



**DU BEHÖVER**

**industriell**

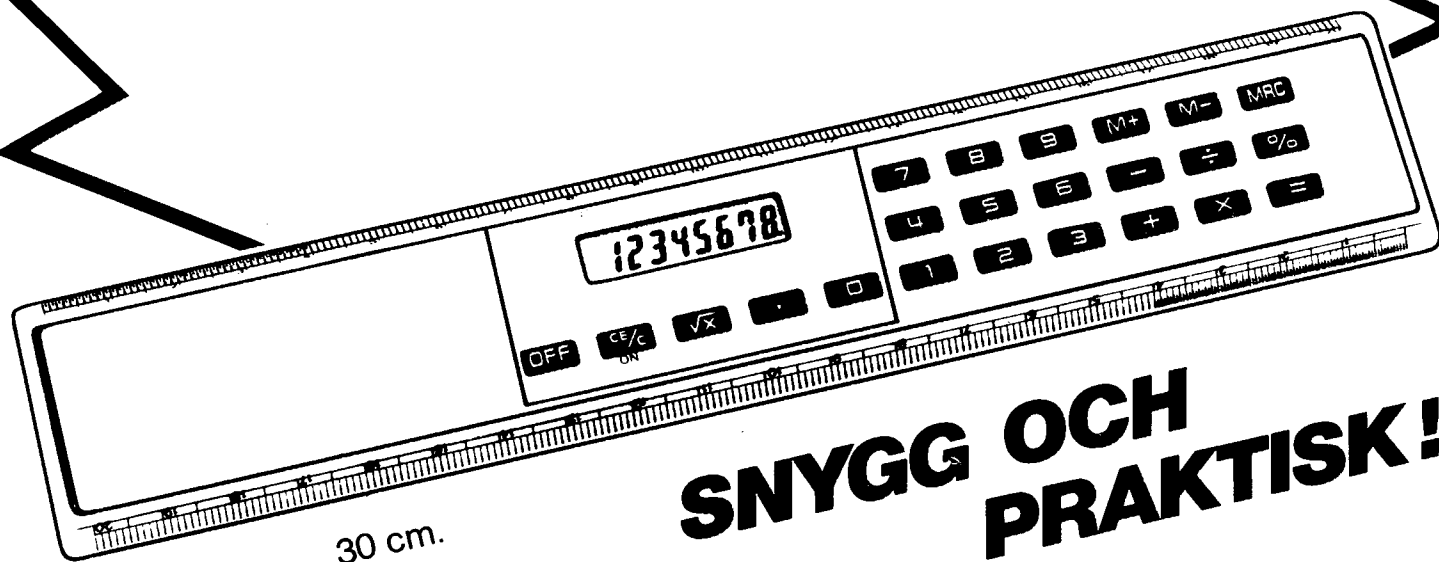
# **Data teknik**

**SVERIGES ENDA SPECIALISERADE  
DATATIDNING FÖR TEKNIKER!**

**PRENUMERERA NU!**

**SNABBHETSPREMIE**

*Skicka in den lösa kupongen i  
tidningen redan idag!*



**SNYGG OCH  
PRAKTISK!**

## Medlemsorgan för ABC-klubben

Vidängsvägen 1, 161 33 Bromma

ISSN 0349-3652

Ansvarig utgivare: Stig Löfgren

Redaktör: Ulf Sjöstrand

I redaktionen: Odd Rolander, Claes Schibler

ABC-klubbens postgiron:

Medlemsavgifter: 15 33 36-3

Publikationer: 62 93 00-5

Q-Zentralen: 43 51 74-8

Bankgiro: 216-25 43

Telefoner:

08-80 15 22 Automatisk telefonsvarare med aktuell klubbinformation

08-80 15 23 Automatisk telefonsvarare med information om monitorsystemet

08-80 17 25 "Prattelefon" till klubblokalen

08-19 44 80 Kansli, kontorstid

Monitorer:

08-80 64 40 300/300 bps V21 eller 1200/75 bps V23

(gruppnummer)

08-80 11 55 1200/1200 bps V22

031-54 75 85 300/300 bps V21

## Annonspriser fr o m nr 1, 1986

1/1-sida 185 × 260 mm . . . . . 3.500:-

1/2-sida 185 × 128 mm, eller 90 × 260 mm . . . . . 2.100:-

1/3-sida 185 × 85 mm, eller 60 × 260 mm . . . . . 1.400:-

2/6-sida 125 × 128 mm . . . . . 1.400:-

1/4-sida 90 × 128 mm . . . . . 1.300:-

1/6-sida 60 × 128 mm . . . . . 900:-

2 st 1/1-sidor i uppslag . . . . . 7.800:-

2:a omslagssida . . . . . 4.375:-

3:e omslagssida . . . . . 4.200:-

4:e omslagssida 185 × 225 mm . . . . . 4.800:-

Begärd placering 10% förhöjning.  
Tidningen ansvarar ej för att införda programlistningar är korrekta.

Upphovsrätt gäller för införda program om inget annat anges.

Tryck: Märstatryck AB 1986

Lämnad till tryck 3 mars 1986.

Upplaga 8 000 ex.

## NUMMER 1, 1986

### INNEHÅLL

|                                                            |    |
|------------------------------------------------------------|----|
| Omslag                                                     |    |
| LEDARE av Stig Löfgren                                     | 4  |
| Brev till styrelsen av John Högdahl                        | 3  |
| Svar från styrelsen                                        | 3  |
| RAM-floppy-projektet av John-Erik Näslund                  | 3  |
| ABC-kassett/diskett nr 18                                  | 4  |
| Manual till OPTLOAD för ABC-serien av Kristoffer Eriksson  | 4  |
| Monitorn med LUX-NET av Bo Kullmar                         | 5  |
| Manual till TED av Anders Franzén                          | 6  |
| Hur fort går det? av Sven Wickberg                         | 8  |
| Manual till KEY av Kristoffer Eriksson                     | 9  |
| Konsten att programmera i BASIC av Sven Wickberg           | 12 |
| Kommentar till ett bokstavsspel av Sven Wickberg           | 14 |
| ABC-klubbens framtid av Bo Kullmar                         | 16 |
| ABC-klubbens årsmöte 1986-02-22 av Bo Kullmar              | 16 |
| Om TVSEARCH av Egil Fjelddahl                              | 16 |
| Om checksumma på RAM-floppy av Harry Peltoniemi            | 16 |
| Musik på ABC80 av Fredrik Tengroth                         | 17 |
| Varför blir det på detta viset? av Olle Wirén              | 17 |
| Angående flexskiveenheter av Bengt Ask                     | 17 |
| VÄXLAPGM.UTL av Anders Franzén                             | 17 |
| Grekerna kunde sudda av Leopold Lundström                  | 18 |
| Ännu mer om primtal av Leopold Lundström                   | 21 |
| Ljus i mörkret av Tomas Wikström                           | 25 |
| Manual till QSORT för ABC80                                | 26 |
| Om sortering av Sven Wickberg                              | 27 |
| OUT-kommandon för ABC99 av Gerry Eriksson                  | 32 |
| Radannonser                                                | 34 |
| SKRIVUT av Ture Pålsson                                    | 36 |
| KLAUS kryptiska sidor av Sven Wickberg                     | 37 |
| ABCVDP, grafik med ABC80 av Lars Karis                     | 38 |
| Mera räknepul med CBS av Sven Wickberg                     | 39 |
| Från 80 till 800 av Sven Wickberg                          | 40 |
| Reaktionstest av Sven Wickberg                             | 42 |
| MSG-utdrag av Stefan Berg                                  | 44 |
| Möte ABC800                                                |    |
| Möte ABC80                                                 |    |
| Möte Medforum                                              |    |
| Möte MONITOR                                               |    |
| Möte Lokalavdelningar                                      |    |
| Möte FORTH                                                 |    |
| Möte CP/M                                                  |    |
| En billig talsyntes av David Andersson                     | 55 |
| Utgivningsplan 1985 för ABC-bladet                         | 57 |
| ALFALIB 1986-02-22                                         | 58 |
| ABC-Öst, årsmötesreferat av Lars Källner                   | 62 |
| ABC-Stockholm                                              | 62 |
| ABC-Stockholm, referat från datorsnack av Marianne Forsman | 62 |
| Förteckning över databaser av Bernt Andersson              | 63 |
| Div notiser av Bo Kullmar                                  | 64 |
| Ang lagerhanteringsprogram av Sten Åsberg                  | 64 |

## Medlemsavgifter för 1986

Styrelsen har mandat från årsmötet 1985 att avisera

Seniorer 160 Skr Juniorer 100 Skr

Avgiften fastställs på årsmötet 1986-02-22

Junior räknas man t o m det kalenderår man fyller 18 år. Ange därför personnummer när Du betalar medlemsavgifter.

Medlemskapet är personligt och avser fysisk person. Medlemskapet räknas per kalenderår och Du får automatiskt det löpande årets förmåner retroaktivt om Du inte markerat annat årtal på talongen när medlemsavgiften betalas in.

Medlem blir Du enklast genom att sätta in medlemsavgiften på ABC-klubbens postgirokonto 15 33 36-3 och ange en entydig avsändare.

## ABC-klubbens styrelse för 1986

(enligt årsmötet 1986-02-22)

Ordförande: Stig Löfgren

Vice ordförande: Bo Kullmar

Ledamöter: David Andersson, Kjell Brealt, Magnus Hedner, Jan Holmberg, Jan Liebe-Harkot, Tom Sjöberg  
Suppleanter: Kent Berggren, John-Erik Näslund, Robert Svedjehammar

# LEDAREN

## Nytt verksamhetsår!!!

Som återvald ordförande får jag tacka för Ert förtroende. Jag skall nu tillsammans med styrelsen göra mitt bästa för att leda klubbens arbete ytterligare ett år.

ABC-klubben har ju inte några politiska värderingar, men oavsett uppfattning, kan det ändå inte hjälpas att att man känner sig en aning tom i huvudet när man sätter sig ned för att skriva ledaren precis när man fått höra att Palme blivit mördad. Tankarna går tillbaka till händelsen och man frågar sig vad det är för tid vi lever i när detta kan hända i Sverige men livet måste ändå gå vidare.

### ABC-ÖSTS ÅRSMÖTE

Vi fick en inbjudan att besöka ABC-Östs årsmöte. Jag var den enda från styrelsen som hade tillfälle att åka ned, men från ABC-Stockholm kunde min son Benny komma med. Trots att det inte är en storstadsregion som Väst och Stockholm, överraskades vi av ett välbesökt årsmöte. Man redovisade en stor mötesaktivitet. Flera rabattavtal hade också slutits under året (se ÖSTs årsberättelse på annan plats i bladet). Observera att en del erbjudanden gäller för alla ABC-medlemmar. Man kunde också, inte utan stolthet, redovisa att man av LUXOR I MOTALA fått ett Luxnät med tillhörande 10 Mb winchester och en ABC-806 och några 802. Man har nu för avsikt att starta en lokal monitor placerad i Linköping. ABC-klubben ställer upp med telefonlinjer och modem.

### ABC-DAGEN

Traditionsenligt hade en utställning arrangerats där ett flertal företag ställde ut. Utställarna gladdes åt mycket folk, och en av dem fick överaskande en utlandskontakt för återförsäljning.

Efter själva årsmötesförhandlingarna tog en expert-panel plats på podiet. Denna panel bestod av Kjell Strömlid vd för NOKIA LUXOR infomations system, Bengt Lönnqvist numera i samma företag, Harrgot Lindmark och Ulf Sandberg från DATAINDUSTRIER, samt Benny Löfgren och Lars-Göran Göransson från Klubben. En livlig diskussion följde, dels om NOKIAS planer för LUXOR-datorernas framtid och dels tekniska frågor. Frågorna var kanske i snällaste laget och möjligen skulle styrelsen förberett med ett frågebatteri. En detaljerad redogörelse kommer i senare nummer av ABC-bladet.

Under paneldiskussionen tycker jag inte det framkom särskilt mycket som kan skingra oron för framtiden. Det som jag uppfattade som väsentligt var att man i motsats till vad som framskymtat i pressen under det gångna året sålt flera ABC-datorer än man hoppats och att konkurrensen från IBM och andra inte är så stor som alla tror. Alla tycks ju vara paralyserade så snart någon nämner IBM. Att sedan deras största återförsäljare gick i konkurs under det gångna året tycks heller inte bekomma någon.

### BASIC II/PC FRÅN DIAB

DATAINDUSTRIER kom med en verkligt positiv nyhet när de meddelade att DIAB skall sälja en MS-DOS kompatibel BASIC som programvara, passande bland annat till IBM maskinerna. Detta måste ju uppfattas som positivt eftersom ABC-programmen då får större användning och efterfrågan. Flera ABC-användare har också ABC hemma och IBM på kontoret.

ABC-klubbens medlemmar fick också direkt från panelen ett erbjudande att köpa BASIC II/PC direkt från DIAB för 1.790:- inklusive manual (moms tillkommer) då är en klubbabatt på 160:- avdragen.

### FRAMTIDEN FÖR ABC-DATORERNA

Det har ju bland medlemmarna spridit sig en känsla av osäkerhet för hur framtiden skall bli med ABC-datorerna. De snabba kasten som skett på senare år inom LUXOR, naturligtvis till stor del ett medärvt problem från LUXOR TV men som i högsta grad också påverkat den utveckling som har varit och kanske i ännu högre grad den kommande.

NOKIA säger nu att det inte skall påverka ABC-datorerna i någon större grad att man lägger ned utvecklingsavdelningen för datorer och flyttningen av försäljningskontoret till Stockholm. Detta är väl lika blåögt som den blå färgen på den finska flaggan. Jag vill se hur många av de ursprungliga entusiastiska LUXOR-anställda som finns kvar på försäljningsavdelningen efter en sådan operation??? Och teknikerna - de skall inte finnas!!! Med så mycket som användare, programvaruhus, och andra företag investerat, i form av hårdvara, kunskande, programvaror och system för ABC-datorerna runt om i landet, får man inte handskas så nonchalant med en produkt. Man får helt enkelt inte ta risken att kontinuiteten äventyras.

Vi vet ju alla vid det här laget att IBM PC och andra MS-DOS kompatibla maskiner oftast kommer till korta när de skall mäta sig med ABC-datorerna och de programvaror som redan nu finns.

Det är därför nödvändigt att vi på alla håll värnar om ABC-datorerna och dess BASIC, inte minst om de programvaruhus som finns i Sverige skall fortsätta att satsa på nya och ännu bättre program.

Under de samtal som jag haft tidigare med Kjell Strömlid och även nu, har han försäkrat att NOKIA inte tänker lägga ned ABC-datorerna utan att hela 800-serien är ett mycket starkt kort som LUXOR har. Han anser sig felciterad när det skrivits om att man skall lägga ned all utveckling.

Jag kan inte hjälpa att jag fortfarande tycker att det är något som inte är klarlagt. Jag har svårt att se en framtid för ABC-datorerna utan utveckling, och med en splittrad försäljningsorganisation. Jag kräver NOKIA/LUXOR på ett omedelbart klarläggande om planerna för framtiden.



<872>

Stig Löfgren

# Brev

# Svar

## Till styrelsen

I en rikstäckande förening har flertalet medlemmar svårighet närvara vid årsmöten - så även jag nordligt boende. Framför därför några synpunkter brevledes.

Utvecklingen på datasidan går fort, från den ursprungliga ABC80-datorn har familjen blivit en heterogen samling. För min del har nu samlats en knippe kassetter, som min ABC802 knappast är betjänt av varken till innehåll eller datortyp. Klubben har också satsat stora belopp på monitorsystem, trots att bara ett fåtal av oss medlemmar har möjlighet eller intresse nyttja monitorn.

ABC-bladet 4,1985 består till mer än hälften av utskrift från möten och liknande på monitorn. En noggrann genomläsning ger en del korn med mest agnar och boss. Direkt beklämmande är Benny Löfgrens angrepp mot Sven Wickberg och de handikappade. Diskussionen om vilka tangentbord som kommer att tillverkas i fortsättningen ger inget svar på frågan - och det är ju intressanta.

Om klubben skall kunna bestå beror i mycket på att ABC-bladet har ett bra innehåll som är lätt att tillgogogöra sig och ge ett brett mått av kunskap - inte bara för ABC-familjen.

Mitt förslag är därför att klubben förstärker den redaktionella delen även om det sker på bekostnad av MONITORN (Jmför text 2064, Sven Wickberg). Jag förutsätter därvid att samarbetet med Luxor förbättras så att vi t ex får veta när tangentbord utgår ur produktionen och vilka åtgärder vi då kan vidtaga. Självklart skall personangrepp därvid undvikas och allraminst angrepp på handikapphjälpmedel. För många handikappade är ju datorn enda möjligheten att kommunicera med omgivningen.

Berätta i stället hur datorn kan nyttjas av de handikappade - för de datorlösningarna har också allmänintresse.

Att fortsättningsvis ge ut band eller skivor med så blandade program som nu sker, förefaller meningslöst. Jag tror det är bättre med rejäla program, som vi får betala en slant för. (Så som sker med KRONSTAT)

Med vänlig hälsning

<3936>

John Högdahl

Tack för Ditt brev, det är tyvärr alltför sällan som vi får någon respons från medlemmarna i form av brev och synpunkter. Det är nödvändigt när man vill veta om man gör ett bra eller dåligt arbete. Dock har denna situation betydligt förbättrats sedan vi fick igång MSG på monitorn.

Först av allt vill vi påpeka att ALLT arbete i klubben görs ideellt och på lediga stunder, alltså även om vi kallar det programredaktion, tidningsredaktion, kassettredaktion monitor mm. Den enda betalda arbetskraft som klubben har är för kansliet, därav den låga medlemsavgiften. Som Du säger så består ABC-datorerna nu av heterogen samling maskiner. Vi har under åren byggt upp en programredaktion som efter bästa förmåga, av gratis inskickade program, försöker producera ett så brett utbud som möjligt, dels till de olika maskinerna och dels olika intresseinriktningar. Meningen med detta är att medlemmarna då och då skall skall hitta program som passar det egna intresset och den egna maskinen.

Klubben har nu börjat ge ut större programpaket till självkostnadspris, de första programmen i denna serie blir KRONSTAT och NEWBAS, och flera kommer med den takt som programredaktionen hinner med.

Tillkomsten av monitorn har kraftigt ökat möjligheterna till ett meningsfullt erfarenhetsutbyte och detta speciellt sedan MSG tillkom. Att bara en minoritet har nytta av monitorn är fel, det senaste året har 1048 st, eller ca:1/4 av medlemmar någon gång varit inloggade, och bara under sista månaden har totala antalet inloggnings varit 6079.

Visserligen är det inte riktigt rättvist med de olika telefonsaxorna, men med televerkets giv "ett rundare Sverige" så har ju situationen förbättrats något.

Sedan måste också påpekas att alla medlemmar faktiskt har nytta av monitorn, eftersom de flesta program som skänks till klubben sänds in till monitorn. Monitorn är också programredaktionens arbetsbas.

Tittar man på utgifterna för monitorn sista året var dessa 40.816:- eller mindre än 5% av budgeten. Att kostnaden är så låg beror att vi fått utrustning för betydande belopp av Luxor, TDX och DIAB.

I tidningsredaktionen har vi ca 4.500 st medarbetare och Du är en av dessa. I praktiken tas allt med i tidningen som sänds in till oss. Det som vi kallar redaktionen är ett par medlemmar som helt ideellt på sin fritid skriver ut och monterar upp tidningen blad för blad och tar dessa till tryckeriet. Allt kan göras bättre, men vi kan nog skryta med att vi har den i särklass proffsigaste tidningen som har framställts helt ideellt.

Vi är klart medvetna om att bladet innehållsmässigt är alldeles för expertinriktat för flertalet läsare, men bladet blir vad medlemmarna gör det till. Därför efterlyser vi också ständigt enkla vardagsnära artiklar och frågor. Vi honorerar också de flesta insända artiklar med en extra flexskiva / kassett tillbaka med den egna insända. Här vill vi också påminna om att med brev och artiklar alltid medsända texten på kassett eller diskett.

Mot denna bakgrund så kan Du se att tidningens innehåll inte minskat för att sidorna om MSG kommit till utan den har tvärtom ökat sitt omfång. Visserligen har också tryckkostnaden ökat något men i gengäld så har vi fått med flera annonser, så man kan nästan säga att MSG sidorna är gratis.

Eftersom dessa texter skrivs in i monitorn i realtid, dvs att texterna lagras i MSG-systemet allteftersom man skriver på tangenterna så finns ingen tid till eftertanke eller redigering i detta moment. När texterna sedan kopieras ut från monitorn (en försvarlig bunt flexskivor) sker det en utgallring och sortering. Vi vill då så mycket som möjligt behålla fräschören i inläggen och ej gå in och redigera. Därför kommer ibland texter som tyckas lösryckta ur sitt sammanhang och kan innehålla stavfel, tappade bokstäver mm. Vi är medvetna om att detta arbete är svårt och tidsödande för vi rymmer ju inte allt material, men vi försöker få med de inlägg som har störst allmänintresse.

Beträffande ärende "ABC-99 tangentbord till ABC-806/802" och "Alternativa kommandon". Klubben är ju ett forum för erfarenhetsutbyte och då skall medlemmarna dryfta sina erfarenheter. Det vore i detta sammanhang tacknämligt om ett klart besked om tangentborden lämnats från LUXOR, och ännu bättre om man sänd frågan på remiss till MSG först. Frågan om alternativa tangentar har väl i sak besvarats men tyvärr i en ton som inte borde höra hemma i MSG. Formuleringen vittnar om arrogans och omognad. Detta tråkiga hade dock ett gott med sig, nämligen vi fick en medlem till av den stora redaktionen att skriva till oss och det tackar vi för.

**ABC KLUBBEN**

## Bygg mera hårdvara/mjukvara till ABC-datorer.

Nu startar ABC-klubbens RAM-floppy-projekt (på 1 Mbyte). Har du ideer eller kunskaper men ej resurser att själv ta fram det, kom med i klubbens projekt!

Skriv till:

ABC-klubben  
RAM-floppy-projektet  
Vidängsvägen 1  
161 33 BROMMA

RAM-floppyn är avsedd för både ABC80 och ABC800

I nästa nummer av ABC-bladet kommer vi att redovisa vilka ideer som inkommit från läsarna och hur långt RAM-floppy-projektet kommit.

Sammanhållande kontaktperson blir i start-en John-Erik Näslund <1105> 08-291126 (hem), 08-7640598 (arb)

## Att fundera på för RAM-floppy folket:

Tillsammans med en 1 Mbytes RAM-floppy så finns det plats för 15-20 minuter digitaliserat ljud. (Se annan artikel i bladet).

- \* Batteri backup ett krav?
- \* Databas i RAM-floppyn?
- \* Mjukvarustöd?
- \* Datasäkerhet i RAM-floppyn!
- \* Drivrutin i ROM?

# ABC-kassett/ diskett nr 18

Producerad 1986-02-15

Manualerna till TED, QSORT, KEY och OPTLOAD finns återgivna i sin helhet på annan plats i detta nummer.

## TED.BAC / TEDI6.BAC

TED är en texteditor för ABC80. Den kan användas tillsammans med kassettbandspelare eller flexskiveenhet som yttre minne. Fungerar även tillsammans med 80-tecken-tillsats (MYAB). TED besitter en mängd finesser som t ex full markörstyrning, bläddring, sökning, utbyte, tabulering och förflyttning av stycken.

## QSORT för ABC80

QSORT för ABC80 QSORT är ett program-paket klubben fått från P-DATA. QSORT-BAC är en sorteringsrutin som skrivits i assembler och är mycket snabb.

Assemblerkoden placeras strax under DOSbuffertarna i ABC80-minnet och den kräver 1 Kbyte. Inladdning sker med RUN QSORT. När QSORT laddats in kan man köra program som använder sorteringen. Att anropa sorteringsrutinen från ett BASIC-program är mycket enkelt: K=CALL(-3703%) Före detta anrop måste man emellertid på något sätt beskriva vilken strängvektor som skall sorteras, sorteringsordningen, antalet strängar, och några parametrar till. För denna "kommunikation" med QSORT använder man sig av en enda variabel, som skall heta P\$\$. Se vidare QSORT.INF.

## KRYSS.BAS för ABC80

KRYSS hjälper till att lösa korsord av typen krypto, d.v.s där varje ruta i korsordet har ett nummer som mostvarar en bokstav.

Börja med att definiera upp nummer i rutorna och ev. markera svarta rutor. Om rutnätet är för litet går man ner till högra hörnet och trycker CTRL-T.

Lös sedan kryptot med att gissa vad det ska vara för bokstäver i rutorna. I listan till höger på skärmen kan man se vilka koder som är vanligast. Felmeddelanden kommer nedtill på skärmen. Tryck "?" för hjälp. Programmet kan för närvarande inte spara data på fil.

## LARMKLOC.BAS för ABC80

Visar en klocka med visare på skärmen. Man sätter tiden med HHMMSS och CTRL-S. Man sätter larmtid med HHMM(SS) och "A". Man stänger larm med "\$". Larm ger tjut i högtalaren samt startar bandspelaren om sådan är ansluten.

## HANOI.BAS för ABC800-serien

HANOI.BAS är ett programmeringstekniskt intressant program, som utnyttjar rekursiva funktioner. Det demonstrerar lösningar till det klassiska problemet "Tower of Hanoi" vilket går ut på att trava block i en viss ordning.

## KEY för ABC800-serien

KEY.BAC är en tangentbordsbuffert med möjlighet till omprogrammering av alla tangenter. Dessutom kan Du med ABC 800 och 802 få klickningar i högtalaren när Du trycker på tangenterna.

Tangentbordsbufferten gör att Du inte behöver sitta och vänta medan datorn laddar ett program eller gör något annat tidskrävande - Det är bara att knappa vidare så läser datorn ifatt så småningom.

Omprogrammering av tangenter innebär att när Du trycker på en tangent får datorn i sig något helt annat än vad som står på tangenten. Det kan vara ett enda tecken eller en hel flod av bokstäver från en enda tangent-tryckning.

## LIB-lista kassett/diskett nummer18

| PROGRAM             | ---- | STL | --- | BESKRIVNING                                                                                                                                                                                                                                             | ----- |
|---------------------|------|-----|-----|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------|
| CASDISK4.BAS        | 14   |     |     |                                                                                                                                                                                                                                                         |       |
| CASDISK2.800        | 36   |     |     |                                                                                                                                                                                                                                                         |       |
| VISA .BAS           | 4    |     |     | För ABC 80/800. Används för att läsa textfiler t ex .TXT, .INF, .REM                                                                                                                                                                                    |       |
| ***** ABC 80 *****  |      |     |     |                                                                                                                                                                                                                                                         |       |
| TED .TXT            | 63   |     |     | Texteditor för ABC80. Utförlig manual. (använd VISA.BAS)                                                                                                                                                                                                |       |
| TED .BAC            | 36   |     |     | Maskinkod för ABC80 med 32 Kbyte minne                                                                                                                                                                                                                  |       |
| TEDI6 .BAC          | 36   |     |     | Maskinkod för normal ABC80                                                                                                                                                                                                                              |       |
| QSORT .INF          | 26   |     |     | Anvisningar till sorteringspaketet (använd VISA.BAS)                                                                                                                                                                                                    |       |
| QSORT .BAC          | 7    |     |     | Maskinkoden som laddas med RUN QSORT                                                                                                                                                                                                                    |       |
| SORTDEMO.BAS        | 5    |     |     | Demo till QSORT                                                                                                                                                                                                                                         |       |
| NEWSEC .BAS         | 5    |     |     | Tillhör QSORT                                                                                                                                                                                                                                           |       |
| KRYSS .BAS          | 20   |     |     | Korsordshjälp. Se vidstående text                                                                                                                                                                                                                       |       |
| ARKEOLOG.BAS        | 37   |     |     | Du är en berömd arkeolog som ska utforska en faraosk pyramidens inre hemligheter....                                                                                                                                                                    |       |
| ARKEOLOG.HLP        | 39   |     |     | Snabbkurs för ARKEOLOGer                                                                                                                                                                                                                                |       |
| LARMKLOC.BAS        | 16   |     |     | Analog-ur med alarm                                                                                                                                                                                                                                     |       |
| ***** ABC 800 ***** |      |     |     |                                                                                                                                                                                                                                                         |       |
| GLIPP .BAC          | 2    |     |     | Det sista i en serie spel-program från                                                                                                                                                                                                                  |       |
| GLPMONO .BAC        | 11   |     |     | GLPFG .BAC 11                                                                                                                                                                                                                                           |       |
| GLPFG .BAC          | 11   |     |     | DIAB som klubben fått av Luxor. Spelet är                                                                                                                                                                                                               |       |
| GLPMONO .ABS        | 22   |     |     | ej avsett för 802 eller 806.                                                                                                                                                                                                                            |       |
| GLPFG .ABS          | 21   |     |     | Filerna utnyttjas olika beroende på vilken 800-maskin man använder.                                                                                                                                                                                     |       |
| LADDA .SYS          | 2    |     |     |                                                                                                                                                                                                                                                         |       |
| HANOI .BAS          | 13   |     |     | Kör 'Tower of Hanoi' på skärmen !                                                                                                                                                                                                                       |       |
| KEY .INF            | 73   |     |     | Manual till KEY.BAC, KEYPROG.BAS, STÄLLPAR .BAS                                                                                                                                                                                                         |       |
| KEY .BAC            | 11   |     |     | Huvudprogrammet                                                                                                                                                                                                                                         |       |
| KEYPROG .BAS        | 10   |     |     | För programmering av tangenter vid start av KEY                                                                                                                                                                                                         |       |
| STÄLLPAR.BAS        | 26   |     |     | Ställer in parametrar i (vissa) program.                                                                                                                                                                                                                |       |
| OPTLOAD .INF        | 22   |     |     | Manual till OPTLOAD .BAC                                                                                                                                                                                                                                |       |
| OPTLOAD .BAC        | 4    |     |     | Programmet gör det möjligt att med hjälp av en ADDOPT-fil ladda REL-filer utan att göra reset. I stället gör man CHAIN eller RUN till programmet OPTLOAD .BAS. Se till att samma option inte laddas två gånger ! Programmet utnyttjar även STÄLLPAR.BAS |       |

```

+-----+
!      OPTLOAD, STÄLLPAR.BAS      !
!      - Laddning av Addopt-filer. !
!      För ABC800-serien. 1985-07-10 !
!      <5357> Kristoffer Eriksson !
!      ABC-klubben !
!      Får kopieras fritt för !
!      ickekommersiella syften. !
+-----+

```

OPTLOAD gör det möjligt att med hjälp av en ADDOPT-fil ladda REL-filer utan att göra RESET. I stället gör man CHAIN eller RUN till programmet OPTLOAD.

## 1. Hämtning från monitorn.

Programmet OPTLOAD.BAC ligger i binär form, och måste hämtas med kommandot "GET,H OPTLOAD.BAC" eller "TYPE,H OPTLOAD.BAC". Därefter ska den överförda hex-filen konverteras tillbaks till binärform med hjälp av programmet HEXTOFIL, som också finns i monitorn. Du bör också hämta STÄLLPAR.BAS som används för att ställa in parametrar i OPTLOAD.BAC om du inte redan har den.

