALVOZ 2, B-4610 BEYNE-HEUSAY.

Cherche DCR. Faire offre à Monsieur Michel LAMBRECHT, Rue J. VERCKRUYST 9, B-4530 HERMALLE S/ARGENTEAU.

Suite à un déménagement, je suis dans l'obligation de me séparer du meuble que j'avais fait faire spécialement pour le DAI (cf. l'article de IDC Bordeaux, dans DAICLIC 4) acheté 3000 FF, je le vends 2000 FF état neuf. Faites vite ! Bruno DELANNAY, Président IDC Bordeaux.

Vends carte PAL DAI: 3000 FB. Christian POELS (Rédaction).

A vendre Ken-DOS 2 x 800K + une quarantaine de disquettes (2ème plus grande programmathèque DAI...): 55000 FB à discuter. Cherche DCR d'occasion à bon prix. Faire offre à Marc VANDERMEERSCH, Avenue du VERT BOCAGE 17, B-1410 WATERLOO.

A vendre interface pour joystick de type Spectravidéo : 500 FB; paddle manche à balai avec 3 ème pot.et event: 500 FB; paddle 3 potentiomètres + event: 500 FB; DAI + DCR + documentation et programmes : prix souhaités 35000 FB (à discuter); DCR + 6 K7 bourrées de programmes : 13000 FB.
Faire offre à Fabrice DULUINS, Allée de la Tour Renard 4, B-1400 NIVELLES.

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.

Dossier assembleur : fin

****
M I C R O P R O C E S S E U R Final
****

JANVIER 1986

D.A.I.c.l.i.c

HP.LEGRY
628 Bd LAHURE
59500 DOUAI,FRANCE

****
* RECAPITULATIF *
****

Tableau recapitulatif des instructions assembleur du 8080A avec leurs code hexadecimal et leur signification.

MNEMONIQUES ET LEUR CODE HEXADECIMAL

I	ACI	CE	I	DCR L	2D	I	MOV C,M	4E	I	POP D	D1	I
I	ADC A	8F	I	DCR M	35	I	MOV D,A	57	I	POP H	E1	I
I	ADC B	88	I	DCX B	0B	I	MOV D,B	50	I	POP PSW	F1	I
I	ADC C	89	I	DCX D	1B	I	MOV D,C	51	I	PUSH B	C5	I
I	ADC D	8A	I	DCX H	2B	I	MOV D,D	52	I	PUSH D	D5	I
I	ADC E	8B	I	DCX SP	3B	I	MOV D,E	53	I	PUSH H	E5	I
I	ADC H	8C	I	DI	F3	I	MOV D,H	54	I	PUSH PSW	F5	I
I	ADC L	8D	I	EI	FB	I	MOV D,M	56	I	RAL	17	I
I	ADC M	8E	I	HLT	76	I	MOV E,A	5F	I	RAR	1F	I
I	ADD A	87	I	IN	DB	I	MOV E,B	58	I	RC	D8	I
I	ADD B	80	I	INR A	3C	I	MOV E,C	59	I	RET	C9	I
I	ADD C	81	I	INR B	04	I	MOV E,D	5A	I	RLC	07	I
I	ADD D	82	I	INR C	0C	I	MOV E,E	5B	I	RM	F8	I
I	ADD E	83	I	INR D	14	I	MOV E,H	5C	I	RNC	D0	I
I	ADD H	84	I	INR E	1C	I	MOV E,L	5D	I	RNZ	C0	I
I	ADD L	85	I	INR H	24	I	MOV E,M	5E	I	RP	F0	I
I	ADD M	86	I	INR L	2C	I	MOV H,A	67	I	RPE	E8	I
I	ADI	C6	I	INX M	34	I	MOV H,B	60	I	RPO	E0	I
I	ANA A	A7	I	INX B	03	I	MOV H,C	61	I	RC	0F	I
I	ANA B	A0	I	INX D	13	I	MOV H,D	62	I	RST 0	C7	I
I	ANA C	A1	I	INX H	23	I	MOV H,E	63	I	RST 1	CF	I
I	ANA D	A2	I	INX SP	33	I	MOV H,H	64	I	RST 2	D7	I
I	ANA E	A3	I	JC	DA	I	MOV H,L	65	I	RST 3	DF	I
I	ANA H	A4	I	JM	FA	I	MOV H,M	66	I	RST 4	E7	I
I	ANA L	A5	I	JMP	C3	I	MOV L,A	6F	I	RST 5	EF	I
I	ANA M	A6	I	JNC	D2	I	MOV L,B	68	I	RST 6	F7	I
I	ANI	E6	I	JNZ	C2	I	MOV L,C	69	I	RST 7	FF	I
I	CALL	CD	I	JP	F2	I	MOV L,D	6A	I	RZ	C8	I
I	CC	DC	I	JPE	EA	I	MOV L,E	6B	I	SBB A	9F	I
I	CM	FC	I	JPO	E2	I	MOV L,H	6C	I	SBB B	98	I
I	CMA	2F	I	JZ	CA	I	MOV L,L	6D	I	SBB C	99	I
I	CMC	3F	I	LDA	3A	I	MOV L,M	6E	I	SBB D	9A	I
I	CMP A	BF	I	LDAX B	0A	I	MOV M,A	77	I	SBB E	9B	I
I	CMP B	B8	I	LDAX D	1A	I	MOV M,B	70	I	SBB H	9C	I

I	CMP C	B9	I	LHLD	2A	I	MOV M,C	71	I	SBB L	9D	I		
I	CMP D	BA	I	LXI B	01	I	MOV M,D	72	I	SBB M	9E	I		
I	CMP E	BB	I	LXI D	11	I	MOV M,E	73	I	SBI	DE	I		
I	CMP H	BC	I	LXI H	21	I	MOV M,H	74	I	SHLD	22	I		
I	CMP L	BD	I	LXI SP	31	I	MOV M,L	75	I	SPHL	F9	I		
I	CMP M	BE	I	MOV A,A	7F	I	MVI A	3E	I	STA	32	I		
I	CNC	D4	I	MOV A,B	78	I	MVI B	06	I	STAX B	02	I		
I	CNZ	C4	I	MOV A,C	79	I	MVI C	0E	I	STAX D	12	I		
I	CP	F4	I	MOV A,D	7A	I	MVI D	16	I	STC	37	I		
I	CPE	EC	I	MOV A,E	7B	I	MVI E	1E	I	SUB A	97	I		
I	CPI	FE	I	MOV A,H	7C	I	MVI H	26	I	SUB B	90	I		
I	CPO	E4	I	MOV A,L	7D	I	MVI L	2E	I	SUB C	91	I		
I	CZ	CC	I	MOV A,M	7E	I	MVI M	36	I	SUB D	92	I		
I	DAA	27	I	MOV B,A	47	I	NOP	00	I	SUB E	93	I		
I	DAD B	09	I	MOV B,B	40	I	ORA A	B7	I	SUB H	94	I		
I	DAD D	19	I	MOV B,C	41	I	ORA B	B0	I	SUB L	95	I		
I	DAD H	29	I	MOV B,D	42	I	ORA C	B1	I	SUB M	96	I		
I	DAD SP	39	I	MOV B,E	43	I	ORA D	B2	I	SUI	D6	I		
I	DCR A	3D	I	MOV B,H	44	I	ORA E	B3	I	XCHG	EB	I		
I	DCR B	05	I	MOV B,L	45	I	ORA H	B4	I	XRA A	AF	I		
I	DCR C	0D	I	MOV B,M	46	I	ORA L	B5	I	XRA B	A8	I		
I	DCR D	15	I	MOV C,A	4F	I	ORA M	B6	I	XRA C	A9	I		
I	DCR E	1D	I	MOV C,B	48	I	ORI	F6	I	XRA D	AA	I		
I	DCR H	25	I	MOV C,C	49	I	OUT	D3	I	XRA E	AB	I		
I			I	MOV C,D	4A	I	PCHL	E9	I	XRA H	AC	I		
I			I	MOV C,E	4B	I	POP B	C1	I	XRA L	AD	I		
I			I	MOV C,H	4C	I			I	XRA M	AE	I		
I			I	MOV C,L	4D	I			I	XRI	EE	I		
I			I			I			I	XTHL	E3	I		

SIGNIFICATION DES MNEMONIQUES
ET DUREE DES INSTRUCTIONS en temps d'horloge

ACI	7	Addition immediate to A with carry
ADC M	7	Addition memory to A with carry
ADC r	4	Addition register to A with carry
ADD M	7	Addition memory to A
ADI	7	Addition immediate to A
ANA M	7	And A with memory
ANA r	4	And register with A
ANI	7	And immediate with A
CALL	17	Call unconditionnel
CC	11/17	Call on carry
CM	11/17	Call on minus (signe)
CMA	4	Compliment A
CMC	4	Compliment carry
CMP M	7	Compare memory with A
CMP r	4	Compare register with A
CNC	11/17	Call on no carry
CNZ	11/17	Call on no zero
CP	11/17	Call on positive
CPE	11/17	Call on parity even

I	CPI	7	Compare immediate with A
I	CPO	11/17	Call on parity
I	CZ	11/17	Call on zero
I	DAA	4	Decimal adjust A
I	DAD B	10	Additionn BC to HL
I	DAD D	10	Additionn DE to HL
I	DAD H	10	Addition HL to HL
I	DAD SP	10	Addition Stack pointer to HL
I	DCR M	10	Decrement Memory
I	DCR r	5	Decrement register
I	DCX B	5	Decrement BC
I	DCX D	5	Decrement DE
I	DCX SP	5	Decrement Stack pointer
I	DI	4	Disable interrupt
I	EI	4	Enable interrupt
I	HLT	7	Halt
I	IN	10	Input
I	INR M	10	Increment memory
I	INR r	5	Increment register
I	INX B	5	Increment BC
I	INX D	5	Increment DE
I	INX H	5	Increment HL
I	INX SP	5	Increment Stack pointer
I	JC	10	Jump on carry
I	JM	10	Jump on minus
I	JMP	10	Jump unconditionnal
I	JNC	10	Jump on no carry
I	JNZ	10	Jump on no zero
I	JP	10	Jump on positive
I	JPE	10	Jump on parity even
I	JPO	10	Jump on parity odd
I	JZ	10	Jump on zero
I	LDA	13	Load a direct
I	LDAX B	7	Load A indirect
I	LDAX d	7	Load A indirect
I	LHLD	16	Load HL direct
I	LXI B	10	Load immediate register pair BC
I	LXI D	10	Load immediate register pair DE
I	LXI H	10	" " " " HL
I	LXI SP	10	" " " SP
I	MVI M	10	Move immediate memory
I	MVI r	7	Move immediate register
I	MOV M,r	7	Move register to memory
I	MOV r,M	7	Move memory to register
I	MOV r1,r2	5	Move register to register
I	NOP	4	No-operation
I	ORA M	7	Or memory with A
I	ORA r	4	Or register with A
I	ORI	7	Or immediate with A
I	OUT	10	Output
I	PCHL	5	Transfert HL to PC
I	POP B,D,H	10	Pop registers BC,DE or HL off stock
I	POP PSW	10	Pop A and flags off stack
I	PUSH B,D,H	10	Push registers BC,DE or HL on stack
I	PUSH PSW	10	Push A and flags on stack
I	RAL	4	Rotate A left throug carry
I	RAR	4	Rotate A right throug carry
I	RC	5/11	Return on carry
I	RET	10	Return
I	RLC	4	Rotate A left

I RM	5/11	Return on minus I
I RNC	5/11	Return on no carry I
I RNZ	5/11	Return on no zero I
I RP	5/11	Return on positive I
I RPE	5/11	Return on parity even I
I RPO	5/11	Return on parity odd I
I RRC	4	Rotate A right I
I RST	11	Restart I
I RZ	5/11	Return on zero I
I SBB M	7	Substract memory from A With borrow I
I SBB r	4	Substract register from A with borrow I
I SBI	7	Substract immediate From A with borrow I
I SHLD	16	Store HL direct I
I SPHL	5	Transfert HL to SP I
I STA	13	Store A direct I
I STAX B	7	Store A indirect I
I STAX D	7	Store A indirect I
I STC	4	Set carry I
I SUB M	7	Substract memory from A I
I SUB r	4	Substract register from A I
I SUI	7	Substract immediate from A I
I XCHG	4	Exchange DE and HL I
I XRA M	7	Exclusive OR memory with A I
I XRA r	4	Exclusive OR register with A I
I XRI	7	Exclusive OR immediate with A I
I XTHL	18	Exchange HL and top of stack I

REPONSE au JEU-CONCOURS du DAICLIC No 5

Le seul problème de cet exercice venait de la manière dont le DAI affiche les nombres supérieurs à 99999. Il fallait donc faire quelques tests pour ajouter quelques zéros où cela était nécessaire !
Voici le programme qui en résulte :

```
IMP INT
1 FOR I=123456 TO 987654
2 I$=STR$(I)
3 I1$=MID$(I$,1,1)
4 I2$=MID$(I$,3,1)
5 I3$=MID$(I$,4,1)
6 IF I3$="E" THEN I3$="0":I4$="0":I5$="0":I6$="0":GOTO13
7 I4$=MID$(I$,5,1)
8 IF I4$="E" THEN I4$="0":I5$="0":I6$="0":GOTO 13
9 I5$=MID$(I$,6,1)
10 IF I5$="E" THEN I5$="0":I6$="0":GOTO 13
11 I6$=MID$(I$,7,1)
12 IF I6$="E" THEN I6$="0"
13 IF I1$=I2$ OR I1$=I3$ OR I1$=I4$ OR I1$=I5$ OR
   I1$=I6$ THEN 17
14 J$=I4$+I5$+I6$+I1$+I2$+I3$
15 J=VAL(J$)
16 IF (J/I)=INT(J/I) THEN PRINT "ELLE = ";I5$;I4$;I4$;I5$
17 NEXT I
```

On vérifie que TACLEF=142457 divise bien LEFTAC=857142.
Donc ELLE = 5885.

Pigeon agile

```
 PAGE 01  --  PIGEON AGILE

1       REM ***  LE PIGEON AGILE  ***
2       REM
3       REM ***  (c) TILT et F.G  ***
4       REM
5       REM
6       REM
7       GOSUB 9000
100     CLEAR 2000:GRD=1:GOSUB 8000:GOSUB 1000:GOSUB 2000:GOSUB 3000:GOSUB 4000

110     REM Initialisation du jeu
120     CHANCE=16:REM Parametre de difficulte
130     GOSUB 2100:GOSUB 1100

131     REM Deplacement du chasseur et de l'oiseau
140     IF INT(RND(CHANCE))=0 THEN GOSUB 1100
150     IF PIGE>0 THEN 130

151     REM S'il y a encore des oiseaux vivants
160     MODE 0:PRINT CHR$(12):POKE #75,95
170     IF PT=0 THEN CURSOR 3,12:PRINT "Vous n'aimez les oiseaux que dans votre
        assiette !!..":GOTO 200
180     IF PT<10.0 THEN CURSOR 4,12:PRINT "Vous avez vu 'Les oiseaux'
        d'Hitchcok, ca se voit...":GOTO 200
190     CURSOR 7,12:PRINT "BRAVO! Voila un defenseur des passereaux !!.."
200     COR=1:CURSOR 15,10:PRINT "Score, vous avez";PT;" points"
205     IF PIGE=0.0 THEN COR=0
210     CURSOR 15,9:PRINT "Il vous restait ";PIGE-COR;" oiseaux"
230     CURSOR 27,2:PRINT "Tapez  [ SPACE ]  pour continuer";
300     POKE #75,95
310     CALLM #D6DA
320     PRINT CHR$(12):GOTO 100
800     END

999     REM *** INITIALISATION DU CHASSEUR ***
1000    TIRX=20:TIRY=2:XB=20:YB=1:DXB=0:RETURN

1009    REM *** GESTION DU CHASSEUR ET DE SON TIR ***
1100    IF DXB=0 THEN IF RND(1)>0.5 THEN DXB=(XO-XB)
1110    CODE=0:X=XB:Y=YB:GOSUB 5000:XB=XB+SGN(DXB):DXB=DXB-SGN(DXB)
1115    IF XB=XO AND YB=YO THEN FIN=1
1116    CODE=3:X=XB:Y=YB:GOSUB 5000
1120    IF INT(RND(10))=0 THEN 1150
1130    IF TIRY<20 THEN 1150
1140    CODE=0:X=TIRX:Y=TIRY:GOSUB 5000:TIRX=XB:TIRY=YB+1
1145    CODE=2:X=TIRX:Y=TIRY:GOSUB 5000
1150    GOSUB 1300:CODE=0:X=TIRX:Y=TIRY:GOSUB 5000
1160    IF TIRY>20 OR TIRY>YO+1 THEN TIRY=20:RETURN
1170    TIRY=TIRY+1:IF INT(RND(CHANCE/4))=0 THEN TIRX=TIRX+SGN(XO-TIRX)
1200    CODE=2:X=TIRX:Y=TIRY:GOSUB 5000
1210    RETURN

1299    REM Regarde si l'oiseau est touche
1300    IF TIRX=XO AND TIRY=YO THEN FIN=1
1310    RETURN
```

```
1999    REM *** INITIALISATION DE L'OISEAU ***
2000    XO=6:YO=20:FIN=0
2010    RETURN

2099    REM *** GESTION DE L'OISEAU ***
2100    GOSUB 1300:CODE=0:X=XO:Y=YO:GOSUB 5000
2110    CL=#4001:CALLM #502D,CL:IF CL<>0 AND Y<20 THEN YO=YO+1
2120    CL=#4002:CALLM #502D,CL:IF CL<>0 AND Y>1 THEN YO=YO-1
2130    CL=#4004:CALLM #502D,CL:IF CL<>0 AND X<43 THEN XO=XO+1
2140    CL=#4008:CALLM #502D,CL:IF CL<>0 AND X>6 THEN XO=XO-1
2150    IF FIN=1.0 OR (XO=TIRX AND YO=TIRY) OR (XO=XB AND YO=YB) THEN GOSUB
        4200:GOSUB 2000
2160    CODE=4:X=XO:Y=YO:GOSUB 5000
2170    A=PEEK(ECRAN+XO*2+1):IF A=8 AND GRAIN=0 AND YO=1 THEN GRAIN=1:CODE=1:X=
        XO:Y=YO-1:GOSUB 5000:GRD=GRD-1:GOSUB 4080
2180    CL=#2020:CALLM #502D,CL:IF XO=6 AND YO=20 AND CL<>0 AND GRAIN=1 THEN
        GOSUB 4100
2190    RETURN

2999    REM +++ MISE EN PLACE DU DECOR +++
3000    PRINT CHR$(12):COLORT 0 10 0 0:POKE #75,32
3010    FILL 296,0 309,177 14
3020    PAS=20:X=0:FOR I=0 TO 40:DRAW 300-X,177+I XMAX,177+I 5
3030    PAS=PAS-1:X=X+PAS:NEXT
3040    FILL 297,160 302,168 9
3080    IF GRD=0 THEN GOSUB 4050:GOTO 160
3085    Y=0:FOR X=0 TO 5:CODE=1:GOSUB 5000:NEXT:GRD=0
3090    FOR X=6 TO 43:CODE=1:IF INT(RND(4))=0 THEN CODE=5:GRD=GRD+1
3100    GOSUB 5000:NEXT
3110    RETURN

3999    REM ... INITIALISATION DU SCORE ET DES OISEAUX ...
4000    PIGE=5:PT=0:CURSOR 0,2:PRINT "Score : ";PT
4010    CURSOR 0,1:PRINT "Nombre d'oiseaux : ";PIGE
4015    CURSOR 30,1:PRINT "L'oiseau dispose de";GRD;"  vers"
4020    RETURN
4050    WAIT TIME 200
4060    RETURN
4080    CUR=49:IF GRD<10 THEN CUR=50
4085    CURSOR CUR,1:PRINT GRD
4090    RETURN

4099    REM ... MISE A JOUR DES POINTS ...
4100    GRAIN=0:PT=PT+1:CURSOR 8,2:PRINT PT;
4110    IF GRD<1.0 THEN PIGE=PIGE+1:GOSUB 3080:GOSUB 4010:IF CHANCE>4 THEN
        CHANCE=CHANCE-4
4120    RETURN

4199    REM ... MISE A JOUR DES PIGEONS ...
4200    PIGE=PIGE-1:CURSOR 19,1:PRINT PIGE:GRAIN=0
4210    IF GRD<1.0 THEN GOSUB 3080
4220    RETURN

4999    REM ... AFFICHAGE EN MODE GRAPHIQUE ...
5000    AFF=(CODE*16+TABLE)*65536+ECRAN+Y*720+X*2+1
5010    CALLM #5000,AFF:RETURN
```

PAGE 03 -- PIGEON AGILE

```
7999    REM ... INITIALISATION DES PROGRAMMES ASSEMBLEUR ...
8000    MODE 6A:COLORG 9 11 5 14:ECRAN=PEEK(#88)+PEEK(#89)*256
8010    TABLE=#5100:FOR I=#5000 TO #5049:READ A:POKE I,A:NEXT
8020    FOR I=TABLE TO TABLE+16*6-1:READ A:POKE I,A:NEXT
8030    RETURN

8099    REM --- AFFICHAGE EN LANGAGE MACHINE ---
8100    DATA #F5,#E5,#D5,#C5,#11,3,0,#19,#5E,#2B,#56,#2B,#4E
8110    DATA #2B,#46,#26,8,#2E,2,#E5,#D5,#A,#12,3,#13,#2D
8120    DATA #C2,#15,#50,#D1,#21,#5A,0,#19,#EB,#E1,#25,#C2,#13
8130    DATA #50,#C1,#D1,#E1,#F1,#C9

8149    REM ---TEST DES TOUCHES ENFONCEES ---
8150    DATA #F5,#E5,#D5,#C5,#11,3,0,#19,#7E,#F3,#32,7,#FF,#3A
8160    DATA 1,#FF,#FB,#2B,#A6,#23,#77,#2B,#36,0
8170    DATA #C1,#D1,#E1,#F1,#C9

8197    REM TABLE DES CARACTERES
8198    REM --------------------
8199    REM Caracteres d'effacage
8200    DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0

8209    REM Caractere du carre plein
8210    DATA 0,255,0,255,0,255,0,255,0,255,0,255,0,255,0,255

8219    REM Dessin du tir du bonhomme
8220    DATA 16,16,16,16,16,16,16,16,16,16,16,16,16,16,16,16

8229    REM Dessin du bonhomme
8230    DATA 0,108,0,40,0,56,198,56,18,2,58,2,18,42,58,2

8239    REM Dessin de l'oiseau
8240    DATA 0,0,0,0,16,16,40,56,68,238,68,0,0,0,0,0

8249    REM Dessin du ver  (pour l'oiseau)
8250    DATA 8,255,4,255,24,255,32,255,0,255,0,255,0,255,0,255
9000    COLORT 5 15 0 0:PRINT CHR$(12)
9010    CURSOR 15,20:PRINT "L E  P I G E O N  A G I L E"
9015    PRINT TAB(15);"*****************************"
9020    PRINT :PRINT :PRINT TAB(14);"Vous disposez de cinq oiseaux,"
9030    PRINT TAB(14);"il s'agit de nourrir la nichee"
9040    PRINT TAB(14);"au moyen de vers . Mefiez-vous"
9050    PRINT TAB(14);"du chasseur et de son gourdin."
9060    PRINT :PRINT TAB(14);"Vous  manipulez  l'oiseau  au"
9070    PRINT TAB(14);"moyen des fleches. Tapez SPACE"
9075    PRINT TAB(14);"pour poser le ver dans le  nid"
9080    PRINT :PRINT :PRINT TAB(14);"                BONNE CHANCE !"
9090    CURSOR 30,2:PRINT "Pour commencer tapez SPACE";
9100    CALLM #D6DA:PRINT CHR$(12):RETURN
```

VIDEO BUG (suite et fin)

(Pascal JANIN, F-73 LA MOTTE SERVOLEX)

En réponse au petit problème vidéo dont nous faisait part E. Boucheron (DAICLIC 4, page 65), voici l'explication du phénomène rencontré.

