N A S C O M N Y T

NR: 6
1. årgang

NASCOM BRUGERGRUPPE
Sidevolden 23
2730 Herlev
Giro 674 26 02

Oktober 1980.

Det er rart på dette sted at bringe meddelelse om, at foreningen har nær 70 betalende medlemmer og 5 interesserede, som ikke endnu har besluttet at knytte sig til foreningen.

Men det kan ikke ses i min postkasse!!!

Nu må det være tiden at fortælle redaktøren om hans linie er god/nogenlunde/dårlig. Bladet bliver kun bedre af medlemsreaktioner, herunder hører også programmer, ideer og tips. Du skal ikke være nervøs for, at det har de andre allerede fundet ud af, for det har de ikke!!!

Derefter må jeg påpege foreningens vedtægt paragraf 2, der siger:" Foreningens formål er at skabe kontakt mellem brugere af Z80 familien og speciel NASCOM's produkter." Denne formålsparagraf bliver blokkeret af dette blads hoved. Jeg efterlyser et nyt hoved til bladet i stil med: Z80 ORIENTERING udgivet af NASCOM BRUGERGRUPPE.

Reaktioner ønskelig.

si'r Asbjørn.

ps. Jeg skal minde om møde på Pædagogisk Central, Rustenborgvej 1, Lyngby. TORSDAG den 9.10.1980 kl. 19.00 .

****

****

AUTOMATISK STOP-START

Styring af båndoptager ved hjælp af loaddioden. LD1 er den lysdiode der lyser op ved funktionerne: W, G, L, CSAVE og CLOAD. Du har med dette kredsløb mulighed for at køre automatisk med din båndcasette. En funktion som: W 1000 2000 vil kunne klares uden brug af det ekstra kredsløb, jeg har påbygget. Straks sværere bliver det med funktionen: W 1000 2000 1000 eller funktionen: G 1000 2000 1000. Båndoptageren vil da stoppe for tidligt, når dioden slukker, og du kan ikke få programmet til at virke. Systemet virker på følgende måde: Ved tryk på f.eks. W 1000 2000 1000 vil spændingen på basis af TR 1 gå lav. Dette får kollektorspændingen på samme transistor til at gå høj. G 1 output er nu lav, og transistor TR 2 vil lede, som får båndoptageren til at starte. Når nu LD 1 slukker vil det få den monostabile 74C221 til at starte, og dens output vil være lav i ca 0,6 til 2 sekunder. Herefter vil begge dioder gå højt og TR 2 vil ikke kunne lede mere. Dette får atter båndoptageren til at stoppe, og vi har opnået den forsinkelse, som var nødvendig. Alle disse funktioner forudsætter naturligvis at S 3 er afbrudt. Med S 3 tændt vil alt være som tidligere, men S 3 er praktisk at have ved arbejdet med båndoptageren.

	Normal	R, W, L	(Normal)	Normal
Basis TR1	høj	lav	høj	høj
Collek -	lav	høj	lav	lav
74C221 p12	høj	høj	lav	høj
G 1 out	høj	lav	høj	høj
Basis TR2	høj	lav	lav	høj
LED 1	Lyser ej	Lyser	Lyser ej	Lyser ej
Båndopt	Kører ej	Kører	Kører	Kører ej

Det er forudsat, at betjeningskanpperne er indtrykket på båndopt. Bemærk at båndoptagerens DIN-stik er omloddet, således at start og stop kan foretages udefra.
S 3 sluttet: Båndoptager i normal kørset.

Ole H.

Se diagram på næste side

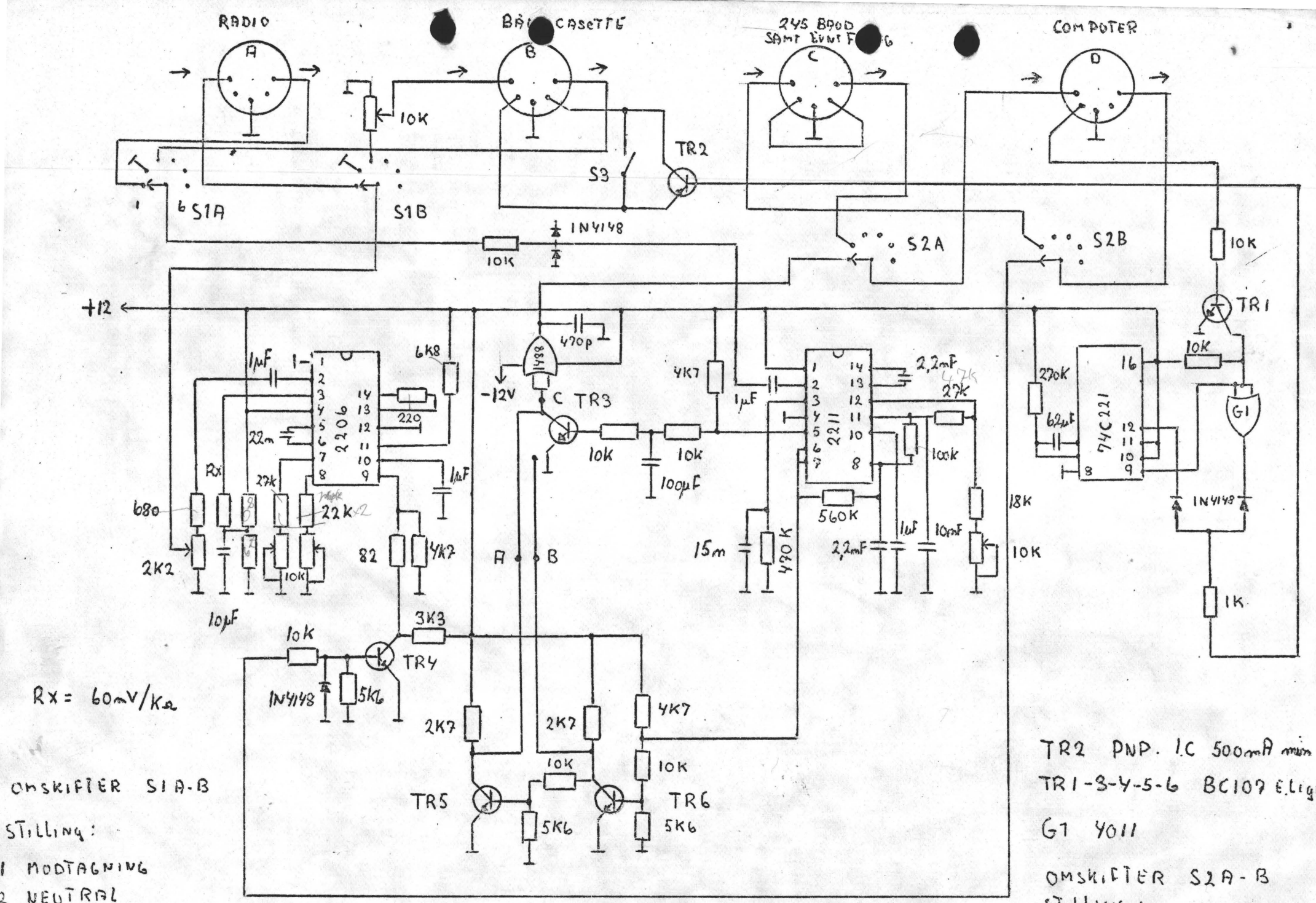
2400 Baud kassette på N2.

Hermed følger en nem lille sag til omskiftning af baudrate på N2.

Forbind TP20 med TP4 og TP21 med TP5. Hvis du vælger external UART clock løber dit kassetteinterface med 2400 baud. Omskifterne på LSW2 skal være:

Omskifter	1	2	3	4	5	6
stilling	-	op	op	-	op	op

Hvor omskifterne 1, 2 og 3 vælger transmissionshastighed til kassetten. 4, 5 og 6 bestemmer modtage hastigheden. 7 skal være nede for at vælge kassette (op vil vælge TTY).



Rx = 60mV/Kø

OMSKIFTER S1A-B

STILLING:

1 MODTAGNING
2 NEUTRAL
3 SENDING

TR2 PNP. IC 500mA min
TR1-3-4-5-6 BC109 E.Lign
G1 4011

OMSKIFTER S2A-B
STILLING:
1 NORMAL
2 NASCOM-STANDART ELLER EVNT

Nascom brugergruppe NASCOM NYT 6 Side 3

ANMELDELSE AF NASPEN.

En 2k wordprossor,der kræver mellem 16k og 60k RAM og som i forbindelse med en printer er et godt og billigt redskab til små kontorer m.v., samt til brevveksling på bånd!

I NASPEN finder man gode faciliteter, der sparer tid, når man er to-fingers maskinskriver. Man undgår en masse retninger med viskelæder, tilbagetastning og gentastning, for ikke at nævne stavefejl og dårlig sætningsopbygning. NASPEN findes i 2 versioner: en til T4/B-bug og en til NAS- SYS 1, og kun i EPROM.

Efter koldstart udskrives følgende overskrift: "NASPEN VS. 1 LINE 1 12000 FREE" og et semikolon nederst til venstre.For at starte indtastningen trykkes -A- for append (her skal det bemærkes, at man aldrig skal bruge -ENTER- . Alle kommandoer virker umiddelbart ved nedtrykningen). Derefter indsætter man den ønskede tekst og -ESCAPE- afslutter indtastningen.

Ved initalisationen placeres en prompt °;°, hvorefter den vil acceptere en kommando,denne bliver holdt under udførelsen og skifter en linie op ved færdiggørelsen. Dette er en god ting, dan den letter ens hukommelse under lange indtastninger.

Når teksten er inde er første trin på editeringens vej at °zeroe° -Z- sin tekst. Derefter -L- der formaterer linielængde til en forudvalgt linielængde (72 - kan ændres af brugeren undervejs). Dernæst at rette eventuelle fejl -c- eller tilføje bogstav -i-, eller tilføje hele sætninger -I-. Man kan tilføje mellemrum, flytte blokke, slette bogstaver -d- eller hele linier -D-,dog kan blokke ikke slettes, kun fra cursor og ud -K-, -Y-.

Takket være autorepeat (valgbar hastighed)kan cursoren flyttes rundt i teksten (både på og udenfor skærmen!) i enkelte trin eller i 10'er skridt. Teksten kan flyttes gennen vinduet (VDU) linie efter linie eller side efter side både frem og tilbage (valgbar sidelængde).

Autorepeat virker på alle taster. Det kan være en ulempe, hvis man falder i staver med fingeren på tasten! Men det er så godt indrette, at der ingen alvorlige skader kan ske. Da -K- skal efterfølges af en bekræftigelse -Y- for at fungere.

Tekststrenge kan finde i bufferen ved -f- fulgt af den ønskede tekst enten fra cursorposition eller fra start -F-. Derefter kan foretages rettelserog tilføjelser, hvorefter man kan fortsætte søgningen ved -a-. Det er bedst at søge efter ord og ikke sætninger, da NASPEN er meget nøjeregnende med mellemrum og orddelinger.