## 2. Användning.

OPTLOAD laddar ADDOPT.ABS som i sin tur laddar en ".REL"-fil, vanligtvis ISAMOPT.REL, OPTROSH.REL, TERMOPT.REL, eller någon kombination av dessa. Det är alltså minst tre steg i denna kedja, det blir därför ett flertal inställningar som måste göras innan man kan utnyttja det hela. Förslagsvis gör du kopior av ADDOPT.ABS och OPTLOAD.BAC med andra namn, ifall du vill ha flera olika varianter på samma diskett.

## 3. Exempel.

I det här exemplet visas hur man gör i ordning ett program med namnet ISAMOPT. BAC som laddar ISAM.

Gör en kopia av ADDOPT.ABS (finns på systemskivan) och kalla den för ADDISAM.ABS. Gör också en kopia av OPTLOAD.BAC och kalla den för ISAMOPT.BAC.

Nu måste ADDISAM.ABS ställas in så att ISAMOPT.REL laddas av den. Detta kan göras med programmet CONFIG som funnits på ABC-kassett/diskett nr 16, men då måste det ändras så att det behandlar den kopian av ADDOPT.ABS och inte originalet. Det är kanske enklare att "göra det själv". På systemskivan finns ett litet program som heter SETOPT.BAC som man lätt kan ändra i så att den bearbetar rätt fil (ADDISAM.ABS i detta exempel) och anger rätt REL-fil (ISAMOPT.REL här). Lagg märke till att REL-filens filnamn ska anges med exakt 8 tecken och filtypen med exakt tre, överblivet utrymme ska fyllas med blanktecken ("ISAMOPT REL" här).

Härnäst måste ISAMOPT.BAC ställas in så att den laddar ADDISAM.ABS. Det gör man genom att köra programmet STÄLLPAR. Ange ISAMOPT.BAC som svar på frågan om filnamn. Då visas nuvarande inställningar, varefter dessa kan ändras. Som "Adoptfil" ska anges "ADDISAM ABS". Även här anges filnamnet med exakt 8 tecken och filtypen med tre, sammanlagt 11 tecken. Ingen punkt mellan filnamn och filtyp. Dessutom kan man ange namnet på ett BASIC-program som ska startas efter inladdningen. Här anges filnamnet på normalt sätt. Till det här exemplet behöver inget namn anges här. Sen ska inställningarna sparas.

Nu är det bara att provköra. Gör det med RUN ISAMOPT. Om man nu provar att skriva ISAM READ \$1 A\$ ska maskinen svara med felkod 32 (filen ej öppnad) och inte 220 (förstår ej). Detta visar att ISAM har blivit inladdat.

#### 4. Användning tillsammans med egna program.

Den största nyttan med OPTLOAD är nog att man när som helst kan ladda in en REL-fil utan att behöva plocka fram en särskild diskett och trycka RESET. Man kan göra iordning kopior för alla olika REL-filer man brukar använda. Men det går också att anropa OPTLOAD från andra program:

Det finns två sätt att använda OPTLOAD tillsammans med egna programsystem. Man kan med STÄLLPAR ställa vilken ADDOPT-fil som ska laddas och vilket BASIC-program som ska CHAIN:as en gång för alla, men om man i stället skriver in bara blanktecken som ADDOPT-fil och en tom sträng (noll tecken) som CHAIN-fil, kan dessa saker variera från gång till gång. Med följande

programsnutt i det program som CHAIN:ar OPTLOAD (eller en kopia därav) görs de installationer som behövs:

```
FOR I=1 TO LEN(Chainfil$)
POKE 63746+I,ASC(MID$(Chainfil$,I,1))
NEXT I
POKE 1,255
POKE 63767,Addoptdev
FOR I=1 TO LEN(Addoptfil$)
POKE 63767+I,ASC(MID$(Addoptfil$,I,1))
NEXT I
POKE 1,255
CHAIN "OPTLOAD"
```

där Chainfil\$ skulle kunna innehålla "NÄSTA.BAC" och Addoptfil\$ "ADDISAM.ABS". Addoptdev är enhetsnumret på den diskett-drive där addopt-filen finns. 255 ger

sökning på alla enheter. I programmet NÄSTA kan man avläsa hur laddningen gick. Om PEEK(64512) ger 255 gick laddningen bra så långt OPTLOAD kan avgöra.

#### 5. Utnyttjat minne.

OPTLOAD utnyttjar dosbuffertar ganska flitigt. POKE-historian som beskrevs ovan hamnar i dosbuffert 4. OPTLOAD ligger delvis och tidvis i buffert 6, och delvis i botten av det lediga RAM-minnet. ADDOPT lägger sig i buffert 3 och 4 samt utnyttjar 1 och 2. Naturligtvis används även buffert 0. "Resultat-byten" 64512 ligger i buffert 7.

<5357>

Kristoffer Eriksson

lika med "LIB,ORDT.S".

## Monitorn

#### LUX-NET

Sedan den 8:e december kör vi det nya monitorsystemet på LUX-NET. En del nya funktioner finns i systemet.

#### Hängningar i LUX-NET

Vi har haft en del problem med att nätkommunikationen i LUX-NET har hängt sig. Hängningarna har berott på att vi har glömt att bygga vissa D-NET modem (Luxor kallar dem för nodadepter). Som en följd av hängningarna har vi resettat centralen och detta har givit problem med den ISAM-databas innehåller MSG:s texter.

#### Strömvavbrott

Vi drabbades av ett strömvavbrott vid 12-tiden den 10 februari. Själva nätet startar om automatiskt med noderna kommer inte igång på grund av att de har en timeout på 5 sekunder. Inom 5 sekunder hinner inte centralen komma igång. Benny Löfgren skall dock på DIAB enligt uppdrag från Luxor göra så att man själv kan ändra timeoutiden.

#### Hur fungerar LUX-NET:et

Det finns en hel del nya funktioner i vårt monitorsystem som inte fanns i CAT-NET systemet. Det beror på att nästan alla förbättringar under det senaste halvåret enbart har gjorts i monitornprogramvaran som körs på LUX-NET.

MSG:s texter i LUX-NET ligger inte som i CAT-NET i vanliga filer utan i en ISAM-bas. I CAT-NET systemet hämtades texterna från filer som låg i underbibliotek, texten lästes in i MEM: och därefter lades den ut på linjen. Detta gjorde att man fick vänta lite innan texten dök upp, men när den väl kom, så kom den direkt.

I LUX-NET systemet ligger alla texter, såväl brev som inlägg i en och samma ISAM-bas. När texten läses upp till MEM: så läggs den ut på linjen samtidigt som den skrivs i MEM:. Detta gör att man får texten på bildskärmen lite tidigare, men ryckigt.

Vårt CAT-NET hade ett 128 KB minneskort som buffert mot disken (cash). Detta gjorde att 50-60 % av alla läsningar på winchestern i stället gick mot minneskortet. En sådan rutin saknas till LUX-NET. Dock kör vi med en Tranfor winchester till LUX-NET med en snabb Tranfor kontrollor.

Själva nätet är snabbare än CAT-NET och disken är alltså också snabbare, men detta gör trots allt att systemet upplevs som lite trögare än vårt gamla CAT-NET system. Det märks också att Tranfor-winchestern till LUX-NET får jobba betydligt intensivare än Mica-winchestern till CAT-NET.

En cash rutin till LUX-NET är i skrivande stund utvecklad av DIAB, men ej avslusad.

Nyligen har Luxor dock beställt rutinen av DIAB vilket innebär att den kommer att göras luddig. Med denna rutin så hoppas vi att LUX-NET snabbt bli minst lika snabbt som det gamla CAT-NET:et.

#### Manual

En ny monitormanualen håller just på att tas fram. Den bör vara tryckt när du läser detta. Just när är den gamla slut och vi trycker inte om den utan väntar tills den nya blir klar. Jag beklagar att några troligen har fått vänta på att få manualen.

#### Programbanken

Programbanken listas i detta nummer på annan plats. Vill du få tag på en fil och om du saknar modem så får du vända dig till en medlem som har modem.

#### Nya medlemmar

Nya medlemmar får inte sitt namn i systemet direkt. Först står det bara medlemsnummer och texten "ny medlem". Jag hämtar namn på nya medlemmar då och då från kansliet på skiva. Är du ny medlem så får du ge dig till tåls lite innan ditt namn hamnar i systemet.

#### Kermit

Vi räknar med att ha en Kermit klar till sommaren, vilket innebär att du då bör kunna hämta och skicka in filer till monitorn med Kermit-protokollet.

#### Funktioner i Monitorn:

##### DUMP

Har du tidigare använt WRITE för att skicka in en fil så skall du normalt använda DUMP. Du kan skicka in en fil på max 32 Kbytes med DUMP. Använder du kommandot "DUMP,D" så kan du skicka in en obegränsat stor fil under förutsättning att ditt terminalprogram tar hänsyn till XON/XOFF som monitorn kan skicka.

##### FIND

FIND använder du för att söka efter filer i programbanken. Du skriver "FIND <filnamn>". Finns filen eller filerna så får du reda på i vilket bibliotek de finns. Du kan inte använda "wildcards" av typen ? och \*.

##### KILL

I systemet finns nu en ägarmarkering för varje fil vilket innebär att du med KILL kan ta bort din egen fil.

##### LIB

Med "LIB,T" får du uppgifter om tid för senaste ändring av filen. "LIB,A" är

#### LOGIN

Loggar du in igen på samma medlemsnummer utan att bryta förbindelsen så får du 2 minuters maxtid.

#### LOGLIST

Utloggningsätt var tidigare den tredje parametern, men är numera en switch. Skriver du "LOGLIST,I" så får du uppgift om alla inloggade men då måste du avbryta med CTRL-C/D för annars så läses loggfilen till början.

#### NEWS

Med "NEWS,L" får du läsa inloggningsfilen igen.

#### Nyheter i MSG:

##### Alla

Du får läsa alla olästa brev och inlägg i en följd. Vill du slippa frågan om tryck return så måste du sätta "sidantal 0". Du använder kommandot om du vill lokalt spara olästa texter.

##### Brev

Brevfunktionen är helt ny. Du får läsa alla olästa brev genom att trycka return när du får ledtexten "(Läsa) nästa brev". Vill du under en körning se om du har fått nya brev så kan du skriva "nästa brev".

Mottagna brev sparas, i regel minst 7 dagar. Du behöver alltså inte själv ta bort brev, eftersom det görs av systemansvariga genom att ett speciellt program körs. Vill du se vilka brev som du har skriver du "återse brev" utan parameter och då får du en lista över breven, bakifrån. Vill du återse ett brev så skriver du kommandot med brevnymret som parameter.

##### Personlig

Du använder kommandot om du vill skicka ett brev till författaren av det inlägg som du just har läst.

##### Radbredd

Radbredd använder du för att ändra radbredd. Det är samma kommando som i monitorn (WIDTH).

##### Vilka

De som kör monitorn kan du se i vilka. Under rubriken möten står det "MONITORN" då.

##### Ärende

Du kan återse ärendet för texter i ett möte med kommandot ärende. Anger du bara ärende så får du en lista bakåt från det senaste inlägget. Anger du ett inläggsnummer får du en lista framåt från angivet inläggsnummer.

# TED texteditor för ABC80

```

+-----+
|      TED      |
| texteditor för ABC80 |
| bruksanvisning |
| September 1985  |
| <5258> Anders Franzen |
| ABC-klubben    |
+-----+

```

## INNEHÅLL

- 1 Beskrivning
- 2 Uppstart
- 3 Meny
  - 3.1 Läsa in filer
  - 3.2 Spara filer
  - 3.3 Editorn
  - 3.4 LIB
  - 3.5 Sluta
- 4 Editorn
  - 4.1 Till meny
  - 4.2 Markörkontroll
  - 4.3 Editering
  - 4.4 Tabulering
  - 4.5 Flytta text
  - 4.6 Klipp och klistra
  - 4.7 Sökning
  - 4.8 Utbyte
  - 4.9 Ordbehandling
  - 4.10 40/80-teckensskärm
  - 4.11 Radlängd
  - 4.12 Tangentklick
  - 4.13 Kontrolltecken
  - 4.14 Hjälpader
- 5 Utrymme

### 1 Beskrivning

TED är en texteditor för ABC80. Det finns två versioner: en för 16K samt en för 32K minne. Det spelar ingen roll vilken checksumma datorn har, TED klarar alla varianter. Du kan ansluta t ex kassettband-spelare eller flexskiveenhet som yttre minne. Om så önskas kan 80-teckentillsats användas (MYAB).

TED besitter en mängd finesser som t ex full markörstyrning, bläddring, sökning, utbyte, tabulering och förflyttning av stycken.

### 2 Uppstart

Då programmet TED startats visas en meny på bildskärmen. Om flexskiveenhet är ansluten består menyn av fem, annars endast av fyra alternativ.

Om programmet bryts (antingen med hjälp av menyn eller CTRL-C) kan det startas igen utan att texten i editorn försvinner. Det sker med kommandot "RUN TED:" i BASIC.

För att ta bort programmet helt ur ABC80 samt göra det möjligt att köra andra program bör RESET-knappen tryckas in. Om inte detta görs finns endast 256 bytes tillgängliga för att skriva program i.

### 3 Meny

Med hjälp av menyn kan man välja ett antal funktioner:

Läsa in en textfil  
Spara innehållet i TED på en fil  
Gå in i själva editorn  
Visa vad som finns på flexskivorna (endast om flexskiveenhet finns)  
Avbryta programmet

Förutom menyn visas även ledigt minnesutrymme för text samt storlek på texten som ev befinner sig i TED.

#### 3.1 Läsa in filer

TED frågar efter namnet på den fil som ska läsas om man från menyn väljer att läsa in en fil. Default extension i filnamnet är "TXT". Om punkt saknas i filnamnet som anges placeras TED alltså "TXT" sist. Om det däremot finns en punkt i filnamnet ändras det ej. Detta medför att ett filnamn som har en extension som utgörs av mellanslag måste anges med en punkt sist, t ex "SLASK."

Om editorn redan innehåller en text frågar TED om den gamla texten ska tas bort eller inte. Om den inte tas bort placeras inläst text efter den gamla. Om man väljer att ta bort den gamla texten rensas editorn innan den nya texten placeras i TED. Ett snabbt sätt att enbart tömma TED på text, utan att läsa in någon fil, är att på frågan om editorn ska tömmas svara ja varefter man endast trycker return på filfrågan (utan att ange något filnamn).

#### 3.2 Spara filer

Då man från menyn valt att spara en fil frågar TED efter namn på utfilen. Nu kan TED erbjuda ett default filnamn inom parentes, detta sker om en fil har lästs in eller sparats tidigare. Genom att endast trycka på retur tangenten sparas filen under detta namn. Om ett annat filnamn är att föredra skrivs detta in varvid filen sparas under det namnet istället. Om ett filnamn erbjuds inom parentes kan det editeras med hjälp av högerpiltangenten.

Även då man sparar filer är extension "TXT" default.

#### 3.3 Editorn

Editorn är naturligtvis ett viktigt alternativ på menyn. Väl där kan man skriva, läsa och ändra text (se kapitel 4).

#### 3.4 LIB

Om en flexskiveenhet är ansluten till datorn erbjuder TED en funktion som visar innehållet på skivorna, en LIB-funktion. Då LIB valts på menyn frågar TED efter önskad drive varvid man svarar genom att trycka på en siffra. När man valt drive visar TED vilka filer som finns på skivan. Om shift tangenten hålles nedtryckt samtidigt som man trycker på siffran visas även storleken på varje fil.

Observera att LIB-funktionen placerats i ett par DOS-buffertar så inget extra minnesutrymme åtgår. Om flexskiveenhet saknas finns ej möjligheten att välja LIB på menyn, dessutom frigörs det minnesutrymme som LIB-programmet upptar så att mer text får plats i TED.

### 3.5 Sluta

Om man väljer att sluta via menyn går man ur TED och hamnar i BASIC. TED finns emellertid kvar i minnet och kan återstartas med "RUN TED:" (se kapitel 2). Om texten i TED har ändrats men ej sparats går det ej att sluta på detta sätt, det kan då bara göras genom att trycka på CTRL-C.

### 4 Editorn

När man valt att gå från menyn in i editorn kan man skriva text. För att kunna kontrollera TED's uppträdande använder man kommandon som man kommer åt med hjälp av CTRL-tangenten. Här följer nu en beskrivning av samtliga kommandon i editorn.

#### 4.1 Till meny

För att återgå till menyn trycker man CTRL-E.

#### 4.2 Markörkontroll

Markören flyttas med vissa tangenter:

|            |                                                               |
|------------|---------------------------------------------------------------|
| CTRL-Q     | till övre vänstra hörnet på sidan (home)                      |
| Return     | till första positionen på nästa rad                           |
| CTRL-W     | uppåt                                                         |
| CTRL-Z     | nedåt                                                         |
| CTRL-A     | vänster                                                       |
| CTRL-S     | höger                                                         |
| vänsterpil | vänster ända till första positionen, sedan upp högerpil höger |
| CTRL-Ö     | första raden i texten                                         |
| CTRL-Å     | sista raden i texten                                          |
| CTRL-Å     | föregående sida                                               |
| CTRL-U     | nästa sida                                                    |

På bildskärmens översta rad visas sidnummer på den sida som syns samt den rad och kolumn där markören befinner sig. Procentsiffran anger hur stor del av maxutrymme texten i TED upptar.

Man kan även erhålla "scroll-mode". Med det menas att istället för att byta sida då en sida är fullskriven scrollas skärmen.

CTRL-P Sätt på / stäng av scroll

Då scrollmode är inställt visas ej sidnr överst på bildskärmen. Det går ej heller att bläddra mellan sidor. Därför får vissa tangenter nya funktioner:

|        |                |
|--------|----------------|
| CTRL-Å | Scrolla bakåt  |
| CTRL-U | Scrolla framåt |

#### 4.3 Editering

Till din hjälp då du ska ändra texten i editorn finns funktioner som nås med vissa tangenter.

|        |                                                                                                                                                                                                                    |
|--------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| CTRL-D | Tar bort det tecken där markören befinner sig samt flyttar resten av raden ett steg åt vänster.                                                                                                                    |
| CTRL-F | Sätter editorn i INS-mode. På översta raden skrivs <INS> ut. Då editorn är i ins-mode skjuts ny text in på den plats där markören befinner sig. Resten av raden flyttas ett steg åt höger. Om editorn redan befin- |

ner sig i INS-mode återställer CTRL-F den till normal mode igen.

**CTRL-E** Tar bort den rad som markören befinner sig på.

**CTRL-R** Skjuter in en ny rad framför den rad som markören befinner sig på.

#### 4.4 Tabulering

Det går fort att skapa tabeller då TED tillhandahåller en lättanvänd tabuleringsfunktion.

**CTRL-T** Tänder/släcker tabuleringsläge i den kolumn som markören befinner sig i. En tänd tab-punkt syns som en fylld ruta på bildskärmens andra rad.

**högerpil** Placerar markör i nästa tabuleringsläge på raden.

#### 4.5 Flytta text

Det går lätt att kopiera en eller flera rader i en text från ett till ett annat ställe. Originalraderna kan sedan tas bort med CTRL-E (se kapitel 4.3). På detta vis kan man lätt flytta stycken i en text.

1 Placera markören på första raden som ska kopieras. Tryck CTRL-O (bokstaven o).

2 Placera markören på sista raden som ska kopieras (kan vara samma som den första). Tryck CTRL-O igen.

3 Placera markören där raderna ska hamna. Då CTRL-O återigen trycks skjuts raderna in framför den rad som markören befinner sig på.

#### 4.6 Klipp och klistra

Ibland vill man kunna dela en rad mitt itu. Detta gör man enkelt i TED. Även motsatsen, klistra ihop text på två rader, kan utföras.

**CTRL-N** Klipper av den rad som markören befinner sig på så att en ny rad bildas. Placera markören på det tecken där klippning ska ske och tryck på CTRL-N. Observera att om RAK-mode är aktiverad justeras den första radhalvan så att rak högermarginal erhålles (se kapitel 4.9).

**CTRL-J** Klistrar ihop text på två rader. Placera markören på önskat tecken i en rad och tryck på CTRL-J. Alla tecken till vänster om markören plus det tecken som markören står på klistras då fast sist på raden ovanför. Om tecknet i kolumn ett inte är ett mellanslag skjuts ett sådant in. Om TED upptäcker att den sammanlagda radlängden kommer att överstiga 160 tecken ignoreras kommandot.

#### 4.7 Sökning

Det går fort att hitta en önskad textsträng i TED.

1 Placera markören där sökning skall börja. Om hela texten skall sökas trycks lämpligen CTRL-Ö varvid markören placerar sig på första raden i editorn.

2 Tryck CTRL-V och svara på frågan om sökt text. Frågan dyker upp på bildskärmens nedersta rad.

3 Om sökt text hittas placerar sig markören där. Ordet <SÖK> syns på bildskärmens översta rad, detta är en markering för att sökning pågår.

4 Så länge ordet <SÖK> syns medför högerpil ett hopp till nästa ställe där sökt text finns. Detta pågår ända till sökt text

ej hittas eller RETURN trycks. Då avbryts sökning och <SÖK> slocknar. Då <SÖK> är tänd fungerar TED precis som normalt med undantag av högerpilstangenten som får en ny funktion enligt punkt 4 ovan. Om markören ändå behöver flyttas åt höger användes lämpligen CTRL-S. Om markören behöver röras nedåt utan att sökningen avbryts tryck CTRL-Z (se kapitel 4.2).

#### 4.8 Utbyte

Man kan lätt byta ut en textsträng mot en annan. De båda strängarna kan vara av olika längd.

1 Placera markören där utbytet skall börja. Om hela texten skall sökas trycks lämpligen CTRL-Ö varvid markören placerar sig på första raden i editorn.

2 Tryck CTRL-B. Ange gammal text på frågan som dyker upp.

3 Ange ny text på nästa fråga.

4 Nu placerar sig markören på första stället där gammal text hittas. Ordet <BYT> syns på skärmens översta rad.

5 Nu finns följande alternativa tangenter att trycka på:

**return** Avbryt utbyte.

**A** Alla gamla texter byts ut mot ny.

**högerpil** Byt ej denna gamla text. Hoppa till nästa ställe där gammal text finns (jämför sökning kapitel 4.7).

**övriga** Denna gamla text byts ut. Hoppa till nästa ställe där gammal text finns.

#### 4.9 Ordbehandling

Det är mycket enkelt att skriva in löpande text i TED. Genom att sätta editorn i ORD-mode kan man skriva in text utan att behöva trycka RETURN före varje ny rad. Detta fixar TED. Om det visar sig att ett ord inte får plats på raden flyttas hela detta ord ned till nästa rad. Raden kapas alltså vid det mellanslag som ligger närmast högerkanten. Det är möjligt att själv välja var högermarginalen ska börja. Defaultvärde för marginalen är sista positionen på raden.

Det är även mycket enkelt att erhålla en rak högerkant i den skrivna texten. Då man sätter editorn i RAK-mode bryts rader av precis som i ORD-mode men med det tillägget att den kapade raden fylls ut med mellanslag mellan lämpliga ord så att den får en fast längd. Högermarginalen kan man givetvis placera själv precis som i ORD-mode.

**CTRL-G** Skiftar editorn mellan normal-läge, ORD-mode och RAK-mode. Som en markering då någon mode är aktiverad visas på bildskärmens översta rad antingen <ORD> eller <RAK>.

**CTRL-Y** Sätter/tar bort egen högermarginal. Placera markören i den position där marginalen ska börja samt tryck på CTRL-Y. Därvid tänds ett "§" på skärmens andra rad (jämför kapitel 4.4). Ytterligare ett tryck på CTRL-Y släcker marginalen. Det har då ingen betydelse var markören befinner sig. Om ingen högermarginal är definierad med CTRL-Y bryter TED texten innan sista positionen på raden. Radlängden kan ändras med CTRL-L (se kapitel 4.11).

#### 4.10 40/80-teckensskärm

TED klarar av både 40- och 80-teckensskärm. Med CTRL-K skiftar man mellan dessa. Om 80-teckentillsats saknas har tangenten ingen funktion.

#### 4.11 Radlängd

Med radlängd menas hur mycket av textraderna i editorn som ska visas på bildskärmen. Radlängden kan anta ett av fyra värden: 40, 80, 120 eller 160 tkn. Den sätts vid uppstart till skärmens bredd (40 eller 80).

Med CTRL-L skiftas radlängden mellan de fyra olika värdena. Om radlängden minskas försvinner ej den bakre halvan av raden. Hela raden (160 tecken) finns alltid i minnet. Om radlängden är längre än skärmbredden visas varje rad på flera skärmlinjer.

Då fil läses in anpassas radlängden efter den längsta inlästa raden. Om en fil som läses in råkar innehålla rader som är längre än 160 tecken så trunckas dessa. TED skriver ej ut något felmeddelande om detta. Observera att rader som är längre än 120 tecken inte kan hanteras av ABC80's BASIC.

#### 4.12 Tangentklick

Om så önskas kan ett klick återges i högtalaren varje gång en tangent nedtrycks. För att sätta på/stänga av klick-mode trycks CTRL-U ned.

#### 4.13 Kontrolltecken

Det kan ibland vara praktiskt att lägga in kontrolltecken i texten för att exempelvis styra utseendet på listan som ska ut på en printer. TED erbjuder möjligheten att skriva tecken vars ASCII-kod är mindre än 32 i texten. Dessa tecken visas på skärmen med ett understrykningstecken (" \_ ") och en bokstav. Exempelvis visas tecknet ASCII 12 (vilket är detsamma som form-feed) som "L".

För att skriva in ett kontrolltecken i texten trycker man på SHIFT-CTRL-O (bokstaven o) varvid TED förväntar sig en bokstav. Om man då skriver ett "L" placeras ASCII 12 på markörens plats.

Observera att det inte går att skriva in tecknen 3,9,10 och 13 då dessa tecken används vid filhantering. Notera också att ASCII 6 och ASCII 23 skrivs ut direkt på skärmen (de visas alltså INTE som "F" resp "W"). Det beror på att de är de tecken som ABC80 använder för "slut grafik" och "start grafik". TED erbjuder sålunda möjligheten att på ett enkelt sätt skapa grafik! Markören försvinner om den placeras på något av grafikstyrtecknen, det får skyl-las på hårdvaran i ABC80.

#### 4.14 Hjälpgrafer

Det kan vara svårt att komma ihåg vad alla CTRL-tangenter betyder. TED erbjuder hjälpgrafer som stöd för minnet. Om CTRL-X trycks ned visas en hjälpgrad längst ned på skärmen. För att se nästa hjälpgrad nedtryckes CTRL-X en gång till. Om bildskärmen visar 80 tecken per rad är hjälp-raderna längre.

#### 5 Utrymme

Det går lätt att se hur mycket av editorns kapacitet som är använd. På meny-n anges ledigt och använt minnesutrymme. Väl inne i editorn visas på översta raden hur stor del av editorns utrymme som är använt uttryckt i procent.

Användaren behöver inte bekymra sig om hur raderna lagras internt i TED. Det kan dock vara värt att veta att flera mellan-slag i en följd packas ihop så att de tar mindre plats.

# Hur fort går det?

Datorn är snabb - mycket snabb om man jämför med manuell hantering av motsvarande manövrer. Med i ett Basic-program begränsas dock hastigheten av att först en Basic-tolk måste undersöka vad som skall göras och översätta till en sträng av ettor och nollor som maskinen kan förstå. Detta tar mycket längre tid än själva maskinspråksrutinen - enligt uppgift 100 till 1000 gånger längre tid.

Basen i ABC-datorerna är erkänt snabb om man jämför med de flesta andra smådatorers. Detta beror på att koden inte sparas i minnet på samma sätt som den ser ut på skärmen (s k list-form eller .BAS-form), utan förvandlas till .BAC-form som är någonting mittemellan listform och maskinkod. Det blir en rad sifferkoder som maskinen mycket snabbare kan läsa än listformen.

Det tar omkring en tusendels sekund (ms) att utföra en instruktion, litet mer eller litet mindre. Om man i ett program låter datorn upprepa en manöver flera hundra eller tusen gånger, spelar det dock en viss roll om man för varje operation kan spara in några bråkdelar av en millisekund. Vissa manövrer är snabbare än andra.

Nedan visas i en tabell en rad operationer som jag undersökte i samband med artikeln om sortering. Det man först lägger märke till är den gamla erfarenheten att heltalsvariabler är mer än dubbelt så snabba som motsvarande flyttalsvariabler. Det beror på att datorn använder 5 byte för varje flyttal, men endast 2 för heltal.

Samma sak gäller all annan hantering av tal. Uttryck som CUR(X,Y) och B\$(I) tar betydligt längre tid (och mera plats i minnet) än CUR(X%,Y%) och B\$(I%). Även om man alltså för utförandets skull inte behöver procenttecknen, bör man ha med dem om de förekommer i en slinga med många varv.

Vidare tar det längre tid att skifta en sträng med många tecken än en liten sträng. Det är litet lustigt med tanke på att det inte är själva strängen man hanterar, utan bara beteckningen av den, men så är det.

Strängar kräver många byte för sin beteckning i datorn, indexerade kräver extra mycket. Men även en enkel operation mellan heltalsvariabler kräver ett oväntat stort antal byte i arbetsminnet. a%=5% representeras i den halvkompilerade ABC-koden av inte mindre än 10 byte!

En räknare som L%=L%+1% med ett GOTO-återhopp tar betydligt längre tid än FOR-NEXT-slingan. Den saken kan man mäta direkt, liksom hur mycket fortare FOR-NEXT går med heltalsvariabel.

Har man bråttom bör man alltså försöka få FOR-NEXT i stället för andra varvräknare, köra med heltal överallt där det går och om möjligt göra jämförelser mellan tal i stället för mellan strängar.

Som framgår av artikeln om sortering kan sådana enkla ändringar spara in 30% av körtiden.

Även REM-satser tar tid. Man bör därför inte ha med dem i slingor som genomlöps ofta.

För undersökningen har jag använt Olle Mauritzons SNABBIS som publicerades i ABC-bladet för mycket länge sedan (1981 någon gång, tror jag). I lätt bearbetad version visas den igen i detta nummer.

Man sätter in den rutin man vill pröva mellan rad 1000 och 2000. Sedan utförs rutinen 1000 gånger och tid tages. Om det tar för lång tid avbryter datorn serien och gör om det bara 100 gånger, eller bara 10 gånger om det är en mycket tidsödande rutin.

Vissa saker kan man inte pröva rakt upp och ner. GOTO kan provas om man hoppar till rad 2000. GOSUB-RETURN kan provas om RETURN ligger utanför provslingan t ex på rad 3000.,

En del andra saker får man bygga in i ett miniprogram, som sedan provas med och utan den del man vill ha tid på.