On ne peut le qualifier de VIDEO-BUG, à la rigueur de VIDEO-ANOMALIE ou de VIDEO-DEMENCE (arrêtons de déconner merci !) ! En fait, cette anomalie apparente dans la vidéo en mode 16 couleurs peut selon les cas être très profitable ou emmerdante comme c'est pas permis. Je m'explique: après avoir parcouru d'un oeil attentif la page 65 relatant cette bizarrerie, j'ai décidé de voir si mon DAI s'avisait de réitérer la même chose. Eh bien oui ! Messieurs les DAIistes, soyez tous solidaires: le problème est commun à toutes vos chères machines !

Après ce long préambule, voici le comment et le pourquoi de cette fantaisie en mode 16 couleurs graphique. Les deux octets successifs définissant un bloc de 8 pixels contiennent d'une part l'organisation des points en 0 ou 1 (adresse A haute) et d'autre part la couleur des bits précédents à 1 et à 0 (dans cet ordre en 2 x 4 bits, adresse B basse). Ce que le manuel ne dit pas, c'est à quel point la gestion de la couleur associée au bit 0 est extravagante ! En effet, la couleur de fond définie en B ne prend effet qu'à partir du moment où un bit à 1 est placé en A ! Sinon, ce sont les couleurs précédentes du fond qui se propagent d'un octet à l'autre. Autrement dit, tant qu'une couleur de fond (bit à 0) ne recontre pas d'"obstacle" (bit à 1), elle se propage d'un octet à l'autre, sinon, c'est la nouvelle couleur de fond (définie avec celle du bit 1 rencontré) qui prend la place de l'ancienne à partir de l'"obstacle". Pigé ? Non ? Alors passons à la pratique...

Allumez votre DAI et rentrez l'exemple de la page 65. Jusqu'au POKE #BB01,#AA, tout marche bien. Exécutez le POKE suivant: un horrible point noir apparait à gauche ! Remède: faites un POKE #BB02,#AA: le point noir est remplacé par de l'orange. De plus en plus fort, faites POKE #BB01,0 et c'est la ligne toute entière qui est orange alors que (notez bien ! j'vais pas tout répéter) initialement la couleur du fond était définie en rouge !

Un petit schéma vaut mieux que toutes les explications du monde:

adresse	BB03	BB02	BB01	BB00
valeur	FF	E0	AA	13
affichage	J J J J J J J J		B R B R B R B R	
			(normal)	

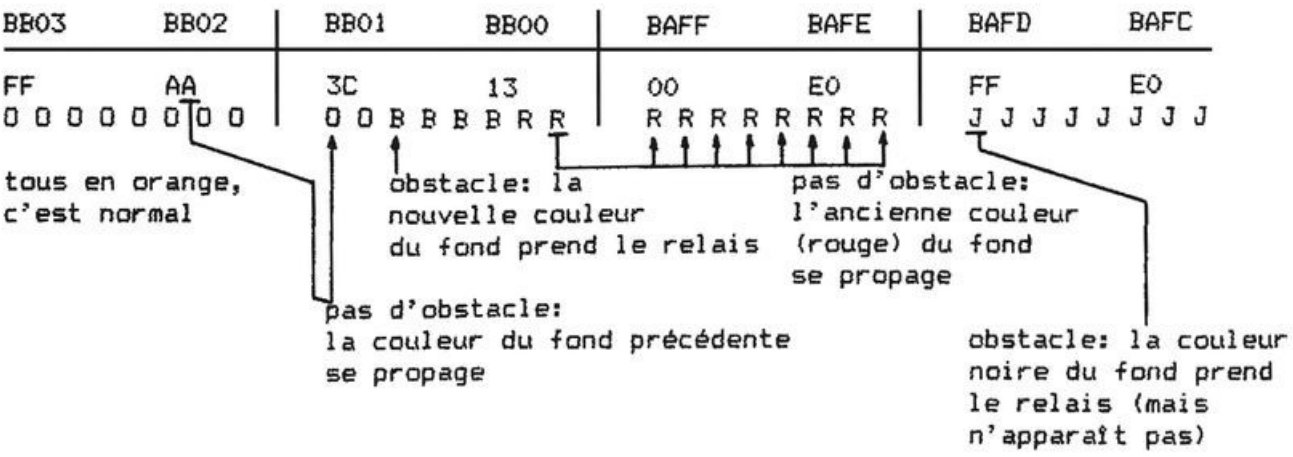
Après un POKE #BB01,#55

BB03	BB02	BB01	BB00
FF	E0	55	13
J J J J J J J J		N B R B R B R B	

couleur du fond précédente qui se "propage"...

obstacle: bit à 1 en BB01 ! la couleur du fond définie en BB00 prend le relais

Après un POKE #BB02,#AA:POKE #BB01,#3C:POKE #BAFF,#00



Vous comprenez à présent ? Oui ? Parfait !

Vous devez maintenant comprendre également pourquoi, en graphique 16 couleurs (avec un COLORG X A B C précédent), les adresses vidéo sont remplies de #FF (adresses hautes) et #X0 (adresses basses), plutôt que #00 et #0X (même effet) ? Parce que, dans le deuxième cas, la moindre tentative de changement de la couleur de fond dans une adresse basse entrainerait la mise au même état de toutes les couleurs de fond en aval ! (essayez pour voir...)

Pour finir, sachez que ce phénomène bizarroide, qui se reproduit sur UNE ligne en graphique 16 couleurs, sévit également sur TOUTES LES LIGNES HORIZONTALES DEFINISSANT UN CARACTERES en mode 16 couleurs/caractère ! Effets imprévus garantis !
A titre d'exemple, rentrez ce court programme BASIC qui affiche le texte "..ANOjpRMAL" en faisant varier les couleurs fond/caractère sur une ligne 16 couleurs caractère / basse résolution / hauteur maximale, et essayez de prévoir puis de comprendre le résultat ! C'est VIDEO-DEMENTIEL, non ???

```
IMP INT

10    PRINT CHR$(12);:COLORT 8 0 0 0
11    POKE #B921,#CF:AD=#B91F
13    READ D$:IF D$="^" THEN END
14    CALLM #D6DA:POKE AD,ASC(D$):AD=AD-3:READ D2:POKE AD,D2:AD=AD+1:GOTO 13
15    DATA ".",#15,".",#E3,A,#AF,N,#37,0,#18
16    DATA j,#2A,p,#40,R,#B9,M,#CD,A,#5E
17    DATA L,#42,^
```

Bonne compréhension... et bon amusement !

DAI QUI RIT (EXCLUSIVITE DAICLIC !):

C L A V I E R B A S I C

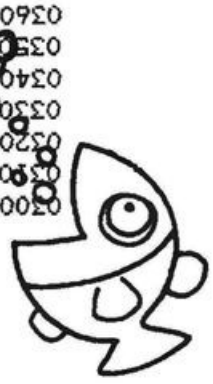
(Paul CALVEZ, PELUSSIN (F))

Voici un des plus courts utilitaires-machine qui permet d'avoir accès directement aux mots du Basic après avoir appuyé sur la touche <TAB> plus une autre touche. Il procure l'équivalent de confort des claviers multi-fonctions, qu'on trouve de plus en plus sur les O.I. familiaux, en évitant au programmeur de se tromper en tapant les ordres basic en toutes lettres. Comme la liste est brève (moins de 250 octets), la meilleure formule pour entrer le programme, consiste à se servir de la commande <S>ubstitue de l'Utility. C'est l'affaire d'un petit quart d'heure. Après cette (pénible) opération, et les vérifications d'usage, il reste à dévier le vecteur RCLOSE, afin que le programme soit rendu auto-start. Ceci se fait en plaçant un JMP 03B0H en 2D4H : toujours avec le <S> du moniteur, faire S2D5 nn-B0 nn-03, et le tour est joué. A la fin du chargement, les pointeurs basic sont ainsi automatiquement ajustés, le vecteur RST 1 <Utility/encode> est détourné vers le programme.

ATTENTION : il faut sauver la routine avant de la lancer. Faire, sous moniteur : W2D4 3D6 CLAVIER BASIC. En effet, la routine d'initalisation est détruite après son fonctionnement.

Pour l'utilisation, il suffit d'appuyer sur <TAB> quand on veut écrire un mot-clé basic. Le curseur se change alors en un 'pavé', signalant que la routine attend la touche assignée au mot-clé désiré. Quand cette dernière est frappée, le mot basic est affiché, et le curseur reprend son aspect habituel. Il ne vous reste plus qu'à découvrir les mots qui se cachent désormais sous chaque touche de votre clavier. Le mieux est de les essayer toutes, une à une !

```
03D0  03 22 64 00 E1 C9
03C0  21 B0 04 22 9F 02 23 22 A1 02 23 22 A3 02 21 73
03B0  E5 CD 45 D4 21 45 D4 22 D5 02 21 B0 03 22 98 02
03A0  4E 2E 71 71 2B 70 CD D4 DA 2E 73 3E 5F C3 83 03
0390  0E C7 F5 05 7B 07 07 07 07 81 81 6F 26 03 46 23
0380  92 3E 7E 32 75 00 01 05 02 1E B6 22 64 00 F1 C3
0370  8C 2A 00 F5 79 D6 04 C2 8E 03 7B 3D C2 8E 03 2E
0360  88 BF 8C 7A 8C 81 8C 89 8C 5C 8C 4A 90 1E 8C 53
0350  8D 56 8D 03 8D 12 8F CF 8C 98 8F C1 8C 91 8D 2B
0340  88 E8 8F F3 8C 01 8F F9 8C AE 8C 42 8C D2 8B F9
0330  90 2B 8C A5 8C C5 90 25 8D 1B 8C 72 8C D9 8F D4
0320  90 7E 8C 2A 8C B6 8D 23 AE 15 8C 68 90 4D 8D 81
0310  90 00 90 06 88 CE 90 99 90 93 8C 3A 8B F2 8B DD
0300  80 01 1B 90 6C 3E 90 89 90 BA 8D 09 8C 7A 8C CB
```



NOUVEAUTES POUR LA CARTE XBASIC

(Christian POELS, janvier 1986, d'après un courrier de Uwe WIENKOP)

Jusqu'à présent, la carte XBASIC de H. Tegethoff ne permettait d'utiliser que les drives Commodore 1541 (ou les cassettes audio...).
Maintenant, que tous les possesseurs de drives Prodata (ou Indata pour ceux qui sont en retard d'une guerre (ou DAI pour ceux qui le sont de deux...)) se réjouissent: La carte est compatible avec leur système !
Mieux... Elle est compatible avec les deux systèmes de disquettes: Prodata et Commodore ! De plus, sachez qu'il est même possible (pour les plus fous...) de connecter SIMULTANEMENT sur votre cher DAI: 4 drives Prodata + 4 drives Commodore + 4 DCR (et oui, ça marche aussi maintenant avec les DCR !) + 2 enregistreurs audio !!!... Pour vous prouver que cette configuration est tout à fait valable, il existe un programme de copie permettant de passer du système Prodata vers le Commodore et l'inverse ! C'est quand même une bonne nouvelle pour ceux que la compatibilité obsède ! Ce premier pas sera bientôt suivi d'autres: Une adaptation du Ken-DOS sur cette carte est en préparation... La carte XBASIC constitue donc un évènement pour le DAI dont les différentes mémoires de masses proposées devenaient de moins en moins compatibles

Les nouvelles instruction du Xbasic pour les drives Prodata sont:

```
#4: DOS - Befehle:
    DISK CAS DIR . EDIR SDIR DLOAD DSAVE DRUN RDSK WDSK
    RSEQ WSEQ OPEN CLOSEA CLOSE GETDIR FIELD GET# PUT# GET
    PUT DEL CREATE LOCK UNLOCK RENAME SIZE DATE FORMAT
    BACKUP VERIFY
#5: COPY TYPE RESETD MESG MAKEJOB DOJOB TSAVE ID SEEK DROP
    FREE
    DISKOUT= DISKIN=
#6: TED - tools  (#12F000)
    BAS>TED BAS.CLOSE TED>DTV TED>REM TED>BAS TED>CPM
    CPM>TED BCKP
    LOW= SPOOL= SPARM=
```

La plupart des instructions des canaux #4 et #5 ressemblent très fort aux instructions disponibles dans le package UDOS de Prodata. Mais voici les différences notables:

L'UDOS sur EPROM n'utilise pas du tout l'espace RAM utilisateur. Par conséquent, vous pouvez utiliser les programmes LM sans aucune restriction mémoire, que vous aviez par contre avec le DOS 3.0 et avec l'UDOS en RAM. De plus, vous ne devrez plus attendre le chargement du système DOS, car il est déjà sur EPROM...

Différences dans les instructions:

DIR / EDIR / SDIR exécutent une instruction DIR. Le point '.' est une abréviation de l'instruction DIR. Il y a deux possibilités:DIR/EDIR/SDIR avec un nombre (ou sans argument: la signification est: DIR du drive par défaut) provoque un DIR normal ou un DIR étendu (EDIR) ou un super DIR (SDIR). EDIR et SDIR affichent des informations supplémentaires sur les fichiers. Par exemple: le type des fichiers de données (INT, FPT, STR), les adresses de début et de fin des programmes 'machine' et aussi le nombre de lignes des fichiers 'source' AHT (seulement SDIR).

La deuxième possibilité est: il est possible de choisir un masque pour les fichiers (cette possibilité n'est pas implémentée dans la version de Prodata !). Par exemple, vous souhaitez n'afficher que les programmes BASIC sur le disque dans le drive 2: DIR "*.BAS" (idem pour EDIR et SDIR).
D'autres exemples:
DIR "DTV *.*"Y - tout ce qui commence par DTV. Le type ne joue pas.
EDIR "D??-doc.BAS" - les 2è et 3è caractères du nom sont ambigus (cf. CP/M).

```
DIR "*.DBS:2":

      DISK NAME: Texte / DBS   DISK DATE: 851014

          DAI-Adr .DBS  R/O    >16.000
          Adressen.DBS           2138
          PrgNamen.DBS         >16.000
          Buecher .DBS           7814
          Software.DBS          10706
          VIP-Adr .DBS           1418
          FREE DISK SPACE:  56 K
```

EDIR et SDIR calculent la longueur réelle des fichiers, tandis que DIR essaie de calculer la longueur des fichiers d'après les informations données dans la directory et cette valeur n'est valide que pour les fichiers jusqu'à 16 K. Ceci est une question de vitesse. Pour les plus grand fichiers, vous aurez la réponse >16.000. Si vous voulez obtenir la longueur réelle, même pour les grands fichiers, vous devez utiliser EDIR ou SDIR:

```
      DISK NAME: Texte / DBS   DISK DATE: 851014

          DAI-Adr .DBS  R/O      24366
          Adressen.DBS            2138
          PrgNamen.DBS           18637
          Buecher .DBS            7814
          Software.DBS           10706
          VIP-Adr .DBS            1418
          FREE DISK SPACE:  56 K
```

DEL, LOCK, UNLOCK, RENAME et COPY fonctionnent de la même façon: vous pouvez utiliser "*" et "?" pour les noms de fichiers ambigus, comme en CP/M: par ex.:
DEL "*.BAS", LOCK "*.*"

RENAME permet aussi d'utiliser les "wildcards":
RENAME "*.ARR","*.SRC". La première astérisque a le même rôle que dans un DELete. La deuxième dit au DOS, que cette partie du nom doit rester la même que dans le nom de fichier originaL. Par ex.: RENAME "*.ARR","*.SRC":

```
          AHT     .ARR        =>        AHT     .SCR
          DTV-1   .ARR        =>        DTV-2   .SCR        ainsi de suite...
```

COPY permet les mêmes possibilités que dans RENAME:

```
COPY fichier$,fichier$
COPY fichier$
COPY "*.*"          ainsi de suite...
```

Si vous n'utilisez qu'un paramètre: le fichier est copié depuis l'autre drive 0 -> 1, ou 1 -> 0. Il faut préciser à la commande de la carte dans quel ordre on utilise les drives. Ex:

```
          0/1 drive de gauche          2/3 drive de droite
ou:       0/2 drive de gauche          1/3 drive de droite
```

La touche crochet (parenthèse carrée) ouvrant est une abréviation de DRUN "FILE". Il suffit de taper (FILE (sans quotes) !

La commande FREE affiche la taille de la place libre sur le drive spécifié ou le drive par défaut.

Les autres commandes sont les mêmes que dans le UDOS PRODATA.

Les commandes du canal #6 (les outils TED) permettent de traduire des fichiers BASIC en format fichier TED ou en format TED lisible via CP/M. Par conséquent, il est possible de traduire des fichiers BASIC DAI pour travailler sous MBASIC ou BASCOM...

Il est aussi possible de commander en supplément, une RAM de 8K qui vient s'ajouter sur la carte XBASIC. Cette RAM peut être utilisée comme buffer d'imprimante, et est activée via la commande SPOOL = parm1 (,parm2).

Le premier paramètre contient le nombre de banques RAM que vous voulez utiliser pour le tampon: 1..4 et le second paramètre (optionnel) contient un nombre qui représente le type de sortie du buffer:

0 - normal (par défaut)	sortie aussi sur écran,	fin de ligne: 0D0A
1 -	sortie uniquement sur buffer,	fin de ligne: 0D0A
2 -	sortie aussi sur écran,	fin de ligne: 0D
3 -	sortie uniquement sur buffer,	fin de ligne: 0D

ou SPOOL = OFF désactive le buffer. POKE #131,X (X>=3) dir ige les sorties vers le spooler !

Cette fonction augmente de 25 DM le prix de la carte.

Dans le package UDOS se trouvent deux disques utilitaires. Ces programmes se trouvent aussi sur EPROM ! Les programmes sont DUMP et SCOPY. Ils permettent une recherche facile d'informations sur le disque.

SCOPY est un super COPY. Ce programme vous permet de copier des fichiers depuis le drive 0 vers le 1 ou vers une cassette ou un drive Commodore. Le programme lit la directory complète du drive 0.