At indsætte en blok gøres nemt på følgende måde: -A-, append, hvorefter teksten tilføjes i slutningen omgivet af "7BH" og "7DH" 's parenteser. Så flytte cursor til det ønskede indsættelsessted og så -M-. Derefter søges efter "7BH" og man dræber -K- fra dette sted og ud. Dette er hurtigere end at bruge -I-, der skal flytte samtlige karakterer i den underliggende tekst efter hver indtastning, hvis det er langt kan det godt føles lang tid hver gang.

Indtil dette tidspunkt har man formateret teksten så den passer til skærmen og det ville nu være en god ide at gemme informationer på bånd. Men her viser sig en begrænsning, idet files ikke kan gemmes og genindspilles efter navne, en utilgivelig fejl!

Ved genindspilning -R- overspilles tidlige tekst, og ved -J- indspilles teksten efter den i bufferen værende.

En artikel eller brev kan efter formatering straks udskri-

ves på printer. Men -S- kommandoen pynter svært på udskriften, idet den retter teksten til i begge sider, ved at indsætte maximalt et mellemrum til hver af de værende, startende skiftevis fra hver side! Støre mellemrumsstykker (tab) berøres ikke af den funktion, og et dobbelt -ENTER- vil blive opfattet som nyt afsnit. Men et enkelt -ENTER- bliver betragtet som sminke ved indtastningen.

Hvis linielængden ikke går op i antallet af mellemrum og længden af ordene udskrives en fejlmeddelelse, hvorefter det angives, hvor der skulle være et mellemrum. Desvære er det ikke nok at tilføje en '-' ved et foranstillet naturligt delested i ordet, man skal også tilføje et mellemrum. Herefter fortsættes med -s-.

Hvis man vil ændre lineafstanden, kan -X- fjerne allede ekstra indsatte mellemrum, men husk at rette orddelingerne tilbage til deres oprindelige form.

Det er muligt at standse printeren efter et bestemt antal linier (opr. 68 - svarende til et A4 ark) ved at tilføje manuelt eller automatisk et symbol (bell), dette kan ligeledes benyttes, hvis man ønsker at skifte skrifttype. Dog slettes disse tegn ved brug af -X-,-S-og -s-, så hvis det er printhovedskift er det en fordel at skifte symbol.

Udover de nævnte er der funktionstaster til mange formål, bl.a. -N- (til NAS-SYS) og en fast inddeling af tab på 8 pladser.

Af svagheder skal også nævnes det spild af øverste linie med meddelelse, som allerede nævnt. Det kunne bruges til noget mere fornuftigt f.eks. aktuelt linienr., sidelængde, cursorposition og iganværende kommando. Det vil forøge skærmen med 2 linier.

Hertil skal føjes cursorens (en venstre pil) forskellige betydninger: referende til den position den er på, pegende på den foranstillede karakter og pegende på 2 karakterer tilvenstre ved deling af ord, alt efter hvilken mode maskinen er i!

Alt i alt en god funktionel grundversion af en wordprossor med begrænsninger, der trods alt ikke er store, når man tager pris og størrelse i betragtning.

Den er simpel i brug og man lærer hurtigt at arbejde med den. Den ville kunne finde anvendelse på mange områder, hvor Nascom allerede eksisterer.

En lidt forbedret udgave ville kunne tilgodese de sidste kritiske bemærkninger, men selv i den nuværende form er den sine 500 kr. (30 pund) værd, samtidig med den er en god introduktionen til word-processing.

Skrevet af Erik Hansen, fortsættes.

Da det ved samtaler m.m. er blevet klart,at der er mange,der har svært ved at komme om ved de udenlandsk sprogede manuals og lærebøger,vil vi her søge at råde bod på manglen af dansk forklaring til u-computerbrugen,da også mange uden de mangeårige erfaringer har meldt sig i klubben,vil vi begynde med det mest elemntære,men kun i beskeden omfang,og så når behovet for en uddybning melder sig give denne.

Forklaringen vil blive søgt holdt i så generelle forhold,som muligt,men vil blive givet med Nascom-computeren,som baggrund.

I første række vil vi holde os til Nacom 1,med uudvidet RAM. og med T 2. monitoren.

Dette gøres for at begrænse forklaringen,og fordi der erfaringsmæssigt er en del,der har denne kombination med de manuals,mens de mere erfarne og dem med de næste udvidelser,OG FOR DEN SAG SKYLD MED ANDRE SYSTEMER,her forhåbentlig vil kunne hente inspiration til sammenstilling med deres systemer,og komme med kommentarer,vejledninger og eventuelle korrektioner,der måtte være begrundede.

I denne forklaring vil der generelt blive anvendt hexadecimal notation UDEN index,mens andre talsystemer vil blive markerede.

således:

10_2=2,binært eller i totalsnotation,mens 10_{10} er ti i den vante titalsnotation,hvor derimod 1Ø er lig med 1_{16} IDET vi jo i computeren anvender denne hexadecimale noation. altså: 1 2 3 4 5 6 7 8 9 A B C D E F 1Ø 11 12 13 14 o.s.v (bør læses 1Ø Enn-nul 11 = Een een 12 = Een to o.s.v.) er den naturlige talrække at anvende,og det skal kort

begrundes i det forhold,at computeren for at kunne arbejde helt præcist kun arbejder med de to elektriske tilstande ÅBEN el. LUKKET,dette medfører,at maskinen egentligt kun behandler 0 og 1'ere,hvad der er meget uoverskueligt for mennesker,der skal betjene apparaterne.

Derfor har man søgt en løsning på at skabe et system,der gjorde maskinsproget til en mere menneskelig notation,en løsning fandt man i BCD - noatationen,idet man ved at skrive de enkelte titals tal i grupper på hver fire binære cifre i en kombination kunne store tal bedre overeses således:

123456 0001 0010 0011 0100 0101 0110 o.s.v.

op til ti = 1010,men dette medførte,at der jo blev nogle muligheder,der blev "forærede",og som hvis de ikke blev benyttede ville betyde væsentlig flere kredsløb og også en større behandlingstid i driften.Altså er det naturligt at anvende:

1011 = elleve,1100 = tolv,1101 = tretten,1110 = fjorten 1111 = femten, Eller vi tæller 1 2 3 4 5 6 7 8 9 A B C D E F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1F 20 (huskede du at sige Een nul, Een een o.s.v.)

Således organiseres en computers memory (hukommelse fra 0000 til FFFF (se Nascom construction manual, herefter kun kaldet NCM. side 31).

Ved lidt regnearbejde findes at FFFF = 65 535_{10} og man kalder 1o $o24_{10}$ for 1.K. enhed,således programmerer Z 80 processoren altså umiddelbart 64 K.hukommelse.

Denne organisation er i Nascom valgt således:

0000 - 03FF =1k = T 2 monitoren

0400 - 07FF =1k = udvidelse (T 4, og Nassys)

Ø8ØØ - ØBFF = 1k = Video RAM

ØCØØ - ØFFF = 1k = Brugers RAM

1ØØØ - FFFF =6Ok = for udvidelser.

Det skal dog bemærkes at T 2 og T4 monitorerne bruger området ØCØØ - ØC49 og Nassys ØCØØ - ØC79,som arbejdsområder,hvorfor disse ikke er til rådighed for brugers programmering (næste afsnit).

Andre computere kan have en anden RAM (Random Access Memory),ROM (Reed Only Memory) organisation.

I Z 80 u-processoeren er der organiseret følgende registre:

A = accumulator (og et "skyggeregister",der kaldes A')

F = Flags (og et F'),der fortæller om status således:

C = Carry (mente eller lån)

N = Add/subtract

P/V= Parity/Overflow (Paritet = 2-lighed)

H = Half-Carry

Z = Zero (Nul)

S = Sign (fortegn)

X = ikke brugt. Flagene fortæller ved deres tilstand i F - registeret om sidste matematiske operations virkninger.Det er et otte-bits register der er organiseret så:

bit	7	6	5	4	3	2	1	0
	S	Z	X	H	X	P/V	N	C

Ved T 2 (og T 4,der dog også viser flagene ved bogstaver)vil ved single step (enkelttrins eksekuering af prgr.) altså flg. F.-tilstand

0 1 X 0 X 0 0 1 vises som

4 1

F A K U L T E T .

Hvad er det største tal man kan tage fakultet af?. Det spørgsmål kan man jo ikke svare på! Det komme da an på, hvilket hjælpemiddel man vil anvende. På en lommeregner kan man tage 69!, da det bliver 1,7112*10↑98, dvs. et 99-cifret tal. Det er jo ikke hver dag man ser så stort et tal, og da slet ikke for de sidste cifres vedkommende.

Kan dette også klares på vores computer? Man leder forgæves efter en !-knap, men må lave et lille program. Dette får da højst 6 cifre og vi ryger ind i overflow om kring de 30.

Der må laves et bedre program: Først sikre man sig en rimeligt udgangsværdi (linie 50 -60), der også gælder for N=0 (linie 80). Derefter går man i gang med at multiplicere med faktorerne i stigende rækkefølge med mente (ME).

```
ASB. LIND
Z
Ok
LIST

10 REM ****************************************
20 REM *   BEREGNER N! MED MULTIPRÆCISION     *
25 REM *          ASBJØRN LIND 29.9.80        *
30 REM ****************************************
35 CLS
40 DIM FAKUL(160)
50 INPUT "HELT TAL N TIL N!";N
60 IF N<0 OR N>100 OR INT(N)<>N GOTO 50
70 REM
80 FAKUL(1)=1: CI=1
90 FOR FK=1 TO N
100 CF=0: ME=0
110 CF=CF+1
120 FAKUL(CF)=FAKUL(CF)*FK+ME
130 ME=INT(FAKUL(CF)/10)
140 FAKUL(CF)=FAKUL(CF)-10*ME
150 IF CF<CI OR ME<>0 GOTO 110
160 CI=CF
170 NEXT
180 REM
190 PRINT
200 PRINT
210 PRINT "Når N=";N;" er"
220 PRINT "    N!="
230 CM=CI-INT(CI/6)*6
235 PRINT TAB(26-CM*3);
240 FOR I=CI TO CI-CM+1 STEP-1
250 PRINT FAKUL(I);
260 NEXT
265 PRINT
267 IF CM=CI THEN END
270 PRINT TAB(8);
280 NR=0
290 FOR I=CI-CM TO 1 STEP -1
300 NR=NR+1
310 PRINT FAKUL(I);
320 IF NR=6 THEN PRINT:NR=0:PRINT TAB(8);
330 NEXT
```

Ved hver multiplikation finder vi den næste mente ved at fjerne det bagerste ciffer i det aktuelle produkt (linie 130), hvorefter vi sikre os, at det alligevel er dette ciffer, der står tilbage ved at fjerne de eventuelle andre cifre i produktet(linie 140). Antallet af cifre vokser (linie 160), men til sidst er vi færdige, og det er nu blot et spørgsmål om at få resultatet til at stå pænt (linie 190-330).