<1384>

Sven Wickberg

TABELL över exekveringstiden för några rutiner

| Tid ms | ! Basic-rutin               |
|--------|-----------------------------|
| 0,14   | ! GOTO                      |
| 0,22   | ! REM                       |
| 0,40   | ! GOSUB-RETURN              |
| 0,54   | ! L%=L%+1%                  |
| 0,60   | ! L%=1500%                  |
| 0,88   | ! IF L%<K% THEN             |
| 1,36   | ! L=L+1                     |
| 1,80   | ! ;CUR(23%,0%)L%;           |
| 1,12   | ! A\$(1%)=A\$(2%)           |
| 1,46   | ! IF A\$(1%)<a\$(2%) THEN   |
| 1,32   | ! IF A\$(1%)<A\$(2%) falskt |
| 1,26   | ! A\$(K%)=A\$(L%)           |
| 1,62   | ! IF A\$(K%)=A\$(L%) THEN   |
| 1,32   | ! IF A\$(K%)=A\$(L%) falskt |

tiderna är beroende av stränglängd  
= och < går lika fort

2,7- ! fullständigt byte mellan  
-3,3 ! två strängar

3 sek ! FOR I%=1% TO 10000%  
11,5 sek! I%=I%+1%;IF I%<10000% THEN

```

10 REM SAVE SNABBIS,1.0,1980-08-08,
   OLLE MAURITZON
20 REM Anger tiden för att exekvera et
   t programavsnitt.
30 REM Testar 1000 ggr. Om tiden >5 se
   k sker nytest 100 ggr, 10ggr, 1 gån
   g
40 REM Utan inlagda testrader skall ti
   den vara 0+-0.02 sek för 1000 tester
   .
50 REM Om variationen är större ändras
   Ö2 i rad 9.
60 REM Det som testas läggs in mellan
   1000 och 2000
70 ; CHR$(12%)'SNABBHETSTEST' : ; STRI
   NG$(40%,61%)
80 Ö1%=1000% : REM --ANTAL TESTER
90 Ö2=85 : REM --INTERNFART(Obs! Flytt
   al)
100 POKE 65008%,255%,255%
110 ; ' Jag testar'Ö1%' gånger' : ;
120 FOR Ö3%=1% TO Ö1%
130 GOSUB 1000
140 IF PEEK(65009%)<>255% Ö1%=Ö1%/10% :
   Ö2=Ö2/10 : IF Ö1%=0% 180 ELSE 100
150 NEXT Ö3%
160 Ö3%=PEEK(65008%)+Ö2
170 ; CHR$(7)Ö1%' test(er) tog '(255-Ö3
   %)/50' sekunder' : GOTO 190
180 ; ' Testet kunde ej genomföras.'
190 ; CUR(20%,0%)'MELLANSLAG= ny test a
   v samma'
200 ; ' RETURN= ny annan test'
210 ; ' . = åter meny'
220 GET W$ : IF W$=' ' 70 : IF W$=CHR$(
   13) : CUR(11,0) : END
230 IF W$='.' CHAIN 'MENY'
240 GOTO 190
1000 REM STARTAR TESTRUTINEN
2000 RETURN

```

ABC-klubbens kansli  
vill framföra följande:

Du kan nu prenumerera på ABC-diskett  
Du kan nu få programmen på diskett i stället för kassett, i vilket dataformat Du  
önskar.

Gör så här:

Om Du har en flexskiveenhet, t ex DataDisc82, (SS/DD 5 1/4") kallar vi detta  
format för "D". Enligt prislistan ser Du att priset för en sådan diskett är 30 kr.  
Sätt in beloppet på ABC-klubbens postgirokonton 62 93 00 - 5 och ange för "ABC-dis-  
kett", så får Du i fortsättningen diskett i stället för kassett.  
I priset ingår kopiering, liblista och porto, emballage och den kommer direkt hem  
till Din brevlåda. Följande disketter betalas i efterskott. Vi använder singleside 48  
TPI-disketter som är testade och av känt fabrikat.

För fullständig prislista se ABC-klubbens annons på sidan 65.

# Manual till

# KEY.BAC

# för ABC800-serien

```

+-----+
! KEY.BAC, KEYPROG.BAS,
! STÄLLPAR.BAS, KEY.INF
! För ABC800-serien 1985-10-15
! <5357> Kristoffer Eriksson
! ABC-klubben
! Får kopieras fritt för
! icke-kommersiella syften
!
+-----+

```

0. Innehåll
1. Vad ska man med KEY.BAC till?
2. Filer
3. Nya BASIC-instruktioner
4. CTRL-tecken och PF-tangenter
5. Tillfällig avstängning av tangentprogrammeringen (CTRL-F)
6. Tömning av bufferten (CTRL-C)
7. Kedjor av programmerade tangenter
8. Knappknäppningar (tangentsklick)
9. SYS(6)
10. Minnesåtgång
11. Autostart
12. Ändring av parametrar i KEY.BAC
13. Det interna arbetet i KEY.BAC

## 1. Vad ska man med KEY.BAC till?

KEY.BAC är en tangentbordsbuffert med möjlighet till omprogrammering av alla tangenter. Dessutom kan Du med ABC 800 och 802 få klickningar i högtalaren när Du trycker på tangenterna.

Tangentbordsbufferten gör att Du inte behöver sitta och vänta medan datorn laddar ett program eller gör något annat tidskrävande - Det är bara att knappa vidare så läser datorn ifatt så småningom.

Omprogrammering av tangenter innebär att när Du trycker på en tangent får datorn i sig något helt annat än vad som står på tangenten. Det kan vara ett enda tecken eller en hel flod av bokstäver från en enda tangenttryckning.

- Du kan få ett markör-paden på ABC99 att ge ifrån sig ABC77-koder eller tvärtom.

- Du kan programmera in ofta använda kommandon på olika tangenter. Ex: Tryck på CTRL-R så skriver den RUN<RETURN>.
- Om Du ska göra någon sak med många tangenttryckningar flera gånger i rad, så kan Du programmera in dessa alla tryckningar som är lika från gång till gång på en eller två tangenter och bara trycka på dem.
- Du kan lägga upp ett slags JOB-filer på tangenterna. Alla tryckningar som behövs för att starta och köra en hel rad program kan läggas in på en enda tangent, och vill Du spara denna programmering för framtiden går det bra att lägga programmeringskommandot i ett vanligt BASIC-program.

Själv brukar jag programmera en tangent till att spara det program jag arbetar med, och slipper på så sätt vara rädd för att skriva fel programnamn. Namnet står nämligen i början av programmet självt på en rad som ser ut så här: "10 ! save NAMN". Tangenten ger ifrån sig "ED", några framåtpilar, CE, flera framåtpilar och RETURN. Jag brukar också programmera en tangent så att den ger ifrån sig ca 5 framåtpilar, vilket är praktiskt vid programredigering.

## 2. Filer

### KEY.BAC

Detta program lägger upp tangentbufferten och övriga rutiner som behövs i datorns minne. Du kan köra det närhelst Du upptäcker att Du behöver dess finesser. Du kan även få det att köras automatiskt när Du trycker på RESET.

### KEYPROG.BAS

I det skick KEY.BAC levereras startas KEYPROG automatiskt när man kört KEY.BAC. Det programmerar om några tangenter så att de avger några vanliga kommandon. Det känner också av om Du an-

vänder ett tangentbord ABC99, och i så fall ställer den om en hel del av dess tangenter.

### STÄLLPAR.BAS

Det här programmet använder Du om någon av de förinställda parametrarna i KEY.BAC inte är till Din belåtenhet.

## 3. Nya BASIC-instruktioner

För att styra programmets arbete finns två nya BASIC-instruktioner som kan användas både som direktkommandon och i program.

### KEY

Denna instruktion har tre varianter. Beror på hur många parametrar man anger kan man ändra en tangents programmering, helt avprogrammera en tangent, eller visa alla programmeringar.

```
KEY "tangent","nytext"
KEY A$,B$
```

När man anger två strängar programmerar man den tangent som anges i första strängen till att avge den text som står i andra strängen. Om minnesutrymmet för tangentprogrammeringar inte räcker till erhålls felkod 201 (Minnet fullt).

```
Exempel:
KEY "_A","RUN LIB"
```

CTRL-A kommer att avge texten RUN LIB när man trycker på den.

```
KEY "tangent"
KEY A$
```

Men bara en sträng innebär den här instruktionen att den i strängen angivna tangenten avprogrammeras och börjar fungera helt normalt igen.

```
Exempel:
KEY "_A"
```

Föregående exempel omintetgörs.

forts på manualen till

# KEY.BAC

## KEY

Om man bara skriver KEY ut en lista på alla programmerade tangenter. Om listan blir så lång att hela inte rymms på bildskärmen får man mata fram slutet av listan rad för rad genom att trycka på valfri tangent upprepade gånger. Som standard inträffar det här om mer minst 21 tangenter är programmerade.

### Exempel:

```
KEY "A","RUN LIB"
KEY "_O","RUN LIB_M2_M"
KEY
```

```
_A= "RUN LIB"
_O= "RUN LIB_M2_M"
```

De sista två raderna (plus en tomrad) skrevs ut som svar på det sista KEY-befälet.

## KEYMODE

Denna instruktion används för att slå på och av tangentprogrammering och buffertanvändning och även för att radera all programmering.

```
KEYMODE <par1> , <par2>
```

Par1 påverkar tangentbufferten och par2 tangentprogrammeringen.

Par1: 0 = Stäng av bufferten.  
1 = Slå på bufferten igen.  
2 = Töm och stäng av bufferten.  
3 = Töm och slå på bufferten.

Par2: 0 = Stäng av tangentprogrammeringarna.  
1 = Slå på programmeringarna igen.  
2 = Radera och stäng av all programmering. Alla tangenter fungerar normalt igen.  
3 = Radera och slå på programmeringen.

När man startar är tangentprogrammeringen och bufferten på.

### Exempel:

```
KEYMODE 0,0 Stäng av alla finesser
men bevara programmeringarna i minnet.
```

```
KEYMODE ,1 Slå på programmeringarna igen.
```

```
KEYMODE 3 Slå på bufferten också,
men töm den först.
```

Som synes behöver man inte nödvändigtvis ange båda talen, utan det räcker med

det som ska ändras. Ett annat sätt att programmassigt tömma bufferten är med den här loopen:

```
WHILE SYS(5) : GET X$ : WEND
```

## 4. CTRL-tecken och PF-tangenter

När man i KEY-instruktionen ska ange CTRL-kombinationer och PF-tangenter kan man alltid använda CHR\$( om man vet tangentens ASCII-kod.

### Exempel:

```
KEY CHR$(1),"RUN LIB"+CHR$(13)
KEY CHR$(192),CHR$(12)+"LIST"+CHR$(13)
```

CTRL-A (CHR\$(1)) ger ifrån sig RUN LIB följt av RETURN (CHR\$(13)) vilket drar igång LIB-programmen med en enda tangenttryckning. PF1 (CHR\$(192)) blankar bildskärmen och listar aktuellt BASIC-program.

Men det finns ett annat sätt. CTRL-koder kan skrivas som ett understryknings-tecken ( \_ ) följt av motsvarande bokstav och PF-tangenter som ett Et-tecken ( & ) följt av tangentens nummer.

### Exempel:

```
KEY "A","RUN LIB M"
KEY "&1","LIST M"
```

Effekten av detta exempel är precis detsamma som exemplet innan.

Eftersom RETURN är detsamma som CTRL-M kan RETURN skrivas som "M". Framåtpil är CTRL-I ("I") och bakåtpil är CTRL-J ("J").

Det går även att skriva PF-tangenter i kombination med CTRL och SHIFT. CTRL anges givetvis med "\_" och SHIFT anges med "&".

### Exempel:

```
KEY "&1","A"
KEY "&5","B"
KEY "&1","C"
KEY "&5","D"
```

PF1 blir A, SHIFT-PF1 blir B, CTRL-PF1 blir C och CTRL-SHIFT-PF1 blir D.

Ordningen mellan "\_" och "&" spelar ingen roll, däremot måste det hela börja med "&".

OBS: Eftersom det som mest finns PF1 - PF15 blir det vissa problem med att ha ytterligare siffror efter PF1. För att ange PF1 + 1 går det inte att skriva "&11" eftersom det tolkas som PF11. I stället måste man skriva "&011" eftersom maximalt två siffror efter "&" räknas in i PF-numret.

### Exempel:

```
KEY CHR$(129),"&012"
```

På ett ABC99-bord blir STOP-tangenten programmerad till PF1 följt av en tvåa.

Eftersom "\_" och "&" har fått specialbetydelser krävs det lite extra knep för

att skriva just de tecknen. Knepet är att dubblera tecknet.

### Exempel:

```
KEY " ","E"
KEY "&&","F"
KEY " _","G"
```

"\_" blir E och "&" blir F. Den sista raden visar hur man ställer CTRL-SHIFT-O (CHR\$(31)) till G. "\*" betecknar ASCII 127, dvs den fylld rutan.

Efter allt detta blir det ändå tangenter över på tangentbord ABC99. De måste anges med CHR\$. Undrar Du vad en viss tangent har för kod kan Du använda den här snutten:

```
GET X$ : ;ASC(X$)
```

Efter att ha tryckt på RETURN ska Du trycka på den tangent du undrar över så visas genast den tangentens ASCII-kod.

## 5. Tillfällig avstängning av tangentprogrammeringen (CTRL-F)

Om man har exempelvis har programmerat "K" till att bli "L" blir det lite svårt av avprogrammera den igen. Det går ju inte att utan vidare skriva KEY "K" eftersom det genast översätts till LEY "L". Om man däremot trycker på CTRL-F före bokstaven K passerar bokstaven opåverkad. <CTRL-F> KEY " <CTRL-F> K" ger rätt resultat.

Vill man mata in ett CTRL-F får man trycka två gånger på CTRL-F.

I stället för CTRL-F kan man med STÄLLPAR-programmet välja någon annan tangent.

## 6. Tömning av bufferten (CTRL-C)

Förutom att CTRL-C stoppar program-exekvering som vanligt tömmer den även tangentbordsbufferten. Eftersom CTRL-C tömmer bufferten kan inte CTRL-C buffras så som de övriga tecknen, utan har alltid omedelbar effekt. Det går heller inte att programmera någon annan tangent till att ha samma effekt som CTRL-C.

## 7. Kedjor av programmerade tangenter

### Exempel:

```
KEY "A","B"
KEY "B","C"
```

Ovanstående exempel visar en kort kedja av programmerade tangenter. Trycker man på A översätts det till B som i sin tur översätts till C. Effekten av att trycka på A blir alltså ett C.

Denna möjlighet är starkt begränsad. Det är bara sista tecknet i en programmering som översätts på nytt.

### Exempel:

```
KEY "A","BBB"
KEY "B","C"
```

Effekten av att trycka på A blir här BBC eftersom det sista, och BARA det sista, B:et översätts till C.

Översättningen av det sista tecknet kan som vanligt undvikas med CTRL-F (eller det tecken som Du har bestämt med STÄLLPAR).

### Exempel:

```
KEY "A","BB_FB"
KEY "B","C"
```

Trycker man på A så får man BBB precis som om översättningen från B till C inte fanns.

Exempel:  
KEY "A","B"  
KEY "B","A"

Ovanstående exempel betyder att A ska översättas till B som ska översättas till A och till B igen i all oändlighet. En sådan kedja är inte tillåten och ger i praktiken heterna A eller B beroende på omständigheterna. Vill man låta två tangenter byta funktion med varandra ska det inte skrivas som ovanstående exempel, utan med CTRL-F.

Exempel:  
KEY "A","\_FB"  
KEY "B","\_FA"

På det här sättet översätts A till B och B till A på ett korrekt sätt.

### 8. Knappknäppningar (tangentklick)

Om datorn har högtalare (ABC 800, ABC 802) hörs ett litet klick varje gång någon tangent trycks ned. Denna finess kan stängas av med programmet STÄLLPAR.

### 9. SYS(6)

SYS(6) brukar lägga tillbaks senast lästa tecken i tangentbordsbufferten genom att slå på tangentbordsflaggan (KEYFLG). Detta fungerar inte när det finns ytterligare tecken i KEY.BAC's tangentbordsbuffert eftersom det tidigare lästa tecknet (KEYCHR) redan ersatts med nästa tecken ur bufferten.

Så är det även i vanliga fall om man hinner mata in ett nytt tecken innan SYS(6) exekveras, men det brukar inte hända så ofta.

Dessvärre använder LIST-rutinen ett slags SYS(6) när den väntar på någon tangenttryckning efter att ha fyllt hela skärmen med programrader, vilket gör att enstaka buffrade tecken kan försvinna när en listning avbryts.

### 10. Minnesåtgång

KEY.BAC roffar åt sig ca 830 bytes plus utrymme för tangentbufferten från datorns RAM-minne. Programmet lägger sig på lägsta oanvända adress och höjer densamma (Y.BOTM) så att andra program inte kolliderar med det. Programmet är relokerbart, vilket innebär att om det ligger andra rutiner i botten av RAM-minnet samsas KEY.BAC med dem och lägger sig längre upp i minnet.

Det går inte att använda KEY.BAC tillsammans med andra program som är "styga" och vill ha samma minne. Ibland kan det gå om KEY.BAC laddas sist, efter det andra programmet. Det går inte heller tillsammans med program som behöver hela datorns RAM-minne. ORD 800 och ORD III är två exempel på program som inte går.

Tangentbuffertens storlek är från början 180 tecken, vilket bör räcka till nästan vad som helst. Tangentprogrammeringarna lagras också i bufferten (från andra hållet). Storleken kan ändras med ställpar, men måste vara minst 25 tecken.

OBS: När en programmerad tangent buff-

ras tar det bara upp en teckenpoition oavsett hur många tecken tangenten är programmerad till. Översättningen görs tecken för tecken i takt med att de blir lästa.

Det gör inget om man kör KEY.BAC flera gånger utan att göra reset emellan. KEY.BAC upptäcker detta och dubbellagras INTE i minnet.

### 11. Autostart

Det är inte nödvändigt att autostarta KEY.BAC, det går bra att starta "manuellt" när som helst, men det kan vara ganska praktiskt. Tyvärr åstadkommer man detta på något olika sätt med olika DOS. Här följer några riktlinjer.

#### Gamla DOS:

Använd programmet PRESTART eller PREABS för att ändra BASICINL.SYS så att BASIC-programmet KEY.BAC autostartas.

#### UFD-DOS:

Ändra i programmet DRINI så att den rad i början av programmet som börjar med Link\$=... blir Link\$='KEY.BAC'.

#### NET med MENYO:

Autostart av KEY.BAC ej lämplig, eftersom MENYO ska autostartas i stället.

I vanliga fall fortsätter uppstarten efter KEY.BAC med programmet KEYPROG.BAC. Vill man starta ytterligare program efter detta kan man ändra en DATA-sats i början av KEYPROG.BAC så att den anger namnet på önskat program. Vill man inte att KEYPROG.BAC ska startas av KEY.BAC går detta att ändra med programmet STÄLLPAR.

Till sist finns det en möjlighet att alldeles före start av KEY.BAC bestämma vad som ska startas direkt efter KEY.BAC: Om man med STÄLLPAR anger att inget program ska startas efter KEY.BAC kommer KEY.BAC att se efter om det på adress 63747 (dosbuffert 4) i datorns minne finns ett filnamn, och i så fall försöka göra CHAIN till det. Filnamnets slut ska markeras med kod 255. Följande programsnutt lägger in F\$ som filnamn i minnet:

```
FOR I=1 TO LEN(F$) : POKE 63746+I,
ASC(RIGHT$(F$),I) : NEXT I
POKE I,255
```

Denna möjlighet finns till för specialfall och behöver inte utnyttjas av den "genomsnittlige" användaren.

### 12. Inställning av parametrar i KEY.BAC

Det finns ett antal parametrar i KEY.BAC som kan ställas in i förväg med hjälp av programmet STÄLLPAR. Dessa parametrar sparas i själva filen KEY.BAC och ligger kvar till nästa gång de ändras med STÄLLPAR. Om man kör KEY och vill ändra någon parameter måste man köra STÄLLPAR, ange rätt parametrar, trycka på reset och ladda in KEY.BAC på nytt.

Det går bra att ha flera kopior (med olika namn förstås om de ligger på samma skiva) av KEY.BAC med olika parametrar i.

När man kör STÄLLPAR får man börja med att ange namnet på den KEY-kopia som ska behandlas. Filtyp behöver inte anges. De nuvarande parametrarna visas, varefter man för varje parameter får frågan om den ska ändras. Svarar man ja får man tillfälle att mata in ett nytt värde.

Följande parametrar finns:

#### Antal rader i KEY-list före stopp:

Ange hur många tangenterprogrammeringar som ska skrivas ut av instruktionen KEY innan den börjar visa en rad i taget och invänta tangenttryckningar mellan varje. Används för att inte en lång lista ska rulla förbi innan man hinner läsa den. Från början 21. Se KEY.

#### Knappklick?

Ange ja eller nej beroende på om högtalaren ska klicka när man skriver.

#### ASCII-kod för avprogrammeringstecknet:

Ange koden för den tangent som ska förhindra att nästa tangenttryckningsprogrammering ska få effekt. Se CTRL-F.

#### Buffertlängd (minst 25):

Ange hur många tecken tangentbordsbufferten ska rymma. Tänk på att tangentprogrammeringarna också lagras i bufferten. Från början är den 180 tecken. Se Minnesåtgång.

#### Fil som chainas efter exekvering:

Ange namnet på ett BASIC-program som ska startas automatiskt efter KEY.BAC. Detta program brukar ställa in en del tangentprogrammeringar. Max 16 tecken. Från början KEYPROG. Se Autostart.

### 13. Det interna arbetet i KEY.BAC

Den vanliga rutinen för interrupt från tangentbordet ersätts av en rutin som vid behov lagrar inlästa tecken i en buffert. Den gamla rutinen används inte alls.

Rutinen för att läsa tecken från tangentbordet (den man kommer till via CONSI) får en rejäl utbyggnad med rutiner för att hämta tecken från bufferten och tolkning av "programmerade" tangenter. Om det finns tecken i bufferten ser dessa rutiner till att KEYCHR fylls på med nya varje gång den gamla CONSI hämtat ett tecken därifrån. Den gamla rutinen används alltid.

Filosofin med ovanstående är att KEYCHR (65507) och KEYFLG (65506) utifrån sett ska fungera som vanligt. Enda skillnaden blir att CONSI MÅSTE anropas om det ska komma nya tecken i KEYCHR; det räcker inte att passivt ligga och avläsa KEYCHR. Det först inkomna tecknet hamnar direkt i KEYCHR, men övriga läggs i bufferten ända tills CONSI anropas.

De två nya BASIC-instruktionerna länkas in i instruktionslistan. Rutinerna som hör till dessa (Lägg till/Ta bort programmeringar, Lista programmeringar, Översätt CTRL- och PF-tecken) tar ganska stor plats.

Alla dessa rutiner förkläds tillsammans med en relokersrutin till ett ".BAC"-program som kan köras direkt med RUN. Programmet går inte att lista eftersom det (nästan) inte innehåller någon BASIC.

De utskrifter som görs (vid uppstart och instruktionen KEY) sker via filsystemet till enheten "CON:", vilket är en ganska "snäll" metod. Om man gjort något finurligt med den enheten fungerar det även för KEY.BAC.

Ha det så trevligt med KEY önskar

<5357>

Kristoffer Eriksson  
Vassvägen 3  
702 31 ÖREBRO

# Konsten att programmera i BASIC

Är Basic ett bra programmeringsspråk? Nej, säger somliga och hänvisar till att programmen ofta blir mycket grötiga och överskådliga. Det lär t o m ha uttalats av någon expert att den som en gång varit i kontakt med Basic för all framtid är förstörd som programmerare.

Samtidigt har Basic blivit normalspråket för mikrodatorerna. Inga andra program-språk har slagit igenom för hobbyvärlden (om man nu undantar att de ivrigaste fackhackarna med frenesi ägnar sig åt assembler).

Basic betyder Beginners All-purpose Symbolic Instruction Code. (Man ansträngde sig säkert extra för att hitta på ett namn som kunde förkortas BASIC, som i sig själv kan tolkas som "grundläggande".) Meningen med språket var att visa programmeringens ide. Språket skulle vara så lättlärt att en nybörjare på några minuter skulle kunna förstå och göra några mycket enkla program och därmed komma underfund med principerna för datorstyrning.

I den byggde också på att språket skulle vara interpreterande, vilket kanske tarvar en förklaring.

En dator kan inte förstå annat än ettor och nollor, och all programkod måste förr eller senare förvandlas till detta binära sifferspråk. I de flesta språk skriver man programkoden med en vanlig texteditor. När man är färdig låter man en s k kompilator läsa av programmet, tolka texten och åstadkomma en maskinläsbar kod. Detta tar en stund, men maskinkoden kan sedan sparas och arbetar kanske tusen gånger snabbare i datorn.

Nackdelarna uppkommer när man skrivit eller tänkt fel. Det visar sig att datorn inte kan kompilera det man skrivit, eller att programmet inte fungerar. Då måste man i de flesta fall gå tillbaka till texteditorn, leta rätt på felet, skriva om, kompilera på nytt, provköra osv. Det hela kan bli en ganska tidsödande process.

Med Basic är det annorlunda. Basic är ett interpreterande (tolkande) språk. I de flesta Basic-datorer sker ingen bearbetning av programkoden förrän man skriver RUN. Då träder tolken in och kompilerar programmet. Detta går naturligtvis betydligt långsammare än om programmet vore skrivet i maskinspråk, men i gengäld är det mycket lätt att få felet markerade, och det är tekniskt mycket enkelt att göra rättelser, som omedelbart kan kontrolleras.

I ABC-Basic sker någonting mittemellan. När man skriver in programkoden omvandlas den i arbetsminnet omedelbart till en sifferkod som datorn relativt lätt kan avläsa. ABC-Basic är därför vanligen mycket snabbare än Basic brukar vara.

Begär man LIST (antingen på skärm eller på yttre minne) tolkas koden tillbaka till textform. Det är därför apparaten förvandlar små bokstäver till stora i instruktioner, lägger in mellanslag och en del andra typografiska finesser.

Men man kommer inte ifrån att Basic nödvändigt dels tar större plats i minnet och dels går bra mycket långsammare att utföra än program som direkt kompilerats skrivits i maskinspråk.

På senare år har man på flera håll gått in för att göra Basic användbart inte bara som en introduktion, utan även som ett universellt användbart programspråk. Det finns många Basicdialekter och åtskilliga ambitiösa utbyggnader, inte minst den för ABC-hackarna välkända Basic II för 800-serien.

**Är då Basic ett bra eller dåligt programspråk?**

Frågan är svår att ge ett generellt svar på. Nästan allt som gör Basic lättanvänt och smidigt kan utpekas som en svaghet när det gäller att göra programmen överskådliga och korrekta.

I Basic kan man skriva följande lilla program:

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

Nästan varje programrad får Basic-motståndarna att se rött.

Den första raden gör alla konservativa matematiker galna. Det ser ut som en meningslös ekvation.

Vidare kan man använda A utan att vare sig tala om vad A är för något, eller vilket utgångsvärde det skall ha. Det kan man t o m ändra på efter hand som programarbetet fortskrider. Varje VÄN AV ORDNING påpekar med all rätt att detta kan leda till katastrofala följder.

Slutligen har vi radnumreringen. I flertalet andra programspråk används eventuella radnummer bara för att editera program-

texten. I Basic möjliggör de att man kan hoppa omkring i programmet hur som helst med GOTO-satser som hatas alldeles speciellt hjärtligt av Basics kritiker. Och de har alldeles rätt i att GOTO måste användas mycket strikt om man inte skall dra på sig spaghetti-program som det senare är praktiskt taget omöjligt att röra i och uppdatera ens för den som ursprungligen skrev dem.

Denna kritik har visst fog för sig. Den har att göra de uppoffringar man måste göra för att få Basic så enkelt och lättlärt som meningen var.

I t ex Pascal inte kan skriva en enda liten programsats utan att först ägna ett ordentligt arbete åt att definiera sina variabler - tala om vad de heter, vilken form de skall ha, utgångsvärde m m. Programarbetet måste tänkas igenom ordentligt i förväg och delas in i ordentligt avgränsade procedurer som sedan snyggt och prydligt kan kallas in. Ovillkorliga hopp är praktiskt taget omöjliga. Språket tvingar fram en struktur som visserligen är bra för ett stort systemarbete, men som av en ovan hobbyist kan upplevas som en besvärande tvångströja, svårlärd och tungrodd. Den som inte håller på med det varje dag glömmar bort hur man kommer i gång och kanske lätt tappat lusten att programmera alls.

**Låt oss undersöka kritiken litet närmare.**

Visst är det sant att Basic tillåter mycket dåligt skrivna program. Vi har säkert alla i vår tidigaste programmeringstid skrivit förfärliga program, som vi numera - flera månader efteråt - inte ens kan tyda, långt mindre förbättra. Och det finns gott om skräckexempel i litteraturen. Jag har haft anledning att flera gånger klaga över programkoden i BASIC COMPUTER GAMES, som tyvärr kommit att tjäna som förlagor åt många noviser.

Men nackdelarna hos Basic kan ju motverkas, både genom hårdvarutricks och genom målmedvetenhet och ökad kunnsighet hos programmeraren.

Om man inte gillar A=A+1 kan man ju göra som det ursprungligen tänktes, skriva LET A=A+1. Då ser man att det inte är fråga om en matematisk ekvation, utan

om en tilldelningssats. I COMAL använder man (precis som i ALGOL) tecknen := för att markera detta. Men varför krångla till det?

Målet är att skriva ett väl överskådligt och strukturerat program i Basic. Hur gör man?

Det bästa är utan tvivel att göra precis som Pascal-programmeraren. Börja med en plan över programmet. Definiera de olika delarna, bestäm vad variablerna skall heta osv. När det är gjort finns ju programstrukturen alldeles oavsett vilket programspråk man använder.

Men nu antar vi för exemplet skall att vår programskrivare (dvs du och jag) inte vill/kan/orkar göra så. Vi vill skriva DIREKT, när inspirationen kommer och medan det är roligt.

Följande tips kan då kanske vara till hjälp.

#### REM-satser

Börja med att skriva en REM-sats med programnamnet på rad 10 och reservera raderna 10-99 för kommentarer om vad programmet gör, sammanhörande program osv.

Strö sedan ut rikligt med förklarande REM-satser överallt där det händer någonting, inte bara i början av varje subrutin.

Bara detta gör arbetet vid en senare uppdatering eller läsning av programmet mycket behagligare.

#### TOP-DOWN

Fortsätt sedan, förslagsvis på rad 100 med att skriva programmet i stora drag genom att ange vad det är som skall göras.

```
100 GOSUB 1000 : REM definitioner och
                    initieringar
150 GOSUB 2000 : REM rita spelplan
200 GOSUB 3000 : REM öppna printer
250 GOSUB 4000 : REM spelarens drag
300 GOSUB 5000 : REM datorns drag
350 GOSUB 6000 : REM spelet slut?
400 END
```

Låt sedan varje subrutin vara en avslutad enhet. Dels blir det överskådligt, dels kan man plocka ut just den delen och använda igen i andra program.

#### Hoppa inte ut genom fönstret!

Gå alltid in i varje programdel från början och gå alltid ut i slutet! Inga godtyckliga hopp hit och dit, som det är omöjligt att följa när man läser programmet.

Om man blir färdig med en subrutin mitt i, hoppa då till RETURN, som skall stå sist! Om man som i detta exempel skall ha flera drag dator-spelare, hoppa då inte från den ena subrutinen till den andra, utan hoppa efter varje dragfölj tillbaka till huvudprogrammets rad 250.

#### Sätt upp synliga staket

Omge varje subrutin med någon form av "staket".

```
2000 REM -----
2010 REM öppna printer
2020 ...
...
2150 RETURN
2190 REM slut printer
2200 REM -----
```

Det är sant att alla dessa REM-satser tar stor plats, men du kommer säkert att bli glad för dem många gånger om. Skulle det visa sig att de tar för stor plats, kan man ju ta bort dem i det program man använder. Men spara originalprogrammet inklusive alla förklaringarna. Bara med dessa enkla regler får man en hygglig ordning i sina program och behöver inte lida av att Basic inte tvingar fram en bra programstruktur.