Vous pouvez ensuite faire défiler la directory sur l'écran et marquer les fichiers que vous voulez copier. Vous pouvez aussi choisir l'ordre dans lequel vous voulez les copier. Lorsque vous avez marqué tous les fichiers à copier, vous pouvez démarrer la procédure de copie. Et maintenant, tous les fichierssont copiés dans l'ordre souhaité. C'est très pratique si vous voulez copier quelque chose sur cassette: vous placez tous les fichiers sur une disquette et ensuite, vous utilisez SCOPY pour les recopier en un bloc. Vous n'êtes pas obligé de rester devant l'ordinateur et d'attendre la fin de chaque action de copie. Ce programme tourne aussi sur drive Commodore et il faut préciser que lors d'une copie, les fichiers sont copiés par blocs de telle façon que la RAM du DAI soit utilisée au maximum. Cela réduit donc le nombre de manipulations dans le cas d'une copie de disquette vers disquette avec un seul drive.

Revenons au système PRODATA: une disquette est aussi fournie avec deux (petites) routine de BOOT et avec quelques autres programmes. La procédure de BOOT est nécessaire car le DAI envoie un signal lors d'un RESET au bus DCE et le floppy répond en lisant le fichier $BOOT. Mais les fichiers EXEC.PRL ne sont plus du tout nécessaires. Le second fichier $BOOT.USR permet d'effectuer un autoexecute d'un fichier appellé "$$$.BIN".

Les autres programmes:
- un programme permettant un DIR (avec tri) sur imprimante en deux colonnes
- un tri de directory et un compresseur de programmes
- un programme qui vérifie si certains blocs sont utilisés doublement sur le disque (cela arrive sous certaines conditions, par ex., si vous changez le disque dans le drive sans un RESETD ! - un bug dans le slave DOS).
- etc...

Comment se procurer la carte XBASIC version drives Prodata ?: Tout d'abord, il faut se procurer (si ce n'est déjà fait) les drives Prodata et posséder la facture de ces drives, ou au moins du "UDOS". Vous envoyer alors votre commande en n'oubliant pas de joindre votre facture (ou une photocopie certifiée conforme...) à l'adresse suivante:

Mr. Uwe WIENKOP
Laerfeldstr., 54
D-4630 BOCHUM (R.F.A.)
T. (0234) 35 61 32

Les prix (à titre indicatif, se renseigner pour les éventuels changements):

XBASIC	287	DM
ComDOS	9	DM (facultatif)
EPROM UDOS	70	DM
RAM SPOOLER	25	DM (facultatif)

Dernière minute: le CP/M fonctionne maintenant sur la carte Xbasic + drive 1541, et permet entre autres, l'accès à la programmathèque CP/M du Commodore 128 !!! Plus de renseignements dans le prochain numéro... (ou sur les serveurs MN2...)

DAICLIC the best for the DAI !

Le réseau belge inter-clubs

"LE" RESEAU BELGE INTER-CLUBS

(Christian POELS, d'après le "Micro-GDV-Press" (1))

Suite à un accord intervenu entre plusieurs clubs belges de micro-informatique, il a été décidé de n'implémenter qu'UN SEUL RESEAU de transmission de données, quelque soit le nombre de clubs et quelque soit le nombre de "SERVEURS" télématiques.

Dans un premier temps, le RESEAU sera wallon et bruxellois, mais rien n'empêche de l'étendre au nord du pays si les clubs wallons sont d'accord et si nos amis néerlandophones le demandent.

Actuellement, les clubs suivants sont d'accord de participer à l'idée:

AMMI / A.I.D.E.S. (LIEGE)
MICRORDI (LIEGE)
C.R.I.V. (VERVIERS)
IDC (INTERNATIONAL)
MOBIL-CLUB (WAVRE)
MICRO-G.D.V. (MICRONET-2) (VERVIERS-LIEGE-BRUXELLES-PARIS)
MICRO-NET-3/4 - WAREMME
MAC BBS (BRUXELLES)
liste non limitative...

Dans la phase "1" (actuelle), seul le transfert des boites postales est opérationel. Chaque utilisateur doit se servir DU MEME PSEUDONYME D'ACCES dans chaque SERVEUR afin de ne pas trop compliquer le "SOFT NOCTURNE" qui doit transférer automatiquement les messages et réponses de chacun, dans son serveur habituel. Ceci est destiné à nous faire gagner du temps et... de l'argent en limitant les appels "interzonaux".

LE "RESEAU WALLON" est surtout actif LA NUIT ! afin de ne pas encombrer les lignes téléphoniques de la R.T.T. et de bénéficier du tarif de taxation réduit. Les SERVEURS "s'appellent" automatiquement, vers 3 ou 4 heures du matin, afin de s'échanger des données (Ordinateur "dispatch").

Un software fonctionnant "en étoile" est opérationel pour tous les serveurs Micro-Net-2.

Si vous avez des suggestions et/ou des idées performantes au sujet du "SOFT-NOCTURNE", contacter notre ami:

Jean PUELINCKX
Rue de la Liberté, 2
B-4840 WELKENRAEDT (BELGIQUE)
T. (0)87/882226

Merci d'avance de votre collaboration.

NB: Pour ceux qui ne possèdent pas encore de modems, ils peuvent se renseigner auprès de la rédaction pour en obtenir à des prix avantageux ! (possibilité aussi de le commander par la poste) De plus, le logiciel "DaiLink V1.1" (distribué par IDC) convient parfaitement pour les connexions sur les serveurs.

SERVEURS MicroNet2 (tous fonctionnent 24h/24 !):

```
* MicroNet2 (LIEGE     "A"):  041/  796666
  MicroNet2 (LIEGE     "B"):  041/  312131
  MicroNet2 (LIEGE     "C"):  041/  582573
* MicroNet2 (BRUXELLES "A"):  02/  2427008
  MicroNet2 (BRUXELLES "B"):  02/  5131111
* MicroNet2 (PARIS     "A"):  331/39718291
  MicroNet2 (VERVIERS  "A"):  087/  883434
  MicroNet2 (VERVIERS  "B"):  087/  883131
```

Les serveurs précédés de '*' contiennent l'index IDC qui fournit toutes sortes d'informations sur le DAI et IDC: programmes, articles, messages, etc. Si la demande le justifie, d'autres serveurs MICRONET2 s'équiperont de cet index.

(1): "Micro-GDV-Press" est une publication mensuelle du club Micro-GDV.

Logiciel : Aladin

A L A D I N

Article original : DAInamic Allemagne
Traduction : Frédéric Bacquet

Voici un nouveau logiciel : "Adventure-Interpreter ALADIN" de Marco van Meegen (soft original de Scott Adams sur TRS 80) proposé par le club DAInamic Germany. Il s'agit d'un programme avec lequel on peut mettre en mémoire des aventures s'éxecutant rapidement puisqu'écrites en langage machine. Il permet une gestion compacte, et donc la création de nombreuses conditions, objets et structures de langage, qui prennent beaucoup moins de place qu'un programme Basic offrant les mêmes possiblités. Aladin offre aussi en version de base la posssibilité d'interrompre et de redémarrer un jeu. Sa technique de traitement de données permet la création de: 254 pièces, 255 verbes et sujets, 256 objets, 8 directions, 65536 messages (texte), 252 flags pour les conditions, 256 compteurs et de nombreuses relations, donc des liaisons (conditionnelles) entre tous les objets. On peut donner des verbes et sujets abrégés. Celui qui veut lui-même écrire une aventure peut utiliser les 32 tests différents, et gagner beaucoup de place en gardant les actions standards (GET/DROP).

Il ne faut désormais plus programmer une aventure sans ALADIN. En plus du programme, 7 aventures prêtes à jouer :

Adventureland
Burglar'adventure
Pirate's adventure
Fun house
Miner's adventure
Savage island
Strange odyssey

Il s'agit pour la plupart de scénarios originaux de Scott Adams, mais repensés et améliorés. Les aventures sont de niveaux de difficulté variables: par exemple, "Adventureland" est accessible aux débutants, alors que "Savage island" est destiné aux aventuriers expérimentés. Le prix de l'interpréteur, avec 7 aventures et 10 pages de doc.: 30 DM. Vous pouvez le commander via IDC en envoyant la somme de 760 FB. Le mode d'emploi est en allemand et le programme en anglais ou allemand.

Toujours plus de rom

COMMENT AVOIR PLUS D'EPROM DANS UN DAI

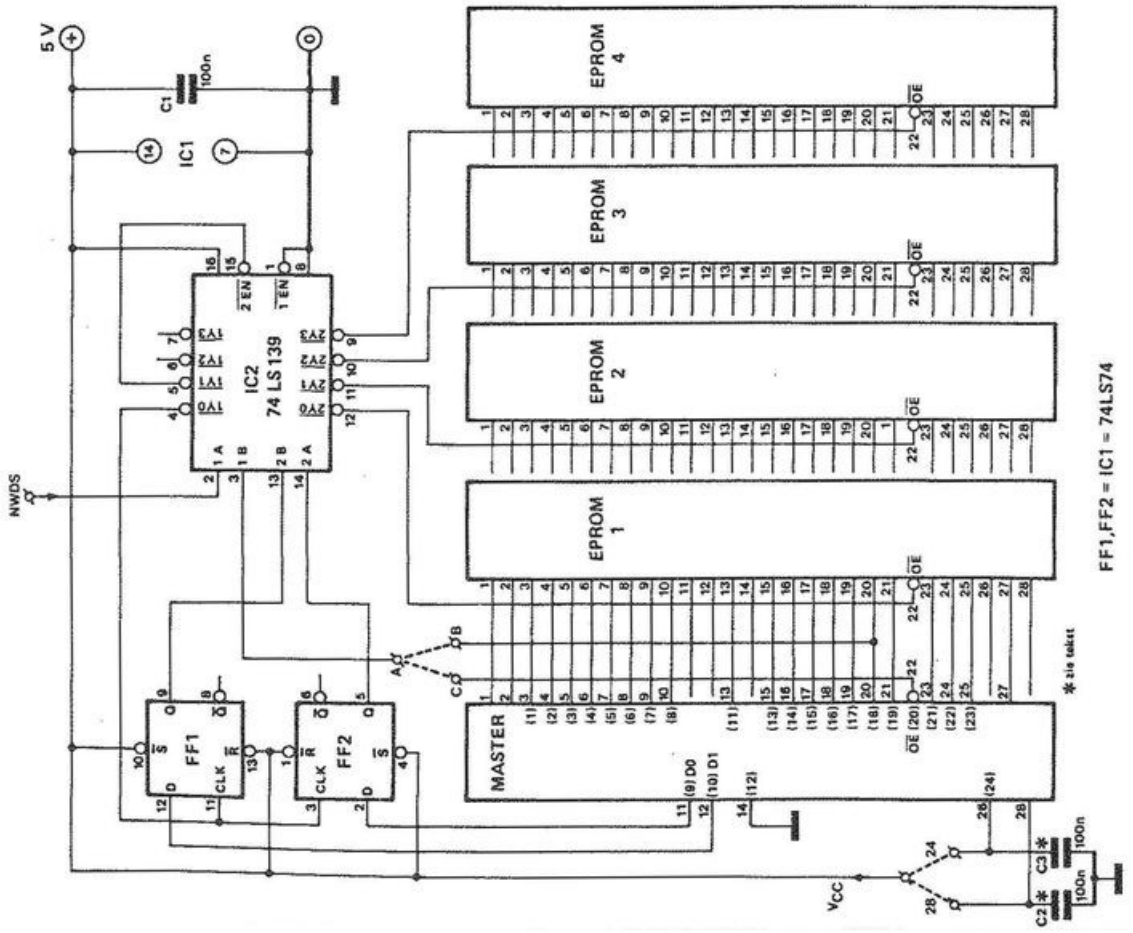
Voici comment, d'une façon simple il y a moyen d'étendre la mémoire EPROM si vous possédez déjà une carte TOS pour votre MEMOCOM. Il y a quelques mois, ELEKTUUR a publié une petite carte d'extension permettant de faire un banking de mémoire avec 4 EPROMs.

Le schéma est assez simple et ne demande pas beaucoup de mots. Quand on adresse la mémoire où se trouve l'EPROM de base (pour le DAI: F000-F7FF), on adresse en fait une des EPROMs selectionnée par le 74S138. C'est en effet la valeur à son entrée qui détermine si on adresse l'EPROM 0, 1, 2 ou 3. A l'allumage ou reset, c'est d'office l'EPROM 0 qui est sélectionnée. Mais comment faire pour lire une des EPROMs 1, 2 ou 3? C'est simple: en écrivant à une adresse entre #F000 et #F7FF, une valeur X (où X = 0, 1, 2 ou 3 selon l'EPROM choisie). Mais attention, avant d'écrire dans une adresse de la zone #F000 - #F7FF, il faut d'abord mettre la valeur #C9 à l'adresse #10 pour déconnecter la génération d'erreurs du STACK OVERFLOW.

Le montage de la carte est assez simple et ne nécéssite pas d'outils spéciaux. Mais attention, il ne faut pas oublier de raccorder la borne NWDR à la borne R/W(neg) du 8080 (borne 18 du 8080).

Ci-joint, le schéma de la carte d'ELEKTUUR (numéro de février 1985).

Dominique CARLIER,
B-1500 HALLE,
Membre du DAIC.



Erratum

Aan het begin van het vorige artikel kwam een programmeerjuweeltje voor waar toch een verontreiniging in bleek te zitten.
Het 'kleiner dan'-teken in regel 100 had natuurlijk een 'groter dan'-teken moeten zijn zoals hopelijk de meeste lezers uit het bijstaande verhaal duidelijk zal zijn geworden.
Dit maal ben niet ik de schuldige zoals sommigen al gniffelend meenden.
Door het gebruik van een fraaie daisywheel printer (margrietwiel afdrukeenheid voor puristen) is de leesbaarheid van o.a. mijn artikel sterk verhoogd. Deze printer heeft echter niet al de mogelijkheden van een matrixprinter. Hij kent b.v. maar 96 karakters op het bloempje en zo kunnen dan ook niet alle symbolen correct weergegeven worden.
De redactieleden zijn er dan ook extra op bezig om de teksten vlekkeloos op papier te krijgen.
Dit lukt dus blijkbaar niet altijd en ook in het laatste programma'tje is de test op ongelijk nul in regel 50 er enigzins verwrongen uitgekomen.

Voor diegenen echter die alleen de programma's intikken volgt hier het hopelijk nu smetteloze sieraad.

```
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
```

Rekenopzet

Zoals in de laatste twee programma's uit de vorige aflevering reeds bleek kan het nuttig en soms zelfs noodzakelijk zijn om de rekenvolgorde van te voren te overdenken.
We moeten ons bedenken dat zolang er geen haakjes gebruikt worden er bij gelijke prioriteit altijd van links naar rechts wordt gewerkt.
Als we hier geen rekening mee houden kunnen we voor vreemde verrassingen komen te staan.
Ik geef een paar voorbeelden en nodig U uit die ook letterlijk op uw eigen DAI mee te maken zodat U overtuigd raakt.

Overflow

Tik in :

```
*IMP INT
*A=2147483647
*?A-10+2
 2147483639
```

Het resultaat is dus 2147483639, waar we eigenlijk geen computer voor nodig hebben om te weten te komen.
Vragen we de DAI echter eerst twee op te tellen en dan pas tien af te trekken zal ons resultaat niet weer 2147483639 zijn maar OVERFLOW.

```
*A=2147483647
*?A+2-10
 OVERFLOW
```

Trek hier echter niet de conclusie uit dat het dus beter is om eerst af te trękken en dan pas op te tellen.
Eenmaal A=-A en de overflow krijgt U nu juist bij eerst aftrekken en dan optellen terwijl eerst optellen en dan pas aftrekken hier juist weer probleemloos gaat.

Het verschijnsel is simpel te verklaren door de interne notatie van de getallen in de DAI. De DAI slaat variabelen en de meeste constanten op in vier bytes of te wel twee en dertig bits. Het eerste bit daarvan wordt gebruikt om het teken aan te geven.
Het grootste gehele getal dat dus in vier bytes past is #7FFFFFFF alle bits met uitzondering van de eerste bit zijn dan 1. In decimale notatie is #7FFFFFFF 2147483647 zoals we de DAI ons zelf kunnen laten vertellen door ?#7FFFFFFF in te tikken.

Ook bij de floiting point getallen kunnen we vanzelfsprekend hetzelfde probleem krijgen. Denk overigens niet dat het probleem zich alleen kan voordoen in de buurt van de grenzen.

Een ieder die de werking van de logaritme (al of niet na het vorige artikel) en de e-macht duidelijk is zal uit het hoofd kunnen zeggen wat er uit de volgende opgave zou moeten komen.

```
*?LOG(EXP(77))
```

Juist 77. Maar de DAI doet niet meer mee. De lezers, die niet achter het toetsenbord zitten mee te werken zien nu in gedachten de OVERFLOW al staan.
Mooi mis. Deze opdracht geeft geen overflow maar de eigenlijk onjuiste foutmelding NUMBER OUT OF RANGE.
Ik dacht deze laatste opmerking in het algemeen te kunnen maar de 9511 mijn trouwe mathchip bleek hier de oorzaak. Met de mathchip aan is het argument van de EXP-functie beperkt tot het gebied tussen -32 en +32.
De grenzen zelf mogen beslist niet meedoen en geeft U een getal buiten dit interval krijgt U de foutmelding NUMBER OUT OF RANGE.
Zonder de rekenhulp kunt U de grenzen wat ruimer leggen in de buurt van 43 in plaats van 32. Deze grens is echter anders dan de 32-grens niet direct gebonden aan de hardware maar aan de opslagmogelijkheid in onze vier bytes. Zonder de mathematische coprocessor krijgt U wel de verwachte OVERFLOW melding.

In het handboek staat dit er ook bij maar bij de LOG staat er niets over en we worden nieusgierig.
We doen enige tests en stellen vast dat de LOG inderdaad geen last heeft van de grenzen die de AMD 9511 stelde bij de EXP.

De gevolgtrekking uit het voorafgaande zou kunnen zijn om bij extreem rekenwerk de mathchip maar uit te schakelen. Niet iets waar de eigenaar hem voor aangeschaft heeft.

Meer problemen

Maar er zijn meer problemen aangaande de nauwkeurigheid. Heeft U enig idee hoelang het duurt voor de DAI OVERFLOW geeft bij het volgende programma'tje ?

```
10   A=A+1
20   GOTO 10
```

Ik zou er maar niet op gaan zitten wachten als ik U was. Ik denk toch gauw dat het rond de twee a drie maanden zal zijn. Maar nu :

```
10   A!=A!+1.0
20   GOTO 20
```

Een stuk langer zult U denken. Ja dat mag men wel stellen maar niet tien of honderd keer zo lang. Dit probleem zal U heel erg lang op een overflow laten wachten. Uw wachten zal zal ook vergeefs zijn. U zult de overflow echt niet meer meemaken en uw nazaten (U moet toch iets doen tijdens dat wachten niet ?) zullen het tot in lengte der tijden ook niet zien. Ik weet namelijk zeker dat dit programma NOOIT zal stuklopen op een overflow.

Ik zal U dat aantonen met het volgende programma :

```
10   A!=1E15
20   INPUT T!:PRINT
30   FOR I=1 TO 100
40   A!=A!+T!
50   NEXT
60   PRINT A!
70   PRINT A!-1E15
80   PRINT
90   GOTO 10
```

Tik het in en run het. Voer bij het vraagteken 100000000 (8 nullen) in. U zult zien dat het eerste resultaat (uit regel 60) een bevredigend resultaat geeft.
Honderd maal vermeerderen met 10 tot de achtste is een toename met 10 tot de tiende. In de notatie met E15 is dat het vijfde cijfer achter de komma en dat is precies wat er staat.
Het resultaat van regel 70 ontnuchterd echter zeer : 6.71089E9 en dat is maar twee derde van tien tot de tiende. De afronding zorgde er nog voor dat het in regel 60 nog schijnbaar goed was.

Met kleinere getallen om er bij op te tellen zal het nog veel eerder schrikbarend fout gaan. Als we nog even aan ons uitgangprobleem denken waar we steeds 1 (een) optelden zult U hopelijk inzien dat dit relatief snel fout zal gaan en na verloop van tijd volledig zal stokken.
Bij een zekere waarde zal door de notatie waarin die waarde staat en waarin onze 1 ook wordt omgezet om er bij te kunnen worden opgeteld de waarde niet meer veranderen.
Best leuk om eens trachten te vinden bij welke kleinste waarde er een (1) bijgeteld kan worden zonder dat die waarde verandert. Het probleem is iets lastiger dan het in eerste

instantie lijkt omdat de DAI nog wat cijfers achter de hand houd die normaal niet afgedrukt worden.

We kunnen de vreemdste resultaten krijgen als we geen rekening houden met de beperkte opslagmogelijkheden in een computer.
Het volgende programma illustreert dat nog eens.

```
10   REM 23-DELING / F.H. Druijff
20   A!=7.0:D!=23.0
30   N=N+1:PRINT N;
40   FOR I=1 TO N
50   A!=A!/D!
60   NEXT
70   FOR I=1 TO N
80   A!=A!*D!
90   NEXT
100  PRINT A!
110  IF A!<0 GOTO 30
```

De werking van het programma zal iedereen wel duidelijk zijn :
De variabele A (=7) wordt N-maal door D (=23) gedeeld en dan weer N-maal met D vermenigvuldigd.
Zou de computer werken zoals je theoretisch zou mogen verwachten zal voor elke waarde van N er altijd 7 uit blijven komen. De computer heeft echter te maken met de opslagmethode in het geheugen.