Her skal bemærkes, at det kan være lige så omfattende som selve beregningen!

Det er ikke det hurtigste program, så hvis man skal skynde sig, kan man spare til ved at bruge alle 6 cifre i hvert element. Men da går overskueligheden tabt.

```
Ok
RUN
HELT TAL N TIL N!? 0

Når N= 0  er
    N!=
                          1

Ok
RUN
HELT TAL N TIL N!? 13

Når N= 13  er
    N!=
                  6  2  2  7
            0  2  0  8  0  0

Ok
```

```
Ok
RUN
HELT TAL N TIL N!? 100

Når N= 100  er
    N!=
                                  9  3
               3  2  6  2  1  5
               4  4  3  9  4  4
               1  5  2  6  8  1
               6  9  9  2  3  8
               8  5  6  2  6  6
               7  0  0  4  9  0
               7  1  5  9  6  8
               2  6  4  3  8  1
               6  2  1  4  6  8
               5  9  2  9  6  3
               8  9  5  2  1  7
               5  9  9  9  9  3
               2  2  9  9  1  5
               6  0  8  9  4  1
               4  6  3  9  7  6
               1  5  6  5  1  8
               2  8  6  2  5  3
               6  9  7  9  2  0
               8  2  7  2  2  3
               7  5  8  2  5  1
               1  8  5  2  1  0
               9  1  6  8  6  4
               0  0  0  0  0  0
               0  0  0  0  0  0
               0  0  0  0  0  0
               0  0  0  0  0  0

Ok
```

DOTS

Der er vist interesse for at få at vide, hvordan en karakter- og tegngenerator er opbygget. Det vil kunne bruges umiddelbart på N2, da den kan modtage 2716 ROM. På N1 skal der ske en ombygning (vi vil offentliggøre et diagram næste gang) for tilføjelse af ny eller anden generator baseret på 2716.

Der vil ikke her blive givet forklaring på, hvordan tegnene kommer ud på TV-skærmen. Men det skal vides, at der bruges 7 addresselinier til formålet.

Derimod er det vigtigt at vide, at hver karakter består af 16 rækker med hver 8 punkter (dots). Hver dot er et bit af en byte, så en hel karakter kan blive repræsenteret af ialt 16 bytes. Da der er 128 karakterer i en karakter-ROM, må den indeholde 2K lager. Dette passer jo lige på en 2716, så nu er problemet, hvordan programmeres den ?

For at konstruere et nyt tegn tag et stykket ternet papir og afgræns et rektangulært felt på 8*16 kvadrater. Udfyld firkanten med den ønskede figur, gå et par skridt bagud og betragt figuren. Er den OK ? - så fortsæt.

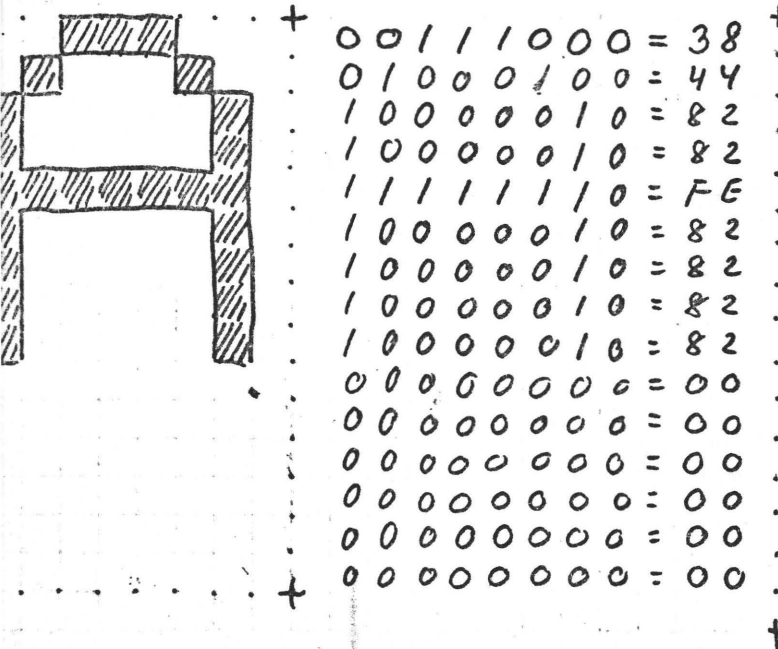
Ved programmeringen skal du huske, at binært '0' er sort og binært '1' er hvid. (Glem heller ikke at TV viser hvid på sort.

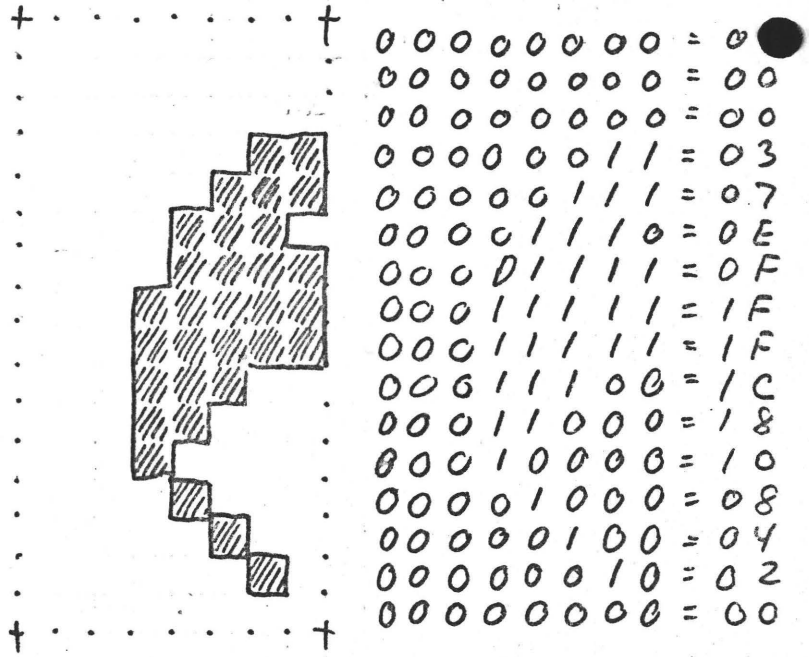
Se vedlagte skitse af et 'A'. Bemærk at højre side af firkanten er udfyldt med blanke, for at undgå at bogstaver flyder sammen. Koden for 'A' er

38 44 82 82 FE 82 82 82
82 00 00 00 00 00 00 00

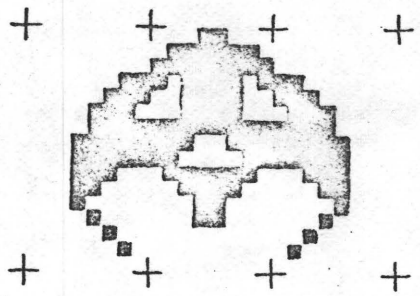
Bemærk også at figuren ser bredere ud end normalt, det skyldes den manglende symmetri på TV-skærmen.

Her følger endnu nogle eks.:





1/3 AF EN "INVADER"



INVADER.

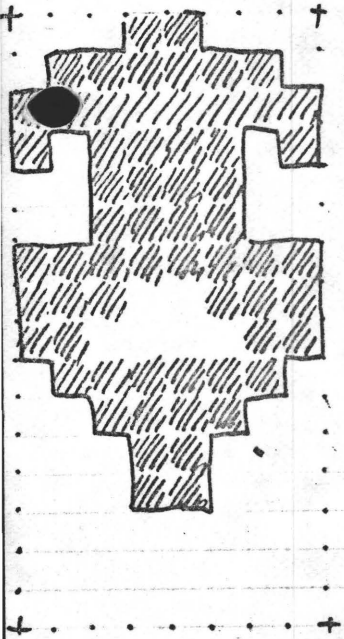
Når man nu har opbygget sine karakterer, skal de jo overføres til 2716. Først konstanterer man at ROM'en er uafhængig af memoryaddresserne! Den første karakter er altid 000H ligemeget, hvor den er placeret i systemet.

Den første karakter fylder 16 byte (10H). Den anden fylder op til 20H osv., den sidste karakter starter ved 7F0H.

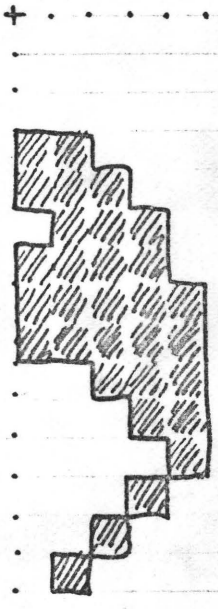
Et lille TRIK: I en karaktergenerator starter karakter 00 i 00H, 01 i 010H, 02 i 020H osv. Det er jo et pænt mønster! I grafikkaraktere starter med 80, så du skal fortsætte med at tælle fra 80 i stedet fra 0.

Karakteren må ind i RAM-lageret, så start ved X800H addresse, så behøver du ikke at lægge sammen hele tiden, for at huske hvor langt du var kommet med indtastningen.

Du behøver ikke at programmere alle 128 karakterer ind på en gang. Man skal blot efterlade alle ubenyttede bytes med FFH.



0 0 0 1 1 0 0 0 = 1 8
0 1 1 1 1 1 1 0 = 7 E
1 1 1 1 1 1 1 1 = F F
1 0 1 1 1 1 0 1 = B D
0 0 1 1 1 1 0 0 = 3 C
0 0 1 1 1 1 0 0 = 3 C
1 1 1 1 1 1 1 1 = F F
1 1 1 0 0 1 1 1 = E 7
1 1 0 0 0 0 1 1 = C 3
0 1 1 1 1 1 1 0 = 7 E
0 0 1 1 1 1 0 0 = 3 C
0 0 0 1 1 0 0 0 = 1 8
0 0 0 1 1 0 0 0 = 1 8
0 0 0 0 0 0 0 0 = 0 0
0 0 0 0 0 0 0 0 = 0 0
0 0 0 0 0 0 0 0 = 0 0



0 0 0 0 0 0 0 0 = 0 0
0 0 0 0 0 0 0 0 = 0 0
0 0 0 0 0 0 0 0 = 0 0
1 1 0 0 0 0 0 0 = C 0
1 1 1 0 0 0 0 0 = E 0
0 1 1 1 0 0 0 0 = 7 0
1 1 1 1 0 0 0 0 = F 0
1 1 1 1 1 0 0 0 = F 8
1 1 1 1 1 0 0 0 = F 8
0 0 1 1 1 0 0 0 = 3 8
0 0 0 1 1 0 0 0 = 1 8
0 0 0 0 1 0 0 0 = 0 8
0 0 0 1 0 0 0 0 = 1 0
0 0 1 0 0 0 0 0 = 2 0
0 1 0 0 0 0 0 0 = 4 0
0 0 0 0 0 0 0 0 = 0 0

FLETSORTERING

Unge Jørgensens problem

Isenkramfirmaet Jørgensen & Søn har et problem. På grund af nogle faglige møder omkring 1. november er en del fakturaer fra sidste uge af oktober ikke blevet lagt på plads i ringbindet med fakturaer for oktober måned. Viggo Jørgensen - også kaldet Unge Jørgensen - får af faderen, Sivert Jørgensen - Jørgensen Senior - tildelt den interessante opgave at få den sidste uges fakturaer sat ind i ringbindet på rette plads blandt de øvrige. Fakturaerne i det pågældende ringbind er ordnet efter kundenumre og indeholder ialt 60 fakturaer fra de tre første uger. Dyngen af fakturaer, som endnu ikke er sat på plads, indbefatter 22 fakturaer. Unge Jørgensen har flere muligheder for at løse opgaven. Han kan tage alle fakturaer ud af ringbindet og derpå stikke de nye fakturaer ind imellem dem én for én. Det gider han ikke!