#### Med hjälp av en ABC800

Har man en ABC800 finns ytterligare hjälpmedel. I stället för de litet formlösa GOSUB-hoppen kan man använda funktioner. Om detta har Bo Kullmar skrivit utförligt i ABC-bladet nummer 4, 1985.

Användardefinierade funktioner har fördelen att kunna anropas från olika ställen i programmet och att ha namn som i sig själva talar om vad de gör. Programexemplet ovan kunde vara

```
100 Z=FNInit      ! Initieringar
200 Z=FNPrinter   ! Öppna printer
300 Z=FNSpelare   ! spelarens drag
400 Z=FNDator     ! datorns drag
500 IF NOT FNEnd THEN 300
600 END
```

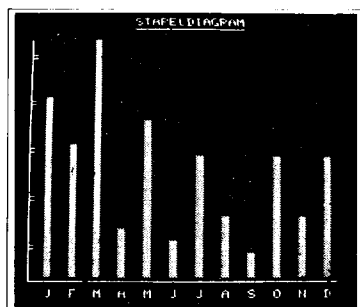
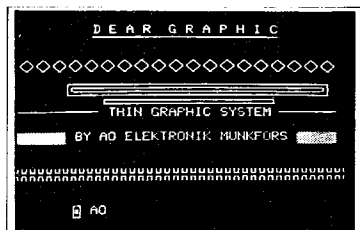
<1384>

Sven Wickberg

## ★ ★ ★ ★ ★ NYHET ★ ★ ★ ★ ★

### DEAR thin GRAPHIC:

feb -86



exempel 1:

"DEAR thin Graphic" är en enkel tillsats för ABC80, med vilken en bättre grafisk presentation kan ske på bildskärm.

Med en enkel modifikation i ABC80 och montering av denna tillsats, erhålles en helt ny teckengenerator. Denna innehåller s.k. "Thin graphic lines" vilket nu gör det möjligt att få extra smala och snygga linjer. Med hjälp av denna kan ett otal kombinationer skapas.

Styrningen sker med ett enkelt OUT 7,1 (för grafik) och OUT 7,0 (för normal). All bildinformation som finns på skärmen växlas vid dessa kommandon, mellan den normala eller nya teckengeneratoren. ABC80:s standard semi-grafik påverkas givetvis dock inte.

**PRIS 296:-** exkl. moms (365:- inkl. moms).

(Reservation för ändringar).

|                 |              |             |
|-----------------|--------------|-------------|
| POSTADRESS      | BESÖKSADRESS | TELEFON     |
| KBAO.ELEKTRONIK | Gamla Bruks- | 0563/509 38 |
| Box 115         | Kontoret     |             |
| 684 01 MUNKFORS |              |             |



### SKIVMINNEN FRÅN A.O.ELEKTRONIK - TILL MYCKET BRA PRISER!

| Fabrikat     | Typ av drive       | Format    | Rek.pris | A.O.pris |
|--------------|--------------------|-----------|----------|----------|
| BASF 6162    | 3,5" microfloppy   | 2x40 spår | 1.775:-  | 1.365:-  |
| BASF 6164    | 3,5" microfloppy   | 2x80 spår | 1.898:-  | 1.460:-  |
| BASF 6128    | 5,25" floppy 0,5Mb | 2x40 spår | 1.859:-  | 1.430:-  |
| BASF 6138    | 5,25" floppy 1,0Mb | 2x80 spår | 2.060:-  | 1.585:-  |
| BASF 6129C   | 5,25" floppy PC    | 2x40 spår | 1.775:-  | 1.295:-  |
| BASF 6139C   | 5,25" floppy PC    | 2x80 spår | 1.976:-  | 1.520:-  |
| BASF 6188    | 5,25" Winchester   | 15 Mbyte  | 7.358:-  | 4.250:-  |
| BASF 6188-R3 | 5,25" Winchester   | 25 Mbyte  | 8.443:-  | 6.495:-  |
| BASF 6185    | 5,25" Winchester   | 27 Mbyte  | 10.237:- | 7.875:-  |
| BASF 6194    | 5,25" Winchester   | 96 Mbyte  | 27.085:- | 20.835:- |
| NEC D3126    | 3,5" Winchester    | 25 Mbyte  | 8.599:-  | 6.615:-  |
| NEC D5128    | 3,5" Winchester    | 25 Mbyte  | 7.527:-  | 5.790:-  |

#### Controllers:

|                |                   |           |         |         |
|----------------|-------------------|-----------|---------|---------|
| Xebec S1410A   | SASI-Winchester   | 1/2 drive | 2.808:- | 2.160:- |
| Xebec S1420    | SASI-Winch/Floppy | 2+2 drive | 3.809:- | 2.930:- |
| Xebec S1210A/C | IBM-Winchester    | 1/2 drive | 2.444:- | 1.880:- |
| Xebec S1220    | IBM-Winch/Floppy  | 2+2 drive | 3.224:- | 2.480:- |

Alla enheter i "slim-line"-utförande med 45 mm höjd, utom:

- BASF-typ: 6185, 6194 & NEC D3126 med 85 mm höjd.
- BASF-typ: 6128/6138 med 33,5 mm höjd.
- Datablad på begäran
- Tekniska Manualer 40:-

#### IBM-SPECIAL! 15 Mbyte Winchesterpaket innehållande:

- BASF 6188, XEBEC-S1210A/C, Kablage, Mjukvara

★ ★ ★ ★ ★ PAKETPRIS: 5.995:- ★ ★ ★ ★ ★

- Kompletta floppysystem (inkl. mjukvara) till ABC80 från 4.795:-
- Professionella kraftfulla CP/M-datorer i utförande från 9.990:-
- DYSAN högkvalitativa disketter till ALLA datorsystem, från 19:-
- Bra Matris skrivare till ABC80 och IBM-maskiner från 1.990:-

**Leveranstid:** Efter beställning ca 1 vecka.

**Leveransvillkor:** Normalt mot postförsäkring. Företag o liknande 15 dagar netto efter avtal.

**Priser:** Alla priser är exkl. moms och frakt

**Garanti:** 6 månader på alla fabriksnya enheter.

(Reservation för oförutsedda prisändringar)

|                      |                 |
|----------------------|-----------------|
| <b>Besöksadress:</b> | <b>Telefon:</b> |
| A.O. Elektronik      |                 |
| Gamla Brukskontoret  | 0563-509 38     |
| MUNKFORS             |                 |

Box 115  
684 01 MUNKFORS

# Kommentarer till ett bokstavspel

De flesta har någon gång sysslat med 15-spelet: En liten låda med 15 numerade brickor och en tom plats för den sextonde. Med hjälp av den tomma platsen skall man flytta om brickorna så att de hamnar i nummerordning.

Här kommer en datorvariant som är MYCKET svårare. Lådan innehåller nu 16 brickor med var sin bokstav. Det gäller fortfarande att ordna dem, från A till P, men eftersom det inte finns någon tom plats, flyttar man genom att rotera fyra intilliggande brickor medurs.

Platserna är numrerade så här:

```
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
```

Ett drag består av ett angivet nummer och innebär att den fyrarutorskvadrat som har dragnumret längst upp till vänster, skiftar medurs.

Anta att fyra brickor står så här:

```
B F
A E
```

Om man väljer draget med platsnumret för B, kommer gruppen att roteras till:

```
A B
E F
```

Det är inte så lätt att ordna till kvadraten. Eftersom tecknen bara kan flyttas i grupper om fyra, och alltid bara medurs, måste man verkligen tänka sig för och planera framåt.

Efter en del övning har jag kommit fram till att man skall kunna fixa en kvadrat på under 40 drag - 33 är mitt rekord. Men räkna med att det första gången tar mycket fler drag.

Programmet börjar med att slumpa ut bokstäverna i rutorna. Detta sker med en liten finess (raderna 222-2222). Först dras ett av de 16 tecknen och får byta plats med tecknet i ruta 16. Därigenom kommer de återstående 15 tecknen att finnas i rutorna 1-15, och det sker en ny dragning. Det dragna tecknet byter plats med tecknet i ruta 15, och så man 14 kvar att välja på. Så fortsätter det tills det bara finns 2 tecken kvar och där kan manövern stanna.

I övrigt är programmet avsett att vara självförklarande, men på begäran kommer här en genomgång av rutinerna.

Raderna 580-800 innehåller instruktionerna för spelet och kan utelämnas eller sparas till sist. Då måste man också hoppa över rad 90 och rad 250.

Rad 60 dimensionerar de indexerade variabler vi behöver och rad 70 innehåller ett tidigare beskrivet sätt att definiera

ett JA-svar med J, j eller RETURN. Om någon av dessa tryckes är FNJ% sann, annars falsk.

B\$(1%) innehåller det tecken som står på ruta nr 1%. På rad 120 definieras bokstäverna A-P (fr o m CHR\$(65%)) och placeras till att börja med i ordning 1-16.

Rad 140-160 blandar om dem slumpvis. RND ger ett slumpstal mellan 0 och 1. Multiplikation med 1% (som är högst 16) ger tal mellan 0 och 15,99999 och när man lägger till ett blir intervallet i stället 1 till 16,99999.

Nu vill vi bli av med decimalerna. Det normala är att skriva INT(), men eftersom resultatet i alla fall skall sparas i en heltalsvariabel X% (för att spara plats och få programmet att få fortare) hoppar jag över INT. Det kommer i alla fall att ske en trunkering (avhuggning) av decimalerna. Denna förenkling spelar ABC800-ägarna ett spratt - se nedan!

I första dragningen är alla 16 rutorna med. Den ruta som dras får sin bokstav placerad i ruta 1%, dvs i sista lediga ruta. Man börjar med att placera innehållet i ruta 1% i slaskvariabeln T\$. Sedan går X-te rutan till I och slutligen flyttas T\$ över till X.

I nästa omgång sker dragning mellan rutorna 1-15 och den dragna rutans bokstav hamnar i ruta 15 osv tills det bara är 2 kvar. Sista rutan behöver man naturligtvis inte dra!

Nu ritas brädet med subrutinen 480-520. Samtidigt uppdateras ett antal räknare: G% håller reda på antalet klarade spel, G2% antalet uppgivna. M% är antalet drag i pågående spel, M1% totala antalet drag i klarade spel och M2% totala antalet drag i uppgivna spel. Efter varje drag genomlöps denna rutin, som rensar skärmen och ritar om ställningen.

Radadd 220 frågar vilken ruta som skall vara bas för rotationen. Samtidigt ges information om vilka alternativa drag som finns. På rad 230 tas inmatningen emot i strängen W\$. Rad 250 undersöker om man dragit RETURN och visar i så fall instruktionerna igen. (Den raden måste bort om det inte finns några instruktioner.) Rad 260 undersöker om man givit upp och om så är fallet uppdateras räknarna och man går till slutsekvensen 390.

Nu gäller det att kontrollera att det föreslagna draget är tillåtet och det sker i 270-280. En ONERRORGOTO-sats ger hopp till rad 220 om det blir fel. Först undersöks om det är ett numeriskt värde som slagits in, sedan om det ligger inom de tillåtna intervallen.

Det där med numeriskt värde fordrar kanske en förklaring. Man skulle klara inmatningen med INPUT X%,Y% och förvänta sig två tal inmatade från tangentbordet. Men om det som matas in INTE är numeriskt (siffror) blir det felmarkering och program-

avbrott. Detta vill vi undvika med den visde metoden. INPUT tar emot två strängar X\$ och Y\$, och de kan se ut i princip hur som helst utan att datorn stannar. Strängarna kan förvandlas till tal med VAL(X\$) osv om de är numeriska, dvs består endast av siffror. Misslyckas detta träder felhanteringen in och skickar tillbaka en till rad 550. (I princip samma metod används på rad 270.) Om allt går bra sker uppdatering av dragräknaren och minnesmarkering av rotationsrutan (290). Sedan sker uppdateringen av de roterade rutorna med samma metod som vid omblandningen i rad 160.

Slutligen ritas det nya brädet (310) och det sker kontroll om alla bokstäverna ligger i ordning. Om så inte är fallet är spelet ännu inte slut och återhopp sker till rad 220.

Just denna rad (340) är skum på ett mycket dolt sätt. När FOR-NEXT-rutinen inleds sparas det aktuella värdet av 1% någonstans i interminnet och uppdateras för varje varv. När det föreskrivna antalet varv genomlöpts avslutas rutinen.

Genom att vi nu hoppar ur rutinen innan den gjort alla varv, avslutas den inte på föreskrivet sätt. Man kan räkna med att någonstans i minnet finns en adress som håller reda på långt varvräknaren hunnit gå. Den kommer inte att användas mera, men den tar fortfarande upp plats.

Har vi tur (och jag tror vi har det) kommer datorn att använda samma minnesadress för nästa FOR-NEXT-slinga, och då gör det ingenting. Men för vissa slag av slingor reserveras i stället en ytterligare minnesadress om den gamla slingan inte är avslutad. Det KAN alltså betyda att man efter en tids spel får ERR 3 (=minnet fullt). Risken här är ytterst minimal. Även om vi inte har den tur jag räknar med är programmet så litet (och spelarens tålamod så kort) att det knappast är troligt att minnet tar slut.

(Man skall aldrig säga aldrig. Skriv följande i början av ROTERA-programmet:

```
1 GOSUB 3      Hoppa till subrutin rad 3
2 N%=N%+1%    Räknare
3 GOTO 1       Avbryt subrutinen före RETURN
4 RETURN      Hit kommer vi aldrig - återhoppadressen sparas
```

och när ERR 3 kommer skriv ;N% så får du se hur många gånger vi kan bryta slingan innan minnet är fullt!) Om alla bokstäver ligger rätt är omgången slut och programmet går vidare till 370. OUT 6,5 är en busvissling som får pågå ungefär en halv sekund.

Lägg märke till att det är flyttat på räknaren denna gång. FOR-NEXT-slingan är här en vänteslinga. Den gör inget annat än räknar upp sig, och den räknar till 1000 på ungefär en sekund. OUT 6,0 stänger av ljudet.

Nu sker utskrift av resultat och litet

statistik. Om man givit upp något spel utan att först göra något drag, kommer någon dragräknare att vara noll och man kan råka ut för ett pogramfel som inte ONERROR kan fixa, näligen division med noll. 390 och 420 skall förhindra det.

Lägg märke till att instruktionen RANDOMIZE inte förekommer. Det betyder att det alltid är samma serie av spel som dyker upp på skärmen. Man kan alltså tävla med sig själv eller med andra om vem som snabbast ordnar upp ett spel.

Vill man ha nya spel kan man antingen välja att "ge upp" ett antal spel i början, eller också kan man naturligtvis skriva in RANDOMIZE på t ex rad 15.

När man blandar om bokstäverna händer det att man får en blandning som inte kan ordnas genom bara rotationer. Hur man än roterar kommer till slut två tecken att ligga i fel ordning. Med kommandot -1 får man tillstånd att byta plats mellan två bredvidliggande rutor i samma rad.

Bland ser det ut som om TVÅ teckenpar ligger fel. Man skulle behöva göra -1 två gånger, men det tillåter inte programmet. Det problemet får man klura ut på något annat sätt - hur det går till skall inte avslöjas här. Något roligt får vi som vanligt lämna till läsaren.

Ha så kul!

#### Tillägg för ABC800-ägare

Det här spelet går (nästan) att köra på 800 också, men man måste göra ett par små justeringar.

Raderna 340 och 390: BASIC II-tolken tål inte kolon och REM i en sats som börjar med IF. Skriv i stället (inget kolon) utropstecken före anmärkningen eller stryk den helt.

ad R370: inte alla 800-varianter levererar ljud via OUT 6,<udda tal>. Om din apparat krånglar, ta bort raden.

Rad 460: END måste stå ensamt på en rad. Antingen kan man byta END mot STOP eller också tar man bort :END och skriver till rad 465 END.

Nu ger programmet inga frågetecken vid inläggning, men datorn vägrar i vändningen vid rd 550. Det beror på ONERRORGOTO 0. I BASIC II heter det ON ERROR GOTO (ingen nolla och två mellanslag). Vissa 800-datorer ordnar till mellanslagen, men nollan måste man ta bort själv.

Rad 550 väntar på inmatning av TVÅ siffervärden. Om man av glömska eller obetänksamhet bara ger ETT värde, kommer ABC80 att skriva nytt frågetecken och snabbt vänta på nästa. Det sker inte i BASIC II, som ger programavbrott och felmarkering.

ON ERROR-funktionen bör därför komma före denna rad för att fånga upp även detta fel och jag föreslår:

```
545 ON ERROR GOTO 545
550 <samma men utan ON ERROR-
satserna>
555 ON ERROR GOTO
```

För att få snyggare skärmskrift vid nyssnämnda fel kunde man börja rad 550 med en extra radmatning:

```
550 ;;;'VILKA etc ...'
```

Slutligen har vi ett mycket lurigt fel på rad 150. Jag har hoppt över INT( ) eftersom X% är heltal och automatiskt tar bort decimalerna. Men medan ABC80 trunkerar (dvs skär bortdecimalerna utan avrundning) sker i 800 en korrekt avrundning. Man kan alltså råka ut för att X% blir 17, och då försvinner ett tecken ur kvadraten.

Den enklaste lösningen är att lägga till INT() som även i BASIC II trunkerar decimalerna:

```
150 X%=INT(RND*I%)+1%
```

Nu skall det fungera! Men osvuret är bäst. Är det någon som hittar fler lustiga (eller mindre lustiga) felkällor vill jag gärna höra om det.

PS. Jo, en sak till - kanske det är någon som undrar varför det står "list ROTERA" i rubrikdelen till programmet?

Det kom av ett tips i något stenålders-nummer av ABC-bladet.

När man jobbar med ett program måste man då och då spara undan det man gjort. Man skriver SAVE <program> eller i det här fallet LIST <program>. Då är det lätt hänt att man av trötthet, utmattning eller bara rent slarv skriver t ex LIST ROTRA eller ROSTERA eller vad som helst i stället för det riktiga ROTERA.

Det programmet försvinner tills vidare spärlost och kan inte återfås hur många gånger man än skriver RUN ROTERA... Någon gav tipset: skriv SAVE <programnamn> på rad 10. När du skall spara undan: gör ED10 och redigera bort radnummer och REM. Sedan kan det bara inte bli fel!

Bara ifall ni undrar, alltså.

<1384>

Sven Wickberg

```
10 REM ...list ROTERA
20 REM id' från ROTATE ur MORE BASIC CO
MP. GAMES
30 REM nyskriven för ABC80 81 06 13 SW
40 REM bearb. Sven Wickberg 85 07 30
50 REM -----
60 DIM B$(16%),B$(16%)
70 DEF FNJ%=W$='J' OR W$='j' OR W$='' :
REM FNJ% sant för JA
80 T%=10% : REM ---vänstermarginal för
brådet
90 ; CHR$(12%)'INSTRUKTIONER (J)'; : IN
PUT W$ : IF FNJ% GOSUB 580
100 REM -----
110 REM göra i ordning brådet
120 FOR I%=1% TO 16% : B$(I%)=CHR$(64%+I
%): NEXT I% : REM fylla rutorna
130 REM blanda om bokstäverna
140 FOR I%=16% TO 2% STEP -1%
150 X%=RND*I%+1% : REM dra ledig ruta
160 T$=B$(I%) : B$(I%)=B$(X%) : B$(X%)=T
$ : REM lägg dragna bokstaven sist
170 NEXT I%
180 REM -----
190 REM spelet börjar; M=drag, S=1 om sk
ift utfört
200 M%=0% : S%=0%
210 GOSUB 490 : REM rita brådet
215 REM -----
220 ; 'Vilken ruta ska roteras?' : ; '(a
lt. 0 =uppgivet -1 =skifta två ruto
r)';
230 ; '(RETURN =instruktionerna)'; : ;
; 'Ditt val'; : INPUT W$
250 IF W$=' ' GOSUB 580 : ; CHR$(12%) : G
OTO 210
260 IF W$='0' THEN ; ; 'OTUR!'M%' för1
orade drag...' : ; G2%=G2%+1% : M2
%=M2%+M% : GOTO 390
270 ONERRORGOTO 220 : W%=VAL(W$) : IF W%
=-1% THEN 540
280 IF W%<1% OR W%=4% OR W%=8% OR W%>11%
THEN ; 'OTILLÄT DRAG!' : GOTO 220
290 M%=M%+1% : T$=B$(W%)
300 B$(W%)=B$(W%+4%) : B$(W%+4%)=B$(W%+5
%) : B$(W%+5%)=B$(W%+1%) : B$(W%+1%)
=T$
310 GOSUB 490 : REM rita brådet
```

```
315 REM -----
320 REM kolla om bokstäverna rätt=slut
330 FOR I%=1% TO 16%
340 IF CHR$(I%+64%)<>B$(I%) THEN 220 : R
EM ---ej slut
350 NEXT I%
360 REM --- alla bokstäver rätt, spelet
slut
370 OUT 6%,5% : FOR I=1 TO 500 : NEXT I
: OUT 6%,0%
380 ; ; 'Du fixade brådet med'M%' drag
' : M1%=M1%+M% : G%=G%+1% : REM ---M
1=totalant.drag;G=antal spel
390 IF G%=0% THEN 420 : REM hindra divis
ion med noll
400 ; ; 'Du har klarat'G%' spel'
410 ; 'med i genomsnitt'M1%/G%' drag per
spel'
420 IF G2%=0% THEN 450
430 ; '- och givit upp'G2%' spel efter'
440 ; 'i genomsnitt'M2%/G2%' drag.'
450 ; ; ; 'Ett spel till (J)'; : INP
UT W$ : IF FNJ% THEN 120
460 ; ; 'SLUT' : END
470 REM -----
480 REM rita brådet
490 ; CHR$(12%)'(Spel'G%+G2%+1%) STÄLLN
INGEN EFTER DRAG'M%': ; ;
500 ; ; FOR I%=1% TO 13% STEP 4%
510 ; TAB(T%)B$(I%) 'B$(I%+1%) 'B$(I%+
2%) 'B$(I%+3%)
520 NEXT I% : ; ; : RETURN
525 REM -----
530 REM ---SKIFTA TVÅ RUTOR OM S=0
540 S%=S%+1% : IF S%>1% THEN ; 'DU FÅR B
ARA GÖRA ETT SKIFTE PER SPEL!' : GOT
O 220
550 ; 'VILKA TVÅ RUTOR SKALL SKIFTAS'; :
INPUT X$,Y$ : ONERRORGOTO 550 : X%=V
AL(X%) : Y%=VAL(Y%) : ONERRORGOTO 0
560 IF X%<>Y%+1% AND X%<>Y%-1% : CHR$(7%
)'MASTE STÅ PÅ SAMMA RAD' : GOTO 550
570 T$=B$(X%) : B$(X%)=B$(Y%) : B$(Y%)=T
$ : GOTO 310
575 REM -----
580 REM ---INSTRUKTIONER
590 ; CHR$(12%) : ; 'Rutorna har följand
e nummer:' : ;
600 FOR I%=1% TO 16% : B$(I%)=I% : NEXT
I%
610 ; ; FOR I%=1% TO 13% STEP 4%
620 FOR J%=0% TO 3%
630 ; TAB(7%+4*J%)B$(I%+J%);
640 NEXT J% : ; : NEXT I%
650 ; ; 'Rutorna fylls slumpvis med bo
kstäverna'
660 ; 'A-P. Du ska ordna dem genom att
rotera'
670 ; 'fyra rutor medurs med utgångspunk
t från'
680 ; 'den övre vänstra rutan i gruppen.
' : ;
690 ; 'Giltiga drag är alltså 1, 2, 3; 5
, 6, 7;samt 9, 10 och 11.' : ; CUR(2
3%,0%)'Tryck RETURN'; : GET W$
700 ; CHR$(12%)'Om t.ex. fyra rutor ser'
TAB(30%)'B F'
710 ; 'ut så här:'TAB(30%)'A E' : ;
720 ; 'kan man genom att trycka'; : 'in
numret för ruta B'TAB(30%)'A B'
730 ; 'rotera dem så här:'TAB(30%)'E F'
740 ; ; ; 'EN GANG under spelet kan
man låta' : ; 'två bokstäver BREDVID
VARANDRA på'
750 ; 'en rad byta plats. Skriv då in -1
'
760 ; 'och därefter rutornas nummer'
770 ; 'med kommatecken emellan.' : ;
780 ; 'Vill du ge upp, skriv in 0 (en no
lla).' : ;
790 ; CUR(20%,0%)'Vill du se intruktione
rna igen skriv I' : ; 'annars tryck
RETURN'; : GET W$
800 IF W$='I' OR W$='i' THEN 580 ELSE RE
TURN
```

# ABC-klubbens framtid?

I samband med samgåendet mellan Luxor Datorer och Nokia Information System i Sverige har det förts en diskussion i MSG-systemet om klubbens framtid.

För att de som inte har tillfälle att följa diskussionen skall få del av den, så försöker jag här ge personlig redogörelse av den.

Som jag tolkar diskussionen så finns det två grupper:

1 ABC-klubben bör öppna sig mot icke ABC80/80X och därmed försäkra att klubben överlever i framtiden. Detta på grund av att dagens ABC-maskiner kan komma att spela en allt mindre roll på marknaden i framtiden.

2 ABC-klubben skall främst ägna sig åt ABC80 och ABC800-serien. När underlaget för dessa maskiner minskar så skall också klubben minska och till slut upphöra. Intrasserade av andra datorer får utöva denna intresse i andra föreningar.

## Medlemmarna bestämmer!

Enligt min uppfattning är det du som medlem som bestämmer vilken verksamhet som klubben skall ha. Visst ger stadgarna en viss vägledning, men dessa kan vi ändra på ett årsmöte om vi vill.

En inriktning på nya maskiner kan aldrig ske om inte någon eller några medlemmar aktivt verkar för detta. Jag känner väl till detta eftersom jag var med om att föra in 800-serien i klubbens verksamhet.

För mig så förefaller det naturligt att vi försöker täcka det produktutbud som Nokia-Luxor kommer att ha. Vi har medlemmar som började med ABC80 och sedan bytte upp sig till ABC800-serien. En del av dessa kommer troligen att byta upp sig till en kommande Nokia-Luxor PC. De kommer därvid att se det som naturligt att fortsätta som medlemmar i klubben. Annorlunda blir det ofta om för medlemmar som köper IBM.

ABC-klubbens verksamhet inriktar sig enligt stadgarna på användare av ABC-Datorer. Detta innebär i praktiken att det är Nokia-Luxor som indirekt definierar verksamheten. Nokia-Luxors vd Kjell Strömli har på årsmötet bekräftat att Nokia-Luxor kommer att behålla ABC i namnet på framtida produkter.

En annan möjlighet kan vara att öppna klubben för användare av BASIC II på IBM PC och IBM PC kompatibla maskiner. Vi har ju stor gemenskap med dessa och vi har dessutom en hel del BASCI II program i programbanken.

Unixgenerationen ABC1600 och ABC9000 har ännu inte påverkat klubbens verksamhet eftersom det är få medlemmar som har tillgång till dessa relativt dyra system.

Kom med synpunkter på klubbens framtida verksamhet! Du kan t ex skriva artiklar till ABC-bladet och berätta hur du ser på framtiden för klubben. Eller så kan du genom moderm delta i debatten i MSG.

<1789>

Bo Kullmar

PS Denna artikel är **inte** förankrad i klubbens styrelse. Det är mina privata åsikter helt och hållet.

## ABC-klubben

### Årsmötet 1986

ABC-klubbens styrelse redovisade boks-lut och årsberättelse som kommer att publiceras i nr 2 av ABC-bladet tillsammans med ett sedvanligt referat av årsmötet.

### Medlemsavgift och motioner

Medlemsavgiften för 1986 fastställdes till 160 kronor för seniorer och 100 kronor för juniorer. Styrelsen fick mandat att för 1987 avisera en högsta medlemsavgift för seniorer om 190 kronor och för juniorer 130 kronor. Styrelsen hade ingen proposition till årsmötet, men två motioner hade inlämnats.

Motionerna var skrivna av Bengt Olsson och ABC-Väst/Bengt Olsson. Tyvärr kom motionerna in sent på grund av postför-sening vilket gjorde att styrelsen inte hade hunnit redovisa synpunkter på de många förslagen som motionären hade ställt. Årsmötet uttalade sin uppskattning och upp-drog åt styrelsen att redovisa sina synpunkter på förslagen i ABC-bladet.

### Funktionärsvalet

Till ordförande respektive vice ordförande omvaldes Stig Löfgren och Bo Kullmar. Till ledamöter omvaldes Jan Liebie-Harkort, Jan Holmberg, Magnus Hedner och Tom Sjöberg samt nyvaldes David Andersson och Kjell Brealt. Till suppleanter i styrelsen nyvaldes Kent Berggren, Robert Svedje-hammar och John-Erik Näslund.

Till revisor omvaldes Kjell Järbin och auktor revisor Lars Gattberg, Bolins Rev Byrå, nyvaldes. Till revisorssuppleant valdes Karl Linström. Lars Gattberg tar ut sedvanligt timarvode för nerlagt arbete, vilket valberedningen uppskattade till 10 \* 350 SEK. Lars Gattberg är inte medlem i ABC-klubben, men det är inget krav.

## DIV:

### Om TVSEARCH och TVREPLACE

På kassett 13 fick vi två utmärkta och efterlängta snuttar - TV search och replace. De funkra utan problem. Varje snutt har 5-6 kommandon för att styra sökandet och ersättningarna. Jag fann det svårt att komma ihåg dessa kommandon, och la därför in följande två rader. Varje gång jag gör "sök" eller "ersätt" får jag en lista över åtgärdskoderna på skärmens botten. Koderna försvinner efter åtgärden.

Så här kan du göra om du har TVSEARCH och TVREPLAC:

### LOAD TVMAIN.

Skriv:

```
1065 F8$=CUR(22%,1%)+C-S=fram
R=bakåt Å=slut G=åter <=stympta"
+CUR(23%,0%)+SPACE$(39%)+
CHR$(13%)
2562 ;CUR(23%,0%);"SP=JA !ALL .only
<GoNext å=SLUT G=bryta";
```

<3098>

Egil Fjeldahl

### "Bästa-program-priset"

Efter själva årsmötet offentliggjorde den avgående styrelsen att Anders Fransen, Stockholm, och Kristoffer Eriksson, Örebro, hade tilldelats pris för bästa program. Priset utgår i form av ett presentkort på 1000 kronor.

Anders har gjort editorn TED och terminalrutinen TEDTERM till ABC80. Kristoffer har gjort en hel del systemprogram till ABC800-serien, bl a KEY, CHAINARE och OPTLOAD.

### Frågestund med en inbjudna

Efter årsmötet ställde följande upp för att besvara frågor från medlemmarna:

Kjell Strömli, VD Nokia-Luxor Information Systems

Bengt Lönnqvist, Nokia-Luxor I S  
Hargort Lindmark, Ulf Sandberg och Benny Löfgren från DIAB  
Lars-Göran Göransson.

Kjell Strömli förklarade bl a att det nya bolaget var ett rent försäljningsbolag och att man skulle forstätta att marknads-föra ABC-konceptet. Närvarande medlem-marna ställde ställde en hel del frågor till panelen, inte minst till Kjell Strömli.