Met of zonder mathchip bij N is 15, ja vijftien, is het resultaat al 0.

N	zonder 9511	met 9511
1	15.0	15.0
2	15.0	15.0
3	15.0	15.0
4	15.0	14.9999
5	15.0	14.9999
6	15.0	14.9998
7	15.0	14.9998
8	15.0	14.9997
9	15.0	14.9996
10	15.0	14.9995
11	15.0	14.9994
12	15.0	14.9993
13	15.0	14.9992
14	15.0	14.9991
15	0.0	0.0

Zonder mathchip lijkt het hier langer goed te gaan maar bedenk dat een andere keuze voor A! en vooral D! het misschien andersom had doen zijn. Met mathchip lijkt ook logischer :
- in eerste instantie een zodanig kleine afwijking dat het bij het printen niet zichtbaar wordt.
- later een kleine afwijking die naarmate we N laten toenemen marginaal groter zal worden.
Maar dan komt bij 15 de grote klap ! Het resultaat is nul. Hoe kunnen we dat verklaren ? Nogal eenvoudig als we de interne notatie beschouwen.
Het is niet nodig voor deze uitleg ook gelijk tot in de kleinste details af te dalen. De zogenaamde E-notatie waarin de DAI zijn floating point getallen aan ons doorgeeft is in feite voldoende.

E-notatie

2.3456E7 betekent 2,3456 x 107 =
2,3456 x 10 000 000 =
23 456 000

Dit laatste getal is niet noodzakelijkerwijs exact. We kunnen bij de opgave van 2.3456E7 alleen zeker weten dat de waarde tussen 23 455 500 en 23 456 500 inligt.
In feite is het laatste getal voor de E een afgeronde waarde.
De notatie van de DAI zou ons echter zes significante cijfers moeten geven.
Is dit laatste cijfer echter een 0 zoals in dit voorbeeld laat de DAI die om discutabele redenen weg.
Omdat we de DAI kennen en dus weten dat er zes significante cijfers moeten zijn kunnen we de grenzen met deze wetenschap enger maken tot : 2.3456E7 ligt tussen 23 455 950 en 23 456 050 in.
Het interval waarin 2.3456E7 ligt is hiermee dus verkleint van duizend tot honderd.
Maar we dwalen af van ons onderwerp.
Delen we bijvoorbeeld 4.88888E6 door twee zal dat 2.44444E6 worden.
Wederom door twee delen geeft nu 1.22222E6 maar weer delen door twee geeft niet 0.61111E6 maar 6.1111E5.
We zien dat het eerste deel NOOIT met een 0 begint en ook NOOIT uit meer dan twee cijfers zal bestaan.
Samenvattend : het deel van het eerste getal voor de komma zal altijd ongelijk aan 0 zijn. Stel dat achter de E maar een getal van een cijfer mag staan. Dit getal/cijfer mag dan wel van een '-' voorzien zijn indien nodig. Delen we 1.22222E-9 door twee wordt dat 6.1111E-10 maar dat kan niet volgens onze afspraak dat achter

de E het getal maar een cijfer mag hebben. Vele ontwerpers van rekenhard en software kiezen nu toch voor de oplossing 0.61111E-9. wordt dit dan door honderd gedeeld is 0.00611E-9 het resultaat.
U ziet dat we 'officieel' nog steeds zes significante cijfers hebben maar de eerste drie zijn nullen en de laatste twee enen zijn al uit de notatie geschoven.
Gaan we nu weer vermenigvuldigen met de getallen waar we zoeven door deelden doen die weggeschoven enen niet meer mee. Dit verklaart de geringe afwijking die U in de mathchiplijst aantrof. Logischerwijze is nu ook in te zien waarom de afwijking met stijgende N toeneemt. Ziet U het ? Ja ? Jammer, want dit is niet de reden van deze afwijking. Het speelt wel een rol maar dat komt pas veel later.

De deling zal vaak veel meer cijfers achter de komma leveren als kunnen worden opgeslagen. Het bewaarde getal zal dus vaak iets afwijken van de echte waarde. Door meer bytes per getal te reserveren verleggen we het probleem maar lossen het niet op.

3 x (1/3)

1 / 3 = 0,333333333333333333333....
we noteren dat als 0,3333333 als we zes cijfers achter de komma werken. Maar dan is 3 x (1 / 3) = 0,999999 en niet aan 1.
De DAI is het met het voorafgaande helemaal eens, maar zal zijn 0,999999 toch als 1.0 op het scherm zetten.
Vragen we hem echter of het resultaat van 3.0*(1.0/3.0) gelijk is aan 1.0 zal hij ontkennen.
Het volgende programma laat dit zien:

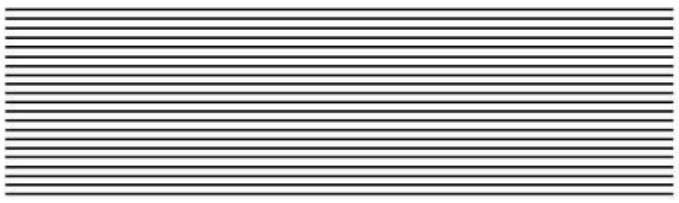
```
10   REM P*(1/P){<1 / F.H. Druijff
20   I!=I!+1.0:PRINT
30   PRINT I!,I!*(1.0/I!),
40   A=I!*(1.0/I!)
50   PRINT A!;
60   IF A!=1.0 GOTO 20
70   PRINT "  Ongelijk";
80   GOTO 20
```

Tik het in en zie dat reeds bij drie er 'Ongelijk' bij staat. Maar wacht eens, er zijn er verderop ook nog waarden waar het ongelijk wegblijft.

Het blijkt uit dit programma dat het alleen gelijk is bij 1,2,4,8,16,32, 64,128,256,512,1024,... dus de machten van twee. En zoals de meesten wel zullen weten werkt de DAI binair en zal bij deling door twee alleen de exponent een verminderen en de mantisse ongemoeid laten.
Bij de vermenigvuldiging gebeurt dat ook en bij een onveranderde mantisse is de afwijking van de oorspronkelijke waarde nul.
Zouden we in het deelprogramma dus niet door 23 maar door 32 (= 2 ^5) gedeeld hebben zouden er ook geen kleine afwijkingen zijn geweest.

Bij de deling onderscheiden we dus twee fases.
- Eerst wordt de deling binnen de nauwkeurigheid van de DAI opgelost en door de notatie vallen de cijfers na (zeg maar) het zesde cijfer weg. De mantisse zal door deze afronding nooit exact op grootte blijven maar wel binnen redelijke grenzen van de echte waarde.
- Dan begint daarna de fase waarin de mantisse er zelf langzaam wordt uitgeschoven. Bij delen door 23 gaat dit met zo'n vier a vijf bits per keer en al heel snel is er niets meer van over. In deze fase kunnen we verwachten dat de einduitkomst ineens grotere verschillen gaat vertonen.
In de praktijk blijkt dit echter mee te vallen daar natuurlijk altijd de belangrijkste cijfers er nog staan.
Ook vermoed ik dat als er teveel bits zijn weggeschoven de waarde automatisch op nul wordt gezet.
In vele praktijk gevallen zal blijken dat er dan 0 uit komt en dat dat bijna iedereen zal opvallen.
Komt er een waarde uit die in feite volledig onbetrouwbaar is loopt men het risico dat niet op tijd in de gaten te hebben. Komt er echter nul uit zullen vele programma's kapot lopen of zulke onzinnige antwoorden geven dat het opvalt.

Frank H. Druijff



64 K ram card

DAI- 64k RAM CARD March 1986

By George Cathcart
12 Evora Park,
Howth
Co.Dublin
IRELAND
Tel. 01-324030

This board connects to the DAI via a motherboard on the DCE bus or (in the future) via a motherboard on the X bus. It has a battery back up can be used in place of EPROM either for DAI or another Micro thereby allowing changes to the program under development to be made very easily. It can also be used instead of a Disk system. The card uses part of one card address from the motherboard (a total of 16 Megabytes RAM (256 X 64K RAM boards) could be connected to each card address). The (DCE bus) motherboard can support 12 card addresses and is supported by the IN and OUT commands in DAI basic. For large data transfers however this is almost as slow as using a cassette recorder. Using a machine code routine transfer rates are about 60 times faster. The X bus version should be even faster.

The mapped position of the board within the card address can be changed by way of eight swiches on the board and the card address can be selected using a wire link.

The board uses eight 8K by 8 bit 6264 CMOS memory chips to achieve 64K. 24 lines of address, generated by the Motherboard are fully decoded using high speed CMOS chips. This keeps the power consumption to a minimum. In standby, not connected to the motherboard, the card uses only about 0.3 ma. This allows data retention for about two weeks using a fully charged onboard 110 ma hour 3.6V Ni Cad battery. The comsumption almost doubles when the card is left connected with the DAI turned off. The battery is charged automatically via the motherboard when the DAI is on and can also be charged by applying 4.4 volts (5 volts via a diode) to pin 49 of the card connector while the DAI is either on or off. The voltage on Pin 49 can also be used to indicate the presence of the RAM card, for example when developing the operating system of a homemade dedicated micro. This voltage monitoring is not supported by the motherboard. Resistor R10 controls the charge rate of the battery. All address lines are tied low whereas data lines and the card enable signal are held high via 100K resistors to stop the CMOS chips oscillating and using excess current when the board is not connected. When on battery backup (+Vbb) the following are also held low via R11 (1K).:-

Pin 26 of ics. 5-12 (6264). This chip enable is active high.
Pin 6 of ic.4 (74HC138). This chip enable is active high.
The switched inputs to ics.1 & 2.(74HC86) used for address line inverting.

POWER comes from the motherboard where it is supplied either by the DAI or an external power supply.

READ/WRITE signals are decoded on the motherboard

ADDRESS DECODING

The address lines are generated by ports A,B & C of the motherboard 8255 PPI. Of the 24 address lines 13 (A0 - A12) go directly to each RAM chip, 3 (A13 - A15) go to ic.4, a three line to eight line decoder (74HC138). This gives eight decoded chip select lines spaced in 8K blocks. This leaves eight (A16 - A23) and these can be inverted using 8 XOR gates -ics.1 & 2 (74HC86). The second inputs of these gates are contolled by 8 onboard switches. The outputs connect to an 8 input 'NAND' gate -ic.3 (74HC30)- which generates one of the chip select signals for the three line to eight line decoder mentioned above when all eight inputs are high. The second chip select is provided by the card enable signal from the motherboard and the third (high) select by the power supply from the motherboard (+Vnb., No battery backup).

CONNECTIONS

Connection is via a 50 pin IDC connector. Pins 1 - 40 are compatable with those from the DCE bus motherboard.

PIN DESCRIPTIONS

1	+5 Volts (in)	2	0 Volts
3	ADDRESS 14	4	ADDRESS 15
5	ADDRESS 12	6	ADDRESS 13
7	ADDRESS 10	8	ADDRESS 11
9	ADDRESS 8	10	ADDRESS 9
11	ADDRESS 19	12	DATA 7
13	ADDRESS 18	14	DATA 6
15	ADDRESS 17	16	DATA 5
17	ADDRESS 16	18	DATA 4
19	ADDRESS 20	20	DATA 3
21	ADDRESS 21	22	DATA 2
23	ADDRESS 22	24	DATA 1
25	ADDRESS 23	26	DATA 0
27	RD (active low)	28	WR (active low)
29	ADDRESS 0	30	ADDRESS 1
31	ADDRESS 2	32	ADDRESS 3
33	ADDRESS 4	34	ADDRESS 5
35	ADDRESS 6	36	ADDRESS 7
37	CARD 7 ENABLE	38	CARD 6 ENABLE
39	CARD 5 ENABLE	40	CARD 4 ENABLE
41	NOT USED	42	NOT USED
43	NOT USED	44	NOT USED
45	NOT USED	46	NOT USED
47	NOT USED (+12 Volts)	48	NOT USED
49	+V battery backup	50	NOT USED (-5 Volts)

NOTE that pins 41 to 50 are not supported by the DCE bus motherboard.

CONSTRUCTION

A double sided PCB measuring 203 by 114mm is used. Track layout and component layout diagrams are supplied. There are three index marks for registration purposes on the drawings. One in each corner at the connector end of the board and one off centre at the other end of the board. The drawings are presented as seen through the board. ie. the printed surface next to the board.

Construction is straight-forward if the following points are noted.:-

Two connections are soldered to the component side of the board. These are Pin 1 (0v) of R.1 and the positive (+Vbb) side of R.6. Note that if the card system is not used, the link labeled 'none' is used and R.6 is omitted.

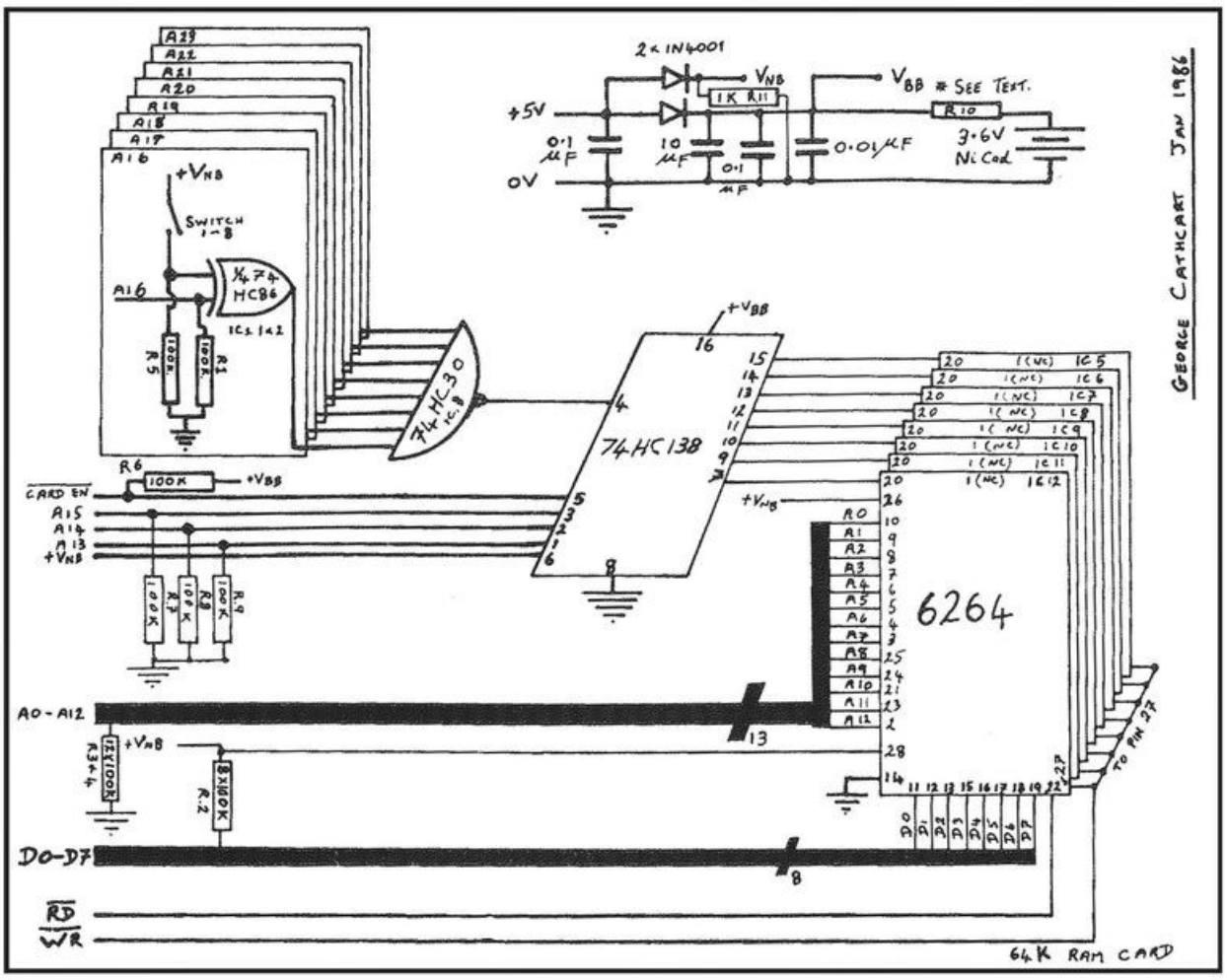
All connections through the board are by way of pins. These should be inserted first as they are in many cases under other components. This saves having to solder the ic. sockets on both sides of the board. Pads on the component side of the poard indicate the positions of the links. Where the pins are very close together they should be inserted from alternate sides of the board.

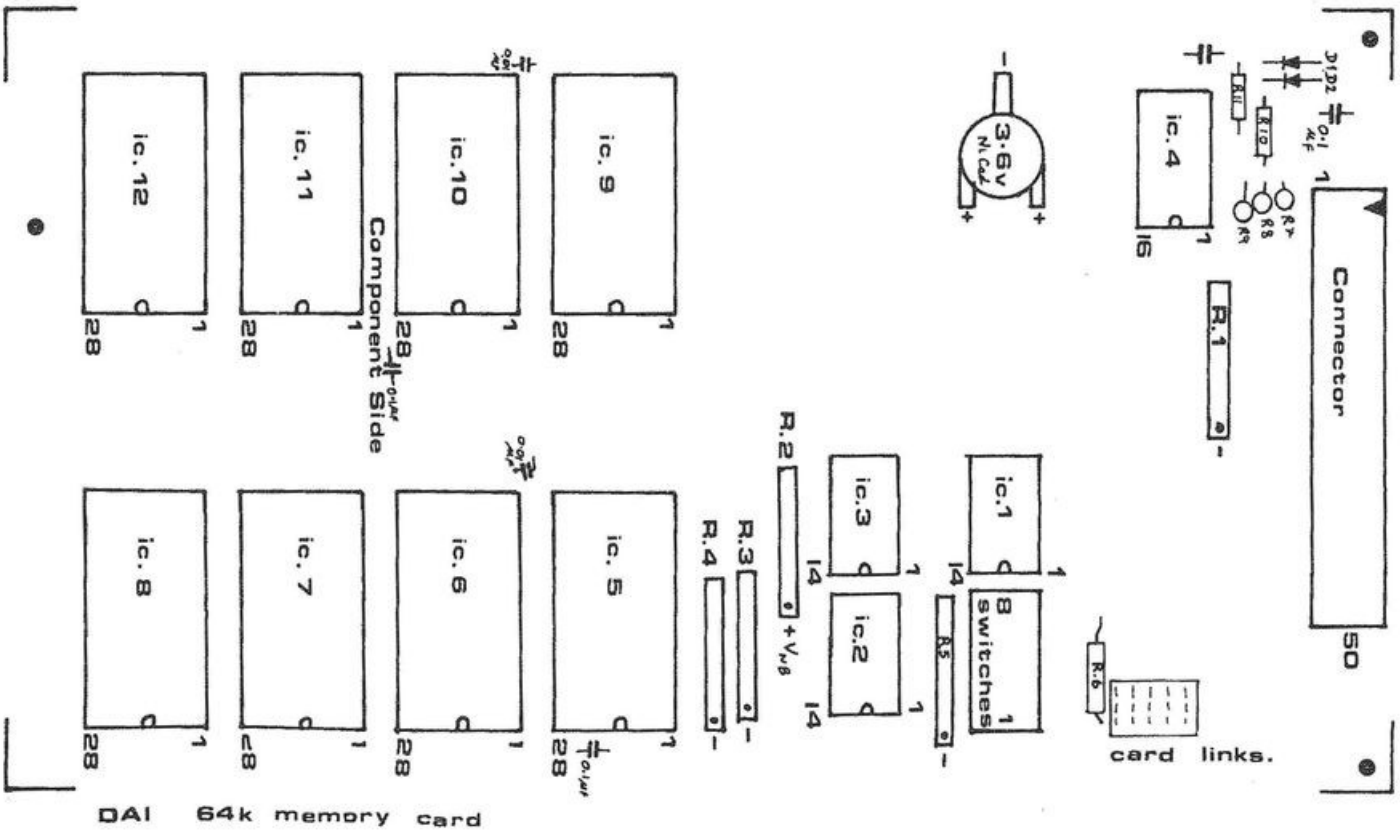
Following this solder in the 50 pin connector and check the board for short circuits before inserting the ic. sockets The battery (if used) should be mounted over an insulator to prevent connection with the lines and pins situated underneath it. R.10 controls the charge rate of the battery. For a 3.6V battery a 470R resistor allows a charge of nearly 1ma for a fully charged battery. The voltage drop across it 5V - 0.6V - Battery voltage. If no onboard battery is used R.10 can be omitted. With the eight address decoding switches switched on, the board maps to Hex 000000 (ic.5) to 00FFFF (ic.12).