Han kan også lade de allerede sorterede fakturaer blive siddende i ringbindet og iøvrigt benytte den lige nævnte fremgangsmåde. I så fald har han hele tiden god styr på de fakturaer, som er sat på plads, men må til gengæld åbne og lukke ringbindet for hver faktura, der skal sættes ind. Det gider han slet ikke! I stedet venter han, til Jørgensen Senior er gået til bestyrelsesmøde i Zero-klubben, og sætter sig så til at tænke over tingene (Jørgensen Senior betragter stillesiddende tænksomhed med mistro som tegn på hang til ikke profitgivende teoretiseren). Viggo Jørgensen skriver nogle kundenumre ned på et stykke papir. Så tænker han lidt. Derpå skriver han nogle andre kundenumre ned. Og tænker lidt igen. Pludselig giver det et ryk i ham. Han streger de sidst skrevne kundenumre ud, og skriver dem op på en anden måde. Så slår han et par streger på papiret, nikker eftertænksomt og går hen og tager fat på arbejdet med fakturaerne.

Hvad Unge Jørgensen skrev

Mens Viggo Jørgensen sorterer fakturaer, kan vi kigge på det, han har skrevet ned.

```
12 15 17 20 23 23 28 31 35 40
44 14 30 34 19 15 23
14 15 19 23 30 34 44
```

Den øverste række skal forestille en række kundenumre, som er sorteret, og repræsenterer dermed fakturaer, som er sat i ringbind. Den næste talrække forestiller kundenumre i den "tilfældige" orden, som fremkommer ved, at de udskrives, efterhånden som ordrerne indgår. Den nederste række af tal er tydeligvis et resultat af, at den midterste række er blevet sorteret. Viggo Jørgensen har altså arbejdet med en model af det problem han skal løse, og ved at betragte denne model, har han fundet ud af en metode til løsning af det reale problem.

- og hvad han tænkte

Den øverste talrække er sorteret og i denne sorterede række skal tallene i den anden række indsættes på en sådan måde, at resultatet bliver en ny, sorteret række. Samtidig skal det helst undgås, at der skal "rodes for meget rundt" i de allerede sorterede tal. Dette sidste skyldes en vigtig egenskab ved de reale objekter, der skal arbejdes med, nemlig at der er tale om papirer i et ringbind. Viggo får nu den idé - måske ved at se på de allerede sorterede numre - at man også kunne sortere den anden række af numre, hvilket han derpå gør. Ved at kaste en sidetanke til det reale problem, indser han, at det ikke vil være så farligt at sortere de fakturaer, der er blevet skrevet i sidste uge af oktober, idet der ikke er flere, end man kan lægge ud på et bord ved siden af hinanden. Ved at se på de to rækker af sorterede kundenumre, får han den afgørende idé, og det tilkendegiver han ved at tegne de to pile og derpå gå hen for at sætte fakturaerne ind i ringbindet. I et "genialt glimt" har han set følgende proces for sig:

Først ser man på de to forreste numre (12 og 14). Man vælger det mindste af disse og sætter det på plads. Nu står numrene 15 og 14 forrest. Det mindste af disse er 14, som derfor står for tur. Sådan går det videre, og en del af processen er afbildet her:

```
                                  12 15 17 20
12 14 15 15 17 19 20 23 23 <
                                  14 15 19 23
```

Den resulterende række er afbildet således, at man let kan se, om et element er kommet fra den øverste eller nederste række.

- og hvad han så gjorde

Efter således at have gennemtænkt problemet, er det let for Viggo Jørgensen at lægge sin arbejdsplan. Først lægger han de 22 løse fakturaer ud på kundebordet - som er let at kende på kaffe- og portvinspletterne - og sorterer dem ved hjælp af en metode, han kender godt fra selskabslivet i Nr. Nusserød, hvor familien har domicil, nemlig den man bruger, når man sorterer en hånd med kort (se kortspilsmetoden, Datalærebladet, Februar 1978, side 12). Derpå tager han ringbindet med de allerede sorterede fakturaer, vender hele indholdet over til højre side og åbner ringene. Og nu bruger han den metode, som han netop har opfundet. Han ser på kundenummeret på de to fakturaer, der ligger øverst i hhv. bunken i ringbindet og bunken på bordet. Hvis den, der ligger øverst i ringbindets højre side, har det mindste kundenummer, vender han den blot over på venstre halvdel af ringene, og hvis det er den, der ligger øverst i bunken på bordet, som har det laveste kundenummer, stikker han dén ind over ringene til venstre. Sådan bliver han ved, indtil én af bunkerne er tom. Resten af den anden bunke kan da blot sættes ind over ringene på én gang. Metoden viser sig at være så god, at Viggo forklarer Senior om den, og efter at have rystet tilbørligt på hovedet beslutter denne, at man for fremtiden vil bruge den systematisk, så man kun sætter fakturaer på plads én gang om ugen.

Fletning

Den proces, som Viggo Jørgensen udfører med fakturaerne, kaldes også fletning. Man kan notere sig to ting i forbindelse med fletning:

1) Der er tale om to rækker, som hver for sig er sorteret.
2) Der ændres ikke noget på den indbyrdes rækkefølge af elementerne i de to rækker hver for sig.

Alle de sorteringsmetoder, vi indtil nu har undersøgt, har virket på tabeller, dvs. datamængder hvis enkelte elementer vi kan læse eller skrive i vilkårlig rækkefølge, og hvor vi får adgang til en komponent ved at angive dens indeks eller postnummer. Vi har da også i alle tilfælde benyttet, at man fx. kan ombytte elementer i sådanne datastrukturer. Hvis de data, man skal sortere, står skrevet i strukturer, hvis elementer man kun kan få adgang til forfra og i rækkefølge, altså fx. data-køer og sekventielle filer, kan sorteringer af de tidligere nævnte typer ikke anvendes. I sådanne tilfælde må man sætte sin lid til metoder, der bygger på fx. fletning. I det følgende skal en sådan metode gennemgås med henblik på sortering af sekventielle filer.

Sortering ved fletning

Lad os tænke os, at vi ønsker at sortere en sekventiel fil, som har dette indhold:

18 32|6 60|14 42 44 68|12 24 30 48|4 8 72|
3 20 58|38 62

Man kan uden videre se, at en sådan fil kan deles op i en række delfiler, som hver for sig er ordnede. På billedet ovenfor er denne opdeling antydet med lodrette streger. De enkelte ordnede delfiler kalder man for løb, og i vort eksempel har vi altså ialt syv sådanne løb. Vi opdeler nu filen i to nye filer, idet vi skriver hvert andet løb over i den ene fil og resten over i den anden:

18 32|14 42 44 68|4 8 72|38 62
6 60|12 24 30 48|3 20 58

Derpå sammenfletter vi de enkelte løb i den ene fil med de tilsvarende løb i den anden til en ny sammenhængende fil - det overskydende løb i den øverste hjælpefil kopierer vi blot med over:

6 18 32 60|12 14 24 30 42 44 48 68|3 4 8 20 58 72|
38 62

Denne fil opdeles atter i parallele løb:

6 18 32 60|3 4 8 20 58 72
12 14 24 30 42 44 48 68|38 62

Den næste fletning giver:

6 12 14 18 24 30 32 42 44 48 60 61|
3 4 8 20 38 58 62 72

Efter endnu en opdeling og en fletning fremkommer den fuldstændigt sorterede fil.

Som eksemplet viser har vi brug for ialt tre filer til processen: en hovedfil og to hjælpefiler. Endvidere kan vi se, at processen i det væsentlige består af en række opdelinger med påfølgende fletninger. Inden vi kan begynde at skrive algoritmerne for processen, må vi også gøre os klart, hvorledes vi kan formulere afslutningen af et løb ved hjælp af en relation, som et datamatisk system kan afprøve. Ser vi atter på eksemplet ovenfor, konstaterer vi, at *det sidste element i et løb er karakteriseret ved, at det efterfølges af et element, der er mindre, eller af filslut.* Det er klart, at processen er *afsluttet, når antallet af løb er lig med 1.* Idet vi kalder hovedfilen for C og hjælpefilerne for hhv. A og B, kan vi nedskrive hovedproceduren for processen samt procedurerne OPDEL og FLETNING for de to vigtigste delprocesser.

```
PROCEDURE FLETTEPROGRAM
  GENTAG
    gør filen A og filen B klar til skrivning
    gør filen C klar til læsning
    OPDEL
    luk alle filer
    gør filen A og filen B klar til læsning
    gør filen C klar til skrivning
    nulstil løbtælleren
    FLETNING
    luk alle filer
  INDTIL løbtælleren er lig med 1
SLUT FLETTEPROGRAM

PROCEDURE OPDEL
  GENTAG
    KOPIERLØB(fra C til A)
    HVIS ikke filen C er slut SÅ
      KOPIERLØB(fra C til B)
    SLUT-HVIS
  INDTIL filen C er slut
SLUT OPDEL

PROCEDURE FLETNING
  GENTAG
    FLETLØB //flet et par af løb//
    tæl løbtælleren én op
  INDTIL filen A eller filen B er slut
  MENS filen A ikke er slut UDFØR
    //et overskydende løb tilbage i A//
    KOPIERLØB(fra A til C)
    tæl løbtælleren én op
  SLUT-MENS
  MENS filen B ikke er slut UDFØR
    //et overskydende løb tilbage i B//
    KOPIERLØB(fra B til C)
    tæl løbtælleren én op
  SLUT-MENS
SLUT FLETNING
```