### Traditionsenlig middag

Under kvällen samlades en tapper skara på 20 medlemmar och åt fläskfile med murkelsås och sallad samt drack vi till detta. Vinet serverades naturligtvis med läns-styrelsens tillstånd, "the same procedure as last year".

<1789>

Bo Kullmar

## Om checksumma på RAM-floppy

Hejsan!!!

Synpunkter på artikeln i första blaskan detta år anno 84 om ABC80 som RAM-maskin av Kjell-Åke Johansson.

Jag har också byggt om min ABC80 till en RAM-maskin (variant RAM-floppy), och när jag tänkte testa KÅJ:s "små snuttar", så upptäckte jag fel i dessa: står för andra checksummor 10042 & 9913 (jag har 9913).

20 POKE 1153,A  
30 POKE 1162,A/2

Det borde stå:

20 POKE 1154,A  
30 POKE 1163,A/2

Jag testade också hans varianter på olika överföringshastigheter och det lyckades ej för mig att vid utläsningshastighet med tidskonstanten 44 läsa in med 37 utan jag måste ha 36 istället (men det gick däremot bra med 37 då hade jag stängt av intern-klockan med POKE 102,237,69).

Jag testade också vad min klarade av för överföringshastighet. Det gav följande resultat:

Säker överföring med tidskonstanten 35 ut och 26 vid inläsning (ca 3000 Baud). (Med säker överföring menar jag inga fel) Jag läste in 5 st program av storleken 20k.

Jag lyckades halvdant med 33,23 fast då var jag tvungen att ta en del omläsningar på en del program innan de kom med, men det gick. Jag använde ett normalt "low-noise"-band.

<4122>

Harry Peltoniemi

# Musik på ABC80

Hör du till dom som undrar hur man gör musik på ABC80? Om så är fallet, är du sannerligen inte ensam om detta. Jag (med flera andra) har läst igenom (nästan) alla böcker som finns för ABC80 utan att hitta en förklaring till hur man får fram toner. Visst stod det om OUT 6, (udda tal mellan 1 och 256), men förutom diverse brus och sirener fick jag bara fram ungefär tre olika (rena) toner.

På ABC-kassett 5 fanns det ett program som producerade musik på kassett. Detta var ju visserligen bra men jag ville ju ha mina låtar i själva programmet och då lämpade sig inte detta program.

Vidare kom det då och då på kassetterna program som spelade låtar typ Sloop John B med flera. Då trodde jag att jag var nära lösningen, men icke. När jag listade programmen fann jag till min stora sorg att en stor del av programmen bestod av datasatser som inte gav mig ett dugg.

Lösningen kom en dag när jag spelade ett spel som heter Missile Command eller nånting åt det hållet. Mellan omgångarna spelade datorn en liten låt, visserligen inte vacker, men ändå en låt. Jag ska nu förklara hur man kan göra sina egna låtar. Börja med att skriva in följande rader:

```
100 ONERRORGOTO 200
110 READ A%,B%
120 FOR C%=1% TO B%
130 OUT 6%,0%,6%,121%
140 FOR D%=1% TO A%
150 NEXT D%
160 NEXT C%
170 GOTO 100
200 END
```

Detta är grundprogrammet. Om du kör det nu kommer det inte att hända nåt, men om du lägger till raden:

```
10 DATA 50,100
```

och sen skriver RUN så börjar datorn tjuta. Vad beror nu detta på? Ja inte vet jag! Första siffran i datasatsen är i alla fall tonens höjd och andra siffran längden på tonen. Tonhöjden bör vara mellan 22 och 78, annars låter det lite falskt (testa gärna ändå!). Tonlängden kan vara hur lång som helst (tror jag).

Här kommer några datasatser som spelar en trevlig låt:

```
10 DATA 50,25,60,20,76,30
20 DATA 60,38,50,46,36,110
30 DATA 27,37,31,35,36,60
45 DATA 60,37,52,45,49,90
50 DATA 49,47,27,120,31,35
55 DATA 36,60,38,110
60 DATA 43,26,38,30
70 DATA 35,60,35,60
80 DATA 49,45,60,40,76,47
85 DATA 50,25,60,20,76,30
90 DATA 60,38,50,46,36,110
95 DATA 27,37,31,35,36,60
97 DATA 38,55,31,70,36,120
```

Skriv nu RUN och njut(?) av musiken!!! Kanske inte helt fulländat, men ändå! Ta väck datasatserna och skriv in följande:

```
10 DATA 70,15,70,18,70,14,52,35,52,35,
45,37,45,37, 33,80,40,25,52,37,40,30,
52,15,63,30,38,90, 43,30,52,15,52,80
```

Ack så vackert! Eller hur? Nu tar vi en sista rolig grej. Skriv NEW och:

```
10 FOR A%=10% TO -10% STEP -1%
: FOR S%=1% TO 10% : OUT 6%,
0%,6%,121% : FOR D%=1% TO A%
: NEXT D% : NEXT S% : NEXT A%
```

Slutligen: Nu vet ni hur ni ska skriva låtar. Skicka gärna in era datasatser till ABC-klubben när ni har komponerat färdigt så att andra kan få njuta av dom!

Hej då!

<5486>

Fredrik Tengroth

## Varför blir det på detta viset?

Jag skulle vara tacksam över om jag kunde få hjälpmed att lösa följande lilla triviala men ändå så förargliga problem.

Jag är tyvärr alltför okunnig i programmering för att kunna lista ut varför följande lilla utskriftsprogram inte fungerar som jag tänkt. Som den intelligenta säkert omedelbart ser och hanterar åt, får jag dubbelt radavstånd vid printerutskriften på text som t ex är inlagd med hjälp av VDO-editorn eller TV-editorn.

Varför blir det så?

```
5 DIM A$=100
10 OPEN "PR:" ASFILE 3
15 PRINT "ANGE FILNAMN";
16 INPUT B$
20 OPEN B$ ASFILE 2
30 INPUTLINE $2,A$
40 PRINT $3,A$
50 GOTO 30
60 CLOSE 3
70 CLOSE 2
80 END
```

Hälsningar

<5621>

Olle Wiren

### Förklaringsförsök

Först måste konstateras att varje gång vi skriver på filnumret \$3, dvs printern, sker en radmatning. Detta kan visas genom att vi skriver ut en tom sträng, med PRINT \$3,"". Sedan kommer Olles variabel A\$ som har läst in från fil \$2 på rad 30 med en INPUTLINE. Den avslutas alltid med CR och LF. Det är dessa som inte skall finnas.

Man kan då t ex ta bort två tecken i slutet av variabeln med raden

```
35 A$=LEFT$(A$,LEN(A$)-2))
```

LEN(A\$) ger aktuell längd av strängen A\$. Vi minskar detta tal med 2, (CR o LF). I LEFT\$(argument 1,argument 2) blir då den ursprungliga strängen A\$ argument 1 och dess längd minskad med 2 argument 2. Resultatet A\$ blir de ursprungliga tecknen i strängen förutom de två sista.

Och skrivs strängen nu ut på printer finns ju nu inte någon "extra" radmatning, varför Olle inte får dubbelt radavstånd i sina utskriften.

<1208>

Ulf Sjöstrand

### Angående flexskiveenheter

Man blir minst sagt bedrövad då man ser vilka priser flexskiveenheter i handeln betingar. Till vissa andra på marknaden förekommande datorer existerar disccinnen som är prissatta till 3-4000 kronor, men till ABC-maskinerna finns ingenting under ca 10 000 kr. (Dock har det börjat dyka upp utrustning kring 6 000 kr.)

Jag har tittat i den senaste medlemsmatirkeln och funnit många medlemmar inte äger flexminne. Detta beror säkerligen helt och hållet på de höga priserna. Jag undrar: skulle ingen av alla klubbens tekniskt sinnade medlemmar kunna konstruera ett billigt alternativ till de dyra flexminnena? Många skulle säkert nöja sig med enbart en drive, men kompatibilitet med exempelvis Luxors 830 skulle naturligtvis vara en fördel. (Detta skulle kanske uppfattas som plagiat?). Det finns säkert också många som skulle kunna utföra själva komponentmonteringen och sammansättningen på egen hand, under förutsättning att ritningar och byggbeskrivning finns tillgängligt. (Kanske en artikelserie i bladet?) Ett pris på ca tre-tusen skulle nog vara överkomligt.

Är detta genomförbart? När jag tänker efter minns jag att jag hört om ett liknade projekt, möjligen hos ABC-Väst. Hur gick det?

Jag inser att det skulle bli svårt med PROM-bränning och kretskortstillverkning, men om tillräckligt många var intresserade kunde kanske en mindre serie PROM och kretskort till verkas av någon firma.

<4166>

Bengt Ask

Ps Hur vore det med ett liknade projekt för modem och minnesexpansion för ABC80?

Utdrag ur MSG angående buggar i programmet VÄXLAPGM.UTL på ABC-kassett nummer 16.

(Text 1191) Anders Franzen <5258>

Ärende: VXLPGM

Programmet VÄXLAPGM.UTL innehåller några allvarliga buggar. Det går inte att läsa filer från kassett och man får inte skriva in för långa rader när VÄXLAPGM är aktivt.

ABC80 dyker när man läser från kassett därför att avbrottsvektorn för kassettinterrupt saknas. Om en BASIC-rad skrivs in och som kompilerad upptar mer än 72 bytes dyker datorn därför att avbrottsvektorn för tangenttryck placerats i kompileringsbufferten!

Man kan åtgärda det första felet på ett enkelt sätt. Det andra är mer svårslutligt. Det bästa vore om maskinkoden till VÄXLAPGM placerades på något annat ställe än i POKE-arean högst upp i minnet. Man kan emellertid flytta avbrottsvektorn till annan plats.

Dessa rader gör VÄXLAPGM lite bättre:

```
151 IF PEEK(65064)=251 POKE 65064,250 :
REM sänk stack
540 POKE 64052,148,255,148,5 : REM interrupt-
vektorer
541 POKE -112,250 : REM rätt värde till
I-reg
```

Rad 540 ersätter gammal rad, de övriga är tillägg. Nu kan man läsa från kassett utan problem. Denna ändring ökar maxlängden på BASIC-rader till 199 bytes vilket dock är för litet ibland.

# Grekerne kunde sudda...

I ABC-bladet 3, 1985 slog klubbens meste(?) lärare till med besked. För dessa sina artiklar har han inhöstat mycket beröm. Jag, en högst ordinär "Svenne Karlsson", med utpräglat dåligt omdöme, tycker dock att mycket är en blandning av okontrollerade påståenden, felaktigheter och, för att använda ett milt uttryck, rent nonsens. Efter att ha läst artikeln om primtal kan man bara utbrista JASÅ. Är det måhända så, att författaren numera tycker sig "behärska läget" och av den anledningen inte anser sig behöva korrekturläsa sina alster? Jag skall villigt erkänna, att jag läst artikeln såsom Fan läser Bibeln, men som professionell pedagog borde SW vara väl medveten om, att ett bra sätt att förlora auktoriteten bland sina elever, är att lämna dåligt underbyggda uppgifter i sin undervisning.

Här kommer, i den ordning de uppträder, de sakuppgifter som förefaller mig felaktiga. (I många fall är det fråga om rent ordryteri, men eftersom SW i andra sammanhang visat sig måna mycket om språket, tar jag mig den friheten.)

Med primtalsberäkning torde avses primtalsbestämning.

Arkimedes räknas förvisso som sin tids störste matematiker, men Euklides stod för beviset, att det finns ett oändligt antal primtal; ett bevis som i sin enkelhet vore synd att undanhålla dem, som inte redan stiftat bekantskap med det.

1/ Utgå från att det finns ett ändligt antal primtal  $P(1), P(2), \dots, P(n)$ .

2/ Betrakta talet  $N = P(1) \cdot P(2) \cdot \dots \cdot P(n) + 1$

3/ Inget av de givna talen är divisor i  $N$ , ty om man dividerar  $N$  med vilket som helst av dessa primtal, blir resten alltid 1.

4/ Alltså är  $N$  antingen ett nytt primtal, eller också har  $N$  en primfaktor skild från alla  $P(i)$ .

5/ Bägge möjligheterna leder till en motsägelse, varav följer att antalet primfaktorer är oändligt stort.

Att 1803 inte kan vara en primtalstvilling inses lätt. Redan här borde SW ha märkt, att programmet inte fungerar som det skall. (Exemplet 11, 13, 17, 19 innehåller förutom primtalstvillingarna, två par primtalstrillingar och utgör dessutom i sin helhet exempel på primtalsfyringar. När det gäller frågan om huruvida antalet primtalstvillingar för-

svinner vid höga tal, torde man ännu inte ha lyckats bevisa någonting, men det finns goda grunder för antagandet att så inte är fallet.)

## Man talar om primfaktorer (SAOL).

Sedan görs en hänvisning till Guido Petz' program PRIM på kassett 1. Det programmet fungerar inte så som SW uppger. Det är riktigt, att det dividerar med 2, 3 och 5, men fortsättningsvis sker division endast med de tal, som  $*\text{inte}*$  är delbara med dessa tal. Detta får jag anledning återkomma till senare. ASCII-aritmetiken tillåter vidare undersökning av tal med upp till 28 siffror.

Eratosthenes är namnet på greken, som gör det möjligt, att på rimlig tid bestämma primtalen i ett intervall. (Dvs så länge talen är måttligt stora.)

Att börja strykningarna med kvadraten på resp. tal, så som SW anvisar, kan i förstone synas smart, men medför 25-30% fler operationer jämfört med att stryka första multipel och låta programmet hoppa över de tal som redan är strukna.

Att programmet måste skrivas med heltalsvariabler för att minnet skall räcka för tal av storleksordningen 14000 är kanske inte helt självklart för nybörjaren.

Programrad 455 sägs vara till för att undvika trassel med de tal, som finns i primtalstabellen. Det enda tal mindre än  $T(N-1)$ , som inte skulle avvisas i rad 470 är 1, och detta borde tas om hand i samband med INPUT-satsen.

Det finns egentligen inget behov av att dimensionera en särskild vektor  $T()$  att lägga primtalen i. Det går lika bra att flytta upp dem till början av den redan befintliga vektorn  $P()$ .

I den rättelse, som dyker upp i nr 4, 1985 ägnas stort utrymme för att rättfärdiggöra det tankefel SW gjorde på programrad 460. Det hela är ju dock mycket enkelt. Vill man inte utnyttja index noll, lägger man räknaren före tilldelningssatsen, eljest efter. Konstigare än så är det faktiskt inte. (Men nog är det väl onödigt att införa division med 0.)

Datorn "konstrar" naturligtvis inte vid ett visst tal, utan den gör vad den blivit instruerad att göra. Att detta skulle hänga samman med storleken av det första primtal, som finns med vid beräkningarna, är förstås rent nonsens. (F ö får vi aldrig

veta inom vilket talområde programmet fungerar.) Vad SW aldrig upptäckt, eftersom han förmodligen alltid matat in ett jämnt starttal, är att programmet går snett redan vid 200001, ty  $200001/2 = \text{INT}(200001/2) = 1E+5$ , vilket medför att 200001 betraktas som ett jämnt tal, vilket på rad 1010 ökas med 1, med följd att starttalet blir jämnt och datorn matar därefter ut jämna "primtal". Det hela hänger förstås ihop med den dåliga siffernoggrannheten hos ABC80. Med 6 siffror kan man inte uttrycka ett decimaltal större än 99999.9, men om man därför ska säga, att datorn vågar utföra operationen  $\text{INT}(A)$  är väl tveksamt. Heltalsdelen av ett heltal är väl just heltalet, eller .....? Men om man nu aldrig behöver utföra en division med 2, då fungerar väl programmet upp till 300000? Nej tyvärr, det går fel för t ex divisionen 224339/223. Nu invänder förstås vän av ordning, att 224339 inte är ett primtal, och det är riktigt, men faktum kvarstår, att 973 primtal försvinner i intervallet 200000-300000 p g a den dåliga räkenoggrannheten.

Vi får veta, att "FAKTOR-metoden" är hopplöst urlusig. Visserligen sägs det inte, att SW's program skulle vara snabbare, men så som skrivningen är gjord, får man ändå intrycket, att genom att sälla ut primtalen kommer man mycket snabbare till målet. Jag har roat mig med att mäta tiden det tar att avverka intervallet 100000-200000. (Varför intervallen rekommenderas vara små övergår mitt förstånd). Det tog 5 h 22 min. För att få en jämförelse, modifierade jag det tidigare nämnda programmet av GP, och visst behövde det längre tid, men skillnaden var förbluffande liten. Det behövde 5 h 29 min (!) för att utföra samma uppgift.

SW rekommenderas till sist att lämna in sin dator för översyn, om den levererar sådana resultat vid inläsning från tangentbordet, som han uppger.

I första hand vänder jag mig nu till andra "Karlssöner". De, som redan passerat detta stadium i utvecklingen, besitter naturligtvis redan de kunskaper, som fordras för att göra ett bra primtalsprogram själva och lär inte ha mycket att hämta i följande text och programförslag.

Hur man införar för att utnyttja hela minnet till Eratosthenes' såll är minst sagt luddigt beskrivet, varför jag här lämnar ett förslag till lösning.

```

100 A%=29000% : DIM A%(A%/2%) :
REM 16k A%=12850
110 FOR I%=3% TO SQR(A%) STEP 2%
120 IF A%(I%/2%) THEN 140
130 FOR J%=I%+I%/2% TO A%/2%
STEP I% : A%(J%)=1 : NEXT J%
140 NEXT I%
150 ; '2' : FOR I%=1% TO A%/2%
160 IF A%(I%) THEN 180
170 ; 2%*I%+1% : IF (INP(56) AND
127%)=32% GET A$
180 NEXT I%

```

Att använda STEP 2 på rad 110 är bara ett bekvämt sätt att komma åt endast de udda talen. Man kunde lika gärna skriva

```

110 FOR I%=1% TO SQR(A%)/2%
120 ...
130 FOR J%=3%*I% TO A%/2% STEP
2%*I%+1% : ...

```

Resultatet blir detsamma. Programmet innehåller inga konstigheter, men eftersom ingen plats reserverats för jämna tal fördubblas, som ju SW påpekat, talområdet. På rad 170 läses tangentbordsporten av, och har mellanslagstangenten tryckts ned stannar programmet och inväntar nytt tangenttryck för att fortsätta. Att jag skriver A%=29000 : DIM A%(A%/2%) i stället för A%=14500 : DIM A%(A%) beror på, att jag i de följande programförlagen, tycker att den första skrivningen medför vissa fördelar och därför har föredragit en konsekvent skrivning.

Nu finns emellertid också möjligheten att tänja ut talområdet en aning genom att höja "taket". Om detta kan man läsa i bruksanvisningen eller Bladet Saml nr 1980:2 s 5. Man tar helt enkelt dos- och/eller kassettbuffertarna i anspråk genom att lägga till en rad

```
90 POKE 65064%,253%
```

Vad man då gör, är att ändra den s k stackpekaren (i bruksanvisningen felaktigt angiven som EOF A-pekaren), från adress 62720 (64256 om man bara använder kassett) till 253\*256 = 64768. (Jfr minneskartan i bruksanvisn.). Genom att på detta sätt höja "taket" kan man ändra A% till 31300 resp 15050.

Men se'n då? Tja, man får väl göra som på sandlådans och griffeltavlans tid. Sudda, sudda, sudda,...

Det problem man ställs inför, när man ska använda Eratosthenes' säll i ett godtyckligt intervall, är att faktorerna inte längre har den obrutna ordningen 3,5,7 ...n. Man kan naturligtvis tänka sig att pröva varje tal med faktorerna i tur och ordning och starta strykningen när man hittar en delare, men även så kommer det att bli ett mycket tidskrävande program. Men, det finns en enkel lösning också här. Ta t ex intervallet 100-200. INT(100/3)\*3=98. Härur inses lätt att 102 är delbart med 3, varför man utgående från 102 stryker vart tredje tal. För 5 hamnar man på 100 och stryker vart femte, för 7 på 105 och stryker vart sjunde osv. Rent generellt kommer att gälla, att  $s = \text{INT}(I(0)/p + 1) * p - I(0)$ , där  $I(0)$  är intervallets första tal. Vidare gäller då, att s alltid blir > 0, dvs s pekar aldrig på I(0). Den inskränkningen passar oss emellertid bra, då vi inte vill ta upp plats för de jämna talen. Eftersom hälften av talen därigenom försvinner, måste s delas med 2 samt en korrigering av det nya startvärdet göras. Alltså: Vi bestämmer oss för, att som I(0) behålla det jämna tal, som föregår intervallet vi vill sälla. Vi beräknar s som tidigare och dividerar med 2. Blir s vid divisionen med 2 ett bråk, behöver vi endast lägga till 0.5 för att få den nya startpositionen, medan vi i det fall s/2 blir ett jämnt tal beräknar INT(s)+p-INT(p/2), för att få rätt startposition.

Programmet PRIMART, som klarar alla primtal <1E+6, bygger på denna ide och är helt skrivet i BASIC.

På rad 260-300 sällas primtalen < 1000 ut. SQR(SQR(999999))=31 och SQR(999999)/2=499.

På rad 310-330 sker utplock och tilldelning.

Rad 340-370 används för inmatning och kontroll av starttal.

På rad 380 adderas 1 om det inmatade starttalet var jämnt.

Rad 390-400 genomlöps om starttalet är mindre än 1000.

Rad 410 Här sker nollställning av det utrymme som reserverats för "flaggor".

Rad 420-470 Här beräknas utgångsvärden att användas vid det intervall, som står närmast i tur för sällning.

Rad 480-570 Här sker själva sällningen. Observera att ASCII-aritmetik endast används när flyttalsnoggrannheten inte räcker till.

Rad 580-600 Här sker utplock, beräkning och utskrift av primtalen. Eftersom talen bör presenteras på ett vettigt sätt, och inte som flyttal, omvandlar vi flyttalen till en teckensträng och plockar bort decimaltecken och exponent på rad 595.

Rad 610 beräknar startvärde för nästa intervall och om detta ligger inom tillåtet talområde, sker hopp till rad 410 där minnet rensas osv

Programmet behöver ca 49.5 min, med 32k minne, för att sälla fram samtliga primtal <1E+6. Rutinen för nollställning tar sin rundliga tid, och jag tyckte att det kunde vara roligt att försöka mig på att skriva den i maskinspråk. Tog därför djupt tag i plånboken och skaffade litteratur. Efter timmar av studier, åtföljda av Trial and Error måste jag ge upp. Jag insåg, att jag inte skulle komma att nå målet, dvs de 15 minuter körtid, som jag hoppats på.

Jag lärde mig en hel del under dessa övningar och eftersom jag som "Karlsson" själv mången gång önskat, att någon skulle visa "hur man gör", skriver jag följande som ett led i det erfarenhetsutbyte, som signaturen 'Svenne' efterlyser.

Vad Du behöver känna till för att förstå det här, är att veta vad en oktett (byte) är, att den består av 8 bitar, att Z80 har ett antal register vardera rymmande en oktett, nämligen B,C,D,E,H,L,A OCH F. Det sistnämnda är ett status-(eller flagg-) register. Av de "flaggor", som sätts vid olika operationer, behöver vi i det här sammanhanget endast bry oss om den bit, som sätts om resultatet av en operation blev noll (Z). Att veta att BC,DE,HL kan användas parvis som 16-bitars register. Det finns ytterligare några register i Z80, men de kommer inte att användas här. Att SWAP(A) här är ungefär detsamma som INT(A/256). Med andra ord, att om Du skriver POKE adress,777,SWAP(777) så laddas cellen, som utpekats av adressen med (777/256-INT(777/256))\*256, dvs 9 och den därpå följande med INT(777/256), dvs 3. Betydelsen av PEEK, POKE och CALL (observera vad som sägs om funktionsvärde) hittar Du i bruksanvisningen.

====>

# VT 52 / 100 / 102

## för ABC 800M, 802, 806 samt FACIT DTC

### Emulatorn klarar följande:

- klarar alla VT52's teckensekvenser
- klarar alla VT100's teckensekvenser
- klarar alla VT102's printersekvenser
- klarar 'KEYPAD' på riktigt sätt
- klarar hastigheter mellan 75—9600 bps
- medger 'privata' funktioner (teckensträngar)
- KERMIT ett snabbt asynkront filöverförings-system, som är fritt tillgängligt

- klarar att ändra parametrar i KERMIT
- Till ABC 800M behövs en ombyggnadssats för att kunna utnyttja 'KEYPAD' på riktigt sätt.

För mer information se ABC-blad 85/4 eller kontakta oss gärna.

**AUTOCODE DATA AB**

Box 1044 • 171 21 SOLNA • Tel. 08-83 43 95

# ABC DATA

## ListSkydd II 800

Listskyddar och "squeezar" vanliga BASIC-program. Spar minne. Gör programmen omöjliga att lista. Ett effektivt skydd. Kraftigt förbättrad version av gamla "ListSkydd.800". De som köpt gamla "ListSkydd.800" och vill byta upp sig behöver bara betala mellanskillnaden i pris.

**Pris 950.- (exkl moms)**

## Edass.800 - Assembler

Sensationellt editor/assembler-program för ABC800/802/806. Med Edass.800 programmerar Du assembler lika lätt som BASIC! Interaktivt med direkt syntaxkontroll av varje rad. Inbyggd bildskärms-editor (liknande "Smartaid" för BASIC-programmering). Blixtsnabb assemblering. Nu för omedelbar leverans! (Finns även för ABC80).

**Pris 1250.- (exkl moms)**

## OBS.800 Ver 2 - Ordbehandling

Ver 2.2 av ABC DATAs ordbehandlingsprogram OBS.800 har nu släppts! OBS.800 ger fullständiga ordbehandlingsmöjligheter till ett sensationellt lågt pris. Skärmentoriserat arbetssätt. Fullständiga möjligheter att styra utskrifterna. Passar till alla datorer i ABC800-serien. I ver 2 tillkom bl. a följande förbättringar: ökad snabbhet (maskinspråk), markerade partier visas med inverterad video, makron, teckenrepetering, merge, automatisk radjustering, mm, mm.

**Pris 1100.- (exkl moms)**

## OkiMate.20 - Skrivare

Mycket finessrik och prisvärd snabbskrivare. 80 tkn/s. Utskrift i färg (100-tals kulörer möjliga). Högupplösningsgrafik. Korrespondensskvalitet. Vanligt papper. Stående A4. Egen teckenuppsättning. Vanligt enstaka tecken kan definieras mjukvarumässigt. Understrykning. Lutande stil. Ytterligare stilar. Subscript. Superscript. Friktions- och piggmätning. 8 K buffert. Svenska tecken. Serieinterface.

**Pris 3295.- (exkl moms)**

☐ Skicka katalog med mer information om produkterna i annonsen och ABC DATAs övriga sortiment.

☐ Jag beställer \_\_\_\_\_

Namn \_\_\_\_\_

(Företag) \_\_\_\_\_

Adress \_\_\_\_\_

Postadress \_\_\_\_\_

Telefon \_\_\_\_\_

Kupongen skickas till:

**ABC DATA, Box 1122, 163 12 KISTA, Tel 08-750 5980**

## PROGRAMVARA

**FÖR ABC 800 / 802 / 806**

### Ordbehandling

OBS.800 1100.-  
OBS.800 NET 2500.-

### Databas/register

Dbas.2/800 1100.-

### Grafik

ABC GRAF 4500.-

### Statistik/matematik

KRONSTAT II 975.-

### Ekonomi

Ekonomipaket.800 295.-

### Systemprogram

Edass.800 1250.-  
Disassembler.800 350.-  
ProgList.800 400.-  
ListSkydd.800 250.-  
ListSkydd II 800 950.-  
CP/M Plus 3.0 1975.-

### Spel mm

Schack.07/800 198.-  
Spel ABC 800 195.-

## PROGRAMVARA FÖR ABC80

### Ordbehandling

OBS.3 700.-

### Databas/register

Dbas.2/80 900.-  
Dbas.1 600.-  
Datreg.80 198.-

### Grafik

Grafia 198.-

### Statistik/matematik

Matematikpaket.1 198.-  
Matematikpaket.2 198.-  
Matematikpaket.3 198.-  
Kurvgraf.80 178.-  
Matrispaket 98.-

### Ekonomi

Ekonomipaket.80 98.-

### System

Edass.80 645.-  
Disassembler.702 200.-  
Debugger.707 155.-  
SYS ABC 375.-  
ProgList.80 250.-  
Dumpareg 98.-  
Programmering.1 148.-  
Programmering.2 148.-  
Autosök 248.-

### Spel mm

Teacher 78.-  
Musik 158.-  
Tips.1 68.-  
Tips.2 98.-  
Schack.07/80 158.-  
Spel ABC 1 98.-  
Spel ABC 2 98.-  
Spel ABC 3 98.-  
Spel ABC 4 98.-  
Spel ABC 5 98.-  
Minispiel 1 68.-  
Minispiel 2 68.-

**Begär ytterligare information!**

# Ännu mer om primtal

## PRIMUS

Att använda en heltalsvariabel (=två oktetter) för att lagra ett tal, som ryms i en oktett, är slöseri. Kan man finna ett sätt att lagra en "flagga" i varje oktett, kan man därmed utöka omfånget av det intervall man vill undersöka till det dubbla. Lösningen på problemet är enkel. Med hjälp av satsen POKE adress,värde markerar man att ifrågavarande minnescell är utnyttjad.

### Variabelroten

För att känna till i vilka celler man skall kunna använda POKE-kommandot, utan att riskera att programmet havererar, måste man veta var i minnet de olika variablerna är lagrade. Uppgift om detta får man genom att titta på den s k variabel-listan. Det är en länkad lista (dvs i uppgifterna om varje variabel finns en pekare till nästa variabel), som skapas samtidigt, som man kör igång sitt program. För att datorn skall veta var denna lista börjar, finns adressen till den lagrad i minnescellerna 65065 och 65066. I listan ligger variablerna lagrade i samma ordning, som de dyker upp i programmet. För varje variabel används 6-14 oktetter för att lagra den erforderliga informationen om variabeln i fråga. De fyra första oktetterna har alltid samma uppgift. Oktett 1 och 2 anger variabeltyp och beteckning medan oktett 3 och 4 ger adressen till nästa variabel i listan. Skulle dessa sista oktetter vara satta till 0 betyder det att variabel-listan är slut.

### För resterande oktetter gäller:

#### Variabeltyp

|                |                     |
|----------------|---------------------|
| Flyttal        | B B B S E           |
| Heltal         | V V                 |
| Sträng         | D D A A L L         |
| Flyttalsvektor | A A M M T T         |
| Heltalsvektor  | A A M M T T         |
| Strängvektor   | Q Q M M T T         |
| Flyttalsmatris | A A M M T T X X Y Y |
| Heltalsmatris  | A A M M T T X X Y Y |
| Strängmatris   | Q Q M M T T X X Y Y |

### I ovanstående sammanställning betyder

|           |                                                                                                                   |
|-----------|-------------------------------------------------------------------------------------------------------------------|
| B B B S E | ett flyttalsvärde (B=2 BCD-siffror, S=tecken, E=10exponent + 128)                                                 |
| V V       | ett heltal (skiftade oktetter, swappade bytar)                                                                    |
| D D       | dimensionerad längd på strängen                                                                                   |
| A A       | adress till strängens (vektorns, matrisens) början                                                                |
| L L       | aktuell längd på strängen                                                                                         |
| M M       | totala antalet element i vektorn (matrisen)                                                                       |
| T T       | tillfällig lagring av mellanresultat                                                                              |
| Q Q       | pekare till tabell över enskilda strängar i en vektor (matris). Varje element får en egen uppsättning D D A A L L |
| X X       | övre gräns för första index i en matris                                                                           |
| Y Y       | övre gräns för andra index i en matris                                                                            |

Med kunskap om denna tabells utseende blir det alltså möjligt för oss, att ta reda på adressen till olika variabler (adresser som ändras så snart vi gör en ändring i vårt program, eftersom listan ligger direkt efter det BASIC-program vi skrivit).