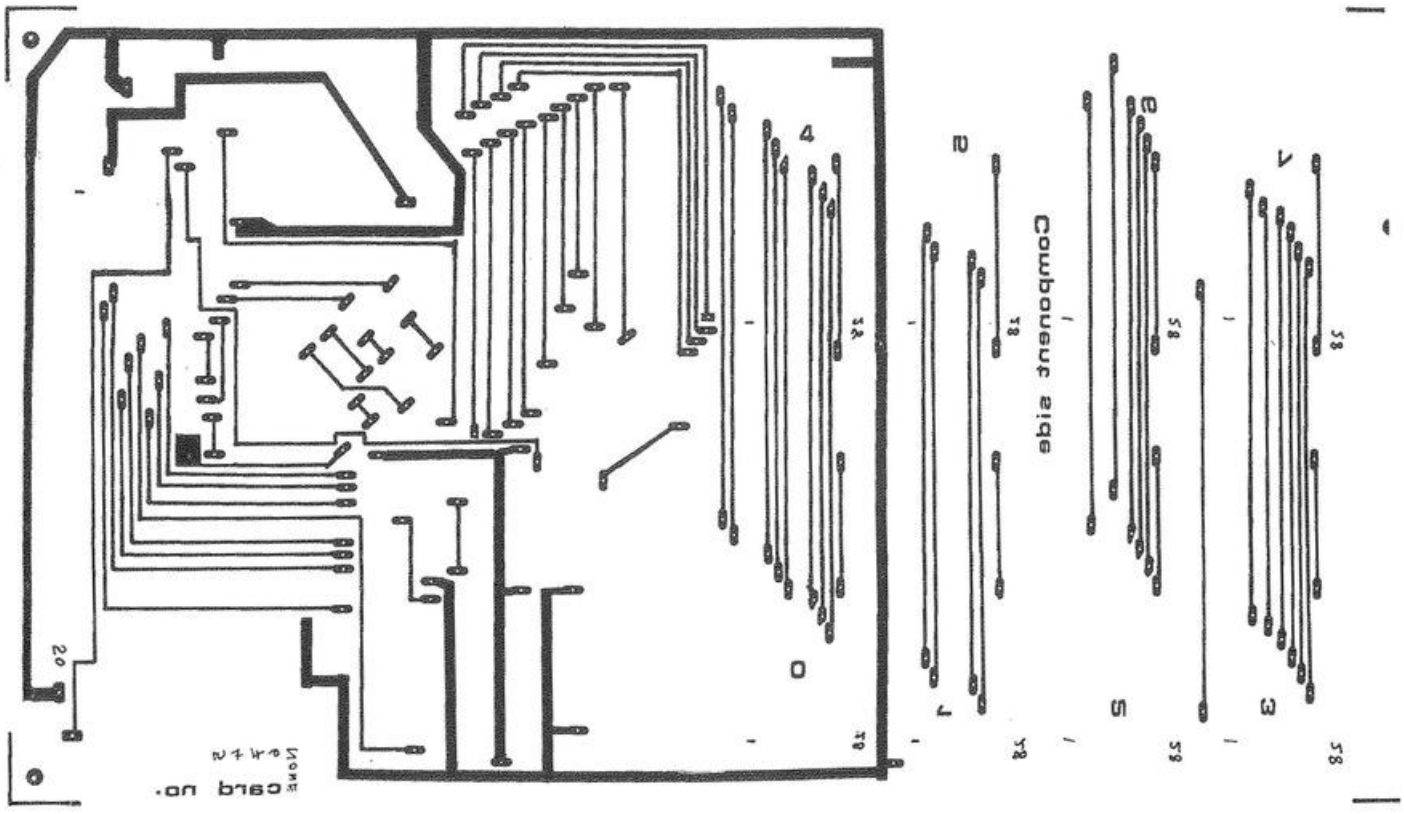
COMPONENTS

R.1-5	100K. 8 commoned resistors in an SIL package
R.6-9	100K 1/4watt.
R.10	see text.
R.11	1K 1/4 watt.
C.1	10 microF.Tant.
C.2-4	0.1 micro F. Ceramic Disc.
C.5-6	0.01 micro F. Ceramic.
D.1-2	1N4001.
ic.1-2	74HC86 Quad XOR Gate
ic.3	74HC30 8 input NAND gate.
ic.4	74HC138 3 to 8 line decoder.
ic.5-12	6264 CMOS 8K X 8 bit memory.

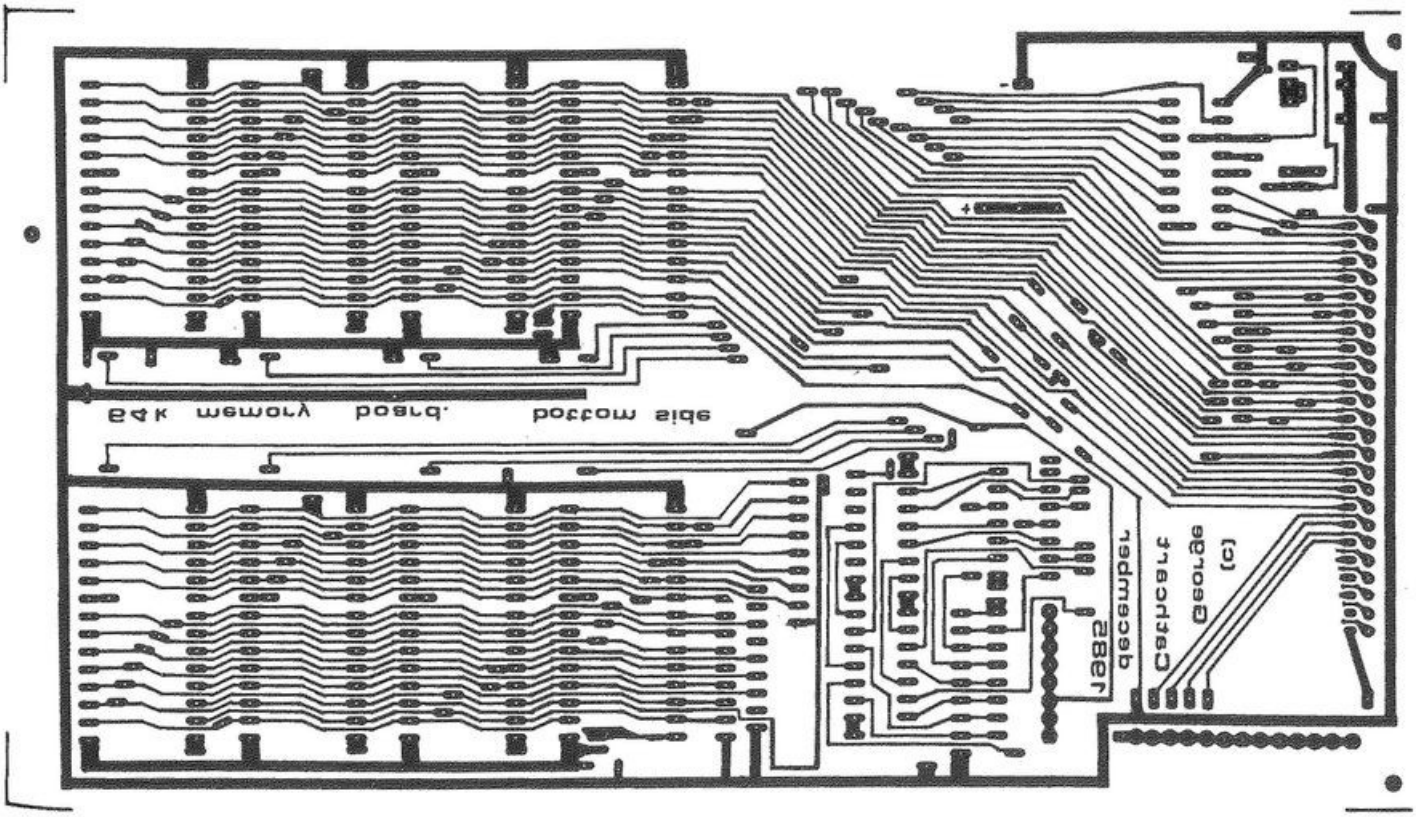
PCB board double sided 203 x 114 mm.
Track pins.
One 3.6V NiCad 10 maH battery.
One 50 way rt angled PCB plug (AWH50 or equivalent).
One 8 way 16 pin DIL switch.
Three 14 pin ic. sockets.
Two 16 pin ic. sockets.
Eight 28 pin ic. sockets.


GEORGE CATHCART JAN 1985
2 x IN4001
V_{NB}
V_{BB} * SEE TEST.
1K R11
R10
+5V
0V
0·1 µF
10 µF
0·01µF
3·6V Ni Cad
A16
A17
A18
A19
A20
A21
A22
A23
+V_{NB}
SWITCH 1-8
X74 HC86
IC1 IC2
R5 100K
R1 100K
74HC30
IC3
74HC138
+V_{BB}
R6 100K
CARD EN
A15
A14
A13
R7 100K
R8 100K
R9 100K
6264
A0-A12
13
R3+4 10X100K
R2 8X100K
D0-D7
8
RD
WR
TO PIN 27
64K RAM CARD


ic.12
ic.11
ic.10
ic.9
ic.8
ic.7
ic.6
ic.5
Component Side
3·6v Ni Cad
D1 D2
0·1 µF
R11
R10
R9
R8
R7
ic.4
R.1
Connector
50
R.2
R.3
R.4
ic.3
ic.2
ic.1
8 switches
R5
R.6
card links.
DAI 64k memory card



DAI 64k memory card top



DAI 64K Memory Card bottom

Address line test

```
1     REM WRITTEN BY George Cathcart 23/2/86
2     FLAG%=0:PRINT CHR$(12);"ADDRESS LINE TEST (DCE BUS MOTHERBOARD)"
3     PRINT :INPUT "Do you wish to Log ERRORS (Answer Y or N)";LG$
6     PRINT :PRINT :INPUT "Motherboard No.";DNO:PRINT :IF DNO<0.0 OR DNO>£F THEN
 PRINT "IMPOSSIBLE ":GOTO 6
7     INPUT "RAM Card No.";CNO:PRINT :IF CNO<4.0 OR CNO>£F THEN PRINT "IMPOSSIBL
E use 4-15":GOTO 7
8     INPUT "How much RAM is present (K)";KBYTE%:PRINT :PRINT :MEM%=KBYTE%*£400
9     CARD=(16.0*DNO)+CNO:PPI=(16.0*DNO)+3.0:GOSUB 1100
10    OUT PPI-3,£A5:OUT PPI-2,0:CIN=INP(PPI-3):IF CIN=£A5 GOTO 13
12    PRINT " NO MOTHERBOARD AT THIS ADDRESS !":GOTO 6
13    DIM FL%(24.0):GOSUB 1100
15    PRINT " Checking first £FF addresses on CARD £";HEX$(CARD)
20    FOR RAM=0.0 TO £FF
30    OUT CARD,XX%:XX%=XX%+1:GOSUB 1500:OUT PPI-3,XX%
40    NEXT RAM
45    GOSUB 1100:REM ADDRESS = £000000
49    REM
50    FOR RAM=0.0 TO £FF
60    II%=INP(CARD)
65    IF XX%<>II% THEN GOSUB 1120
70    XX%=XX%+1:GOSUB 1500:OUT PPI-3,XX%:NEXT RAM
79    REM
80    IF FLAG%=0 THEN PRINT " LOWER ADDRESS LINES  OK!"
90    IF FLAG%>0.0 THEN PRINT FLAG%;" (£";HEX$(FLAG%);")";" ERRORS FOUND":FLAG1%
=1
95    IF FLAG%>£FD THEN PRINT " IS THERE ANY MEMORY AT THIS ADDRESS ?"
100   MEM%=MEM%/£FF:PRINT :PRINT " Checking one of each 256 bytes on CARD £";HEX
$(CARD)
109   REM
110   GOSUB 1100:FLAG%=0
120   FOR RAM=0.0 TO MEM%-1.0
130   OUT CARD,YY%:YY%=YY%+1:GOSUB 1520:OUT PPI-2,YY%
200   NEXT RAM
299   REM
300   GOSUB 1100:REM ADDRESS £000000
320   FOR RAM=0.0 TO MEM%-1.0
330   II%=INP(CARD):IF YY%<>II% THEN GOSUB 1130
340   YY%=YY%+1:GOSUB 1520:OUT PPI-2,YY%:NEXT RAM
359   REM
360   IF FLAG%=0.0 THEN PRINT " UPPER ADDRESS LINES OK!"
370   IF FLAG%>0.0 THEN PRINT FLAG%*£100;" (£";HEX$(FLAG%*£100);") PROBABLE ERRO
RS DETECTED"
380   IF FLAG%=0.0 AND FLAG1%=0.0 THEN PRINT " ALL OK!":GOTO 1000
385   PRINT
390   FOR LINE%=0 TO 23
392   IF FL%(LINE%)>0.0 THEN PRINT "CHECK ADDRESS LINE ";LINE%
395   NEXT
1000  PRINT :END
1100  XX%=0:YY%=0:ZZ%=0
1110  OUT PPI,£80:OUT PPI-1,ZZ%:OUT PPI-2,YY%:OUT PPI-3,XX%:RETURN
1120  FLAG%=FLAG%+1:IF LG$="n" OR LG$="N" THEN GOTO 1123
1121  PRINT " ERROR AT £";HEX$(ZZ%);" ";HEX$(YY%);" ";HEX$(XX%);
1122  PRINT "       OUTPUT =£";HEX$(XX%);"         FOUND =£";HEX$(II%)
1123  IP%=II%-XX%:IF IP%<0 THEN IP%=-IP%
1124  ER=LOG(IP%)/LOG(2.0):FL%(ER)=1:RETURN
1130  FLAG%=FLAG%+1:IF LG$="N" OR LG$="n" THEN GOTO 1133
1131  PRINT " ERROR AT £";HEX$(ZZ%);" ";HEX$(YY%);" ";HEX$(XX%);
```

```
1132  PRINT "       OUTPUT =£";HEX$(YY%);"          FOUND =£";HEX$(II%)
1133  IP%=II%-YY%:IF IP%<0 THEN IP%=-IP%
1134  ER=LOG(IP%)/LOG(2.0):FL%(ER+8.0)=1:RETURN
1500  IF XX%>£FF THEN XX%=0:YY%=YY%+1
1510  IF YY%<£100 THEN OUT PPI-2,YY%
1520  IF YY%>£FF THEN YY%=0:OUT PPI-2,YY%:ZZ%=ZZ%+1:OUT PPI-1,ZZ%
1560  RETURN
```

```
1     REM WRITTEN BY George Cathcart 22/2/86
2     FLAG1%=0:XX%=0:YY%=0:ZZ%=0:ENVELOPE 0 15,10;0,10;:PRINT CHR$(12);"NON-DES
RUCTIVE RAM TEST (DCE BUS MOTHERBOARD)"
3     PRINT :INPUT "Do you wish to log errors Y/N ";L$
4     IF L$="Y" OR L$="N" THEN GOTO 6
5     GOTO 3
6     PRINT :PRINT :INPUT "Motherboard No.";DNO:PRINT :IF DNO<0.0 OR DNO>£F THE
 PRINT "IMPOSSIBLE ":GOTO 6
7     INPUT "RAM Card No.";CNO:PRINT :IF CNO<4.0 OR CNO>£F THEN PRINT "IMPOSSIB
E ":GOTO 7
8     INPUT "How much RAM is present (K)";KBYTE%:PRINT :PRINT :MEM%=KBYTE%*£400
9     CARD=(16.0*DNO)+CNO:PPI=(16.0*DNO)+3.0
10    PRINT "Do NOT ABORT or you may corrupt a memory location.":PRINT
14    OUT PPI,£80:OUT PPI-1,ZZ%:OUT PPI-2,YY%:OUT PPI-3,XX%:REM ADDRESS=£000000
15    FOR RAM=£0 TO MEM%-1.0
17    FLAG%=0
20    IF XX%=0 THEN GOSUB 119:REM Print 'CHECKING....etc.
40    B=INP(CARD):Z=£FF-B:OUT CARD,Z:Z1=INP(CARD):Z2=£FF-Z1:OUT CARD,B
45    GOSUB 245:REM Check if OK.
50    FLAG1%=FLAG1%+FLAG%:FLAG2%=FLAG%:FLAG%=0
55    XX%=XX%+1:IF XX%>£FF THEN YY%=YY%+1:XX%=0:GOSUB 150:REM Print OK or BAD
56    IF YY%>£FF THEN ZZ%=ZZ%+1:YY%=0
57    OUT PPI-3,XX%:IF XX%=0 THEN OUT PPI-2,YY%:IF YY%=0 THEN OUT PPI-1,ZZ%
60    NEXT
80    IF FLAG1%=0.0 THEN PRINT :PRINT "ALL OK!":GOTO 100
90    PRINT :PRINT FLAG1%;" ERRORS DETECTED."
100   END
119   PRINT "CHECKING £";HEX$(CARD);":";HEX$(ZZ%);" ";
120   PRINT HEX$(YY%);"00 TO ";HEX$(YY%);"FF  ";:RETURN
150   IF FLAG2%=0.0 THEN PRINT "OK.":RETURN
160   IF FLAG2%<>0.0 THEN PRINT "BAD.":SOUND 0 0 15 0 FREQ(1000.0):WAIT TIME 30
SOUND OFF :FLAG2%=0:RETURN
245   IF Z2<>B THEN FLAG%=1
246   IF L$="N" THEN RETURN
247   IF Z2<>B THEN PRINT :PRINT :PRINT "ADDRESS = xxyy ";HEX$(XX%):PRINT
251   IF Z2<>B THEN PRINT "RAM ORIGINAL = ";HEX$(B);" COMPLEMENT OUTPUT = ";HEX
(Z)
253   IF Z2<>B THEN PRINT "RAM RETURN = ";HEX$(Z1);" RETURN COMPLEMENT = ";HEX$
Z2)
254   IF Z2<>B THEN SOUND 0 0 15 0 FREQ(1000.0):WAIT TIME 30
255   RETURN
```

DCE bus motherboard

DAI - DCE Bus Motherboard.
March 1986.

by George Cathcart
12 Evora Park,
Howth
Co.Dublin
Ireland.
Tel. 01-324030

This board connects to the DCE bus via. a 34 way connector. The DAI - DCE bus pin 1 is on the top row away from the ON/OFF switch. The pins are compatable with the standard DCE Bus connections and so input and output through the board are supported by DAI Basic. The board acts as a buffer and decoder for memory cards, an EPROM programmer, LED boards, switches etc. The address of the Motherboard is decided by four switches on the board with possible addresses of 0 to 15. It is possible therefore to connect more than one motherboard at at a time if desired. Each motherboard decodes 16 card addresses, 12 of which are available to the user. addresses 0 to 3 are used by the onboard 8255 PPI which is wired for mode 0 operation. This can be used - for example to latch 24 address lines thereby supporting up to 16 Mega byte of RAM or EPROM on a single card address.

HOW IT WORKS.

The motherboard address, is compared to the address on the DCE bus by a 74LS85 comparitor. This 74LS85 is enabled by the BE signal (from the DCE bus) using the cascade input. The inverted 74LS85 output is used as an Enable signal and gates the RD (low) and WR(low) signals - to avoid the low glitch, (from the DAI 8255 PPI during DATA update) from generating both signals simultaneously. It also enables a 4 bit to 16 line decoder (74LS154) which decodes the card addresses.

As already mentioned four of the 16 available card addresses are used on the board by an 8255 PPI chip. These four enable signals (for card addresses 0,1,2 and 3) are combined by half a dual four input NAND gate (74LS20) and then inverted. Address lines 0 and 1 from the DCE Bus are used directly for the 8255. Data from the DCE bus is buffered by a bidirectional buffer (74LS245).

The onboard 8255 is reset at power up via a 56K resistor and 0.47 micro Farad capacitor connected to pin 35 of the 8255.

The following describes the connections to and from the board.:-

INPUT

34 PIN INPUT from DCE Bus

NB. No Connection REFERS TO THIS BOARD AND NOT THE DCE BUS.