* * *

Procedurerne

Vi skal nu til at se nærmere på de procedurer, der hedder FLETLØB og KOPIERLØB, og vi kommer ikke længere uden om at overveje et par

tekniske detaljer. Vi er ved at sortere filer, og filer består af poster. Indholdet af en enkelt post kan efter omstændighederne være ret så broget. Posten kan bestå af mange felter med mange forskellige informationer. Et af disse felter må imidlertid indeholde den nøgle, man sorterer efter. Det kan fx. være et felt med et kundenummer, et felt med et lønningsnummer, eller et felt med et efternavn. For ikke at gøre denne gennemgang af fletsortering altfor afhængig af en særlig anvendelse, vil jeg nøjes med at tale om en nøgle (key), som indeholder den information, der danner grundlag for sorteringen. Læseren må så selv modificere programmet efter sit eget behov og herunder selv sørge for, at nøgleinformationen for hver af de tre filer bliver læst og skrevet på rette sted i programmet. I det følgende vil jeg anvende betegnelsen nøgle (X) for værdien af det sidst læste felt med nøgleoplysningen fra filen X. Hvis man altså fx. benytter en persons efternavn som nøgle for sorteringen, må man sørge for, at nøgle (X) indeholder efternavnet fra den sidst læste post i filen X, hvis sorteringen skal virke korrekt. Endvidere vil jeg arbejde med en buffer, som kan indeholde den næstsidste værdi af nøgle (X), og som kan bruges fx. i det tilfælde, hvor man skal undersøge en relation mellem to på hinanden følgende nøgleoplysninger. I såvel FLETLØB som KOPIERLØB kaldes proceduren KOPIER, som på en måde udfører "det grove" for de to procedurer, fra hvilke den bliver kaldt. Denne procedure kopierer en post fra én fil, som vi vil kalde X, til en anden, som vi vil kalde Y, og dernæst undersøger den, om et løb er afsluttet eller ej. Proceduren kan beskrives således:

```
PROCEDURE KOPIER(fra X til Y)
  buffer:=nøgle(X)
  SKRIV post(X) i filen Y
  LÆS post(X) i filen X
  udtag nøgle(X)
  HVIS filen X er slut SÅ
    slutløb:=sand
  ELLERS
    slutløb:=(buffer>nøgle(X))
  SLUT-HVIS
SLUT KOPIER
```

Ved procedurens start har vi den senest læste post fra filen X - post (X) - stående i arbejdslageret, og nøgle (X) indeholder værdien af det felt, vi sorterer efter. Nøgleoplysningen tildeles den variable buffer til senere brug. Derpå finder den egentlige kopiering sted, idet hele posten fra X skrives i filen Y, hvorpå en ny post læses i filen X. Denne læsning giver naturligvis også nøgle (X) en ny værdi, nemlig den fra den nye post. Der er nu flere muligheder: Det kan vise sig, at filen X er slut, og så er et løb naturligvis også slut, og det noterer vi i den Boolske variabel slutløb ved at tildele den værdien sand. Hvis filen ikke er slut, er der atter to muligheder: enten er et løb slut, eller også er det ikke slut. Det afgøres af sandhedsværdien af det Boolske udtryk: buffer > nøgle (X). Vi husker, at den variable buffer indeholder nøglen til den post, som netop er blevet skrevet over i filen Y, og at nøgle (X) indeholder nøglen til den post, som netop er blevet læst i filen X. Dersom det viser sig, at denne sidst læste nøgle er mindre end den foregående, betyder det netop, at et løb er afsluttet. Hvis det derimod ikke er tilfældet, fortsætter løbet. Den rigtige sandhedsværdi bliver tildelt den variable slutløb. Når vi forlader KOPIER, "ved" den variable slutløb altså, om et løb er slut eller ej, og denne oplysning kan vi bruge i de øvrige procedurer.

Efter at have skrevet KOPIER er det en let sag at skrive procedurerne KOPIERLØB og FLETLØB:

```
PROCEDURE KOPIERLØB(fra X til Y)
  GENTAG
    KOPIER(fra X til Y)
  INDTIL slutløb
SLUT KOPIERLØB

PROCEDURE FLETLØB
  GENTAG
    HVIS nøgle(A)<nøgle(B) SÅ
      KOPIER(fra A til C)
      HVIS slutløb SÅ
        KOPIERLØB(fra B til C)
      SLUT-HVIS
    ELLERS
      KOPIER(fra B til C)
      HVIS slutløb SÅ
        KOPIERLØB(fra A til C)
      SLUT-HVIS
    SLUT-HVIS
  INDTIL slutløb
SLUT FLETLØB
```

Bemærk, hvorledes den fra KOPIER styrede variabel slutløb anvendes i de to procedurer.

Program FLETSORTERING

Som bilag til denne artikel findes et COMAL program, som udfører sortering ved fletning. Programmet følger nøje de angivne algoritmer, men på grund af visse mangler i COMAL, har det været nødvendigt at føje nogle hjælpesætninger til de oprindelige algoritmer for at få processerne udført af et COMAL-system. De tre filvariable A, B og C er realiseret som numeriske variable med værdierne hhv. 1, 2 og 3. De fleste af de nuværende COMAL versioner tillader desværre ikke parameter-overføring mellem procedurer, og i programmet er denne overføring af værdier ved parametre erstattet af simple tildelinger, som foretages inden kaldet af proceduren. Vi kan fx. se på PROC OPDEL, hvor man finder tildelingerne: LET X=C (linje 300) og LET Y=A (linje 320) inden kaldet af proceduren KOPIERLØB. Når vi via KOPIERLØB når frem til KOPIER, har X altså værdien 3 og Y værdien 1, og det betyder, at der kopieres fra filen C ind i filen A, hvilket netop er det, vi ønsker. Ved at give X og Y andre værdier, udføres kopieringen med andre afsender- og modtagerfiler. For at holde styr på posterne fra de tre filer, er vi også nødt til at bruge nogle strukturerede buffere, og til dette formål har jeg valgt at bruge fire tabeller: NAVN$, ADR$, POSTDSTR$ og NØGLE$. Hver af dem har tre komponenter: 1. komponenterne bruges som buffere for poster fra A, 2. komponenterne som buffere for poster fra B, og 3. komponenterne som buffere for poster fra C. Den valgte poststruktur skal blot tjene et demonstrationsformål, og læseren må selv indrette buffere til sit specielle formål. Man skal blot huske at få NØGLE$ tildelt værdi af det felt, hvis indhold danner grundlag for sorteringen. Der er desuden indsat en "service-procedure" LISTFIL, som kan

bruges ved afprøvningen af programmet. Programmet startes fra afsnittet i linje 1010 - 1120, hvor der også kan indtastes test-værdier til filen C. Også dette afsnit må naturligvis indrettes efter det aktuelle behov.

Afslutning

Hermed slutter artikelserien: ORDEN SKAL DER TIL. Min væsentligste kilde til de i serien gennemgåede algoritmer har været:

N. Wirth: Algorithms + Data Structures = Programs.
(Prentice-Hall 1976)

Det er som bekendt Wirth, der har defineret programmerings-sproget Pascal, og det var med dette sprog som forbillede, COMAL i sin tid blev defineret.
Det kan derfor heller ikke undre, at COMAL programmerne i artiklerne har en påfaldende lighed med Wirths Pascal-programmer. Af andre bøger, som kunne have interesse for mine læsere, kan jeg nævne:

E. Horowitz & S. Sahni: Fundamentals of Data Structures (Pitman 1977)
Peter Naur: Concise Survey of Computer Methods (Studentlitteratur 1974)
Kenneth L Bowles: Problem Solving Using Pascal (Springer Verlag 1977)

Tønder *Børge R. Christensen*

```
0010 PROC LISTFIL
0020   OPEN FILE (C,3),"CFIL"
0030   READ FILE (C),NAVN$(C),ADR$(C),POSTDSTR$(C)
0035   LET NØGLE$(C)=NAVN$(C)
0040   WHILE NOT EOF(C) DO
0050     PRINT NØGLE$(C),
0060     READ FILE (C),NAVN$(C),ADR$(C),POSTDSTR$(C)
0065     LET NØGLE$(C)=NAVN$(C)
0070   ENDWHILE
0080   PRINT
0090   CLOSE FILE (C),
0100 ENDPROC LISTFIL
0110 REM //---------------//
0120 PROC KOPIER
0130   LET BUFFER$=NØGLE$(X)
0140   WRITE FILE (Y),NAVN$(X),ADR$(X),POSTDSTR$(X)
0150   READ FILE (X),NAVN$(X),ADR$(X),POSTDSTR$(X)
0155   LET NØGLE$(X)=NAVN$(X)
0160   IF EOF(X) THEN
0170     LET SLUTLØB=TRUE
0180   ELSE
0190     LET SLUTLØB=(BUFFER$>NØGLE$(X))
0200   ENDIF
0210 ENDPROC KOPIER
0220 REM //---------------//
0230 PROC KOPIERLØB
0240   REPEAT
0250     EXEC KOPIER
0260   UNTIL SLUTLØB
0270 ENDPROC KOPIERLØB
0280 REM //---------------//
0290 PROC OPDEL
0300   LET X=C
0310   REPEAT
0320     LET Y=A
0330     EXEC KOPIERLØB
0340     IF NOT EOF(C) THEN
0350       LET Y=B
0360       EXEC KOPIERLØB
0370     ENDIF
0380   UNTIL EOF(C)
0390 ENDPROC OPDEL
0400 REM //---------------//
0410 PROC FLETLØB
0420   LET Y=C
0430   REPEAT
0440     IF NØGLE$(A)<NØGLE$(B) THEN
0450       LET X=A
0460       EXEC KOPIER
0470       IF SLUTLØB THEN
0480         LET X=B
0490         EXEC KOPIERLØB
0500       ENDIF
0510     ELSE
0520       LET X=B
0530       EXEC KOPIER
0540       IF SLUTLØB THEN
0550         LET X=A
0560         EXEC KOPIERLØB
0570       ENDIF
0580     ENDIF
0590   UNTIL SLUTLØB
0600 ENDPROC FLETLØB
0610 REM //---------------//
0620 PROC FLETNING
0630   REPEAT
0640     EXEC FLETLØB
0650     LET L=L+1
0660   UNTIL EOF(A) OR EOF(B)
0670   LET Y=C
0680   WHILE NOT EOF(A) DO
0690     LET X=A
0700     EXEC KOPIERLØB
0710     LET L=L+1
0720   ENDWHILE
0730   WHILE NOT EOF(B) DO
0740     LET X=B
0750     EXEC KOPIERLØB
0760     LET L=L+1
0770   ENDWHILE
0780 ENDPROC FLETNING
0790 REM //---------------//
0800 PROC FLETTEPROGRAM
0830   REPEAT
0840     OPEN FILE (A,1),"AFIL"
0850     OPEN FILE (B,1),"BFIL"
0860     OPEN FILE (C,3),"CFIL"
0870     READ FILE (C),NAVN$(C),ADR$(C),POSTDSTR$(C)
0875     LET NØGLE$(C)=NAVN$(C)
0880     EXEC OPDEL
0890     CLOSE
0900     OPEN FILE (A,3),"AFIL"
0910     OPEN FILE (B,3),"BFIL"
0920     OPEN FILE (C,1),"CFIL"
0930     READ FILE (A),NAVN$(A),ADR$(A),POSTDSTR$(A)
0935     LET NØGLE$(A)=NAVN$(A)
0940     READ FILE (B),NAVN$(B),ADR$(B),POSTDSTR$(B)
0945     LET NØGLE$(B)=NAVN$(B)
0950     LET L=0
0960     EXEC FLETNING
0970     CLOSE
0980   UNTIL L=1
0990 ENDPROC FLETTEPROGRAM
1000 REM //---------------//
1005 RANDOMIZE
1010 LET A=1; B=2; C=3
1020 LET TRUE=1; FALSE=0
1022 DIM NAVN$(3,30),ADR$(3,20),POSTDSTR$(3,20)
1024 DIM NØGLE$(3,30)
1030 OPEN FILE (3,1),"CFIL"
1040 FOR I=1 TO 10
1050   INPUT "NAVN:          ",NAVN$(C)
1052   INPUT "ADRESSE:       ",ADR$(C)
1054   INPUT "POSTDISTRIKT:",POSTDSTR$(C)
1060   WRITE FILE (C),NAVN$(C),ADR$(C),POSTDSTR$(C)
1070 NEXT I
1080 CLOSE
1090 EXEC LISTFIL
1100 EXEC FLETTEPROGRAM
1110 EXEC LISTFIL
1120 END
*
```