I programmet PRIMUS används några variabler för att direkt kunna adressera programmet till olika delar av minnet. De adresser som i första hand är intressanta är:

B% = adressen till det utrymme vi reserverat för att sätta flaggor för icke primtal.

B1% = adressen till primtalslistan minskad med 1, dvs sista cellen i det nyss nämnda utrymmet.

C% = adressen till Q\$, som dels tjänar som minnesarea för adresser vid utplockningen av primtalen, dels bidrar med mellanrum vid utskriften. För att vara säkra på att det finns ett mellanrum som första tecken i Q\$, stoppar vi dit ett på rad 160. Q\$ tilldelas ju aldrig något innehåll med någon tilldelningssats i programmet och kan därför innehålla vad som helst. Används programmet efter en kallstart är 50% chans att där ligger en nolla, som inte kommer att störa utskriften, annars kan finnas vad som helst mellan 0 och 255.

C1% = den del av Q\$ där vi kommer att lägga primtalet i form av en sträng före utskriften.

D% = adressen till värdet av N1, dvs primtalet i flyttalsform.

D1% = adress till exponenten för N1

### Nollställning

För att nollställa det utrymme A%(), som reserverats för 'flaggor' så snabbt som möjligt är det lämpligt att använda en maskinspråksrutin. En lämplig instruktion att använda härvid är LDDR (Load, Decrement and Repeat). Denna instruktion arbetar på registren enligt följande:

```
(DE) <-- (HL);
DE <-- DE-1;
HL <-- HL-1;
BC <-- BC-1;
```

Detta upprepas till BC=0. Med en registerbeteckning inom parentes förstås att de data, som skall användas finns på den adress, som finns lagrad i ifrågavarande register. Den sista cellen i vårt reserverade utrymme kommer alltid att förbli noll under programkörningen, om vi inte skriver i den, och därför laddar vi HL med adressen till denna cell och DE med adressen till cellen före. Räknaren BC laddas med intervallets längd. Vi får

```
LD BC,n      ; 1,n,n   intervallets längd
LD DE,n      ; 17,n,n  första cell som skall nollställas
LD HL,n      ; 33,n,n  cell som innehåller en nolla
LDDR         ; 234,184 flytta data till BC=0
RET          ; 201
```

Detta återfinns på rad 170. Vi ser här att instruktionerna varierar i längd. En instruktion kan vara 1 - 4 oktetter lång. n står förstås för talvärden, som beräknas av programmet.

**Primfaktorer**

Största primfaktor är  $\leq \text{SQR}(\text{IE}+6)$ . Största primfaktor vid sällning av dessa blir därför  $\text{SQR}(\text{SQR}(\text{IE}+6))$ , dvs 31. Vi gör en slinga med vars hjälp vi stryker (sätter flagga för) de tal, som är multiplar av 3,5,...,31. För att snabba upp det hela gör vi så här:

|               |          |                                                                                                                                                   |
|---------------|----------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| LD B,n        | ; 6,n    | Ladda B med det antal p som ryms i intervallet                                                                                                    |
| LD C,0        | ; 14,0   | Ladda C med höga delen av antalet p                                                                                                               |
| LD HL,nn      | ; 33,n,n | Ladda HL med startposition intervallet                                                                                                            |
| LD A,0        | ; 62,0   |                                                                                                                                                   |
| HIT LD (HL),E | ; 115    | Ladda adressen som utpekats av HL med innehållet i E. (Flaggan = låga delen av primfaktorn, som ju aldrig kan bli noll eftersom alla P() är udda) |
| ADD HL,DE     | ; 25     | Räkna upp HL ett "hopp"                                                                                                                           |
| DJNZ          | ; 16,252 | Hoppa till HIT så länge B<>0 (Decrement, Jump relative if NonZero) Följande bit behövs för den egentliga sällningen                               |
| CP C          | ; 185    | Är C=0? (ComPare)                                                                                                                                 |
| RET Z         | ; 200    | Return om C=0 (Zero)                                                                                                                              |
| DEC C         | ; 13     | Minska C med 1 (Decrement)                                                                                                                        |
| JR            | ; 24 247 | Hoppa till HIT (Jump Relative)                                                                                                                    |

De celler som ändras för varje nytt primtal tilldelas sina värden på rad 280 (540), övriga får sina värden på rad 170.

Hur vi hittar startposition för våra strykningar har beskrivits tidigare. Nu måste vi emellertid också hålla reda på, hur många gånger vi behöver sätta en flagga för att inte skriva över programmet, så att det därmed havererar. Detta antal får vi genom att ta heltalsdelen av intervallets längd minskad med startpositionen, delad med primfaktorn och öka kvoten med 1. Alltså:  $\text{INT}((\text{LEN}-s)/p)+1$ . Detta sker på rad 540 där också startpositionen adderas till intervallets begynnelseadress i minnet.

Vid beräkningen av s är det nödvändigt att ta hjälp av ASCII-räkning, eftersom vi i vissa delar rör oss i talområdet över 200000. Som en garantering mot att avrundning uppåt sker (ASCII-aritmetiken avrundar ju korrekt), begär vi ett antal decimaler i kvoten, vilka sedan plockas bort (trunkeras). (Rad 500-510)

För att ta ut de tal, som inte blivit strukna, utnyttjar vi en annan instruktion, som är repeterande nämligen CPIR (Compare Increment and Repeat). Den utför följande A-(HL); HL<-HL+1; BC<-BC-1 Detta upprepas tills BC=0 eller A=(HL)

|            |              |                                                                  |
|------------|--------------|------------------------------------------------------------------|
| LD DE,nn   | ; 17,n,n     | Ladda DE med adressen till intervallets början                   |
| LD BC,(nn) | ; 237,75,n,n | Ladda BC med intervallets längd, som finns i den utpekade cellen |
| LD HL,(nn) | ; 42,n,n     | Ladda HL med adress i intervallet                                |
| LD A,0     | ; 62,0       | Ladda ackumulatorn med det värde, som skall jämföras med         |
| CPIR       | ; 237,177    |                                                                  |
| LD (nn),BC | ; 237,67,n,n | Spara ny resterande intervalllängd                               |
| LD (nn),HL | ; 34,n,n     | Spara ny adress                                                  |
| SBC HL,DE  | ; 237,82     | Minska HL med innehållet i DE (Subtract with Carry)              |
| RET        | ; 201        |                                                                  |

Detta återfinns på rad 180

**Omvandling från flyttal till sträng**

Vid återhoppet från en rutin skickas värdet av innehållet i HL med som argument. Vi har nu sett till, att där ligger antalet steg som gjorts genom att minska adressen, som ligger i HL, med startadressen, som ju ligger i DE. På rad 580 beräknas värdet av primtalet och ligger i form av ett flyttal i variabeln N1. Text 1.00289E+5. För att kunna presentera detta som 100289 kan man ju omvandla till sträng med NUM\$ och plocka bort decimaltecken och exponent med LEFT\$ resp MID\$ (se rad 460). Det är emellertid ganska tidsödande operationer och det är här som (hjälp)-

variabeln Q\$ kommer in i bilden. Med hjälp av maskinspråk skall vi "plockas isär" de BCD-kodade (Binary Coded Decimal) siffrorna och flytta dem till Q\$.

För att uttrycka siffrorna 0-9 åtgår endast 4 bitar. En sådan grupp kallas halvoktett (nibble). En oktett är alltså 2 halvoktetter och rymmer därmed två siffror. För att komma åt varje siffra för sig, använder vi instruktionen RLD (Rotate Left Decimal). Den fungerar så, att de 4 högsta bitarna i den cell, som utpekats av HL överförs till ackumulatorns lägre del. På deras plats läggs de 4 lägsta bitarna i cellen, och dessa ersätts i sin tur av det, som fanns på motsvarande plats i ackumulatorn. Allt detta sker samtidigt.

Så här gör vi:

|            |           |                                                   |
|------------|-----------|---------------------------------------------------|
| LD DE,nn   | ; 17,n,n  | Ladda DE med adressen till oktett 2 i Q\$         |
| LD HL,nn   | ; 33,n,n  | Ladda HL med adressen till N1                     |
| LD A,48    | ; 62,48   | Ladda ackumulatorn med 48 för att få ASCII-värdet |
| LD C,3     | ; 14,3    | Ladda C med 3 (antalet oktetter med BCD-siffror)  |
| NEX LD B,2 | ; 6,2     | Ladda B med 2 (varje oktett har 2 siffror)        |
| DIT RLD    | ; 237,111 | Flytta en nibble till ackum.                      |
| LD (DE),A  | ; 18      | Lägg talets ASCII-värde i Q\$                     |
| INC DE     | ; 19      | Öka DE med 1                                      |
| DJNZ DIT   | ; 16,250  | Hoppa till DIT                                    |
| INC HL     | ; 35      | Peka på nästa oktett med BCD-siffror              |
| DEC C      | ; 13      | En oktett mindre att behandla                     |
| JRNX NEX   | ; 32,244  | Om obehandlad oktett hoppa till NEX               |

Vid relativa hopp betyder ett tal >129 ett hopp bakåt medan ett tal mindre än eller lika med samma tal innebär ett hopp framåt.

**Utskrift**

Vill man använda ABC-tolkens rutin för utskrift lägger man adressen i HL och antalet tecken, som skall skrivas ut i BC före anropet av rutinen, som finns på adress 11D

Så här blir det:

|           |            |                                                                             |
|-----------|------------|-----------------------------------------------------------------------------|
| LD A,(nn) | ; 58,n,n   | Ladda ackumulatorn med exponenten. (nn) är adress, swap%(adress)            |
| AND 127   | ; 230,127  | Maska bort bit 7                                                            |
| INC A     | ; 60       | Öka A med 1. Nu ligger strängens längd+1 i ack.                             |
| LD C,A    | ; 79       | Överför längden till C. Obs att B blev 0 redan vid omvandlingen till sträng |
| LD HL,nn  | ; 33,n,n   | Adress till strängen                                                        |
| CALL 11   | ; 205,11,0 | Anropa utskriftsrutinen                                                     |

För att kunna hinna med att läsa, måste vi ha en möjlighet att göra paus. Det gör man ju normalt med en GET-sats, men då måste man också pröva om någon tangent tryckts ner, och IF ... THEN tar tid. Därför är det lämpligt att låta den proceduren utföras, medan vi fortfarande är på maskinspråksnivå.

|               |           |                                                                 |
|---------------|-----------|-----------------------------------------------------------------|
| GET IN A,(56) | ; 219,56  | Ta in värdet från port 56 (tangentbordet)                       |
| AND 127       | ; 230,127 | Maska bort bit 7, (som sätts när en tangent trycks ned)         |
| CP 32         | ; 254,32  | Var det ASCII 32 dvs mellanslag som trycktes ned                |
| JR GET        | ; 40,248  | Lägg datorn i en slinga så så länge ingen ny tangent trycks ned |
| RET           | ; 201     | Åter till BASIC                                                 |

Detta återfinns på raderna 200-230. Vill man att programmet skall fortsätta när mellanslagstangenten hålls nedtryckt, kan man utesluta operationskoden för AND 127 (230 127).

### Övriga rader

På raderna 410-470 sker en initiering före varje sällningsomgång. Först nollställs vårt reserverade minnesområde. Därefter beräknas högsta tal i omgången. Skulle detta tal bli större än  $1E+6$  så sätts maximalvärdet till 999999 och bestäms hur stort intervallet då blir ( $=A2\%$ ). På rad 430 matar vi in intervallens längd + 2. När detta värde uppnås under utplockningen, är ju intervallet slut och återhopp till huvudprogrammet skall ske. Rad 440-450 används för att plocka ut de primfaktorer vi behöver under nästa sällningsomgång. Observera att utplocket inte startar från 0 varje gång, utan att endast de nya tillkommande talen plockas ut. På rad 460-470 räknas värdet ut för det jämna tal, som föregår intervallet.

Rad 610 räknar ut starttalet för nästa intervall, och så länge detta understiger  $1E+6$  sker hopp till 410 för en ny omgång.

På detta sätt har det alltså varit möjligt att konstruera ett program, som tar ut alla primtal  $<1E+6$  på en tid understigande 17 minuter (16k 22 min). Och under de omständigheterna behöver man ju inte ha så stort tålamod, om man vill låta programmet löpa till sitt slut.

### PRIMUSA

Låt oss gå vidare

Eftersom ABC80 kan hantera heltal upp till 32767, är det möjligt att sälla primtal av storleksordningen 1 miljard. Ett sådant program blir med nödvändighet långsammare, för i det fallet behöver ASCII-aritmetik att tillgripas i stor utsträckning och den är som bekant mycket långsam.

Eftersom antalet primfaktorer, som skall användas, blir ganska stort, måste man också minska kraftigt på den area, som avsättes för flaggor. Primtalen, som skall användas vid den fortsatta sällningen, plockas ut på precis samma sätt som tidigare. Nu tar det emellertid ca 30 sek för att sälla fram dessa, och det förefaller som en "oändlig" tidrymd när det inte syns att datorn arbetar. Därför är det lämpligt att t ex skriva ut primtalen på skärmen följt av en vagnretur, CHR\$(13), vilken gör att talen kommer att skrivas på samma ställe, innan det blir dags att mata in starttalet.

Så långt det varit möjligt har jag använt samma radnummer i alla programmen, för att det skall bli lättare att se skillnaderna. Därigenom finns luckor i den jämna numreringen genom att vissa rader blivit överflödiga, liksom även udda nummer som tillkommit genom att nya rader skjutits in.

Rad 410-470 är som tidigare initiering före själva sällningen, där det då blir nödvändigt att tillgripa ASCII-räkning.

Rad 480-570 för bestämning av startpositionen skiljer sig inte mycket från det förra programmet. Dock måste man sätta in ett villkor, som medför att programmet hoppar om s faller utanför det tillgängliga minnesintervallet. Att så sker är ju en naturlig följd av, att primfaktorerna nu kan vara större än det aktuella intervallens längd och därför inte återfinns som faktor i det intervall man just sällar.

De utplockade primtalen skulle i denna version utan vidare kunna skrivas ut, eftersom de ligger i N9\$, men även om detta program är avsevärt långsammare, behövs möjligheten att göra paus och för att inte slöa ner i onödan, har jag föredragit att behålla utskriften som en subrutin i maskinspråk. Strängens aktuella längd avläses i variabelistan och ökas med 1 för att få med det mellanrum, som ligger sist i Q\$. (Skillnad mot förra programmet.)

```
LD BC,(nn) ; 237,75,n,adda BC med längden av N9$
INC BC ; 3 Öka längden med 1
LD HL,nn ; 33,n,n Låt HL peka på cellen före N9$
```

Resten är densamma som i föregående program.

Räkna med långa körtider. Visserligen tar ett varv i början av talområdet bara 82-85 sek, men ju fler primfaktorer som är inblandade, desto tydligare märker man hur långsam ASCII-räkningen är. I slutet av talområdet handlar det om dryga 8 minuter för ett varv. Men hur långsamt är det egentligen? Det största primtalet i mitt program är 1072999987. För att fastställa att detta är ett primtal, behöver Guido Petz' program 12 min 37 sek. PRIMUSA klarar det på 8 min 18 sek. (Detta program går inte att köra med 16 k minne.)

### PRIMELVA

Det är naturligtvis möjligt att skriva ett program, som med utnyttjande av Erathostenes' säll klarar att plocka ut ännu större tal än de hittills beskrivna programmen kan. Det sker då lämpligen genom att utnyttja samma princip som Guido Petz använt.

Skriver man upp alla tal  $1,2,3,4,...,n$  och stryker alla tal, som är delbara med 2 och 3 finner man, att differenserna mellan de kvarvarande talen bildar en periodisk talföljd  $4,2,4,2,4,2,...$ . Längden för en period blir som synes 2. I programmet PRIM har använts den talföljd, som man får när talen 2,3 och 5 stryks. Då uppkommer en talföljd med 8 element, nämligen  $6,4,2,4,2,4,6,2$ . Väljer man att stryka även talet 7 blir längden 48 och om även 11 tas med fås en talföljd med längden 480. Denna är lämplig att utgå ifrån vid konstruktion av ett program för sällning av primtal med primfaktorer större än 32767. Nästa talföljd innehåller nämligen hela 5760 element, och då blir kvarvarande minnesutrymme så litet, att tiden för sällning inte längre står i rimlig proportion till storleken av det intervall som sällas.

Sällningen börjar med att talen delbara med 3,5,7 och 11 stryks, varefter programmet får gå in i en slinga där man börjar tar ca 13, varefter variabeln ökas med tillskott ur talföljden till dess faktorn blir större än kvadraten ur sista talet i sällintervall.

P g a de långa varvtiderna vid större tal (sista varvet tar ca 12 timmar att sälla) har det inte känts lika angeläget att lägga utskrift och paus som maskinspråk. Spännvidden på programmet motiverar dock en snabb nollställning.

Den begränsande faktorn i detta program är rotutdragningen på rad 440. Roten får ju inte överstiga värdet 999999. Men även den begränsningen går att komma runt genom att använda ASCII-aritmetik vid rotutdragningen. Man använder då t ex den metod SW pekar på i nr 4.1985 (som dock i den form den presenterades fungerar sämre än den inbyggda rutinen för rotutdragning). I så fall kommer man upp till tal, som på grund av sin storlek kommer att kräva så långa körtider, att det knappast kan vara meningsfullt att sälla primtal med ABC80, trots dess relativa snabbhet. Så här skulle man dock kunna göra:

```
435 Z$='I' : Z1$='I'
440 X$=DIV$(N$,Z$,0%)
441 X$=ADD$(X$,Z$,0%)
442 Z$=DIV$(X$,Z$,0%)
443 IF COMP%(Z$,Z1$)<>0% THEN Z1$=Z$ : GOTO 440
444 ; ; 'SQR 'Z$
445 IF COMP%(P$,Z$)=1% THEN 580 ELSE GOSUB 500
```

Dessutom ändras naturligtvis rad 240, 350 och 370 till med för programmet lämpliga värden.

Sammanfattningsvis kan man alltså konstatera, att det egentligen varken är minnets storlek, flyttalsnotationen eller någon annan faktor, som är avhängig av datorn, som är den begränsande faktorn. Det är tiden, som i praktiken sätter gränserna.

<2694>

Leopold Lundström



```

10 REM ++++++
11 REM ! Program .... PRIMÄRT
12 REM ! Utgåva 1.0 1985-07-01
13 REM ! av L Lundström <2694>
14 REM ! Minne 32 (16) Kbytes
15 REM ++++++
100 A%=26500% : DIM A$(A%/2%),P$(166%) :
REM 16k A%=10200
240 ; CHR$(12%);TAB(7%)"*** PRIMAL < 10
00000 ***"
250 ; ; 'Strax ...';
260 FOR I%=3% TO 31% STEP 2%
270 FOR H%=I%+I%/2% TO 499% STEP I%
280 A$(H%)=1%
290 NEXT H%
300 NEXT I%
310 FOR I%=1% TO 499%
320 IF A$(I%) THEN 330 ELSE P$(K%)=2%*I%
+1% : K%=K%+1%
330 NEXT I%
340 ; CHR$(13%);TAB(6%)"Använd mellansla
g för paus" : ;
350 ONERRORGOTO 350 : N$=' ' : 'Startta
1 : <' ;CUR(PEEK(65011%)),8%); : I
NPUT N1$
360 FOR I%=1% TO LEN(N1$) : Z%=ASC(MID$(
N1$,I%,1%)) : IF Z%>47% AND Z%<58% T
HEN N$=N$+CHR$(Z%) : NEXT I%
370 N=VAL(N$) : IF N<1000 THEN N=1001 :
GOTO 390 ELSE IF LEN(N$)>6 THEN ; 'F
ör stort tal' : GOTO 350
380 N$=DIV$(N$,'2',1%) : IF RIGHT$(N$,LE
N(N$))='0' THEN N=N+1 : GOTO 410 ELS
E 410
390 ; 'Då får Du allt från början ...'
400 ; '2' : FOR I%=0% TO 166% : ; P$(I%
) : ; NEXT I%
410 ; ; 'Vänta...' : FOR I%=0% TO A%/
2% : A$(I%)=0% : NEXT I% : ; CHR$(13
%);SPACE$(8%);
420 M=N+A% : IF M>=1E+6 THEN M=9.99999E+
5 : A%=M-N
440 Z%=SQR(M)
450 FOR I%=J% TO 166% : IF P$(I%)<Z% TH
EN J%=J%+1% : NEXT I%
460 N$=NUM$(N) : IF INSTR(1%,N$,'.') THE
N N$=LEFT$(N$,2%)+MID$(N$,4%,5%)
470 N1$=SUB$(N$,'1',0%) : N1=N-1
480 FOR I%=0% TO J%-1%
490 P%=P$(I%) : ; CHR$(13%);P%;
500 N$=DIV$(N1$,NUM$(P%),3%)
510 N$=LEFT$(N$,LEN(N$)-4%)
520 S=((VAL(N$)+1)*P%-N1)/2
530 IF INT(S)<>S THEN S=S+.5 ELSE S=S+
P%-P%/2%
550 FOR H%=S%-1% TO A%/2% STEP P%
560 A$(H%)=1% : NEXT H%
570 NEXT I%
580 ; CHR$(13%); : FOR I%=0% TO A%/2%-1%
590 IF A$(I%) THEN 600 ELSE P$=NUM$(N+2*
I%)
595 IF INSTR(1%,P$,'.') THEN ; LEFT$(P$,
2%)+MID$(P$,4%,5%); ELSE ; P%;
596 IF (INP(56) AND 127%)=32% THEN GET Ä
$
600 NEXT I%
610 N=N+A% : IF N<9.99999E+5 THEN GOTO 4
10
620 END

10 REM ++++++
11 REM ! Program .... PRIMUS
12 REM ! Utgåva 1.0 1985-11-15
13 REM ! av L Lundström <2694>
14 REM ! Minne 32 (16) Kbytes
15 REM ++++++
100 A%=25500% : DIM A$(A%/2%),P$(166%),Q
$(13%) : N1=PI : REM 16k A%=9000
110 DIM N$=11,N1$=11
120 B%=PEEK(65065%)+SWAP$(PEEK(65066%))
130 C%=PEEK(B%+32%)+SWAP$(PEEK(B%+33%))
: D%=B%+40% : D1%=B%+44% : C1%=C%+9%
: C2%=C%+1%
140 B1%=PEEK(B%+20%)+SWAP$(PEEK(B%+21%))
-1%
150 B%=PEEK(B%+10%)+SWAP$(PEEK(B%+11%))
: A1%=A%+1% : A2%=A% : B2%=B%-1%
160 POKE 65408%,1%,B1%-B%,SWAP$(B1%-B%),
17%,B1%-1%,SWAP$(B1%-1%),33%,B1%,SWA
P$(B1%),237%,184%,201%
180 POKE 65427%,62%,0%,115%,25%,16%,252%
,185%,200%,13%,24%,247%
190 POKE 65438%,17%,B%,SWAP$(B%),237%,75
%,C1%,SWAP$(C1%),42%,C1%+2%,SWAP$(C1
%+2%),62%,0%,237%
200 POKE 65451%,177%,237%,67%,C1%,SWAP$(
C1%),34%,C1%+2%,SWAP$(C1%+2%),237%,8
2%,201%
210 POKE 65462%,17%,C2%,SWAP$(C2%),33%,D
%,SWAP$(D%),62%,48%,14%,3%,6%,2%,237
%,111%,18%,19%,16%,250%
220 POKE 65480%,35%,13%,32%,244%,58%,D1%
,SWAP$(D1%),230%,127%,60%,79%,33%,C%
,SWAP$(C%),205%,11%,0%
230 POKE 65497%,219%,56%,230%,127%,254%,
32%,40%,248%,201%
240 ; CHR$(12%);TAB(7%)"*** PRIMAL < 10
00000 ***"
250 ; ; 'Strax ...';
260 FOR I%=3% TO 31% STEP 2%
270 S=B%+I%/2%+I%
280 POKE 65420%,6%,499%/I%,14%,0%,33%,S%
,SWAP$(S%)
290 Z%=CALL(65420%,I%)
300 NEXT I%
310 FOR I%=1% TO 499%
320 IF PEEK(B%+I%) THEN 330 ELSE P$(K%)=
2%*I%+1% : K%=K%+1%
330 NEXT I%
340 ; CHR$(13%);TAB(5%)"Använd <mellansl
ag> för paus" : ;
350 ONERRORGOTO 350 : N$=' ' : 'Startta
1 : <' ;CUR(PEEK(65011%)),8%); : I
NPUT N1$
360 FOR I%=1% TO LEN(N1$) : Z%=ASC(MID$(
N1$,I%,1%)) : IF Z%>47% AND Z%<58% T
HEN N$=N$+CHR$(Z%) : NEXT I%
370 N=VAL(N$) : IF N<1000 THEN N=1001 :
GOTO 390 ELSE IF LEN(N$)>6 ; 'FÖR ST
ORT TAL' : GOTO 350
380 N$=DIV$(N$,'2',1%) : IF RIGHT$(N$,LE
N(N$))='0' THEN N=N+1 : GOTO 410 ELS
E 410
390 ; 'Då får Du allt från början ...'
400 ; '2' : FOR I%=0% TO 166% : ; P$(I%
) : ; NEXT I%
410 Z%=CALL(65408%)
420 M=N+2*A% : IF M>=1E+6 THEN M=9.99999
E+5 : A2%=(M-N)/2% : A1%=A2%+2%
430 POKE C1%,A1%,SWAP$(A1%),B%,SWAP$(B%)
440 Z%=SQR(M)
450 FOR I%=J% TO 166% : IF P$(I%)<Z% THE
N J%=J%+1% : NEXT I%
460 N$=NUM$(N) : IF INSTR(1%,N$,'.')>0%
THEN N$=LEFT$(N$,2%)+MID$(N$,4%,5%)
470 N1$=SUB$(N$,'1',0%) : N1=N-1
480 FOR I%=0% TO J%-1%
490 P%=P$(I%)
500 N$=DIV$(N1$,NUM$(P%),3%)
510 N$=LEFT$(N$,LEN(N$)-4%)
520 S=((VAL(N$)+1)*P%-N1)/2
530 IF INT(S)<>S THEN S=S+.5 ELSE S=S+
P%-P%/2%
540 T%=(A%-S%)/P%+1% : S%=B2%+S%
550 POKE 65421%,T%,14%,SWAP$(T%),33%,S%,
SWAP$(S%)
560 Z%=CALL(65420%,P%)
570 NEXT I%
580 Z%=CALL(65438%)
590 IF Z%>=A2% THEN GOTO 610 ELSE N1=N+Z
%*2
600 Z%=CALL(65462%) : GOTO 580
610 N=N+2*A% : IF N<9.99999E+5 THEN GOTO
410
620 END

10 REM ++++++
11 REM ! Program .... PRIMUSA
12 REM ! Utgåva 1.0 1985-12-05
13 REM ! Av L Lundström <2694>
14 REM ! Minne 32 Kbytes
15 REM ++++++
100 A%=19500% : DIM A$(A%/2%),P$(3511%),
Q$=5%
110 DIM N9$=17%,N$=12%,N1$=12%,M$=10% :
Q$=SPACE$(5%)
120 B%=PEEK(65065%)+SWAP$(PEEK(65066%))
130 C%=PEEK(B%+32%)+SWAP$(PEEK(B%+33%))
: D%=PEEK(B%+42%)+SWAP$(PEEK(B%+43%))
-1% : D1%=B%+44%
140 B1%=PEEK(B%+20%)+SWAP$(PEEK(B%+21%))
-1%
150 B%=PEEK(B%+10%)+SWAP$(PEEK(B%+11%))
: A1%=A%+1% : A2%=A% : B2%=B%-1%
170 POKE 65408%,1%,B1%-B%,SWAP$(B1%-B%),
17%,B1%-1%,SWAP$(B1%-1%),33%,B1%,SWA
P$(B1%),237%,184%,201%
180 POKE 65427%,62%,0%,115%,25%,16%,252%
,185%,200%,13%,24%,247%
190 POKE 65438%,17%,B%,SWAP$(B%),237%,75
%,C%,SWAP$(C%),42%,C%+2%,SWAP$(C%+2%
),62%,0%,237%
200 POKE 65451%,177%,237%,67%,C%,SWAP$(C
%),34%,C%+2%,SWAP$(C%+2%),237%,82%,2
01%
210 POKE 65462%,237%,75%,D1%,SWAP$(D1%),
3%,33%,D%,SWAP$(D%),205%,11%,0%,219%
,56%,230%,127%,254%,32%,40%,248%,201
%
240 ; CHR$(12%);TAB(2%)"*** PRIMAL, 1E+
6 < p < 1.073E+9 ***" : ;
260 FOR I%=3% TO 181% STEP 2%
270 S=B%+I%/2%+I%
280 POKE 65420%,6%,16383%/I%,14%,SWAP$(1
6383%/I%),33%,S%,SWAP$(S%)
290 Z%=CALL(65420%,I%)
300 NEXT I%
310 FOR I%=1% TO 16383%
320 IF PEEK(B%+I%) THEN 330 ELSE P$(K%)=
2%*I%+1% : K%=K%+1%
325 ; P$(K%-1%);CHR$(13%);
330 NEXT I%
340 ; TAB(5%)"Använd <mellanslag> för pa
us" : ;
350 ONERRORGOTO 350 : ; 'Starttal <' ;CUR
(PEEK(65011%)),8%); : INPUT N$
370 IF VAL(N$)<1E+6 OR VAL(N$)>=1.073E+9
THEN ; 'Tal utanför tillåtet interva
ll' : GOTO 350
380 N1$=DIV$(N$,'2',1%) : IF RIGHT$(N1$,
LEN(N1$))='0' THEN N$=ADD$(N$,'1',0%)
410 Z%=CALL(65408%)
420 M$=ADD$(N$,NUM$(2*A%),0%)
425 IF VAL(M$)>=1.073E+9 THEN M$='107300
0001' : A2%=VAL(SUB$(M$,N$,0%))/2% :
A1%=A2%+2%
430 POKE C%,A1%,SWAP$(A1%),B%,SWAP$(B%)
440 Z%=SQR(VAL(M$))
450 FOR I%=J% TO 3398% : IF P$(I%)<Z% TH
EN J%=J%+1% : NEXT I%
470 N1$=SUB$(N$,'1',0%)
480 FOR I%=0% TO J%-1%
490 P%=P$(I%)
500 N9$=DIV$(N1$,NUM$(P%),6%)
510 N9$=ADD$(LEFT$(N9$,LEN(N9$)-7%),'1',
0%)
511 N9$=MUL$(N9$,NUM$(P%),0)
512 N9$=SUB$(N9$,N1$,0)
520 S=VAL(N9$)/2
530 IF INT(S)<>S THEN S=S+.5 ELSE S=S+
P%-P%/2%
540 IF S<=A2% THEN T%=(A1%-S%)/P%+1% :
S%=B2%+S% ELSE 570
550 POKE 65421%,T%,14%,SWAP$(T%),33%,S%,
SWAP$(S%)
560 Z%=CALL(65420%,P%)
570 NEXT I%
580 Z%=CALL(65438%)
590 IF Z%>=A2% THEN 610 ELSE N9$=ADD$(N$
,NUM$(2*A%),0%)
600 Z%=CALL(65462%) : GOTO 580
610 N$=ADD$(N$,NUM$(2*A%),0) : IF COMP$(
N$,'1072999999')=-1 THEN 410
620 END