PIN		PIN	
34	No connection	33	No connection
32	Card Address bit 2	31	Card Address bit 1
30	" " " 0	29	No connection
28	RD (low to read)	27	WR (olow to write)
26	BE (high turns on bus)	25	Card Adddress bit 3
24	Device Address bit 0	23	Device Address bit 1
22	" " " 2	21	" " " 3
20	No Connection	19	No Connection
18	" "	17	" "
16	Data bit 0	15	Data bit 7
14	Data bit 1	13	Data bit 6
12	Data bit 2	11	Data bit 5
10	Data bit 3	9	Data bit 4
8	No Connection	7	No connection
6	" "	5	" "
4	0 Volts	3	" " (-5 Volts)
2	(+12 V) No connection	1	+5 Volts

OUTPUTS

There are two output plugs on the board:-

20 pin		40 pin	
8	buffered data	8	buffered data
2	power	2	power
2	RD & WR	2	RD & WR
8	card enable (8,9,10,11, 12,13,14,15)	4	card enable(4,5,6,7)
		24	(3x8 bit) 8255 ports

ONBOARD 8255 PPI

Port A	bidirectional - latched	(Card Address 0)
Port B	bidirectional - latched	(Card Address 1)
Port C	bidirectional - latched	(Card Address 2)

The 8255 Control Byte is at Card address 3

20 pin output - pin descriptions

PIN		PIN	
20	0 Volts	19	+5 Volts
18	Data bit 6	17	Data bit 7
16	Data bit 4	15	Data bit 5
14	Data bit 2	13	Data bit 3
12	Data bit 0	11	Data bit 1
10	Card 14 Enable	9	Card 15 Enable
8	Card 12 Enable	7	Card 13 Enable
6	Card 8 Enable	5	Card 11 Enable
4	Card 10 Enable	3	Card 9 Enable
2	RD (low to read)	1	WR (low to write)

40 pin output - pin descriptions

PIN		PIN	
1	+5 Volts	2	0 Volts
3	8255 Port B bit 6	4	8255 Port B bit 7
5	8255 Port B bit 4	6	8255 Port B bit 5
7	8255 Port B bit 2	8	8255 Port B bit 3
9	8255 Port B bit 0	10	8255 Port B bit 1
11	8255 Port C bit 3	12	Data bit 7
13	8255 Port C bit 2	14	Data bit 6
15	8255 Port C bit 1	16	Data bit 5
17	8255 Port C bit 0	18	Data bit 4
19	8255 Port C bit 4	20	Data bit 3
21	8255 Port C bit 5	22	Data bit 2
23	8255 Port C bit 6	24	Data bit 1
25	8255 Port C bit 7	26	Data bit 0
27	RD (low to read)	28	WR (low to write)
29	8255 Port a bit 0	30	8255 Port A bit 1
31	8255 Port A bit 2	32	8255 Port A bit 3
33	8255 Port A bit 4	34	8255 Port A bit 5
35	8255 Port A bit 6	36	8255 Port A bit 7
37	Card 7 Enable	38	Card 6 Enable
39	Card 5 Enable	40	Card 4 Enable

CONSTRUCTION.

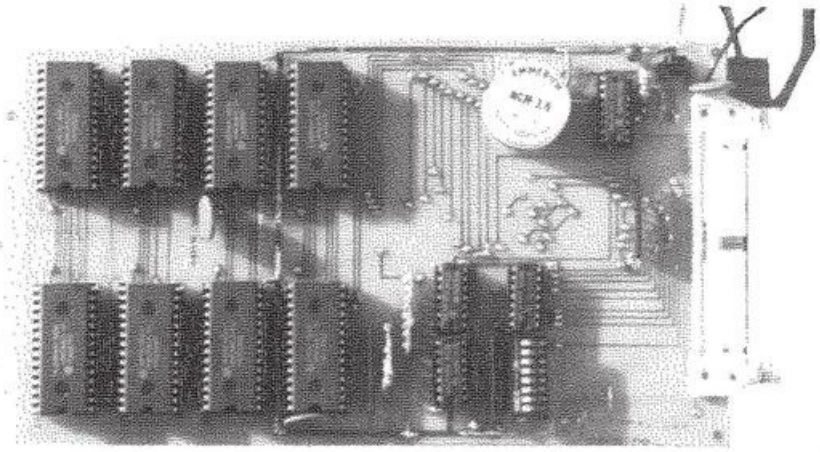
The motherboard can be made on either a double sided or a single sided (with wire links) PCB measuring 100 X 160mm. If a single sided board is used, two links which pass under the 20 pin header are best soldered on the underside of the board. If a double sided board is used, registration is by three index marks on each drawing. To date I have made two motherboards, both on single sided boards and the only problems encountered were due to too long a connecting lead. This sometimes caused the DAI to reset while reading from memory connected to the motherboard.

On my boadrs I have soldered a 0.1 micro Farad capacitor across the power supply pins of the 8255 PPI (pins 26 and 7), this supresses spikes on the ports of the 8255.

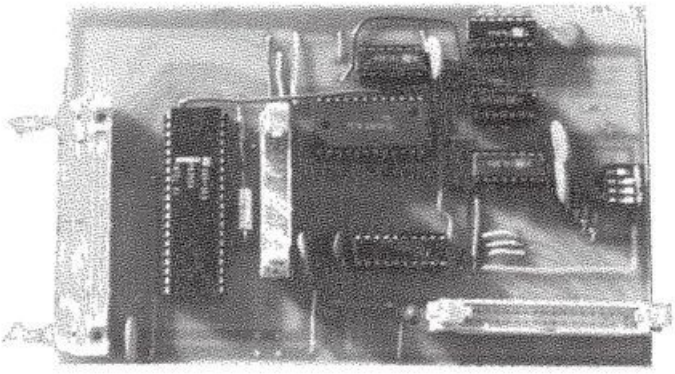
It is possible to either use the DAI 5 volt power supply or an external one. If using power from the DAI insert link "A" and solder the 10 micro Farad Tantalum capacitor in position "A", if using an external supply leave out the link and use position "B" for the capacitor. External power is input on two pins near this capacitor.

Note that the PCB artwork is presented as seen through the board ie. the writing is the correct way around on the finished board.

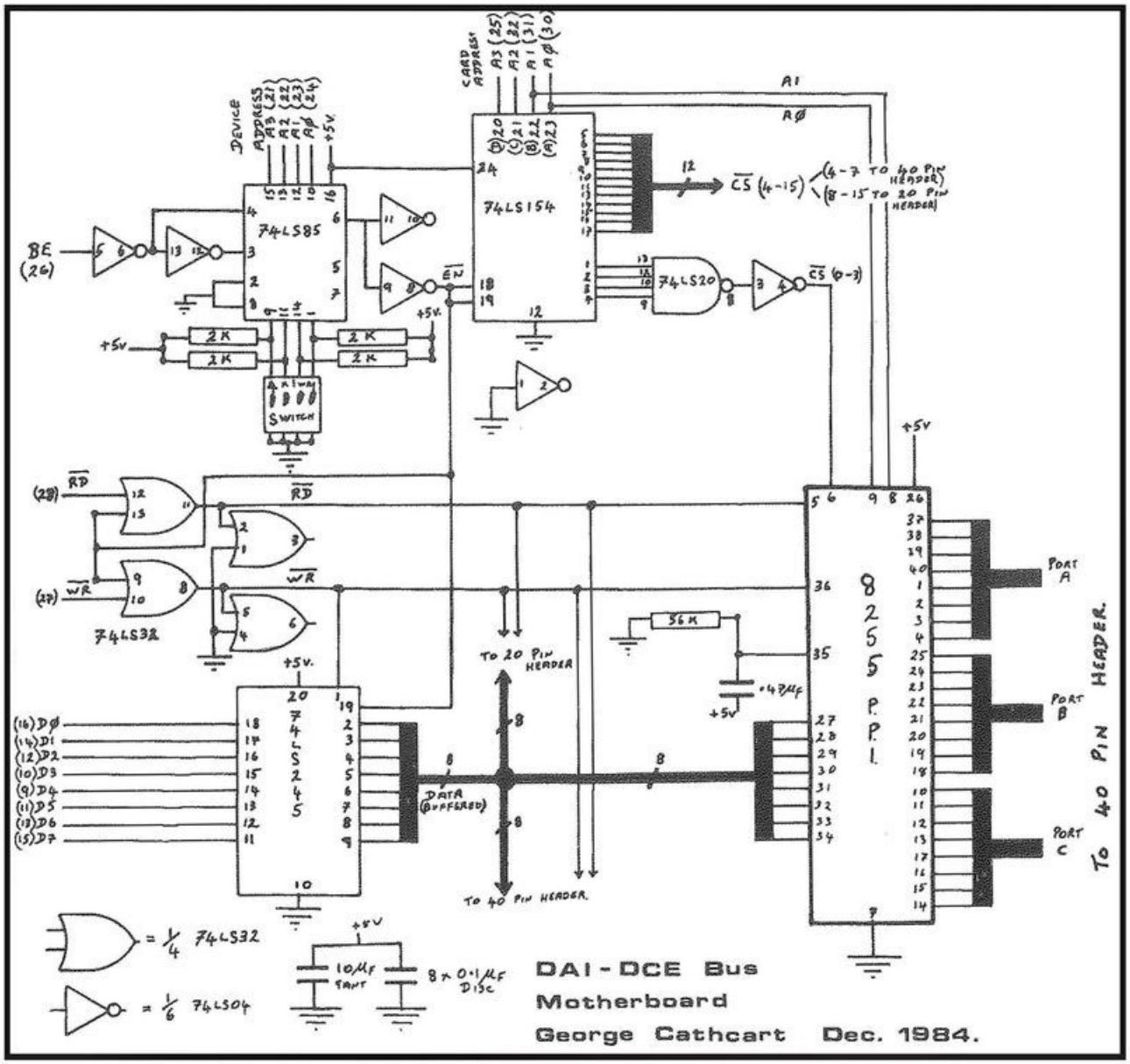
```
Resistors.
      4     2K 1/4 watt.
      1    56K    "    "

Capacitors
      1     0.47 micro Farad.
      8     0.1      "      "    disk.
      1     10       "      "    tantalum.

Integrated Circuits.
      1     74LS85 Comparitor
      1     74LS04 Hex inverter
      1     74LS32 Quad 2 input OR gate
      1     74LS245 Octal bidirectional buffer
      1     74LS154 4 to 16 line decoder
      1     74LS20 Dual 4 input NAND gate
      1     8255 PPI

Miscellaneous.
      1     40 pin DIL socket.
      1     24 pin DIL socket.
      1     20 pin DIL socket.
      1     16 pin DIL socket.
      3     14 pin DIL sockets.
      1     4 spst DIL switches.
      1     34 way straight PCB plug (like DCE BUS plug).
      1     20 way straight PCB plug.
      1     40 way PCB plug.(straight or right angle.
      1     100 X 160mm PCB.
            PCB pins if double sided board is used.
```

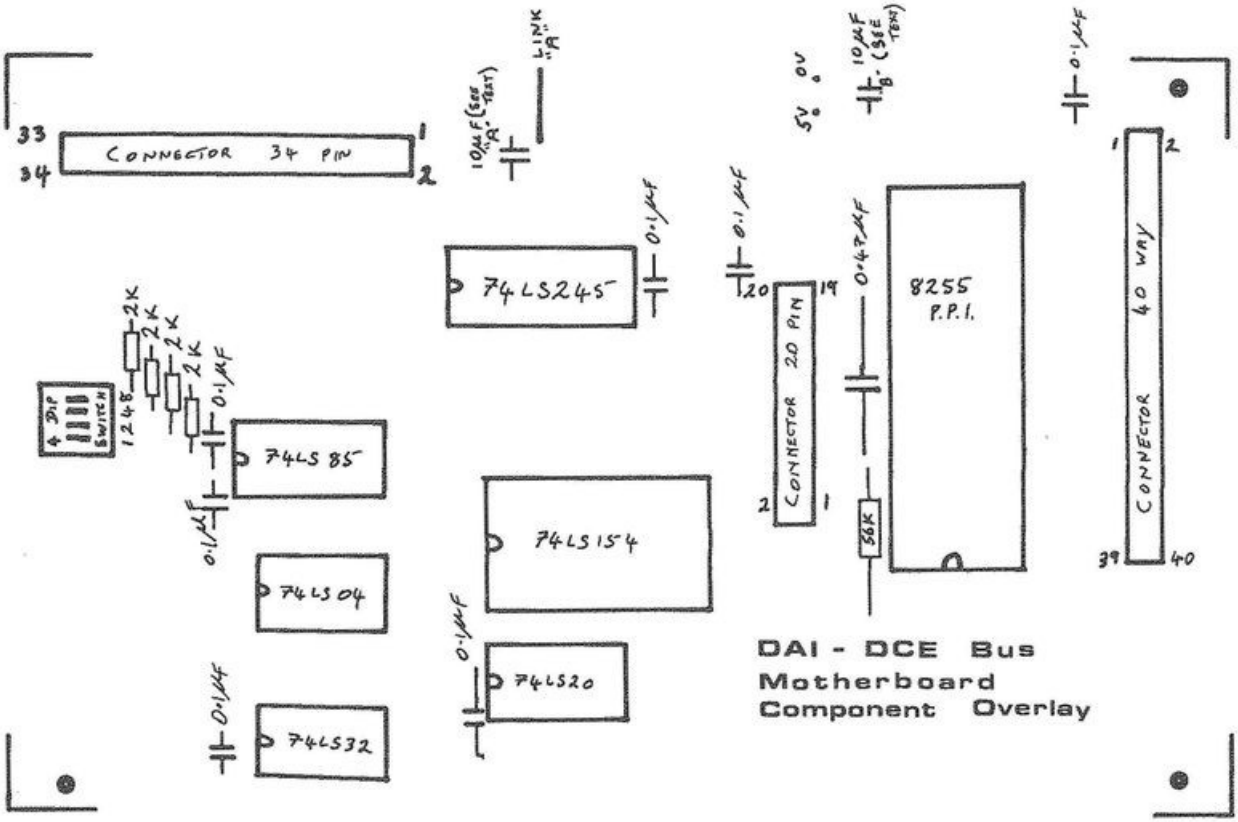


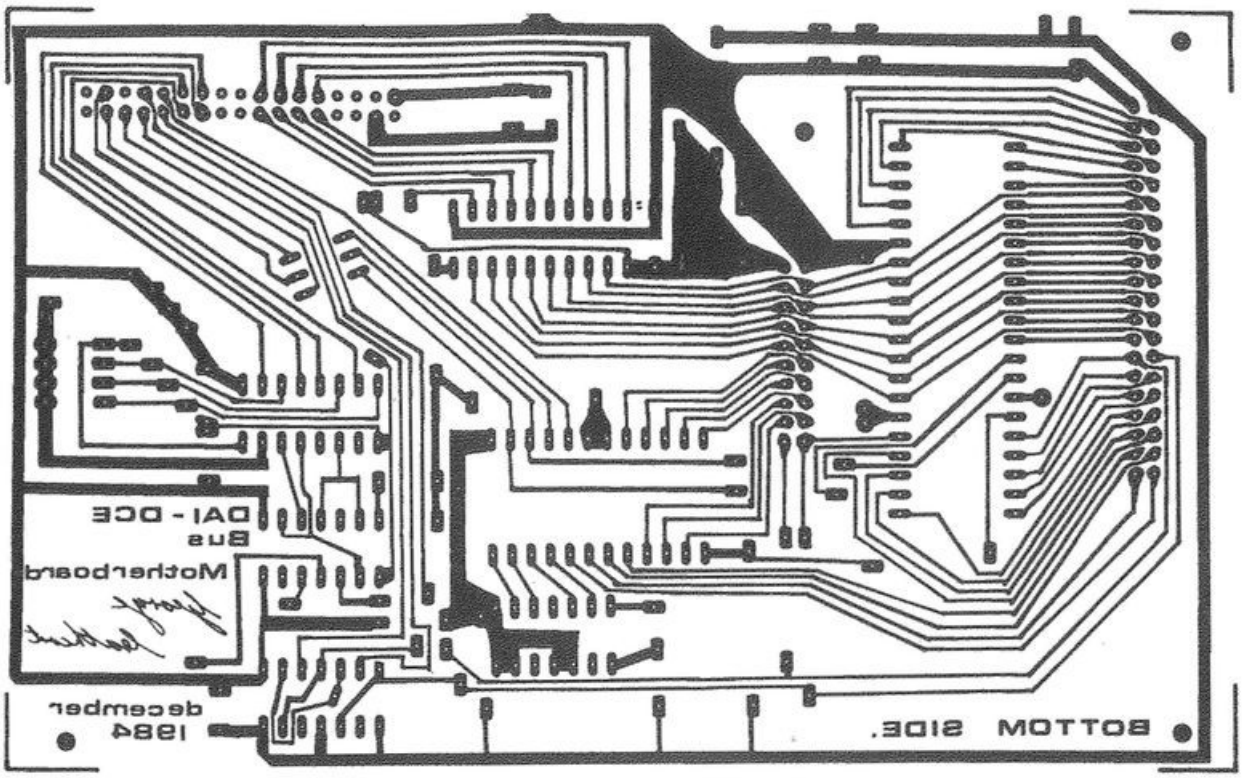
64 K RAM CARD

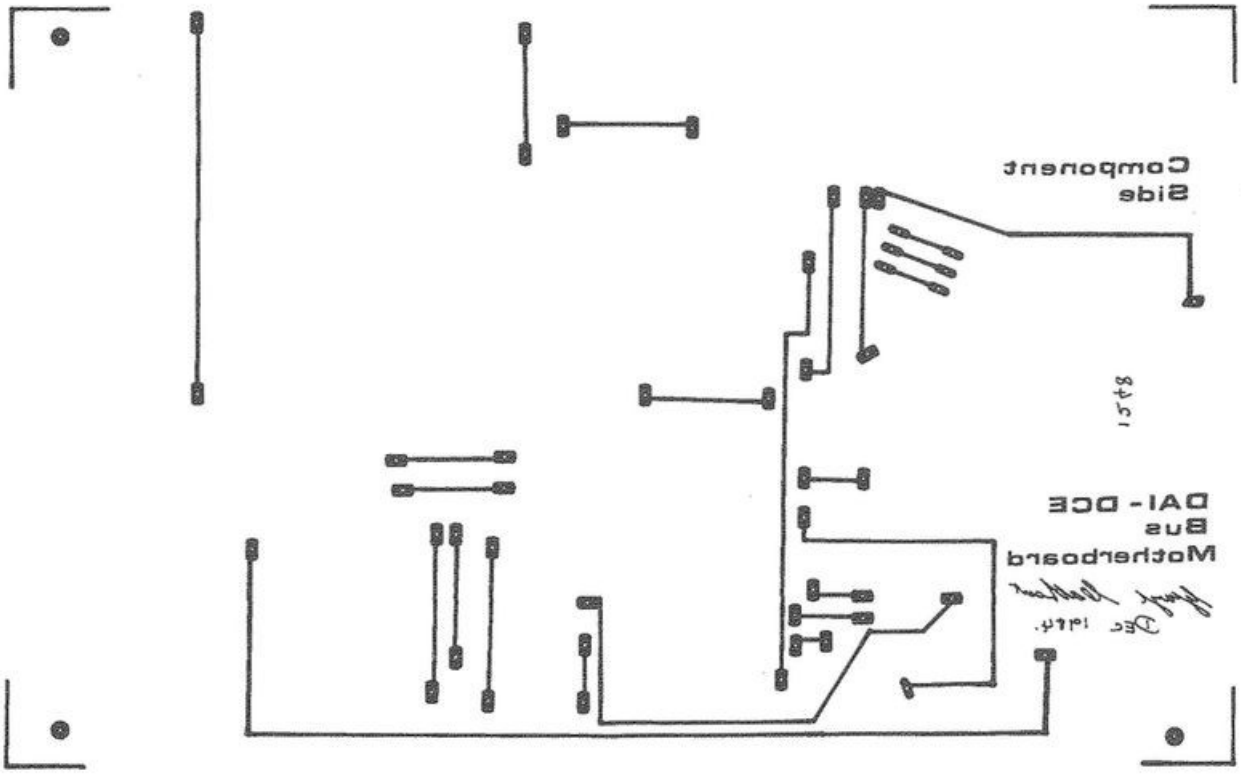


DCE-BUS MOTHERBOARD


DEVICE ADDRESS A3 (21) A2 (22) A1 (23) AØ (24) +5V.
CARD ADDRESS A3 (25) A2 (32) A1 (31) AØ (30)
74LS85
74LS154
74LS20
74LS32
74LS04
SWITCH
2K
+5V
BE (26)
EN
A1
AØ
12 CS (4-15) (4-7 TO 40 PIN HEADER) (8-15 TO 20 PIN HEADER)
CS (0-3)
(28) RD
(27) WR
RD
WR
+5V.
TO 20 PIN HEADER
56K
.47µF
8255 P.P.I.
PORT A
PORT B
PORT C
TO 40 PIN HEADER.
(16) DØ
(14) D1
(12) D2
(10) D3
(9) D4
(11) D5
(13) D6
(15) D7
74LS245
DATA (BUFFERED)
TO 40 PIN HEADER.
= 1/4 74LS32
= 1/6 74LS04
10µF TANT
8 x 0.1µF DISC
DAI - DCE Bus
Motherboard
George Cathcart Dec. 1984.


CONNECTOR 34 PIN
33
34
LINK "A"
10µF (SEE TEXT)
74LS245
0.1µF
74LS85
74LS154
74LS04
74LS20
74LS32
4 DIP SWITCH
1 2 4 8
2K
2K
2K
2K
0.1µF
0.1µF
0.1µF
0.1µF
0.1µF
5V 0V
10µF (SEE TEXT)
0.47µF
8255 P.P.I.
CONNECTOR 20 PIN
56K
CONNECTOR 40 WAY
1
2
39
40
DAI - DCE Bus
Motherboard
Component Overlay


DAI - DCE
Bus
Motherboard
december
1984
BOTTOM SIDE.