L I F E .

Life er et spil opfundet af lektor John Horton Conway. Navnet skyldes lighed med biologiforsøg om bakteriers udvikling.

HVAP ER LIFE?

Life kan beskrives som et matematisk spil foretaget på et skakbrædt, bestående af ene hvide felter. Et begyndelsesmønster fastlægges og computeren ændre mønstret efter bestemte regler.

REGLER.

Forestil dig et uendeligt skakbrædt. Hver kvadrat kaldes en celle. Men af praktiske grunde bliver man nød til at foretage en begrænsning af brædtet, til en endelig størrelse.

Hver celle kan enten død eller overleve, og det er kun afhængig af dets naboceller.

Efter hver generation skifter mønstret efter disse enkle regler:

1: Enhver celle (både de døde og levende) vil fødes, hvis de er omgivet af 3 levende celler.

2: Enhver celle vil forblive uændret, hvis den er omgivet af 2 levende celler.

3: Hvis der er mere end 3, dør man af sult. Er der under 2 dør man af kedsomhed!!

Programmet er skrevet i Z80 kode. Alle hexadecimale tal er efterfulgt af "H", andre tal er decimale. DB 18H tilknytter værdien "18H" til den bestemte byte. DS 1 reserverer en byte til en variabel. DB 'XXXXXX' gemmer strengen i ASCII værdi i RAM lageret. EQU tilknytter en bestemt værdi til en variabel uafhængig af ORG (begyndelsesadresse).

HVORDAN SPILLER MAN LIFE ?

Man starter med E1000 (NL). Derefter tegner man et mønster ved hjælp af "SPACE", "B/S" og "RETURN/NEWLINE". "R" starter forfra på nyt mønster. "Q" afslutter tegningen.

Når man er færdig med at tegne, trykkes "Q" og man bliver spurgt om auto "A" eller manual "M". I auto skifter generationerne med 0.2 sek. mellemrum (2MHz), så det ser ud som film. Ved "M" skal trykkes tast for hver ny generation, der ønskes.

Under begge ("A" "M") kan man skifte status ved at trykke det modsatte. Ved "Q" forlades run mode og der spørges om ny "N" eller fortsættes "C".

Ved en stor populatuion skal man sørge for at placere mønstret omkring midten af skærmen.

God fornøjelse med spillet. Gå nu ikke for sent i seng!!!

```
ASB. LIND
E1002
0001                ;*******************************************
0002                ;*       ET SPIL OM LIV OG DØD.            *
0003                ;*******************************************
0004
0005                ;Skrevet af M. Kuczynski TIL NASCOM 1
0006                ;Modificeret TIL NASCOM 2 af Asbjørn Lind
0007
0008 1000           ORG 1000H
0009
0010
```

```
0011 1000 213030   START:  LD HL,3030H        ;3030=ASCII "00"
0012 1003 22AE11          LD (NUMBER),HL      ;RESET TÆLLER
0013 1006 3E0C            LD A,0CH            ;RYD SKÆRMEN
0014 1008 F7              RST 30H
0015 1009 11D00B          LD DE,0BD0H         ;ØVERSTE LINIE
0016 100C 219711          LD HL,TITLE         ;"GAME OF LIFE"
0017 100F 010C00          LD BC,12            ;ANTAL BOGSTAVER
0018 1012 EDB0            LDIR                ;SKRIV TITEL
0019 1014 21D811   INIT:  LD HL,BORDER        ;BEGYNDELSEN AF BRÆDT
0020 1017 0619            LD B,25             ;ANTAL CELLER I TOPPEN
0021 1019 0E0F            LD C,15             ;ANTAL RÆKKER
0022 101B 36FF     TOP:   LD (HL),0FFH        ;UDFYLD ØVERSTE RÆKKE
0023 101D 23              INC HL              ;NÆSTE CELLE
0024 101E 10FB            DJNZ TOP            ;FORTSÆT TIL SLUT
0025 1020 36FF     LEFT:  LD (HL),0FFH        ;ØDFYLD VENSTRE KANT
0026 1022 23              INC HL              ;NÆSTE CELLE
0027 1023 0618            LD B,24             ;ANTAL CELLER PR. RK.
0028 1025 3600     ROW:   LD(HL),0            ;UDFYLD MED BLANKE
0029 1027 23              INC HL              ;
0030 1028 10FB            DJNZ ROW            ;
0031 102A 0D              DEC C               ;SIDSTE RK.?
0032 102B 20F3            JR NZ,LEFT          ;NEJ NÆSTE RK.
0033 102D 061A            LD B,26             ;CELLER I NEDERSTE KANT
0034 102F 36FF     BOTTOM:LD (HL),0FFH        ;UDFYLD
0035 1031 23              INC HL              ;
0036 1032 10FB            DJNZ BOTTOM         ;
0037 1034 11E40B   LOADER:LD DE,0BE4H         ;MIDTEN AF ØVERSTE LINIE
0038 1037 21B011          LD HL,NEWPTN        ;"NEW PARRERN"
0039 103A 010D00          LD BC,13            ;
0040 103D EDB0            LDIR                ;
0041 103F DF7B     SCAN:  SCAL 7BH            ;AFVENT KARAKTER
0042 1041 FE0D            CP 0DH              ;NEW LINE?
0043 1043 2004            JR NZ,BS            ;NEJ TIL B/S
0044 1045 DF6A            SCAL 6AH            ;CRLF
0045 1047 18F6            JR SCAN             ;NÆSTE KARAKTER
0046 1049 FE08     BS:    CP 8                ;BACKSPACE ?
0047 104B 2004            JR NZ,SPCE          ;NEJ TIL SPACE
0048 104D F7              RST 30H             ;CURSOR TIL VENSTRE
0049 104E F7              RST 30H             ;   -    -    -
0050 104F 18EE            JR SCAN             ;
0051 1051 FE20     SPCE:  CP 20H              ;SPACE ?
0052 1053 2005            JR NZ,QUIT          ;NEJ TIL QUIT
0053 1055 F7              RST 30H             ;CURSOR TIL HØJRE
0054 1056 DF69     SPCE1: SCAL 69H            ;   -    -    -
0055 1058 18E5            JR SCAN             ;
0056 105A FE51     QUIT:  CP 51H              ;QUIT ?
0057 105C 280F            JR Z,RUN            ;GÅ TIL RUN MODE
0058 105E FE52            CP 52H              ;RESTART ?
0059 1060 289E            JR Z, START         ;RETUR TIL START
0060 1062 2A290C          LD HL,(CURSOR)      ;FYLD CURSOR POSITION
0061 1065 3680            LD (HL),80H         ;MED EN FIRKANT
0062 1067 23              INC HL              ;
0063 1068 22290C          LD (CURSOR),HL      ;GEM POSITION
0064 106B 18E9            JR SPCE1            ;NÆSTE KARAKTER
0065 106D 11E40B   RUN:   LD DE,0BE4H         ;MIDTEN AF ØVERSTE LINIE
0066 1070 21BE11          LD HL,CHOOSE        ;MANUAL/AUTO ?
0067 1073 010D00          LD BC,13            ;
0068 1076 EDB0            LDIR                ;
0069 1078 CF       AGAIN: RST 8               ;AFVENT KARAKTER
0070 1079 FE4D            CP 4DH              ;MANUAL ?
0071 107B 2004            JR NZ,AUT           ;HVIS EJ SÅ AUTO
0072 107D 3E00     MANREF:LD A,MANUAL-JUMP-1  ;UDREGN DISPLACEMENT
0073 107F 1806            JR LDJUMP           ;
0074 1081 FE41     AUT:   CP 41H              ;AUTO ?
```