```

```

10 REM ++++++
11 REM ! Program .... PRIMELVA
12 REM ! Utgåva 1.0 1986-01-05
13 REM ! av L Lundström <2694>
14 REM ! Minne 32 Kbytes
15 REM ++++++
100 A%=25000% : DIM A%(A%/2%),P%(483%)
120 B%=PEEK(65065%)+SWAP%(PEEK(65066%))
140 B1%=PEEK(B%+20%)+SWAP%(PEEK(B%+21%))
    -1%
150 B%=PEEK(B%+10%)+SWAP%(PEEK(B%+11%))
    : B2%=B%-1%
170 POKE 65408%,1%,B1%-B%,SWAP%(B1%-B%),
    17%,B1%-1%,SWAP%(B1%-1%),33%,B1%,SWA
    P%(B1%),237%,184%,201%
180 POKE 65427%,62%,0%,115%,25%,16%,252%
    ,185%,200%,13%,24%,247%
240 ; CHR$(12%);TAB(4%)"*** PRIMTAL 225
    < p < 1E+12 ***"
250 ; ; 'Strax ...';
260 FOR I%=1% TO 1155% : A%(I%)=2%*I%+1%
    : NEXT I%
265 FOR J%=1% TO 5%
270 IF A%(J%)=0% THEN 295
275 P%(L%)=A%(J%) : L%=L%+1%
280 FOR K%=J% TO 1155% STEP A%(J%)
285 A%(K%)=0%
290 NEXT K%
295 NEXT J%
300 X%=1%

```

```

310 FOR I%=1% TO 1155%
315 IF A%(I%)=0% THEN 330
320 P%(L%)=A%(I%)-X% : L%=L%+1% : X%=A%(
    I%)
330 NEXT I%
340 ; CHR$(13%);TAB(6%)"Använd mellansla
    g för paus" ; ;
350 ONERRORGOTO 350 : N$=' ' : 'Startta
    1:'SPACE$(12%)<'>;CUR(PEEK(65011%),8%
    ); : INPUT N1$
360 FOR I%=1% TO LEN(N1$) : Z%=ASC(MID$(
    N1$,I%,1%)) : IF Z%>47% AND Z%<58% T
    HEN N$=N$+CHR$(Z%) : NEXT I%
370 IF VAL(N$)<225 OR COMP$(N$,SUB$( '999
    999999999',NUM$(2*A%),0%))=1% THEN ;
    'Utanför tillåtet talområde' : GOTO
    350
380 N1$=DIV$(N$,'2',1%) : IF RIGHT$(N1$,
    LEN(N1$))='0' THEN N$=ADD$(N$,'1',0%
    )
410 Z%=CALL(65408%)
420 M$=ADD$(N$,NUM$(2*A%),0%)
440 Z=SQR(VAL(M$)) : ; ; 'SQR=';INT(Z)
445 N1$=SUB$(N$,'1',0%) : ;
450 FOR I%=0% TO 3% : P$=NUM$(P%(I%)) :
    ; CHR$(13%);RIGHT$(P$,2%); : GOSUB 5
    00 : NEXT I%
455 P$='1'
460 FOR I%=4% TO 483% : P$=ADD$(P$,NUM$(
    P%(I%)),0)

```

```

462 ; CHR$(13%);P$;
465 IF VAL(P$)>Z THEN 580 ELSE GOSUB 500
470 NEXT I% : GOTO 460
500 N9$=DIV$(N1$,P$,6%)
510 N9$=ADD$(LEFT$(N9$,LEN(N9$)-7%),'1',
    0%)
511 N9$=MUL$(N9$,P$,0%)
512 N9$=SUB$(N9$,N1$,0%)
520 S$=DIV$(N9$,'2',1%)
530 IF RIGHT$(S$,LEN(S$))='5' THEN S=VAL
    (S$)+.5 : GOTO 540
531 P1$=DIV$(P$, '2',1)
532 P1$=LEFT$(P1$,INSTR(1%,P1$,'.')-1%)
533 P1$=SUB$(P$,P1$,0)
534 S=VAL(S$)+VAL(P1$)
540 IF S>A% THEN RETURN ELSE T%=(A%-S)/
    VAL(P$)+1% : S%=B2%+S
550 IF T%>1% THEN POKE 65420%,6%,T%,14%,
    SWAP%(T%),33%,S$,SWAP%(S%) : Z%=CALL
    (65420%,VAL(P$)) : RETURN
570 POKE S$,1% : RETURN
580 ; CHR$(13%) : FOR I%=B% TO B%+A%-1%
590 IF PEEK(I%) THEN 600 ELSE P9$=ADD$(N
    $,NUM$(2*(I%-B%)),0)
595 ; P9$;' ' : IF (INP(56) AND 127)=32
    % GET A$
600 NEXT I%
610 N$=ADD$(N$,NUM$(2*A%),0) : GOTO 410
620 END

```

( Text 1058 ) Tomas Wikström <1398>  
Ärende: Ljus i mörkret - Eureka!  
Efter att ha glatt sig åt paketet med Ge-  
Jo's ramfloppy som jag hämtade ut på  
posten härom veckan, blev allt snart svart  
i dubbel bemärkelse. Vad hände ? - Jo  
efter installationen verkade allt fungera, till  
en början, men då jag började använda ram-  
floppyn, dvs körde med den initierad, och  
\*samtidigt\* hade Supersmartaiden aktiverad  
började konstigheterna. Reset !? - Va? -  
Loop - Allsköns smörja skrollande på  
skärmen - Den lilla högtalaren tjutande som  
en stungen ... Nåväl, efter 7 sorger och 8  
bedrövelser och en hel del funderande, +  
\*mycket och goda\* råd från vänner i den  
här eminenta klubben började vägen öppna  
sig ur eländet.  
Om Ni bygger ut ABC80 (speciellt då den  
med nya 'lilla separata' nätdelen, se till att  
göra som följer nedan. FÖR ABC:n DRAR  
MER STRÖM ÄN SUPPLYET KAN LEVERERA.  
Den vetenskapen har kostat mej 328 gråa  
hår, minst.

Så här gjorde jag:  
Besökte ELFA för att skaffa 2 st. nya  
spänningsregulatorer: ELFA's best-nr:  
73-0956-0 Pris 12:45 + moms. Dessa  
regulatorer är dubbelt så kraftiga som origi-  
nalet i ABC80 (de sitter förresten på kyl-  
flänsen i tangentbordet, och är bara att  
byta ut mot de nya, lätt om man har lite  
lödvana). Man bör också, speciellt om man  
har den lilla nya nätdelen, byta likriktar-  
bryggan till en kraftigare dito. (min likriktar-  
brygga gick sönder i ett mycket tidigt sta-  
dium, långt innan 80'an blev utbyggd till  
vad den är nu)

Det finns också en viss risk att inte trans-  
formatorn kan leverera den ström som kan  
tänkas behövas (på min maskin ligger jag  
på bara litet över 8 volt innan regulatorerna,  
och det bör upp litet, helst till minst 9  
volt (men ej heller för högt då överspän-  
ningen då måste kylas bort). Detta med  
tanke på att man måste räkna med viss

spridning på komponenter, gör att man bör  
kanske undersöka sin egen spänning innan  
likriktning för att se om det räckte med  
det ovanstående ändringarna eller om man  
måste se sig om efter ett kraftigare nät-  
aggregat.

Det blev mycket skrivet, men det var inte  
ett ord för litet efter allt besvär och alla  
konstigheter jag haft på sista tiden.

Många och mycket välmenande hälsningar  
från Tomas i Solna.

( Text 1058 )

Mottagare: ABC80

Ärende: Ljus i mörkret - Eureka!  
Bra skrivet! Jag har haft samma problem  
och kommit på att det är spänning/ström  
som inte räcker. Däremot har jag inte haft  
konkreta uppgifter om VAD som måste  
göras. Nu ska det bli gjort här också!

Men jag tycker Gejo m fl borde tala om  
för att lå som köper tillsatserna att den  
nya kraftburken är för klen. Jag har hittills  
klarat Superbasic+32K+80tkn utan besvär,  
men Gejos ramfloppy blev för mycket.

# Ljus i mörkret- EUREKA

( Brev 4296 ) Tomas Wikström <1398>  
Mottagare: Sven Wickberg <1384>  
Ärende: Strömförsörjningsproblem ABC80  
Nu har jag äntligen fått stil på mitt system.  
Det tog sig en tur till, som jag vill varna  
Dej för: Normalt ska 5-volten ligga på  
just 5 V (eg 4.75 - 5.25), men av någon  
anledning ville inte mina problem helt upp-  
höra förrän jag gick upp \*över\* 5.2 V!  
Så nu ligger jag på 5.4 (vid regulatorn -  
sjunker något på vägen till kapslarna) som  
är något för högt, men det måste ju fungera.  
Dessutom är det inget problem då garante-  
rad breakdown-voltage ligger på 7 V (för  
TTL).

Nå till saken - jag gjorde på ett något  
annat sätt än som jag beskrev i MSG, då  
jag var tvungen att kunna finjustera spän-  
ningen. Komponenter till det har jag, inte i  
överflöd men i alla fall så att jag kan  
dela med mig.

Hör av Dej till mig, jag hjälper gärna till  
om det behövs, eller skriver ned enkla instruk-  
tioner som visar hur jag har (varit tvingad  
till att) gå tillväga.

Hälsningar Tomas.  
( Brev 4296 )

# QSORT sorteringsrutin för ABC80

## 1. QSORT - SORTERINGSRUTIN FÖR ABC80

QSORT är en sorteringsrutin som skrivits i assembler och är mycket snabb. Assemblerkoden placeras strax under DOSbuffertarna i ABC80-minnet och den kräver 1 Kbyte. Inladdning sker med RUN QSORT. När QSORT laddats in kan man köra program som använder sorteringen. För att ta bort assemblerkoden trycker man på resetknappen till vänster på tangentbordets baksida.

Att anropa sorteringsrutinen från ett BASIC-program är mycket enkelt:

```
K=CALL(-3703%)
```

Före detta anrop måste man emellertid på något sätt beskriva vilken strängvektor som skall sorteras (man vill eventuellt sortera flera olika strängvektorer i samma program). Antag att vi har deklarerat:

```
DIM B$(200)=25,C$(20)
```

Sorteringsrutinen använder sig av en enda variabel, som skall heta P9\$. Denna enkla sträng får sedan beskriva vilken strängvektor som skall sorteras. Antag att vi vill sortera först B\$ och sedan C\$ enligt DIM-satsen ovan. Vi matar då in följande satser:

```
P9$="B$":K=CALL(-3703%)
P9$="C$":K=CALL(-3703%)
```

P9\$ ändras inte vid sorteringen, och behöver alltså bara sättas vid ett tillfälle om man sorterar samma strängvektor flera gånger.

Tillsammans med sorteringsrutinen levereras ett testprogram SORTDEMO, som visar denna enkla tillämpning.

Funktionsvärdet (K) blir 0 om sorteringen gick bra och negativt annars.

Följande fel kan uppstå (K=):

- 1 Det finns ingen variabel i P9\$
- 2 P9\$ innehåller ingen beskrivning, dvs P9\$ har ej tilldelats någon strängvektor
- 3 P9\$ beskriver ingen strängvektor, dvs tilldelningen är felaktig

I avancerade tillämpningar dessutom:

- 4 Försök att sortera fler strängar än vad som finns i vektorn
- 5 Heltalsvektorn är för liten
- 6 P9\$ innehåller en felaktig beskrivning efter strängvektorn
- 7 Tabellen över ny sorteringsordning är ej 64 tecken lång

## 2. AVANCERADE TILLÄMPNINGAR

### 2.1 FULLSTÄNDIGT ANROPSFORMAT

Sorteringsrutinen kan egentligen anropas med fyra olika parametrar enligt följande format:

```
P9$="B$,N%,X%,C$":K=CALL(-3703%)
```

B\$ = En strängvektor som skall sorteras, dvs nycklarna

N% = En heltalsvariabel, som anger antalet använda strängar i strängvektorn B\$

X% = En HELTALSVEKTOR, som efter sorteringen anger ordningsnumret för de olika strängarna i strängvektorn B\$

C\$ = En STRÄNGVARIABEL som definierar den sorteringsordning (koalitionsssekvens) som skall gälla.

Variablerna N%, X% och C\$ behöver ej anges, jfr avsnitt 1. Det är även möjligt att endast ange en av variablerna vid anropet.

Varibelnamnen kan anges fritt.

### 2.2 LÄNGDANGIVELSE AV ANTAL ELEMENT

Många gånger dimensioneras en strängvektor för att rymma fler strängar än vad som normalt används. Man vill då kunna begränsa sorteringen till färre antal strängar i vektorn. Detta antal läggs i en heltalsvariabel och variabeln namges i P9\$. Antag att N% innehåller strängvektorn B\$ aktuella antal strängar i vektorn.

```
P9$="B$,N%":K=CALL(-3703%)
```

Här sorteras endast de N% första strängarna i vektorn.

### 2.3 NYCKELTABELL

Om man vill sortera ett stort antal poster med hjälp av en nyckel, dvs om man vill sortera en del av posten, skall naturligtvis endast nyckeln ingå i posten. Efter sorteringen av dessa nycklar, måste man då veta de till nycklarna motsvarande posternas relativa ordningsnummer. Detta är möjligt genom att samtidigt sortera en heltalsvektor i samma ordning som strängvektorn med nycklarna blir sorterad.

Denna heltalsvektor ger på detta sätt en referens till posternas adresser. Man kan således låta posterna ligga oförändrade och endast använda heltalsvektorn för att ange deras inbördes ordning.

Denna metod är användbar även oavsett om posterna är lagrade i primärminnet eller på skivminnet. I det senare fallet anger heltalsvektorn relativa postnummer.

Antag att vi har skrivit:

```
DIM B$(500)=10,X$(500)
```

Sedan vi överfört 500 nycklar till B\$ skriver vi till exempel:

```
FOR K=1 TO 500:X$(K)=K:NEXT K
```

Vi har nu tilldelat heltalsvektorn X% värdena 1,2,3,.....

Därefter sorterar vi B\$ och X% med:

```
P9$="B$,X%":K=CALL(-3703%)
```

Efter sorteringen anger då först talet i X%, vilken sträng som ska komma först. Det andra talet i X% anger vilken sträng som skall komma som nummer två, och så vidare.

### 2.4 SORTERINGSORDNING

Sorteringsordningen är densamma som för tecken enligt ASCII-kod utom för intervallet 64-127 (stora och små bokstäver). Där gäller följande ordning:

```
AaBbCcDdEeEeFfGgHhIiJjKkLlMmNnOoPpQqRrSsTtUuVvWwXxYyZzÄäÖö*
```

(\* = sista tecknet är CHR\$(127) dvs fylld ruta)

Om två strängar börjar likadant men är av olika längd, sorteras den kortaste först. Om två strängar är helt lika bibehålles den ursprungliga ordningen mellan dem (sorteringen är stabil).

### 2.5 ÄNDRING AV SORTERINGSORDNING

Sorteringsordningen kan, om så önskas, modifieras. Man måste då specificera en ny tabell för intervallet ASCII 64-127. För detta ändamål används en strängvariabel.

Antag att C\$ innehåller denna tabell. Sorteringsprogrammet kontrollerar att tabellen är 64 tecken lång. Med basicprogramet NEWSEQ kan du kontrollera att alla tecken finns med i tabellen och att inga tecken förekommer två gånger. På detta sätt är det till exempel möjligt att definiera sortering i fallande ordning.

### 2.6 DEMOPROGRAMET SORTDEMO

Ladda sorteringsrutinen med RUN QSORT. Sorteringsrutinen laddas då in och ABC80 skrivs ut på skärmen. Kör DEMO-programmet genom att skriva RUN SORTDEMO och trycka på RETURN. Programmet genererar 200 slumpmässiga strängar med längden 3 tecken. Strängarna skrivs ut på skärmen.

Starta därefter sorteringen genom att skriva RETURN. Programmet exekveras och de sorterade strängarna samt tid för sorteringen skrivs ut på skärmen.

Du kan lätt ändra antal strängar för att köra andra exempel än det som finns angivet i demoprogramet.

Antal strängar ändras på rad 10. N% sätts då till önskat antal strängar.

# Om sortering

I datormötet Beställ på ABC-klubbens monitor har framförts önskemål om presentation av olika sorteringsrutiner. På klubbens senaste programskiva finns programmet SORTERA som visar 6 olika sätt med tidtagningsmöjlighet. Programexemplen hänvisar till detta program, om ej annat sägs.

För den som inte kan eller vill jobba med SORTERA-programmet från kassett 16 kan använda bifogade program SORT.BAS som grund. Med MERGE kan man komplettera den med olika sorteringsrutiner och få tidtagning.

## Vad är sortering?

Sortering kallas i datorsammanhang rutiner som ordnar listor eller register i t ex bokstavsordning eller nummerordning efter olika nyckelord, som efternamn, ort eller postnummer.

## Hur gör man?

När man manuellt sorterar en bunt registerkort gör man en rad saker på en gång. Man kommer ihåg vilken bokstav man just nu letar efter och var man senast hade ett kort med den bokstaven. Man kan lägga ut korten i olika högar på bordet, bläddra fram och tillbaka i en kortlåda eller föra anteckningar på ett papper vid sidan om.

Att exakt beskriva detta i ett program är inte lätt.

## Tidigare behandling

Detta ämne tas upp då och då i dator-tidningar. Jag har själv sett två artikelserier i Mikrodatorn (1980:7-8 och 1983:7) till vilka jag hänvisar källforskarna.

Där fick jag Basic-koden till ett antal sorteringsrutiner, men förklaringarna var minst sagt knapphändiga, och rutinerna var ingalunda lättlästa. Av nyfikenhet och intresse ägnade jag ett par semesterdagar åt att komma underfund med hur programmen fungerade. Resultat av dessa efterforskningar redovisas i denna artikel.

För den intresserade ett par tips: Gör små numrerade lappar för talen 1-9. Blanda och lägg ut på bordet, och försök sedan efterlikna datorprogrammets åtgärder. Nio lappar säger kanske mer än tusen ord!

Ett annat sätt att följa sorteringen är att sätta in "pausfåglar" i programmen. Skriv

```
;CUR(23%,0%)I%;
```

eller liknande varje gång den aktuella räknaren ändras. (Men kom ihåg att detta saktar ned sorteringen högst avsevärt!)

## Ordförklaringar och sakupplysningar

Det förutsätts i alla programexempel att det finns ett antal strängar B\$(X) som skall läggas i bokstavsordning. Varje sträng kallas en term. "A" är den "minsta" termen och den skall vara längst "till vänster", och sedan kommer termerna i storleksordning med alla på "Ö" längst "till höger".

Ibland behöver man plocka ut någon av termerna för jämförelse. Denna jämförelse-term placeras i T\$. (Dessa variabelnamn stämmer med SORTERA-programmet.)

I verkliga livet är det ofta grupper av termer som skall följas åt vid sorteringen. En adressuppgift kan ha namn, gatuadress, postadress, telefon - 4 termer som bildar en post, och det är egentligen posterna som skall sorteras.

Algoritm kallas en rutin eller ett program som löser en viss uppgift. I detta fall handlar det om sorteringsalgoritmer.

Alla sorteringsalgoritmer arbetar med tre underrutiner:

Byte  
Tilldelning  
Jämförelse

## Nu går MINIKALKYL även på IBM PC ... MINI-PROGRAM för

ABC 80, ABC 800, FACIT DTC och IBM PC (och kompatibler)

### MINIKALKYL

Kr 525:--

Ett robust och lättanvänt kalkylprogram av matristyp där du själv bygger upp din beräkningsmodell i ett rutnät (för 40/80 tecken). Lättredigerad utskrift av kalkylen.

### MINIKALKYL 2

Kr 675:--

Kan utöver vad MINIKALKYL kan bl.a. ...  
... att samköra delkalkyler till ett antal huvudkalkyler. Praktiskt taget obegränsad uppbyggnad och lagring av data.  
... att tillsammans med MINITEXT redigera vackrare och fylligare utskrifter av hela eller delar av en kalkyl oberoende av hur kalkylen ser ut i rutnätet.

### MINITEXT

Kr 525:--

Lättanvänt textbehandlingsprogram med obegränsad dokumentlängd och en mängd nyttiga funktioner och kopieringsmöjligheter (för 40/80 tecken).

### ADRESS

Kr 350:--

Adressregister med sökning på valfritt begrepp, utskrifter på lista, kuvert eller etiketter (för 40/80 tecken).

### FAKTURA

Kr 725:--

Faktureringsprogram med kundregister och fakturajournal samt rutin för utskrift på fakturablankett (för 80 tecken, 32 Kb).

Samtliga program för kronor 1.750:--.

### KONVERTERING

av disketter och program till IBM PC-miljö hjälper vi gärna till med. Begär offert.

Priser inkl. frakt och moms. 15% rabatt till enskild ABC-medlem. (ej företag och skolor)

Ring EEA HB, 08/768 80 08

## Disketter – specialpriser

5''DP ES 40 spår 6:—

5''DP DS 80 spår 9:90

8''DP 11:—

Min. 10 st/beställning

Priserna är inkl. moms plus frakt och postförskottsavgift.

Vid betalning på pg 64 18 12-3

bjuder vi på frakten.

Beg. BASF med 3 mån. garanti.

E-PROM-programmerare

2758-27256 E-PROM

Ring för prospekt

**F&G** ORSDATA ARDELL AB

TELEFON:  
0141-165 00

**Bytet (ca 3 millisekunder (ms))**

Om två termer skall byta plats måste man först flytta på den ena. Eftersom den måste vara någonstans, placerar vi den i en "slaskterm" T\$. Nu är dess plats ledig och där kan vi placera den andra termen. Slutligen låter vi den första termen gå från T\$ till den andra termens plats.

$$T\$ = B\$(1\%):B\$(1\%) = B\$(K\%):B\$(K\%) = T\$$$

Denna rutin är hjärtat i varje sorteringsalgoritm. Men eftersom detta byte tar ganska lång tid bör man söka klara sig med så få som möjligt. Om det är poster med många termer som skall sorteras kan tidsåtgången för varje byte vara den helt dominerande faktorn när det gäller sorteringshastigheten.

**Tilldelningen**

Denna kan vara en tredjedel av bytet:  $T\$ = A\$(1\%)$  som separat tar drygt 1 ms. Men hit hör också uttryck som  $J\% = K\%$  (0.44 ms) och räknaren  $L\% = L\% + 1\%$  (0.54 ms).

**Jämförelsen**

IF A\$(I%) < A\$(J%) THEN ... tar ca 1.3 ms medan

IF L% < K% THEN...tar endast 0.88 ms

**Sorteringsalgoritmerna**

Nu är vi mogna för en titt på de olika sorteringsalgoritmerna. Om inget annat sägs syftar jag i texten på SORTERA-programmets rutiner.

**Bubbelsortering**

Detta är den vanligaste och kortaste av alla sorteringsalgoritmerna. Det är alltid den som förekommer i elementära läroböcker, eftersom den är enklast att beskriva och förstå och dessutom har den enklaste koderna.

"Som sorteringsmetod är den tämligen värdelös, men den har ett pedagogiskt principiellt intresse just därigenom att den är så enkel och enfaldig", skrev Dan Jonsson i Mikrodatorn 1983.

I korthet kan man säga att den letar rätt på den minsta termen och placerar den längst till vänster. Den fortsätter med den näst minsta osv tills alla termer är insorterade.

Bubbelsortering har fått sitt namn av att den minsta termen "bubblar upp" till ytan (längst till vänster).

Man börjar med att låta första termen vara jämförelseterm och jämför den med var och en av de efterföljande. Så fort man hittar en mindre term, byter den plats med första termen och blir ny jämförelseterm. När man gått igenom hela lagret finns den minsta termen längst till vänster.

Sedan gör man om proceduren med term nummer 2, 3 osv tills alla är genomgångna.

Denna metod är alltså enkel och kort. Om det inte är mer än hgst 50 termer att sortera går det på ett par sekunder.

Men sorteringstiden växer med kvadraten på antalet termer, eller annorlunda uttryckt: har man dubbelt så många termer tar det fyra gånger så lång tid. Tio gånger fler termer tar hundrafalt längre tid.

Sortering av 100 termer kan ta omkring 15 sekunder och 1000 termer kan ta en halvtimme!

Vi måste hitta snabbare metoder.

**Onödiga byten**

En analys av vad som händer i programmet visar att det sker ganska många onödiga byten. Varje term kommer till jämföras med varje annan term, och det blir byte varje gång en mindre term påträffas.

Skall man spara tid måste man i första hand minska antalet byten, i andra hand minska antalet jämförelser.

**Urvalsmetoden**

Denna är i grunden densamma som bubbelmetoden, men man minskar ner bytandet en smula. Man lagrar jämförelsetermen i T\$ och börjar jämförandet. Hittas en mindre term kan den omedelbart lagras i T\$ i stället för den tidigare, som inte behöver sparas, eftersom alla termerna ligger kvar på sina ursprungliga platser. För att komma ihåg var man hittade den mindre termen, spar man dess index i K%.

Först när hela raden är genomgången gör man det fullständiga bytet. T\$ hamnar på första platsen och den tidigare första termen intar dess plats i raden K%.

Det blir alltså bara ett fullständigt byte. De andra ersätts av tilldelningar i T\$. Denna manöver spar i genomsnitt 30% av sorteringstiden.

I SORTERA-programmet finns en metod som kallas DELAYED REPLACEMENT (fördröjt utbyte), som egentligen gör precis samma sak som urvalsmetoden. Skillnaden i den befintliga programkoden är bara att man inte använder T\$ som jämförelseterm, utan placerar den för tillfället minsta termens index i J2%. Man måste ha ett antal räknare (M2% och L2%) för att hålla reda på var man är i raden.

Resultatet har blivit så olyckligt att denna metod är en aning långsammare än urvalsmetoden, trots att man sparar in en rad strängtilldelningar.

**Se upp med räknarna!**

Man får tydligen vara försiktig med sina räknare. Även en operation som  $J = J + 1$  eller  $J = L$  tar sin tid. Den snabbaste räknaren är FOR-NEXT-slingan. Om man skriver om urvalsmetoden med två sådana slingor i stället, ökar sorteringshastigheten med ytterligare en tredjedel.

Bifogade programexempel URVAL SW kan sättas in i stället för DELAYED REPLACEMENT och visar sig gå nästan på halva tiden i jämförelse med bubbelmetoden!

**Insticksmetoden**

Denna fungerar på liknande sätt (se bifogat programexempel som kan sättas in i stället för BATCHER). Här arbetar man "baklänges" så att man lägger den aktuella jämförelsetermen i T\$ och jämför med föregående term. Om denna är större, flyttas den fram ett steg och man numrerar om räknaren så att samma procedur kan upprepas tills man hittar en term som är mindre än jämförelsetermen T\$. Då "sticker man in" T\$ på den aktuella platsen.

Första jämförelsetermen är term 2, sedan 3, 4 osv. Här har man nytta av den ordning som redan finns bland termerna. Det har man ju inte vid bubbelsorteringen, som skickar omkring för tillfället icke aktuella termer litet hur som helst, även om de från början stod nära sina rätta platser.

Bäst går insticksmetoden om termerna redan är sorterade och man vill sticka in en ytterligare term. Då går det undan värre! Inga onödiga jämförelser, inga onödiga byten - den perfekta sorteringsalgoritmen.

**Överlägsen metod?**

Egendomligt nog visar sig insticksmetoden i det här aktuella fallet inte så överlägsen min variant av urvalsmetoden som jag hade väntat mig. Kanske beror det på att den endast har en slinga av FOR-NEXT-typ medan den andra (rad 450) har en vanlig räknare. Det kanske också blir för många strängtilldelningar på osorterat material? Ett intressant mysterium för den intresserade att lösa.

**Onödiga byten**

Med URVAL och INSTICK sparar man in ett antal onödiga byten. Men fortfarande jämförs alla termer med alla. Antalet jämförelser ökar med kvadraten på antalet termer, varför tiden för sorteringen ökar i stort sett lika fort som med bubbelmetoden.

Har man mer än ett hundratal termer är det nödvändigt med radikalare grepp.

**Tolka koderna**

Det är inte lätt att tolka koderna för de mer avancerade metoderna. Det beror framför allt på att datorn måste ha så detaljerade instruktioner för att "komma ihåg" var de poster finns som skall jämföras eller bytas.

Man måste ha minnesvariabler för minsta och största term och för aktuell jämförelseterm. Vidare behövs pekare för den del av sorteringen man är klar med och andra för de termer man just håller på med.

**Förstå och uppskatta**

Kanske man måste vara matematiker för att riktigt förstå och uppskatta de teoretiska finesserna med de mest avancerade sorteringsalgoritmerna, men även en vanlig hempulare kan ha viss glädje och nytta av att i stora drag få reda på hur det kommer sig att de går fortare.

**Shellsortering**

Shellsortering har fått sitt namn efter SHELL som uppfann den. Idén är att minska sorteringstiden genom att låta termerna komma till rätt position genom så få byten som möjligt. För att uppnå detta låter man dem börja med "långa hopp".

Det sker genom att man först jämför första och mittersta termen och byter så att den minsta kommer till vänster. Sedan gör man likadant med alla efterföljande par.

Har man från början 8 termer, jobbar man alltså med 1 och 5, 2 och 6, 3 och 7 samt 4 och 8.

Nu har vi fyra färdigsorterade listor om vardera två termer.

I nästa omgång delar vi termerna i fyra delar som jämförs två och två. Vi arbetar först med 1 och 3, sedan 2 och 4, 3 och 5 osv.

Här tillkommer att ifall man behöver byta mellan 5 och 7 sker omedelbart en ny jämförelse mellan 3 och 5, och därefter mellan 1 och 3 ända tills man hittar en mindre term. Då behöver man inte jämföra mer, eftersom man vet att de övriga ligger i ordning. När hela mängden är genomgången har vi två sorterade listor om vardera 4 termer.

Slutligen jämför vi nu 1 med 2, 2 med 3 och. Precis som förut leder varje byte till att man fortsätter jämförelsen "åt vänster" tills alla bitar ligger rätt. Men det blir inte många fler byten, eftersom grovgörat redan är klart och ordningen ganska god.

Shell-sortering går märkbart fortare än de tidigare nämnda redan med 50 termer. Om man skriver den med en FOR-NEXT-slinga (programexemplet) snabbar man ytterligare omkring 20%, och då klarar den 600 termer på ungefär samma tid som urvalsmetoden (SW) klarar 200!

Matematikerna har räknat ut att tiden är proportionell mot antalet termer upphöjt till 3/2. I praktiken kommer vid dubbelt så många termer sorteringstiden att bli omkring 2.5 gånger längre (mot 4 gånger längre för bubblsortering).