Component
side
DAI - DCE
Bus
Motherboard

Power on initialisatie

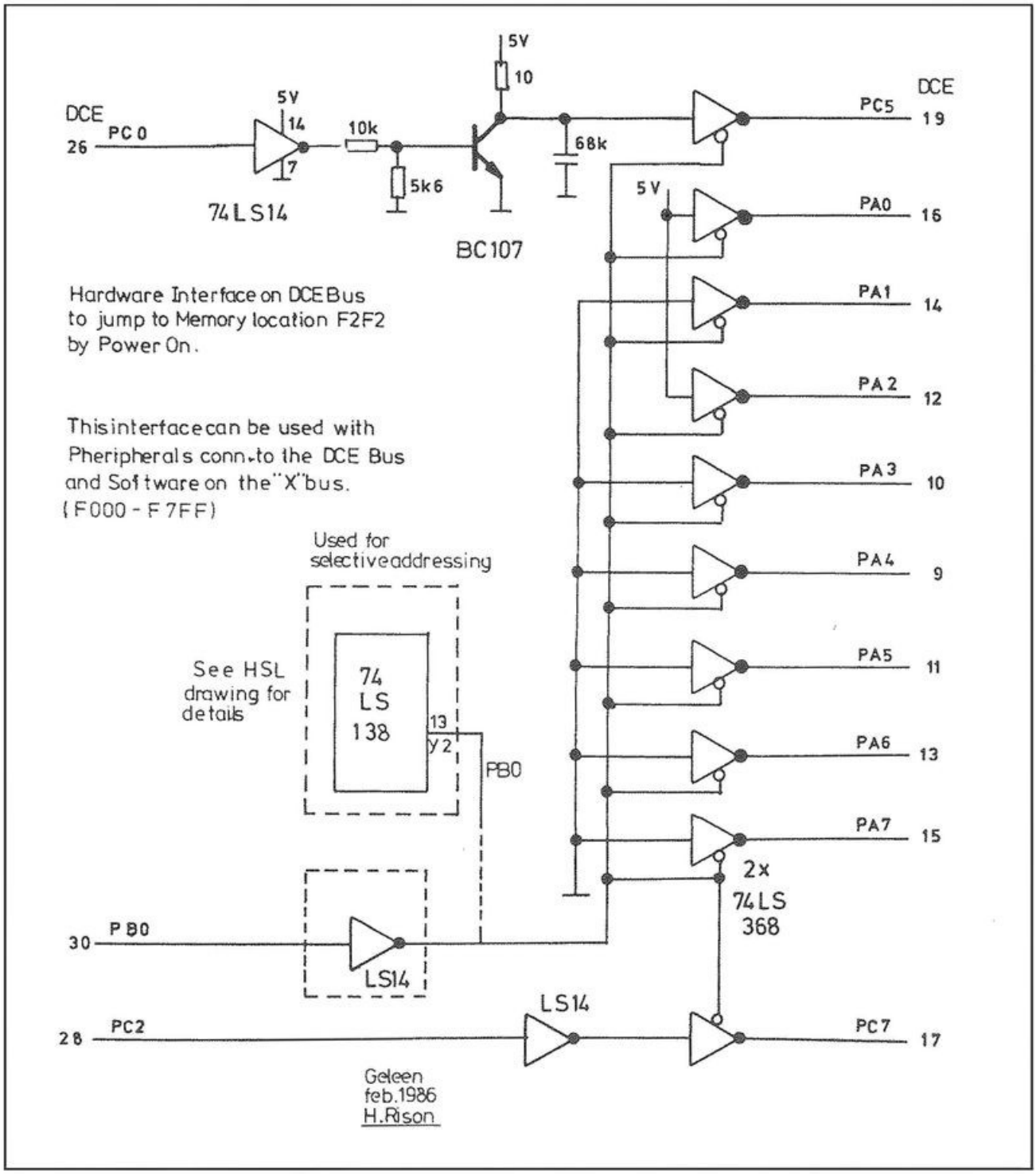
Om bij Power-on te controleren of er iets is aangesloten op de "DCE" bus beschikt de DAI over een routine die controleerd of op de "DCE" bus data staat.
Voor ons geval nemen we aan dat dit #F2F2" moet zijn.
Is dit niet het geval dan gaat hij normaal verder en vergeet de "DCE" bus.
Ziet hij echter wel #F2F2 dan springt hij naar deze memory locatie.
#F2F2 is n.l. niet alleen een Jump instructie maar ook een locatie in een vrij beschikbaar gedeelte van het geheugen waarin we een programma kunnen plaatsen d.m.v. een Eprom of ander geheugen element.
Deze vrije geheugen ruimte loopt van #F000 tot #F7FF en is dus twee Kbyte groot.
We kunnen deze geheugen ruimte "Hardware matig" inschakelen door op de "X" bus connector binnen in de DAI een steekkaartje te plaatsen waarop zich een adres decoder en een Eprom bevinden.
Met deze "X" bus connector zijn nog meerdere mogelijkheden voorhanden die we hier niet zullen bespreken.
Mocht U nu b.v. de High Speed Loader willen bouwen dan kunt U de software hiervoor in dit "X" bus kaartje onderbrengen.
Om het programma automatisch te starten moet U dus bijgaande schakeling op de "DCE" bus aansluiten.

Beschrijving schema

De schakeling bestaat uit een aantal TTL Low Power IC's.
Het wordt geactiveerd door het signaal PB0.
Als deze hoog gaat ("H")worden de inverters van de twee gebruikte 74LS368 actief.
Deze zijn zo geschakeld dat op de "A" poort van de "DCE" bus #F2 komt te staan.
Dit signaal blijft staan totdat PB0 weer laag gaat.
Voor selectieve adressering b.v. voor externe apparatuur kunt U ook een 74LS138 gebruiken zoals beschreven in het artikel over de High Speed Loader.
Het bit PC0 wordt gelijktijdig gezet met PB0 en een time counter wordt gestart.Gedurende de tijd dat deze time counter loopt wordt data van poort A ingelezen en bit PA5 hiervan gecontroleerd of hij "H" is.
Is dit niet het geval dan wordt uit de DCE routine gesprongen en teruggekeerd naar de normale Power-On routine.
We nemen aan dat dit laatste niet het geval is.
PC2 wordt "H"(Request voor data) en de inputs van poort C worden ingelezen. Hier wordt gekeken of bit PC7 "H" is. Is dit het geval dan is de data die nu op de A poort staat correct . Deze data moet nu de instructie bevatten met wat we gaan doen.
In ons geval moeten we naar #F2F2 springen. Deze procedure blijft doorgaan totdat PC5 "H" wordt. Dit gebeurt korte tijd nadat PC0 "L" is geworden.

Dit komt door de vertraging die ontstaat door de transistor BC107 in combinatie met de 10K weerstand en 68K condensator.
PC5 moet zolang "H" blijven totdat we tenminste 4 keer #F2 gelezen hebben n.l. #F2F2 voor de jump instructie en #F2F2 voor het adres.
Dit stukje "Hardware heeft U maar een keer nodig.
Beschikt U reeds over een DCR c.q. Floppy interface of een andere aansluiting op Uw DCE bus dan heeft U dit niet nodig.
Voor de juiste afhandeling van de initialisering van DCE peripherie staat een beschrijving van J.Boerrigter in DAInamic nr 14. blz.28 van jan-febr. 1983.

Geleen, febr. 1986
H.Rison



Micro-ordinateur et douane

Micro-ordinateur et douane : rien à déclarer ? OCT.85

Ce n'est pas tous les jours qu'un parlementaire, fut-il national ou européen, s'intéresse, sans la moindre arrière-pensée électorale, à la vie quotidienne du bon peuple qui l'a élu et aux tracasseries administratives dont il est la victime plus souvent qu'à son tour. Aussi valait-il la peine de signaler la récente intervention de Madame Raymonde Dury, Député au Parlement Européen, à propos des difficultés administratives en matière de douane auxquelles sont - assez fréquemment semble-t-il - astreints les possesseurs d'un «personal computer» - entendez un de ces petits ordinateurs allant du «micro» au «home computer» et que l'on range très facilement dans une valise - lorsque leur vient l'idée, saugrenue aux yeux des gabelous, de vouloir l'emporter au-delà des frontières.

D'où la question de Madame Dury :

La Commission peut-elle confirmer que le «personal computer» destiné à accompagner l'utilisateur lors des voyages à l'étranger soumis à des réglementations douanières comme par exemple établir une déclaration d'exportation provisoire au moment où l'on passe la frontière et au retour établir une déclaration d'importation, cela signifie qu'il est pratiquement impossible de transporter son «personal computer» d'un pays à un autre alors qu'il s'agit précisément de l'intérêt de leur utilisateur. La Commission compte-t-elle prendre des initiatives pour faciliter ce passage de frontière ?

Cette question nous a paru intéressante à deux titres au moins.

D'abord parce que la réponse donnée par Lord Cockfield, le 3 octobre dernier, au nom de la Commission est révélatrice d'un lent - trop lent sans doute - changement des mentalités et des usages en cette matière, et ensuite, parce que la réponse de la Commission intéresse également les possesseurs de matériel vidéo.

Voici cette réponse :

En règle générale, les effets personnels qui accompagnent un voyageur et qui sont destinés à son usage privé bénéficient, à l'entrée d'un Etat membre, sans que le voyageur doive déposer une déclaration écrite, d'une franchise des taxes... La Commission estime que ce principe, déjà acquis en ce qui concerne des articles tels que les appareils de photographie, les caméras, les postes de radio, etc... parfois de valeur relativement élevée mais destinés à l'usage personnel, devrait s'appliquer également aux micro-ordinateurs. Si cela n'est pas encore le cas, cette situation pourrait résulter du caractère relativement nouveau et inconnu de ces appareils.

En d'autres mots, si vous vous partez en vacances avec votre petit matériel vidéo (magnétoscope, caméra, moniteur et, pourquoi pas, matériel de montage) point n'est donc besoin de vous munir du sacro-saint carnet ATA qui a déjà valu quelques cheveux gris à plus d'un confrère cinéaste. Le seul risque auquel vous devrez, peut-être, faire face est d'avoir à convaincre un douanier trop zélé que ce que lui considère comme des engins diaboliques et pernicieux - et donc producteurs de taxes et de paperasserie - ne sont rien d'autres que des effets personnels que vous avez l'habitude de trimballer avec vous...

Un conseil cependant, ayez toujours avec vous les factures qui vous ont été délivrées au moment de l'achat de vos appareils. Croyez-moi, c'est souvent utile.

Cher Monsieur Hermans

Beaucoup de membres de nos Clubs ont eu des difficultés aux frontières entre nos pays Européens, lorsqu'ils emportaient leur Ordinateur Personnel avec eux.
Madame Dury Parlementaire Européen veut nous défendre.
Notre PC doit être considéré comme un BAGAGE NORMAL.
Je suis en contact avec le bureau de Mm Dury : Plus je recevrai de lettres de soutien de ce projet, Plus le dossier de demande que nous allons préparer avec des Services sera Fort devant la Commission Européenne.
Si cet appel peut passer dans la revue nos membres et amis des Pays Bas et d'Allemagne, d'Angleterre et d'Italie pourront également me contacter.
Ceci s'adresse à TOUS ceux qui ont un PERSONAL COMPUTER ou assimilé et qui veulent le transporter dans l'Europe ENTIERE sans Difficultés.
Que ceux qui ONT EU des Difficultés m'écrivent et disent ce qui s'est passé Avec tous ces témoignages et lettres, j'irai chez Mme Dury et nous batirons un dossier puissant.
J'ai déjà contacté la FRANCE, mais toutes les aides seront bienvenues. Un texte d'appel sera prévu pour passer dans la Presse Informatique Spécialisée et traduit dans différentes langues.
Que ceux qui se sentent bons rédacteurs veuillent bien me contacter.
MERCI pour TOUS.

ERIC NEVE
25 HOOGSTRAAT
1512 DWORP
BELGIUM

3-D 0 x 0

```
1     REM ....3D OXO adapted by Lloyd Bailey
3     CLEAR 3000:DIM CHAR$(125),M%(3,3,3),N%(3,3,3),A%(8),B%(8),SP%(4),SR%(
4),SC%(4)
4     BO%=0:C3%=5:C4%=0:XC%=0
5     PRINT CHR$(12):MODE 6A:MODE 6A:COLORT 10 0 0 0:COLORG 10 0 5 15
7     FOR X%=32 TO 125
9     READ CHAR$(X%):NEXT
10    A$="NOUGHTS AND CROSSES":D%=0:CS%=18:L%=0:GOSUB 50000
20    A$="ENTRE Plain,":D%=0:CS%=1:L%=2:GOSUB 50000:A$="Row, Column.":D%=0:
CS%=1:L%=3:GOSUB 50000
30    P2%=RND(3)+49:R2%=RND(3)+49:C2%=RND(3)+49
32    A$=CHR$(P2%)+","+CHR$(R2%)+","+CHR$(C2%):D%=0:CS%=4:L%=4:GOSUB 50000
35    GOSUB 2000:DA%=16:P%=P2%-49:R%=R2%-49:C%=C2%-49:GOSUB 2000
37    INPUT "PRESS [RETURN] TO CONTINUE";V$:PRINT CHR$(12):C3%=10:GOSUB 200
0:XC%=0:C3%=5:DA%=1
40    A$="YOUR ARE NOUGHTS":D%=0:CS%=53:L%=2:GOSUB 50000:A$="I'AM CROSSES":
D%=0:CS%=58:L%=3:GOSUB 50000
50    FOR P%=0 TO 3:FOR R%=0 TO 3:FOR C%=0 TO 3
60    IF P%=R% AND P%=C% THEN 110
70    IF P%=C% AND P%=3-R% THEN 110
80    IF P%=3-C% AND R%=C% THEN 110
90    IF P%=R% AND P%=3-C% THEN 110
100   GOTO 120
110   N%(P%,R%,C%)=10
120   M%(P%,R%,C%)=1:NEXT C%:NEXT R%:NEXT P%
130   FOR N%=0 TO 8:READ A%(N%),B%(N%):NEXT N%
140   PRINT "ENTER YOUR MOVE ";
150   INPUT P%,R%,C%:PRINT :P%=P%-1:R%=R%-1:C%=C%-1
160   IF P%>3 OR R%>3 OR C%>3 OR P%<0 OR R%<0 OR C%<0 THEN 140
170   IF M%(P%,R%,C%)>1 THEN PRINT P%+1;",";R%+1;",";C%+1;" IS OCCUPIED":GO
TO 140
180   M%(P%,R%,C%)=3:GOSUB 1000:GOSUB 2000
185   GOSUB 3000:IF M%(P%,R%,C%)>1 THEN 185
190   PRINT "MY MOVE IS ";P%+1;",";R%+1;",";C%+1:MC%=MC%+2
200   M%(P%,R%,C%)=2:GOSUB 1000:GOSUB 2000
210   IF MC%=64 THEN PRINT "THE GAME IS A DRAW":END
220   GOTO 140
999   REM FIND ON WHICH LINE THE MOVE CELL LIES
1000  FOR Q%=1 TO 3:GOSUB 4000:NEXT Q%
1020  IF P%<>R% AND P%<>C% AND R%<>C% THEN 1060
1030  IF P%=R% THEN Q%=4:GOSUB 4000
1040  IF P%=C% THEN Q%=5:GOSUB 4000
1050  IF R%=C% THEN Q%=6:GOSUB 4000
1060  IF P%<>3-R% AND P%<>3-C% AND R%<>3-C% THEN 1130
1070  IF P%=3-R% THEN Q%=7:GOSUB 4000
1080  IF P%=3-C% THEN Q%=8:GOSUB 4000
1090  IF R%=3-C% THEN Q%=9:GOSUB 4000
1100  IF P%=R% AND P%=3-C% THEN Q%=10:GOSUB 4000
1110  IF P%=C% AND P%=3-R% THEN Q%=11:GOSUB 4000
1120  IF P%=3-C% AND R%=C% THEN Q%=12:GOSUB 4000
1130  IF P%=R% AND R%=C% THEN Q%=13:GOSUB 4000
1140  RETURN
```

```
1999  REM SET UP DATA TO DRAW BOARD
2000  IF BO%=1 THEN 2090:FOR B%=0 TO 3
2010  Y%=200-B%*50:Y1%=160-B%*50
2019  REM DRAW VERTICAL LINES
2020  FOR X%=110 TO 190 STEP 20
2030  X1%=X%+40
2040  DRAW X%,Y% X1%,Y1% 15:NEXT X%
2049  REM DRAW HORIZONTAL LINES
2050  FOR X%=110 TO 150 STEP 10
2060  X1%=X%+80
2070  DRAW X%,Y% X1%,Y% 15
2080  Y%=Y%-10:NEXT X%:NEXT B%
2085  BO%=1:RETURN
2089  REM SET UP DATA FOR NOUGHTS AND CROSS
2090  X1%=116:X2%=122:X3%=128:X4%=134:P2%=P%:R2%=R%:C2%=C%
2100  Y1%=198:Y1%=Y1%-P2%*50:Y2%=Y1%-6:R2%=R2%*10:C2%=C2%*20+R2%
2105  IF DA%=16 THEN 2120
2107  IF DA%=81 THEN 2130
2109  REM DRAW EITHER A NOUGHT OR A CROSS
2110  XC%=XC%+1:IF XC% MOD 2=1 THEN 2130
2119  REM DRAW A CROSS
2120  DRAW X1%+C2%,Y1%-R2% X4%+C2%,Y2%-R2% C3%:DRAW X2%+C2%,Y2%-R2% X3%+C2%
,Y1%-R2% C3%
2125  RETURN
2129  REM DRAW A NOUGHT
2130  DRAW X1%+C2%,Y1%-R2% X2%+C2%,Y2%-R2% C4%:DRAW X3%+C2%,Y1%-R2% X4%+C2%
,Y2%-R2% C4%
2140  DRAW X1%+C2%,Y1%-R2% X3%+C2%,Y1%-R2% C4%:DRAW X2%+C2%,Y2%-R2% X4%+C2%
,Y2%-R2% C4%
2150  RETURN
2999  REM FIND CELL WITH HIGHEST PRIORITY VALUE
3000  HV%=0:FOR P%=0 TO 3:FOR R%=0 TO 3:FOR C%=0 TO 3
3020  IF M%(P%,R%,C%)>1 THEN 3060
3030  IF N%(P%,R%,C%)<HV% THEN 3060
3040  IF N%(P%,R%,C%)>HV% THEN HV%=N%(P%,R%,C%):P1%=P%:R1%=R%:C1%=C%:GOTO 3
060
3050  IF RND(1)>0.5 THEN HV%=N%(P%,R%,C%):P1%=P%:R1%=R%:C1%=C%
3060  NEXT C%:NEXT R%:NEXT P%:P%=P1%:R%=R1%:C%=C1%:RETURN
3999  REM ADD PRIORITY VALUES TO TOTAL PRIORITY VALVES
4000  FOR T%=0 TO 3:P1%=P%:R1%=R%:C1%=C%
4020  ON Q% GOTO 4050,4060,4070
4030  P1%=T%
4040  ON Q%-3 GOTO 4060,4070,4100,4110,4120,4130,4140,4150,4160,4170
4050  P1%=T%:GOTO 4180
4060  R1%=T%:GOTO 4180
4070  C1%=T%:GOTO 4180
4100  P1%=P%:R1%=T%:C1%=T%:GOTO 4180
4110  R1%=3-T%:GOTO 4180
4120  C1%=3-T%:GOTO 4180
4130  P1%=P%:R1%=T%:C1%=3-T%:GOTO 4180
4140  R1%=T%:C1%=3-T%:GOTO 4180
4150  R1%=3-T%:C1%=T%:GOTO 4180
4160  R1%=3-T%:C1%=3-T%:GOTO 4180
4170  R1%=T%:C1%=T%
4180  IF F%=1 THEN N%(P1%,R1%,C1%)=N%(P1%,R1%,C1%)+S%:GOTO 4200
4185  SP%(J%)=P1%:SR%(J%)=R1%:SC%(J%)=C1%:J%=J%+1:IF J%=4 THEN J%=0
4190  DA%=DA%*M%(P1%,R1%,C1%)
4200  NEXT T%:IF F%=0 THEN F%=1:GOSUB 5000:GOTO 4000
4210  F%=0:RETURN
```

```
4999  REM FIND NEW PRIORITY VALUES TO BE ADDED
5000  IF DA%=16 THEN C3%=15:GOSUB 6000:FILL 73,205 225,212 10:A$="* * I WIN
 * *":D%=0:CS%=24:L%=0:GOSUB 50000:GOTO 6010
5020  IF DA%=81 THEN C4%=15:GOSUB 6000:PRINT CHR$(12);"you must have won by
 LUCK or you're a very good player":END
5030  IF DA%=6 AND M%(P%,R%,C%)=2 THEN S%=-10:GOTO 5070
5040  IF DA%/M%(P%,R%,C%)=6 THEN S%=0:GOTO 5070
5050  FOR N%=0 TO 8:IF DA%=A%(N%) THEN S%=B%(N%):GOTO 5070
5060  NEXT N%
5070  DA%=1:RETURN
6000  FOR J%=0 TO 3:P%=SP%(J%):R%=SR%(J%):C%=SC%(J%):GOSUB 2000:NEXT J%:RET
URN
6010  COLORG 10 0 3 15:WAIT TIME 10
6020  COLORG 10 0 5 15:WAIT TIME 10:GOTO 6010
50000 IF (XMAX+25)/4=24 THEN P%=#BFEB-264*L%
50010 IF (XMAX+25)/4=46 THEN P%=#BFEB-506*L%
50020 IF (XMAX+25)/4=90 THEN P%=#BFEB-990*L%
50030 P1%=P%
50040 FOR CA%=0 TO LEN(A$)-1
50050 I$=MID$(A$,CA%,1)
50060 CX%=ASC(I$)
50070 C$=CHAR$(CX%)
50080 FOR CN%=0 TO LEN(C$)-1 STEP 2
50090 NO%=VAL(MID$(C$,CN%,2))
50100 POKE P%-CS%,NO%
50110 P%=P%-(XMAX+25)/4
50120 NEXT CN%
50130 IF D%<>1 THEN P%=P1%-(2*(CA%+1))
50140 IF D%=1 THEN P%=P%-(XMAX+25)
50150 NEXT CA%
50160 RETURN
50200 DATA 00000000000000,04040404000004,10101000000000,10103110311010,0415
2014053004,25250204081919,04101014211813,02040800000000
50210 DATA 04081616160804,04020101010204,00042114210400,00040431040400,0000
000000080816,00000031000000,00000000002424,01010204081616
50220 DATA 14171921251714,04122004040431,14170102040831,30010204021714,0206
1018310202,31163001011714,07081630171714,31010204080808
50230 DATA 14171714171714,14171715011714,00121200121200,0000000800080816,03
040816080403,00003100310000,24040201020424,14170102040004
50240 DATA 14171714161615,14171731171717,30171730171730,14171616161714,2818
1717171828,31161630161631,31161630161616,14171619171715
50250 DATA 17171731171717,14040404040414,07020202021812,17182024201817,1616
1616161631,17272121171717,17252521191917,14171717171714
50260 DATA 30171730161616,1417171717211401,30171730201817,15161614010130,31
040404040404,17171717171714,17171717171004,17171721212717
50270 DATA 17171004101717,17171710040404,31010204081631,14080808080814,1616
0804020101,14020202020214,04142104040404,00000000000031
50280 DATA 08090200000000,00002802141831,16162818181828,00001416161614,0101
0709090907,00001417301614,06080828080808
50285 DATA 000013191717150130
50290 DATA 16163017171717,04001204040414,0002000602020228,16161820282018,12
040404040414,00002621212121,00002226181818,00001417171714
50300 DATA 000022251725221616,000013191719130101,00002224161616,00001516140
130,08083008080906,00001717171714,00001717171004
50305 DATA 00001717212717
50310 DATA 00001710041017,000017171719130130,00003102040831,06080824080806,
04040404040404,12020203020212
50399 REM SET UP DATA FOR BOARD
50400 DATA 3,10,2,14,9,98,4,100,27,900,8,1000,6,-14,18,-98,12,-100
```