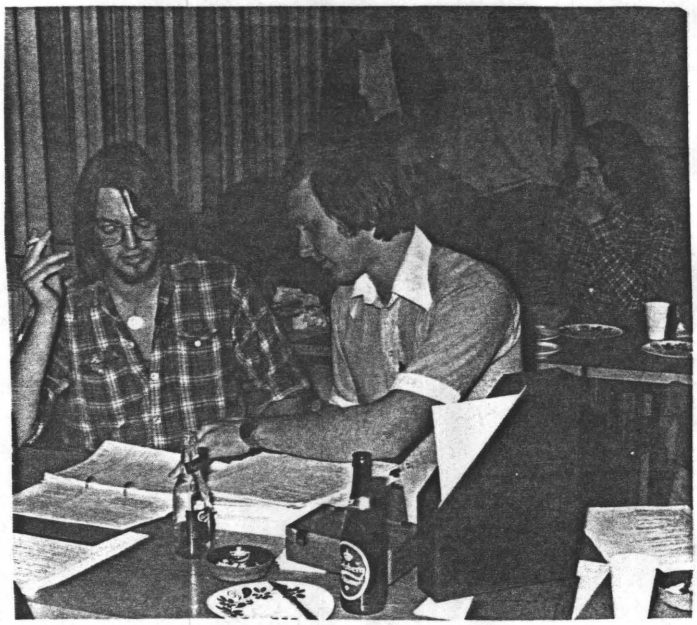
```
0075 1083 20F3            JR NZ,AGAIN          ;HVIS EJ PRØV IGEN
0076 1085 3E08   AUTREF:  LD A,AUTO-JUMP-1     ;
0077 1087 326211 LDJUMP:  LD (JUMP),A          ;SKRIV DISP.
0078 108A 11E40B GEN1:    LD DE,0BE4H          ;
0079 108D 21A311          LD HL,RUNSTR         ;GENERATION XX
0080 1090 010D00          LD BC,13             ;
0081 1093 EDB0            LDIR                 ;
0082 1095 2A290C CPYVDU:  LD HL,(CURSOR)       ;ADR. CURSOR
0083 1098 36FF            LD (HL),0FFH         ;NEUTRALISER CURSOR
0084 109A 210A08          LD HL,80AH           ;1. LINIE PÅ VDU
0085 109D 11F211          LD DE,BOARD          ;START AF BRÆDT
0086 10A0 7E      BACK:   LD A,(HL)            ;TEST CELLEN
0087 10A1 3C              INC A                ;FÆRDIG MED SKÆRMEN ?
0088 10A2 CA6111          JP Z,REFL            ;GÅ TIL RUN MODE
0089 10A5 3D              DEC A                ;TEST CELLEN
0090 10A6 2009            JR NZ,WRITE          ;FORTSÆT HVIS EJ SLUT
0091 10A8 D5              PUSH DE              ;GEM REGISTRET
0092 10A9 111000          LD DE,16             ;LINIE OFFSET
0093 10AC 19              ADD HL,DE            ;ADD OFFSET TIL LINIE
0094 10AD D1              POP DE               ;GENDAN REGISTER DE
0095 10AE 13              INC DE               ;NÆSTE CELL
0096 10AF 18EF            JR BACK              ;FORTSÆT TIL LINIESLUT
0097 10B1 CB6E    WRITE:  BIT 5,(HL)           ;LEVENDE CELLE ?
0098 10B3 2008            JR NZ,BLANK          ;TEST IGEN HVIS DØD
0099 10B5 3E01            LD A,1               ;"1" = LEVENDE CELLE
0100 10B7 12      CELL1:  LD (DE),A            ;SÆT CELLEN PÅBRÆDT
0101 10B8 23              INC HL               ;
0102 10B9 23              INC HL               ;NÆSTE SKÆRM POS.
0103 10BA 13              INC DE               ;  -   BRÆDT  -
0104 10BB 18E3            JR BACK              ;NÆSTE CELLE
0105 10BD AF      BLANK:  XOR A                ;"0" = DØD CELLE
0106 10BE 18F7            JR CELL1             ;CELLE TIL BRÆDT
0107 10C0 DD21F211 INIT2: LD IX,BOARD          ;START PÅ BRÆDT
0108 10C4 217601          LD HL,374            ;ANTAL CELLER
0109 10C7 0600    LOOP:   LD B,0               ;RYD NABOCELLE
0110 10C9 DDCB007E        BIT 7,(IX+0)         ;BRÆDT CELLE ?
0111 10CD 2045            JR NZ,NEXT           ;IGNORE DEN
0112 10CF DD7EE6          LD A,(IX-26)         ;TEST CELLEN TIL NV
0113 10D2 CD8F11          CALL CHECK           ;CHECK OM DEN ER LEVENDE
0114 10D5 DD7EE7          LD A,(IX-25)         ;TIL NORD
0115 10D8 CD8F11          CALL CHECK           ;
0116 10DB DD7EE8          LD A,(IX-24)         ;TIL NØ
0117 10DE CD8F11          CALL CHECK           ;
0118 10E1 DD7EFF          LD A,(IX-1)          ;MOD VEST
0119 10E4 CD8F11          CALL CHECK           ;
0120 10E7 DD7E01          LD A,(IX+1)          ;MOD Ø
0121 10EA CD8F11          CALL CHECK           ;
0122 10ED DD7E18          LD A,(IX+24)         ;MOD SV
0123 10F0 CD8F11          CALL CHECK           ;
0124 10F3 DD7E19          LD A,(IX+25)         ;MOD S
0125 10F6 CD8F11          CALL CHECK           ;
0126 10F9 DD7E1A          LD A,(IX+26)         ;SØ
0127 10FC CD8F11          CALL CHECK           ;
0128 10FF 78      NEIBRS: LD A,B               ;FLYT NABOTÆLLER
0129 1100 FE03            CP 3                 ;TRE NABOER ?
0130 1102 2006            JR NZ,SAME           ;HVIS EJ, TEST FOR 2
0131 1104 DDCB00CE ALIVE: SET 1,(IX+0)         ;CELLEN VIL VÆRE I LIVE
0132 1108 180A            JR NEXT              ;NÆSTE
0133 110A FE02    SAME:   CP 2                 ;TO NABOER
0134 110C 2006            JR NZ,NEXT           ;HVIS EJ, TIL NÆSTE CELLE
0135 110E DDCB0046        BIT 0,(IX+0)         ;LEVER DEN ?
0136 1112 20F0            JR NZ,ALIVE          ;DET BLIVER DEN VED MED
0137 1114 DD23    NEXT:   INC IX               ;NÆSTE CELLE
0138 1116 2B              DEC HL               ;NEDSKRIV CELLETÆLLER
0139 1117 7C              LD A,H               ;ER ALLE UNDERSØGT ?
```

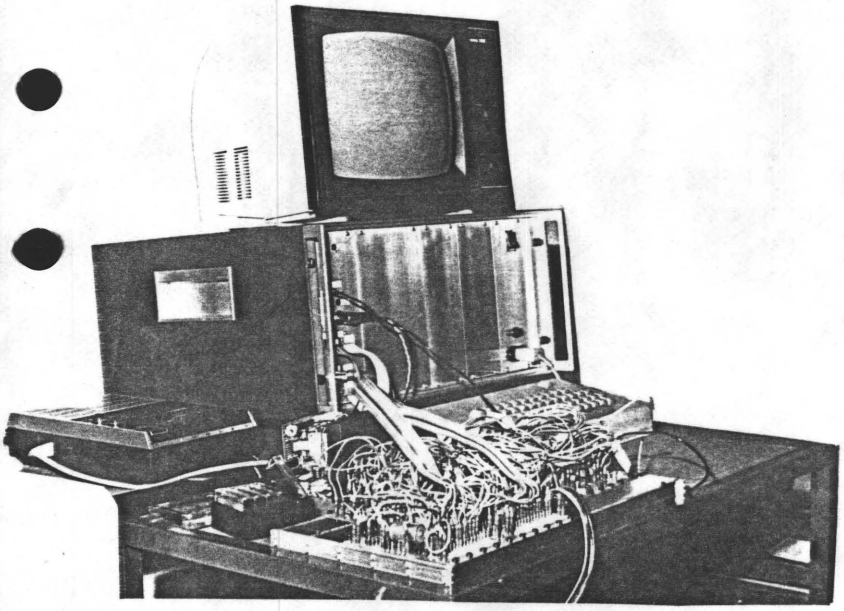
```
0140 1118 B5              OR L                ;
0141 1119 20AC            JR NZ,LOOP          ;HVIS EJ,SÅ NÆSTE
0142 111B 016801   GEN2:  LD BC,360           ;ANTAL CELLER
0143 119E 110A08          LD DE,80AH          ;1. LINIE
0144 1121 21F211          LD HL,BOARD         ;START AF BRÆDT
0145 1124 CB7E     TEST:  BIT 7,(HL)          ;KANT ?
0146 1126 2808            JR Z,REGEN          ;HVIS EJ REGENERER
0147 1128 23       NEWLIN: INC HL             ;NÆSTE CELLE
0148 1129 E5              PUSH HL             ;GEM REGISTRET
0149 112A 211000          LD HL,16            ;LINIE OFFSET
0150 112D 19              ADD HL,DE           ;ADD OFFSET TIL LINIE
0151 112E EB              EX DE,HL            ;BYT REGISTRE
0152 112F E1              POP HL              ;GENDAN HL
0153 1130 CB3E     REGEN: SRL(HL)             ;REGENERER
0154 1132 CB46            BIT 0,(HL)          ;TEST CELLE
0155 1134 2004            JR NZ,LIVE          ;ER DEN I LIVE ?
0156 1136 3E20     DEAD:  LD A,20H            ;DØD CELLE = BLANK
0157 1138 1802            JR LDSCRN           ;SPACE SKÆRM
0158 113A 3E80     LIVE:  LD A,80H            ;LEVENDE CELLE=FIRKANT
0159 113C 12       LDSCRN: LD (DE),A          ;FYLD SPACE PÅ SKÆRM
0160 113D 13              INC DE              ;
0161 113E 13              INC DE              ;
0162 113F 23       NXTCEL: INC HL             ;NÆSTE CELLE
0163 1140 0B              DEC BC              ;NEDSKRIV CELLETÆLLER
0164 1141 78              LD A,B              ;ER ALLE CELLER FLYTTET ?
0165 1142 B1              OR C                ;
0166 1143 20DF            JR NZ,TEST          ;HVIS EJ, SÅGØR DET
0167 1145 E5       GEN:   PUSH HL             ;GEM REG.
0168 1146 21F00B          LD HL,0BF0H         ;"ENER" TÆLLER
0169 1149 34              INC (HL)            ;OPSKRIV DEN
0170 114A 7E              LD A,(HL)           ;TEST FOR
0171 114B FE3A            CP 3AH              ;STØRRE END 9 ?
0172 114D 200B            JR NZ,COUNTR        ;HVIS EJ, GEM I RAM
0173 114F 3630            LD (HL),30H         ;NULSTIL ENERE
0174 1151 2B              DEC HL              ;"TIER" TÆLLER
0175 1152 34              INC (HL)            ;OPSKRIV
0176 1153 7E              LD A,(HL)           ;ER DE STØRRE
0177 1154 FE3A            CP 3AH              ;END 9 ?
0178 1156 2002            JR NZ,COUNTR        ;HVIS EJ, GEM I RAM
0179 1158 3630            LD (HL),30H         ;NULSTIL TIERNE
0180 115A 2AEF0B   COUNTR: LD HL,(0BEFH)      ;FLYT NR TIL SKÆRM
0181 115D 22AE11          LD (NUMBER),HL      ;GEM DET I RAM
0182 1160 E1              POP HL              ;RESTORE HL
0183 1161 18       REFL:  DB 18H              ;18 = KODE FOR "JR"
0184 0001          JUMP:  DS 1                ;HER PLACERES DISP.
0185 1163 CF       MANUAL: RST 8              ;AFVENT KARAKTER
0186 1164 FE41            CP 41H              ;AUTO ?
0187 1166 CA8510          JP Z,AUTREF         ;JA, GÅ TIL AUTO MODE
0188 1169 1807            JR ENDRUN           ;TEST FOR "Q"
0189 116B DF62     AUTO:  SCAL 62H            ;"FØLER" PÅ TASTATURET!
0190 116D FE4D            CP 4DH              ;MANUAL ?
0191 116F CA7D10          JP Z,MANREF         ;JA TIL MANUAL MODE
0192 1172 FE51     ENDRUN: CP 51H             ;QUIT ?
0193 1174 C2C010          JP NZ,INIT2         ;FORTSÆT HVIS IKKE Q,M,A
0194 1177 11E40B   CHOICE: LD DE,0BE4H        ;
0195 117A 21CB11          LD HL,NEWCON        ;NEW/CONTINUE ?
0196 117D 010D00          LD BC,13            ;
0197 1180 EDB0            LDIR                ;
0198 1182 CF       CALLKB: RST 8              ;AFVENT KARAKTER
0199 1183 FE43            CP 43H              ;CONTINUE ?
0200 1185 CA8A10          JP Z,GEN1           ;JA FORTSÆT
0201 1188 FE4E            CP 4EH              ;NEW ?
0202 118A CA0010          JP Z,START          ;JA TIL RESTART
0203 118D 18F3            JR CALLKB           ;HVIS INGEN PRØV IGEN
0204 118F CB7F     CHECK: BIT 7,A             ;KANT  CELLE ?
```

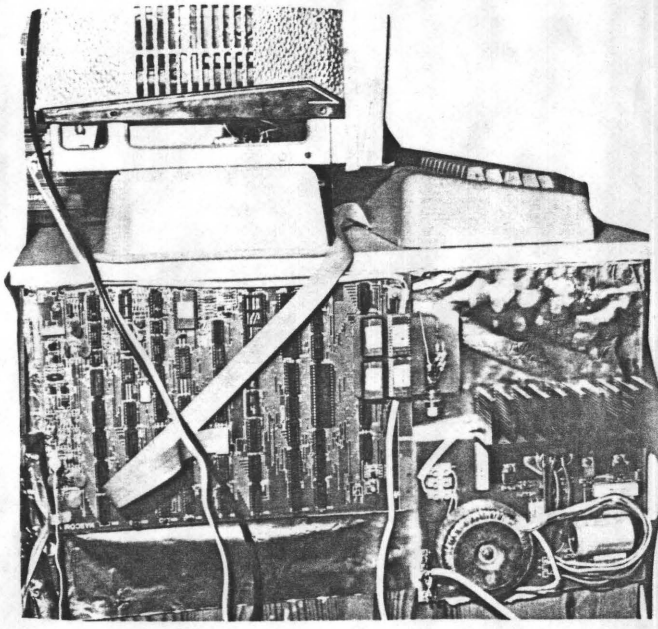
```
0205 1191 C0                    RET NZ              ;TIL NÆSTE CELLE
0206 1192 CB47                  BIT 0,A             ;DØD CELLE
0207 1194 C8                    RET Z               ;TIL NÆSTE CELLE
0208 1195 04                    INC B               ;ADD EN ANDEN NABO
0209 1196 C9                    RET                 ;OG RETUR
0210 1197 47414D45 TITLE:   DB'GAME OF LIFE'
0211 119B 204F4620
0212 119F 4C494645
0213 11A3 47656E65 RUNSTR: DB'Generation '
0214 11A7 72617469
0215 11AB 6F6E20
0216 0002          NUMBER: DS 2
0217 11B0 4E657720 NEWPTN: DB'New pattern '
0218 11B4 70617474
0219 11B8 65726E20
0220 11BC 2020
0221 11BE 4D414E55 CHOOSE: DB'MANUAL/AUTO? '
0222 11C2 414C2F41
0223 11C6 55544F3F
0224 11CA 20
0225 11CB 4E45572F NEWCON: DB'NEW/CONTINUE?'
0226 11CF 434F4E54
0227 11D3 494E5545
0228 11D7 3F
0229 001A          BORDER: DS 25+1          ;26 BYTES FOR ØVERSTE KANT
0230 0190          BOARD:  DS 400           ;400 BYTES FOR BRÆDT
0231 0C29          CURSOR: EQU 0C29H
0232
0233 1382          END
```

```
ASB. LIND
T 1000 11D8
  1000 21 30 30 22 AE 11 3E 0C BC
  1008 F7 11 D0 0B 21 97 11 01 C5
  1010 0C 00 ED B0 21 D8 11 06 D9
  1018 19 0E 0F 36 FF 23 10 FB C1
  1020 36 FF 23 06 18 36 00 23 FF
  1028 10 FB 0D 20 F3 06 1A 36 B9
  1030 FF 23 10 FB 11 E4 0B 21 8E
  1038 B0 11 01 0D 00 ED B0 DF 93
  1040 7B FE 0D 20 04 DF 6A 18 5B
  1048 F6 FE 08 20 04 F7 F7 18 7E
  1050 EE FE 20 20 05 F7 DF 69 D0
  1058 18 E5 FE 51 28 0F FE 52 3B
  1060 28 9E 2A 29 0C 36 80 23 6E
  1068 22 29 0C 18 E9 11 E4 0B D0
  1070 21 BE 11 01 0D 00 ED B0 1B
  1078 CF FE 4D 20 04 3E 00 18 1C
  1080 06 FE 41 20 F3 3E 08 32 60
  1088 62 11 11 E4 0B 21 A3 11 E0
  1090 01 0D 00 ED B0 2A 29 0C AA
  1098 36 FF 21 0A 08 11 F2 11 24
  10A0 7E 3C CA 61 11 3D 20 09 0C
  10A8 D5 11 10 00 19 D1 13 18 C3
  10B0 EF CB 6E 20 08 3E 01 12 61
  10B8 23 23 13 18 E3 AF 18 F7 DA
  10C0 DD 21 F2 11 21 76 01 06 6F
  10C8 00 DD CB 00 7E 20 45 DD 40
  10D0 7E E6 CD 8F 11 DD 7E E7 F3
  10D8 CD 8F 11 DD 7E E8 CD 8F F4
  10E0 11 DD 7E FF CD 8F 11 DD A5
  10E8 7E 01 CD 8F 11 DD 7E 18 57
  10F0 CD 8F 11 DD 7E 19 CD 8F 3D
  10F8 11 DD 7E 1A CD 8F 11 78 73
  1100 FE 03 20 06 DD CB 00 CE AE
  1108 18 0A FE 02 20 06 DD CB 09
  1110 00 46 20 F0 DD 23 2B 7C 1E
  1118 B5 20 AC 01 68 01 11 0A 2F
  1120 08 21 F2 11 CB 7E 28 08 D6
  1128 23 E5 21 10 00 19 EB E1 57
  1130 CB 3E CB 46 20 04 3E 20 DD
  1138 18 02 3E 80 12 13 13 23 7C
  1140 0B 78 B1 20 DF E5 21 F0 7A
  1148 0B 34 7E FE 3A 20 0B 36 AF
  1150 30 2B 34 7E FE 3A 20 02 C8
  1158 36 30 2A EF 0B 22 AE 11 D4
  1160 E1 18 FF CF FE 41 CA 85 C6
  1168 10 18 07 DF 62 FE 4D CA FE
  1170 7D 10 FE 51 C2 C0 10 11 00
  1178 E4 0B 21 CB 11 01 0D 00 83
  1180 ED B0 CF FE 43 CA 8A 10 A2
  1188 FE 4E CA 00 10 18 F3 CB 95
  1190 7F C0 CB 47 C8 04 C9 47 CE
  1198 41 4D 45 20 4F 46 20 4C 9D
  11A0 49 46 45 47 65 6E 65 72 76
  11A8 61 74 69 6F 6E 20 30 30 54
  11B0 4E 65 77 20 70 61 74 74 C4
  11B8 65 72 6E 20 20 20 4D 41 FC
  11C0 4E 55 41 4C 2F 41 55 54 1A
  11C8 4F 3F 20 4E 45 57 2F 43 E3
  11D0 4F 4E 54 49 4E 55 45 3F 42
```

Der vil være mulighed for at få LIFE til N1 fra pogrambiblioteket.







Almindelige oplysninger om foreningen

BESTYRELSENS SAMMENSÆTNING:

Formand: (redaktør)	Asbjørn Lind Sidevolden 23 2730 Herlev 02 91 71 82
Næstformand:	Jesper Skavin Broholms Alle 3 2920 Charlottenlund 02 64 03 14
Kasserer:	Søren Sørensen Højlundvej 13 3500 Værløse 02 48 31 01
Teknisk redaktør:	Ole Hasselbalch Vibeskrænten 9 2750 Ballerup 02 97 70 13
Medlemsmødeleder:	Erik Hansen Lyngby Kirkestræde 6,1 2800 Lyngby 02 88 60 55 (dg. 8 - 15.30)

HENVENDELSE TIL FORENINGEN:

Indmeldelser, adresseændringer o.l. til kasseren
Programbiblioteket til næstformanden.

Øvrige henvendelser til formanden.
(herunder annoncer/stof til NASCOM NYT)

Årskontigent for hele 1980: 75 kr.
Indmeldelsegebyr: 25 kr.

Reduceret kontingent for indmeldelser efter 1.8.80

Nascom brugergruppe.

Gode ven.

Hermed forvarsles,at der afholdes medlemsmøde d. 16. Nov. (søndag).
Sted: Pæd. central. Rustenborgvej 1. 2800. (se om.)
Tid: Kl. 11.00.
Forplejning: som sidst, der er mulighed for,at skaffe til billigste dagspris (leverpostej.spegepølse og dagens tilbud,kaffe,øl,vand og the.)
Emner: Der vil blive assisteret ved udbygning af keyboard iflg.artikel i bladet,der kommer. (v.Jesper Skavin.)
Matematik og højere programmering ved Anders.(kendt fra Poly Data.)
Assemblerprogrammering ved Erik Palsbo.
Andet: Efter dit eget ønske.
Varighed: til vi er færdige,eller ikke orker mere.

Maskinel: Det ku.`være sjovt,hvis du vil deltage med DIT. ang. dette så ring og aftal med ECH. 02 88 60 55. dgl. kl.8. - 15.30. To. til 17.30.

Glæder os til vi ses

Bestyrelsen.


Her
LYNGBY HOVEDGADE
LYNGBY KIRKE
APOSTOLSKE KIRKE
Lyngby Lokalstation
LYNGBYGÅRDSVEJ
RUSTENBORGVEJ
JERNBANEVEJ
KLAMPENBORGVEJ
NØRGAARDSVEJ
RÅDHUSET
LYNGBY TORV
JERNBANEPLADSEN
HOVEDVEJ A5
FOLKEPARKEN
BAPTIST KIRKEN
BUDDINGEVEJ
ENGELSBORGVEJ
SAGSVÆRDVEJ
ENGELSBORGSKOLEN
KONGENS
LYNGBY
ULRIKKENBORG ALLE
PIPERSVEJ
STATSSKOLE
CHRISTIANS KIRKEN
JOHAN WILMANNSVEJ
HOLLANDSVEJ
CHRISTIAN X's ALLE