PROGRAMMING THE DAI IN MACHINE AND ASSEMBLY LANGUAGES
by C W Read

Part 3 - WRITING A FILE IN DAI's EDITOR

In previous parts of this series two different instructions, LXI H and LHLD were used to load the HL register pair. To demonstrate their differences, suppose an address or value ABCD is stored, low byte (CD) at 320 and high byte at 321. Then:
LXI H 320 (object code 21 20 03) puts 03 in register H and 20 in L.
LHLD 320 (object code 2A 20 03) puts AB in register H and CD in L.
Thus although the LHLD operand was only 320, the instruction has loaded the memory byte from address 320 into L and then automatically stepped to 321 for the memory byte to load into H.
LHLD is useful for loading address pointers but sometimes it is necessary to work with two addresses. On such occasions the XCHG instructon (op code EB) can help. XCHG means Exchange the contents of the two register-pairs DE and HL.
Example:

```
    LHLD    0320H       ;Load HL with address 1 stored at 320/1.
    XCHG                ;Address 1 moves from  HL to DE and anything
                        ; in DE moves to HL.
    LHLD    0300        ;Load HL with address 2 stored at 300/1.
```

LHLD and SHLD are for loading and storing 2-byte addresses. We now consider how to load and store 1-byte values. The instructions are LDA address (Load Accumulator Direct) and STA address (Store Accumulator Direct). Op codes are 3A and 32 respectively.
Example: to move a byte that is in memory at adddress 0075 into the accumulator, decrement the byte and return it to 0075:

```
  Source code      Object code        Comment
    LDA  0075H     3A 75 00         ;Load accumulator with byte from 0075
    DCR A          3D               ;Decrement accumulator
    STA  0075H     32 75 00         ;Store decremented byte at 0075
    RET            C9               ;Return
```

Try it, at say #300. It changes the flashing cursor symbol.

Instructions for indirect loading and storing the accumulator are Load A eXtended and Store A eXtended. They are LDAX B (op code 0A), LDAX D (1A), STAX B (02) and STAX D (12). B and D are register-pairs BC and DE. There is no LDAX H or STAX H instruction. With a valid address in registers BC instruction LDAX B will load the accumulator with the byte at that address, and STAX B will store the accumulator at that address. Here is an example which copies 16 bytes from one memory block (500-50F) to another (520-52F):

```
    MVI L  10H      2E 10          ;16 (#10) into L for byte counter
    LXI B  0500H    01 00 05       ;Source address 500 into BC
    LXI D  0520H    11 20 05       ;Destination address 520 into DE
NXT LDAX B          0A             ;Move byte from source into A
    STAX D          12             ;Move byte to destination
    DCR L           2D             ;Count=count-1
    JZ     End      CA (End)       ;When L=0. Thus no more bytes
    INX B           03             ;Point BC at next source address
    INX D           13             ;Point DE at next destination address
    JMP    NXT      C3 (NXT)       ;Loop to get next byte
End RET             C9             ;Return to main programme
```

In Part 2 we developed an assembly language routine to display a programme menu on the screen. If you now re-load the source code of that programme into your assembler we can add a routine to it. The routine is to allow letters, text or data files to be written in the DAI's edit-mode. Even for those without a printer such a programme is useful for writing files of listed items or data. Writing in the editor has the added advantages of a functioning TAB key and use of the arrowed cursor control keys (with and without SHIFT) for text alterations and screen scrolling. A minor snag is the different effect of the CHARacter DELete key which, instead of deleting the last character typed, deletes whatever is at the flashing cursor signal; the use of a different cursor symbol can serve as a reminder that one is in edit mode. Files created in the editor can be saved and reloaded in Utility as type 1 files, in the same way as one deals with machine language programmes. Machine language routines for loading and saving could be added to the programme; see examples in DAInamic 16 and 22.

A file writing programme will need subroutines to:-

1 Initialise the edit mode.
2 Fetch characters from keyboard.
3 Enter characters in file and display them on screen.
4 Leave the edit mode and return to the menu.

An area of memory will be required for writing, editing and storing the file; let us call it the File Buffer. Its location must be clear of the programme being written (6000-6FFF) and, at least during programme development, it must also be clear of the Assembler and its working area; #400 to #FFF can be used with either assembler. It only provides a small buffer, holding about 60 screen lines of text, but it will suffice for the present. It can be enlarged and/or moved when we have finished with the assembler. We will need to store the addresses of start and end of the File Buffer and of the end of text or data in the buffer. A labelled store can be reserved for each, using an assembler instruction that accepts a conventional 2-byte address in the source programme and converts it to low-high byte order for the object code. The instruction is DBL, or DW, depending on the assembler. Insert the three lines appropriate to your assembler in the source programme immediately before the END instruction.

```
   DNA assembler          SPL assembler          Comment
BufSt   DBL :0400     BufSt  DW  0400H     Address, start of file buffer
BufEnd  DBL :0FFF     BufEnd DW  0FFFH     Address, end of file buffer
TxtEnd  DBL 0         TxtEnd DW  0H        Address, buffer end-of-text
```

We do not know where the end of text will be but that TxtEnd entry will reserve a 2-byte store for its address, initially filling it with 0000.

Initialising the editor and making entries or changes in it involves a special set of machine instructions known as the Restart group (mnemonic RST). The group consists of RST 0, RST 1 etc, to RST 7. These are calls which the microprocessor directs to one of eight permanently reserved addresses in the first 64 bytes of the RAM. At each reserved address 8 bytes are available; not enough for a complete routine but enough to instruct the programme to jump elsewhere to run a routine. RST 0 calls RAM address #0000, RST 1 address #0008 and the one we will be using, RST 5 (op code EF), calls #0028. At #0028 is a JMP to #C6FD for switching to bank 2 of the 4 ROM banks all labelled the same, E000 to EFFF. Bank 2 services the screen and the editor. The RST 5 instruction is followed by a byte of data to indicate which of the screen or editor routines is required. When bank 2 has been selected the data byte is automatically added to E000 to make the address of the required routine. Thus with

instruction RST 5 Data 2A, programme control passes to E000+2A = E02A which is the address of the editor initialisation routine in ROM. All RST instructions save the address to which control must return after completion of the Restart service; also on completion, the E bank in use prior to the RST instruction is brought back into service; for a programme running in Utility that would be E bank 3.

When using the BASIC Editor the DAI creates an edit buffer and moves the Basic programme into it. Our programme will not do that; instead it will give the editor the addresses of our File Buffer, saying in effect, the File Buffer is now the edit buffer. LHLD and SHLD instructions will load and store the addresses. The DAI's editor pointers are at addresses 00A2 (start of buffer), 00A4 (end of text) and 00A6 (end of buffer). The corresponding addresses of our File Buffer have to be stored at those pointers. Here is the routine, labelled InitEd, for setting the pointers and initialising the editor. It too can be inserted in the souce programme, before END. If you want to include the programme line comments in your source programme you will have to shorten them, otherwise the assembler will truncate them.

```
;SET EDIT POINTERS & INITIALISE EDITOR:
InitEd  LHLD    BufSt     ;Buffer start address into registers HL
        SHLD    0A2H      ;Store it in editor pointer, 00A2 and
        SHLD    8AH       ; in screen character-start pointer, 008A.
        LHLD    BufEnd    ;Load end address of file buffer,and
        SHLD    0A6H      ; store it as end of edit buffer.
        LHLD    TxtEnd    ;Load current end-of-text address,and
        SHLD    0A4H      ; store it as editor end-of-text
        RST 5             ;Call Restart 5 service to
        DB      2AH       ; initialise the editor.
        RET               ;Return to main programme
```

The RST 5 call to initialise the editor will display the text it finds in the file buffer, on the screen. The byte after the last text character in the buffer is a zero, which marks the end of text. When a new file is to be written the first two bytes in the buffer are made zero; thus the editor finds an empty buffer and starts with a clear screen. As characters are entered into the buffer the 0 is moved on, thus still marking the end-of-text. We will programme that now. Label the routine NEW. Load the buffer address into register-pair HL and put 0 into memory at that address and at that address+1. Put the routine at the end of the source programme.

```
;WRITE A NEW FILE:
NEW     LHLD    BufSt     ;Start address of file buffer into HL pair
        XRA A             ;Zero the accumulator
        MOV M,A           ;Move zero to byte 1 of file buffer
        INX H             ;Move HL to next byte
        MOV M,A           ; and insert another zero
        SHLD    TxtEnd    ;Save this address in the TxtEnd store
```

This routine can be continued with a call to the previous one to initialise the editor, followed by a sequence to fetch characters from the keyboard and enter them in the buffer. Label the keyboard sequence KEYBD3. It can use the ROM call GETC that we used earlier but this time the BREAK key will be used to escape from the editor, via a routine labelled BREAK. Another RST 5 instruction is used to insert a character in the buffer but this one requires the character to be waiting in the accumulator. The GETC routine returns with the keyed character in the accumulator. Then the RST 5 data 2D instruction puts that character in

the buffer and on the screen. Here is the contination of the routine labelled NEW:

```
        CALL    InitEd    ;To set edit pointers & initialise editor
KEYBD3  CALL    GETC      ;Get key pressed. Return with character in A
        JC      BREAK     ;When BREAK key pressed to leave editor
        JZ      KEYBD3    ;Loop back and look again when no key pressed
INSERT  RST 5             ;Keyed character is in register A
        DB      2DH       ;Insert it into Edit buffer.
        JMP     KEYBD3    ;Go back for next character
```

The BREAK routine extracts the end of text address from the edit pointer and saves it in the TxtEnd store, then jumps back to MENU to await the next command.

```
BREAK   LHLD    0A4H      ;Get current end-of-text address
        SHLD    TxtEnd    ; and save it.
        JMP     MENU
;
```

Having completed routine NEW it is necessary to gain access to it from the Menu. Insert in the KEYBD1 routine, at the appropriate place:

```
        CPI     '4'
        JZ      NEW
```

and in the MENU$ data:

```
        DB      0DH
        DB      '  Key 4   TO WRITE A NEW FILE'  or similar phrase.
```

Before assembling the programme check it for errors and save the source on cassette; then assemble, switch to Utility and run (G6000). If you switch to Utility after having written a few words of text you will be able to see your text stored in ASCII form by giving the Display command D400 4FF. There is no way of getting the actual text back on the screen yet as Menu item 3 is still inoperative. Can you write the necessary routine for item 3? It will only require a dozen lines of additional source code because some of the sub-sections needed have already been written for the NEW routine. All the instructions needed have already been described. The source programme should still be in the assembler. Remove the existing EDIT1 NOP line. Label your routine EDIT1 and enter it after the BREAK routine. Your first instruction should put the buffer start address in HL, as done in NEW. The routine should then go through the buffer file a character at a time looking for the 00 which indicates end-of-text; can you remember from Part 2 how the ROM routine PMSG uses ORA A to look for 00 at the end of a string? Having found the required end, store its address in TxtEnd, initialise the editor and jump to KEYBD3. Check for errors, assemble, switch to Utility and you should be able to display and/or edit the first text. If a blank edit screen appears there are probably a couple of 00s at the start of the text area in the buffer, caused through trying in vain to display the text by calling NEW. Go to Utility and check with D400 40F. If 00s are there, use the Substitute facility to replace them with 0Ds (Carriage Returns). The programme should now work and the extra carriage returns can be removed by editing them out with the CHAR DEL key. If you have difficulty in writing the routine you will find a version at the end of this article.

There now follows a few general hints that you may find useful when developing programmes and working in Utility:

(a) Trial runs of a programme are frequently followed by switching back

to assembler, to make small changes to the source code. The inclusion of a few temporary lines in the programme can speed return to the assembler. For example, in our current programme the Menu routine could include: CPI 'A', JZ 8500H (SPL start address) or JZ :1100 (DNA start address). Thus, while the menu is on screen, key A provides an immediate jump to the assembler.

(b) When using the Utility Monitor the normal Escape from a Utility command is via the Cursor Left key. However some commands can actually start operating as soon as the cursor key is pressed and before Escape becomes effective. This is the sort of action which may overwrite programmes with rubbish and turn the screen into a kaleidoscope. Escape from a Go command with an incomplete address is particularly prone to such calamities. If you must escape from a situation where the DAI is waiting for another address digit, it is safer to type in a letter, say Q, (not A to F) and let Utility's error system throw out the command on the grounds of an invalid address.

(c) If running a machine language programme results in a STACK OVERFLOW message the cause may be the same that can arise with a BASIC programme, namely a Jump out of a called subroutine instead of a RETurn. Another possible cause is a PUSH without its necessary POP. Inspection of the source programme will usually reveal the fault, as long as the programme is short and uncomplicated. For long programmes it is easier to switch immediately to Utility and display the Stack with the command DF800 F8FF. If you now see an address that is repeated very many times it is either the return address for which your programme has not given a RET, or it is the contents of a PUSHed register which was never POPped. In either case the address should guide you to that part of the source programme wherein the error lies.

(d) If you are doubtful about a section of programming it is always worthwhile entering a short example at say #300 with Utility S commands, and then using the Look facility to see if the registers are being used as you intended. Do not forget to key Z3 before giving the L comand!

Finally, here is the promised EDIT routine:

```
EDIT1   LHLD  BufSt     ;Load HL with start address of the buffer
EDIT2   INX H           ;Step to next address
        MOV A,M         ;Move character from there to A
        ORA A           ;Test for a character or 00
        JNZ   EDIT2     ;Character found, so try the next address
        SHLD  TxtEnd    ;00 found. End-of-text found. Save its address
        CALL  InitEd    ;Start editor, display text
        JMP   KEYBD3    ;Ready for editing.
```

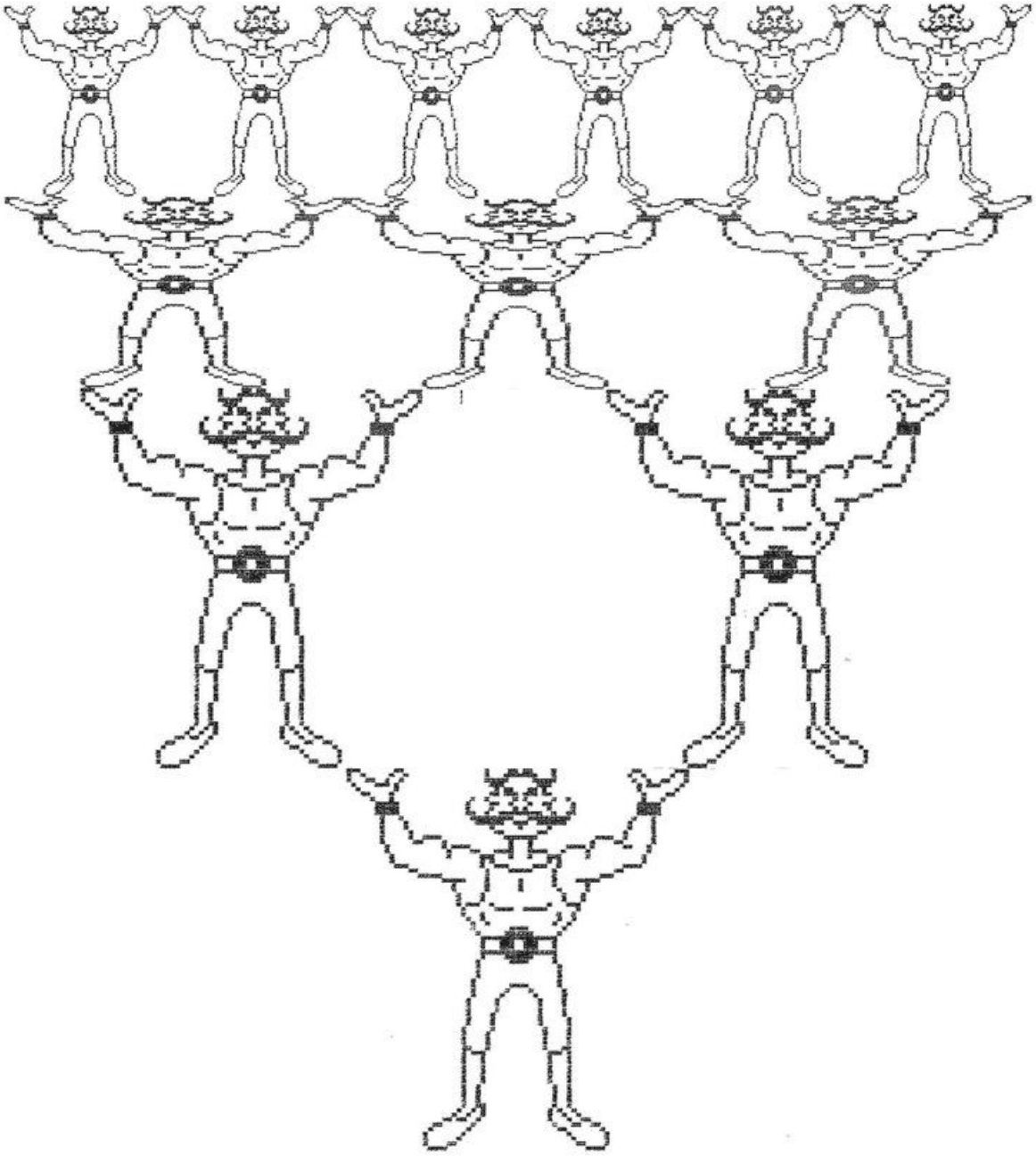
We should ensure that the above routine does not search for 00 beyond the end of the buffer, into the rest of the RAM. That can be done by putting 00 in the last byte of the buffer. Do it at the start of the programme, after the ORG directive, with the following entry:-

```
        ORG
        LHLD  BufEnd  ;Load buffer-end address into HL pair
        XRA A         ;Zero the accumulator
        MOV M,A       ;Move the zero into end of buffer.
        ...
```

As the BREAK key is used for escaping from both NEW and EDIT routines to redisplay the menu, you may wish to display a note to that effect at the foot of the menu; it would add a "user-friendly" touch.

(to be contined)

SUPERFONT



SUPERFONT .. SOURCE OF IMAGINATION!

SUPERFONT .. SOURCE OF IMAGINATION!

SUPERFONT .. SOURCE OF IMAGINATION!

SUPERFONT .. SOURCE OF IMAGINATION!

SUPERFONT .. SOURCE OF IMAGINATION!

SUPERFONT .. SOURCE OF IMAGINATION

SUPERFONT .. SOURCE OF IMAGINATION

GHOSTBUSTERS