Shell-sorteringen har alltså klara fördelar. Koden är inte alltför omfattande, tiderna kortare. Dessutom förlorar man inte märkbart med tid för "organiseringen" av sorteringen. Jag tänker på alla pekare och hjälpvariabler som behövs för mera avancerade sorteringsmetoder. Shell hävdar sig mot bubbel och instick även för ett litet antal termer.

Med mycket stora postmängder blir dock varje ytterligare tidsvinst värdefull. De kloke har därför spekulerat ut nästa metod:

#### QUICK-SORT

Den vane hackaren ser med en gång att QUICKSORT har 9 bokstäver, dvs en för mycket för att få rum i ett filnamn. Förkortningarna är många och leder den fåkunnige in i osäkerhet om hur ordet egentligen skall stavas - så ta en god titt på rubriken. (Där stod det rätt när jag skrev detta...)

En vanlig förkortning är Q-sort som jag skall använda här.

För att ytterligare öka sorteringshastigheten behövs mycket radikala medel. Det gäller dels att få bort onödiga byten och att dessutom kraftigt nedbringa antalet jämförelser.

Det första söker man uppnå genom att göra så "långa" hopp som möjligt, det andra genom att inte behöva på nytt jämföra de termer som man efter första jämförelsen vet sitter rätt.

Som var och en ser innehåller Quicksort-algoritmen i SORTERA-programmet ett virrvarr av pekare och hjälpvariabler, som det är mycket svårt att ta sig igenom när man skall tolka den för att se vad som egentligen händer.

Det beror på att Q-sort egentligen är en rekursiv formel, dvs den anropar sig själv flera gånger under sorteringslopp. Detta framgår tydligt om man i stället skriver koden i Basic II (se QSORT.800). Om man tar fram de nummerlappar jag rekommenderade tidigare och försöker följa anvisningarna i Basic II-formen kanske det är lättare att se vad som händer.

I korthet kan man säga följande. Välj ett jämförelseelement T\$. Börja sedan "från vänster" och jämför med term efter term tills det kommer en term som är större än T\$. Denna hör hemma längre till höger i raden. Denna terms läge markeras med en pekare.

Börja sedan igen, denna gång från höger, och leta efter en term som är mindre än T\$. Denna hör hemma längre till vänster, och nu låter man de båda markerade termerna byta plats.

(Observera att man inte kan använda T\$ som slaskvariabel vid bytet, eftersom den innehåller en jämförelseterm som skall användas igen. Jag har använt Y\$ till slask, utom i QSORT.800 där X\$ blev jämförelseterm när jag skrev av från ursprungsversionen - se nedan.)

När det är klart fortsätter man på samma sätt från de båda pekarna in mot mitten

och gör så många byten man kommer åt innan pekarna möts.

Därefter väljer man ett nytt jämförelseelement och upprepar proceduren.

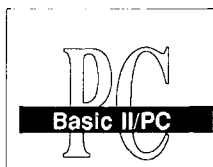
Den springande punkten är att välja jämförelseelement så att med tiden elementen kommer i korrekt ordning från vänster. Det gäller att hålla reda på vilka termer som ligger rätt och inte behöver undersökas ytterligare. På detta sätt minskar man drastiskt antalet jämförelser, även om vinsten av detta märks först vid ett större antal poster.

#### Bättre i BASIC II än i COMAL?

Anledningen till att det blir en redigare algoritm med BASIC II är framför allt att WHILE-WEND slingan minskar behovet av pekare, och de två sista raderna som på nytt anropar den användardefinierade funktionen ersätter en skog av hjälpvariabler.

Jag har förresten knyckt algoritmen från Skolan och datorn 3/85, där den är skriven i Comal. Man skryter där med att Compis Comal är så snabb att man klarar att sortera 5000 heltal på 7 minuter med denna Q-sort. Jag kan inte neka mig nöjet att konstatera dels att rutinen nästan utan ändringar kunde överföras till Basic II, dels att den då (på min ABC80 med Basic II i RAM) går på 5 minuter 29 sekunder... Man börjar undra om skolpolitikernas stora satsning på en helt ny skoldator var så nödvändig. Finessen att ersätta BASIC med COMAL som datorspråk i skolan blir alltmer tveklaktig.

# Basic II/PC är här!



Med den stora bas som ABC-datorerna är för applikationer skrivna i Basic II har nu denna välkända produkt tagit steget till MS/PC-DOS. Basic II/PC har utöver en rad nyheter alla de gamla välkända funktioner-

na och är som vanligt en snabbis när det gäller att bearbeta program. Basic II/PC är som tidigare skriven i assembler.

Till dig som är medlem i ABC-klubben erbjuder vi dig att köpa denna nya produkt till ett mycket förmånligt pris. Vi drar helt enkelt av din medlemsavgift (160:-) från det ordinarie priset på 1.950:- vilket innebär att du betalar 1.790:-. Vill du ha options-paketet så ger vi dig ytterligare 100:- rabatt på detta. Istället för normalt 500:- betalar du bara 400:-. Moms tillkommer.

Beställning sker skriftligen genom ABC-klubben och erbjudandet är enbart öppet för medlemmar. Sänd alltså in kupongen till ABC-klubben. Leveranstiden är ca två veckor från det att klubben har fått din beställning.

**DATA  
INDUSTRIER AB**

Box 2029, 183 02 Täby.

#### Exempel på nya funktioner:

- o använder 8087 flyttals-processor om sådan finns maskinen.
- o definiera dina egna funktionstanger
- o sätt start-up parametrar i fil
- o sätt upp din egen prompt
- o CD - CHDIR - förflytta dig i biblioteks-strukturen
- o RMDIR-MKDIR - ta bort/lägg till ibibliotek
- o STAT - programstatus
- o CAT - lista katalog över .bac-filer
- o SCAN - lista rader med visst innehåll
- o EOF - testa på filslut
- o CLS - rensa bildskärm
- o CRT(x) för att sätta 'bold, reverse, underline, flash' mm
- o CRT(x) - sätta bak/förgrunds-färg på färgskärm
- o FIELD - omdefiniering av strängar till 'sub-strängar'
- o REQUEST - använd MS-DOS systemanrop
- o DEF SEG - sätt segmentpekare
- o Optioner: ISAM, ABC806 färggrafik kommandon, Squeeze, Cross

Jag beställer härmed

- ☐ Basic II/PC specialpris 1.790:-  
☐ Optionspaket specialpris 400:-

Namn \_\_\_\_\_

Adress \_\_\_\_\_

Postadress \_\_\_\_\_

Medlemsnummer \_\_\_\_\_

SÄND KUPONGEN TILL ABC-KLUBBEN

**Pekarna igen?**

Nu är det faktiskt inte så stor vinst att köra Q-sort med koden i SORTERA-programmet - det går inte fortare än med den snabbare varianten av Shell (jag har provat upp till 600 termer).

Av nyfikenhet letade jag fram versionen från Mikrodatorn (se programexempel). I den visade versionen kan den MERGEas in sist i sorteringsprogrammet och köras om man på rad 230 ändrar sista hopp-raden till 1080 (eller bättre 1180).

Den rutinen visar sig vara nästan 20% snabbare, men jag har inte närmare undersökt varför. Något skall väl vara kvar även för den vetgirige läsaren att göra?

Vad beträffar Basic II-varianten har jag inte jämfört den med de andra på samma material. Men SORT-rutinen med tillägg av ett par rader som läser klockan i ABC800 (anpassad för ABC80) tycks visa att i Mikrodatorns QSORT går en hårsman fortare än den rekursiva funktionen. Skillnaden är dock härfin (3 sekunder av 50 vid 600 termer).

**Kan köras i .BAS-format**

Man kan faktiskt köra alla de här visade programmen i alla ABC800-varianterna, om man laddar in dem i .BAS-form. Det kunde vara intressant att höra resultatet om någon försökt jämföra rutinerna i den större maskinen.

**Vilken sorteringsmetod är bäst?**

Det kan man inte svara på utan vidare. Om man inte har så många termer att sortera går även bubblsorteringen så fort att man knappast blir avbruten. Eftersom den har enklaste koden kan den kanske duga lika bra som de andra.

Om de poster man jobbar med i stort sett är färdigsorterade och bara någon eller några enstaka nya tillkommer mellan varje sortering, kan insticksmetoden vara något att fundera på.

Har man en mycket stor mängd färdigsorterade poster lönar det sig med sekvensiell sökning efter den plats där den nya skall stickas in. Man börjar alltså inte i ena ändan av raden, utan i mitten, och ser om man kommit för högt eller för lågt osv. Då behövs bara 10 jämförelser för att hitta rätt bland 1000 poster.

Programexemplet BINÄR INSTICKSSORTERING bygger på en rutin från Mikrodatorn 1983, som dock friserats för ökad snabbhet. Där delas rutinerna upp i tre delar: först kollar om T\$ behöver flyttas, sedan söker man rätt plats för T\$ och sist gör man omflyttning av alla aktuella termer. Det är i mittdelen som binär sökning kommer till användning. Metoden speedar visserligen upp insticksmetoden 30%, men är kanske ändå inte så praktisk i förhållande till resultatet. För den idoge läsaren kan koden ha visst pedagogiskt värde på samma sätt som bubblsorteringen. Möjligen kan den vara fördelaktig om man i en redan sorterad mängd (flera hundra termer) skall sortera in ytterligare en. Detta har jag dock inte provat.

Jag upptäckte under loppet av mina undersökningar att om man använder de här visade metoderna på en redan sorterad post-

mängd, bläddrar insticksmetoden genom mängden med rekordhastighet, och Q-sort är en av de långsammaste! Då har man ingenting för all exercis med pekarna, som bara tar en massa onödig tid!

Detta gäller dock bara okomplicerade poster med bara en term i varje. Ju mer det finns i en post, desto mer tidskrävande blir både byte och tilldelning. En sorteringsalgoritm som har få postförflyttningar blir då mycket lönsam.

I detta fall, och där det finns ett stort antal poster, verkar Q-sortering vara den överlägsna och kan löna sig trots att koden är invecklad, tar stor plats och använder ganska mycket tid och utrymme till "bokföring" av posterna med pekare och hjälpvariabler av många slag.

**Det finns fler**

Det finns fler metoder än dessa. I SORTERA-programmet visas BATCHER-metoden, som inleds med ett till synes mycket invecklat sätt att välja vilka poster som skall jämföras.

Den bygger på någon teori om maximalt lämpligt avstånd mellan termerna för de första "hoppen". Det lär inte vara helt utrett vilka avstånd man skall ha för snabbaste sortering, men man har intresserat sig för tal av typen (2 upphöjt till P) minus 1 som för P=10 ger 1023, 511, 255 osv - en inte helt okänd serie för datorhobbyister.

Även Shell-sorteringar finns som använder sådana talserier i stället för att ta hälften av antalet termer i första omgången.

Eftersom BATCHER-sorteringen går avsevärt långsammare än SHELL (SW)-varianten, har jag inte känt mig motiverad att göra en närmare analys av den.

**Finter och finesser**

I kommersiella sorteringsprogram - som vanligtvis finns som delrutiner i t ex registerhanterare - används åtskilliga extra knep. Ett vanligt är att inte sortera posterna i sig själva, utan i stället en fil som talar om i vilken ordning posterna står. Fördelen är att denna fil, som är numerisk, kan sorteras mycket fortare än de verkliga posterna. När man sedan skall söka en post får man titta i sorteringsfilen var den står.

**Ännu snabbare**

Ännu snabbare går det givetvis om man överför sina sorteringsrutiner till maskinspråk, som är minst 100 och ibland 1000 gånger snabbare än BASIC. Det har jag inte haft tillfälle att testa.

Helt klart finns det i denna sak fortfarande gott om utrymme för någon annan att skriva sin vers. Min slutar här.

**Tillägg**

Om du inte vill använda SORTERA-programmet på ABC-kassett 16 ges här en enkel rutin att prova med, SORT.BAS.

Skriv in den och MERGEa sedan med den sorteringsrutin du vill undersöka. Sätt in rätt radnummer efter GOSUB på rad 200 ifall du inte gillar min lilla nödlösning.

```
405 REM ++++++
410 REM ++ URVAL ++ SW vers
415 REM ++++++
420 FOR I%=2% TO N%
430 T$=B$(I%)
440 L%=I%
450 IF B$(L%-1%)>T$ THEN B$(L%)=B$(L%-1%)
   : L%=L%-1% : GOTO 450
460 B$(L%)=T$
470 NEXT I%
480 RETURN
```

```
405 REM ++++++
410 REM ++ INSTICKSSORTERING ++ SW
415 REM ++++++
420 FOR I%=2% TO N%
430 T$=B$(I%)
440 L%=I%
450 IF B$(L%-1%)>T$ THEN B$(L%)=B$(L%-1%)
   : L%=L%-1% : GOTO 450
460 B$(L%)=T$
470 NEXT I%
480 RETURN
2000 REM binär insticksortering
2010 REM -----
2020 FOR K%=1% TO N%-1%
2030 T$=B$(K%+1%)
2040 IF T$>B$(K%) THEN 2080
2050 R%=1%
2060 GOSUB 3000 : REM stick in T$
2070 GOTO 2110
2080 IF T$>B$(K%) THEN 2110
2090 GOSUB 2140
2100 GOSUB 3000
2110 NEXT K%
2120 RETURN
2125 REM -----
2130 REM ---sök plats för T$
2140 L%=1%
2150 R%=K%
2160 GOTO 2220
2170 M%=(R%+L%)/2%
2180 IF T$<B$(M%) THEN R%=M% ELSE L%=M%
2220 IF R%-L%>1% THEN 2170
2230 RETURN
2240 REM -----
3000 REM --- stick in T$
3010 FOR J%=K% TO R% STEP -1%
3020 B$(J%+1%)=B$(J%)
3030 NEXT J%
3050 B$(R%)=T$
3060 RETURN
```

```
555 REM ++++++
560 REM ++ SHELL-SORTERING ++ SW version
565 REM ++++++
570 M1%=N%
575 M1%=M1%/2% : IF M1%=0% THEN RETURN
580 K7%=N%-M1%
585 FOR J1%=1% TO K7%
590 I1%=J1%
595 IF B$(I1%)>B$(I1%+M1%) T$=B$(I1%)
   : B$(I1%)=B$(I1%+M1%) : B$(I1%+M1%)=
   : T$ : I1%=I1%-M1% : IF I1%>=1% THEN 5
   : 95
600 NEXT J1%
605 GOTO 575
```

&lt;1384&gt;

Sven Wickberg

```

1080 REM ...SAVE QUICKSORT...82 02 22
1090 ; 'SORTERING MED QUICK-SORT'
1100 REM Ur Mikrodatorn 80/8
1110 REM
1120 REM B$( )=listans element
1130 REM N% =antal element
1140 REM I%,K%=index
1150 REM T$ =jämförelseelement
1160 REM F% =flagga
1170 REM R%,L%=indexgränser
1180 DIM L$(N%),R$(N%),F$(N%)
1190 F%=1% : L$(1%)=1% : R$(1%)=N%
1200 L%=L$(F%) : R%=R$(F%) : F%=F%+1%
1210 I%=L% : K%=R%
1220 T$=B$(L%)
1240 IF B$(I%)>T$ THEN 1250 ELSE I%=I%+
1% : GOTO 1240
1250 IF B$(K%)<=T$ THEN 1260 ELSE K%=K%-
1% : GOTO 1250
1260 IF I%>K% THEN 1290
1270 Y$=B$(I%) : B$(I%)=B$(K%) : B$(K%)=
Y$
1280 I%=I%+1% : K%=K%-1%
1290 IF I%<=K% THEN 1240
1300 REM ...ändra index
1310 IF K%-L%>R%-I% THEN 1360
1320 IF I%>=R% THEN 1340
1330 F%=F%+1% : L$(F%)=I% : R$(F%)=R%
1340 R%=K%
1350 GOTO 1390
1360 IF L%>=K% THEN 1380
1370 F%=F%+1% : L$(F%)=L% : R$(F%)=K%
1380 L%=I%
1390 IF L%<R% THEN 1210
1400 IF F%>0% THEN 1200
1410 RETURN

```

```

10 ! qsort
20 ; 'Hur många'; : INPUT N
30 DIM B$(N)=3
40 ;
50 FOR I=1 TO N
60 B$(I)=CHR$(25*RND+65)
70 ; B$(I) ' ';
80 NEXT I
90 ;
100 ! sortera -----
110 Z=FNSort(1,N)
120 ! skriv ut -----
130 FOR I=1 TO N
140 ; B$(I) ' ';
150 NEXT I
160 ;
170 END

```

```

10 REM sort
11 REM program för att pröva sortering
12 REM av trebokstaviga ord.
20 ; 'Hur många ord'; : INPUT N%
30 DIM B$(N%)=3%
40 REM skapa orden -----
50 FOR I%=1% TO N%
60 B$(I%)=CHR$(25%*RND+65%,25%*RND+65%,
25%*RND+65%)
70 ; B$(I%) ' ';
80 NEXT I%
85 ;
90 REM tidtagning -----
100 POKE 65008%,255%,255%,255%
110 GOSUB 200
120 T=(PEEK(65008%) XOR 255%)/50+5.12*(P
EEK(65009%) XOR 255%)
130 ; 'Sorteringstiden ' ; T' sekunder'
140 ; ; '(Tryck RETURN för att se den
sorterade listan)' : GET W$
145 REM utskrift -----
150 FOR I%=1% TO N%
160 ; B$(I%) ' ';
170 NEXT I%
180 END
200 REM

```

```

10 REM QSORT.800 = Qsort i Basic II
1499 REM Q-sort -----
1500 DEF FNSort(L,R)
1510 I=L : J=R
1520 X$=B$((L+R)/2)
1530 WHILE I<=J
1540 WHILE B$(I)<X$
1550 I=I+1
1560 WEND
1570 WHILE X$<B$(J)
1580 J=J-1
1590 WEND
1600 IF I<=J THEN T$=B$(I) : B$(I)=B
$(J) : B$(J)=T$ : I=I+1 : J=J-1
1610 WEND
1620 IF L<J THEN Z=FNSort(L,J)
1630 IF I<R THEN Z=FNSort(I,R)
1640 RETURN 0
1650 FNEND

```

ABC-bladets redaktion  
behöver hjälp med inknappning av bandinspelningar av årsmötesförhandlingarna och  
den paneldiskussion som hölls efteråt.

Hör av dig snarast!

Rent praktiskt går det till så att Du får en kassett med delar av förhandlingarna  
och det vi önskar få tillbaka är en maskinläsbar version.  
Även kassettbaserade utrustningar är utmärkta för denna uppgift.  
Vi kan nog tänka oss en mindre gengäva som erkänsla för nedlagd möda.

Hör av dig snarast till

"ÄMF"  
ABC-bladet  
Vidängsvägen 1  
161 33 BROMMA

FÖR LUXORS  
ABC-DATORER:

# LUX-NET

som  
**ALLTERMINAL**  
med central  
kommunikation  
som nås från  
alla terminalerna.

IBM 2780/3780 eller  
asynkron kommunikation  
från lokal terminal.

Centralskrivare

**LUX-NET CENTRAL**

- 52 Mbyte winchester
- bandstreamer
- floppy

Fungerar även som styrenhet för flera kommunikationsprotokoll samtidigt!

Tex:

- IBM 3270 SNA/SDLC
- IBM 3270 BSC
- Univac UTS 4000
- X21 Datex
- X25 Datapac

ABC

*Vi kan nätverk!*

**T-D-X** Box 227  
191 23 Sollentuna  
Sollentunav 225  
T-D-X DATORER AB 08-92 03 30

**ABC CENTER** Stockholm  
Kungsgatan 79  
08-50 68 75

**ABC CENTER** Gävle  
Engelbrektsg 2  
026-10 33 55

# OUT-kommandon för tangentbordet ABC99

Koderna sänds från BASIC till tangentbordet som "OUT 34,Kod".

| Kod       | Funktion                                      |
|-----------|-----------------------------------------------|
| 0000 0111 | alarm-signal                                  |
| 0000 0001 | slå av larmfunktion                           |
| 1000 0001 | på        "-"                                 |
| 0000 0010 | slå av kort tangent-klick                     |
| 1000 0010 | på        "-"                                 |
| 0000 0100 | slå av långt        "-"                       |
| 1000 0100 | på        "-"                                 |
| 0000 0101 | slå av repetering                             |
| 1000 0101 | på        "-"                                 |
| 0000 0110 | ASCII-mode (normalt)                          |
| 1000 0110 | UP/DOWN-mode                                  |
| 0000 1000 | CAPS LOCK normalt                             |
| 1000 1000 | CAPS LOCK mode                                |
| 0000 1001 | INS normalt                                   |
| 1000 1001 | INS mode                                      |
| 0000 1010 | ALT normalt                                   |
| 1000 1010 | ALT mode                                      |
| 0001 1000 | sänd identifikationskod *1                    |
| 0011 1001 | skriv data till musens positions-<br>register |
| 0011 1010 | skriv data till musens prescale<br>register   |
| 0011 1011 | skriv data till musens gränsregister          |
| 0011 1100 | läs data från musen                           |
| 0011 1110 | läs data från musen auto mode                 |
| 0011 1111 | slå av auto mode för musen                    |

```

0xxx 0000 tänd  lysdiod  x  (1=000, 2=001,
                        3=010 osv)
1xxx 0000 släck  "-"  x

```

**\*1 Identifikationskod**

Tangentbordet svarar med 7 bytes:

| Byte | Kod       | Betydelse                |
|------|-----------|--------------------------|
| 1    | 0001 1011 | ESC                      |
| 2    | 001x xxxx | nationalitet version     |
| 3    | 001x xxxx | LED-typ                  |
| 4    | 0010 xxxx | LED's 5-8 1=tänd         |
| 5    | 0010 xxxx | 1-4                      |
| 6    | 0010 0xxx | INS, ALT, CAPS LOCK      |
| 7    | 0010 0000 | 1=påslagen<br>spare byte |

| Version   | (byte 2)  |
|-----------|-----------|
| 0010 0001 | Svenskt   |
| 0010 0010 | Norskt    |
| 0010 0011 | Danskt    |
| 0010 0100 | USA       |
| 0010 0101 | Engelskt  |
| 0010 0110 | Spanskt   |
| 0010 0111 | Franskt   |
| 0010 1000 | Tyskt     |
| 0010 1001 | Isländskt |

| Tgb-type  | (byte 3)             |
|-----------|----------------------|
| 001x x001 | ABC 55               |
| 001x x010 | ABC 77               |
| 001x x011 | ABC 99               |
| 001x 1xxx | UP/DOWN-mode         |
| 001x 0xxx | ASCII-mode (normalt) |
| 0011 xxxx | Mus ansluten         |
| 0010 xxxx | ej ansluten          |

## ABC99 MUS

Tangentbordet avkodar pulserna från musen och håller reda på dess position. Följande gäller både för x- och y-förflyttning. Puls-erna från musen är ansluten till ett pre-scale register. När skalfaktorn är nådd så adderas/subtraheras increment registret till/från positions registret. Om musen går utanför gränserna uppdateras inte positionsregistret förrän man ändrar riktning.

**Kodning:**

Alla register är 12 bit långa. De delas upp i två bytes med off-set 20H vid sändning till/från tangentbordet. Varje byte har ett värde mellan 20H till 5FH. Den högsta byten sänds alltid först. Puls-jitter måste filtreras. Även tryckknapparna har en off-set av 20H.

| Register    | Initialvärde (hex) |
|-------------|--------------------|
| x-position  | 000                |
| y- "        | 000                |
| x-min       | 000                |
| x-max       | FFF                |
| y-min       | 000                |
| y-max       | FFF                |
| x-scale     | 001                |
| y- "        | 001                |
| x-increment | 001                |
| y- "        | 001                |

**Sätta register:**

**Position:**

0011 1001, x-pos hi, x-pos low, y-pos  
hi, y-pos low

**Scale & increments:**

0011 1010, x-scale hi, x-scale low, y-scale hi, y-scale low, x-increment hi, x-inc low, y-increment hi, y-inc low

**Limits:**

```
0011 1011, x-min hi, x-min low, x-max
hi, x-max low, y-min hi,
y-min low, y-max hi, y-max
low
```

**Avläsning av musen:**

Det finns två sätt att läsa data från musen.

- När koden 0011 1100 sänds till tangentbordet, så svarar den EN gång med:

```
1001 0000, x-pos hi, x-pos low, y-pos
          hi, y-pos low, knappstatus
```

Ingen tangent får avbryta sekvensen.

- När koden 0011 1110 sänds till tangentbordet, så svarar den med:

1001 0001, x-pos hi, x-pos low, y-pos  
hi, y-pos low,

1001 0010, knappstatus

Efter denna första sekvens så sänds positionsregister eller knappstatus endast när dessa förändras. De kan sändas separat. Efter varje sekvens (6 alt. 3 bytes) kan tangentbordet sända tangentkoder.

För att avsluta denna auto read kommando, sänds koden 0011 1111 till tangentbordet.

**Mus-kontaktens funktioner (15 pol. D-sub):**

| Pin  | Funktion                                      |
|------|-----------------------------------------------|
| 1    | 5V/50 mA                                      |
| 2    | XA                                            |
| 3    | XB                                            |
| 4    | YA                                            |
| 5    | YB                                            |
| 6    | Vänstra knappen                               |
| 7    | Mitt "                                        |
| 8    | Högra "                                       |
| 9-13 | NC                                            |
| 14   | Mus-indikator, aktiv låg när mus är ansluten. |
| 15   | Jord                                          |

Överföringshastigheten ökar från 600 Baud till  $\geq 4800$  Baud när musen är ansluten.

**Koder för plattan på ABC99:**

Koderna som plattan avger kan avkodas på följande sätt:

bit

```

7  6  5  4  3  2  1  0
!  1  !  0  !  1  !  s  !  x  !  x  !  y  !  y  !

```

```

!      !      !      !      !
----!----      !      !      !      !      --- Upp/Ned
      !      !      !      0      1      Upp
      !      !      !      1      1      Ned
      !      !      !
      !      !      --- Höger/Vänster
      !      0      1      Höger
      !      1      1      Vänster
      !
      !      ----- SHIFT
      !
      ----- Kod för plattan

```

----- Kod för plattan

Exempel:  
1010 1101 = uppåt, vänster  
1011 0100 = höger, SHIFT nedtryckt

<4842>

**Gerry Eriksson**

# Noteringar till kassett nr 16

Jag har nyligen mottagit kassett nr 16 och har efter att jag gjort några provkörningar ett par frågor och kommentarer som jag tror kan vara av intresse även för andra klubbmedlemmar.

1. Det intressanta programmet SORTERA bildar först med ABC80:s slumpvalsgenerator ett önskat antal ord av 3 till 6 eller 7 teckens längd. Programmet jämför sedan sorteringstiden för denna ordmängd med 6 olika sorteringsmetoder. De 6 metoderna är:

1. BUBBLE-sortering
2. URVALS-sortering
3. DELAYED REPLACEMENT-sortering
4. BATCHER-sortering
5. SHELL-METZNER-sortering
6. QUICKSORT-sortering

Resultatet visar klart skillnaden i hastighet mellan olika sorteringsmetoder - N.B. om man startar upp programmet med RUN före varje nytt prov! Men programmet inbjuder på rad 580 till att direkt fortsätta med nästa sorteringsmetod, och då sorteras i fortsättningen DET REDAN SORTERADE ordbeståndet, inte det ursprungliga!

För att undvika detta bör programmet utökas med en ny rad 355:

355 FOR I%=1% TO N% : B\$(I%)=A\$(I%) : NEXT I%

varigenom det efter varje sortering återgår till samma osorterade ordföljd före varje nytt sorteringsalternativ.

Vill man sedan också ha slumpvis start vid bildningen av ordmängden tillfogas en rad:

265 RANDOMIZE

Det ursprungliga programmet gav emellertid en del oväntade resultat som motiverade en närmare undersökning. Den följer här.

Jag bildar t.ex. 200 ord och jämför med dessa olika sorteringsmetoder, dels som första metod och dels som andra, dvs sedan den första redan sorterat materialet. Då får jag de tider (i sek) som framgår av den övre delen av tabellen nedan, där MOL = max ordlängd, ST = sorteringstyp, AOB = antal ordbyten

|     |    | Sorteringsmetod |      |      |      |      |      |      |
|-----|----|-----------------|------|------|------|------|------|------|
| MOL | ST | AOB             | 1    | 2    | 3    | 4    | 5    | 6    |
| 6   | OS | 0               | 67.2 | 49.8 | 55.0 | 13.5 | 11.3 | 9.4  |
| 7   | OS | 0               | 66.9 | 49.7 | 54.9 | 13.0 | 11.2 | 8.8  |
| 7   | S  | 0               | 35.3 | 48.8 | 54.0 | 10.1 | 4.8  | 55.8 |
| 7   | S  | 4               | 37.5 | 48.8 | 54.0 | 10.4 | 5.9  | 35.3 |
| 7   | S  | 10              | 38.4 | 48.8 | 54.0 | 11.3 | 7.7  | 18.8 |
| 7   | S  | 25              | 44.9 | 48.9 | 54.1 | 11.9 | 8.6  | 15.7 |
| 7   | S  | 50              | 50.0 | 48.9 | 54.2 | 12.1 | 9.9  | 10.7 |
| 7   | OS | 0               | 593  | 437  | 487  | 53.1 | 43.1 | 32.6 |
| 7   | S  | 0               | 318  | 433  | 484  | 41.0 | 18.9 | 490  |
| 7   | S  | 3x10            | 326  | 433  | 484  | 43.3 | 22.3 | 243  |

Resultatet är intressant. De tre sista metoderna är påfallande snabbare än de tre första, och långsamheten hos dessa ökas kraftigt (något mindre än kvadratisk) med antalet ord som sorteras.

Skillnaden i snabbhet mellan de tre senare metoderna är inte särskilt stor och lustigt nog går det för alla metoder något fortare när största ordlängden ökas från 6 till 7 tecken.

Men när uppgiften blir att sortera en redan sorterad ordmängd går QUICKSORT bet. Medan SHELL-METZNER-metoden minskar tiden till mindre än hälften tar QUICKSORT på sig ungefär lika lång tid som de tre långsamma metoderna! Om nu tio slumpvis valda ordpar i den sorterade ordmängden byter plats sjunker sorteringstiden för QUICKSORT till ca 20 sek, dvs fortfarande mer än dubbla tiden för sortering med SHELL-METZNER. Först vid ca 50 ordbyten blir sorteringstiden lika, ca 10 sek.

Slutsatsen blir att metod 5, SHELL-METZNER, är något långsammare än QUICKSORT vid mycket osorterade ordmängder, men att QUICKSORT bör undvikas vid omsortering av en förut sorterad mängd som får ett tillskott av mindre än ca 20% nya ord. Och QUICKSORT skall tydligen skys som pesten vid små tillskott, sådana som ofta görs vid uppdatering i registerprogram. Det kan möjligen förklara varför ett välkänt registerprogram vid omsortering av några tusen poster efter tillskott av några få poster tar ett par timmar på sig för ny sortering.

Då SHELL-METZNER-metoden dessutom har en mycket enkel uppbyggnad, speciellt jämfört med QUICKSORT, och därtill tar mindre plats i minnet, bör den avgjort föredras när medlemmar (eller företag) behöver en subrutin för sortering.

Den undre delen av tabellen visar tiderna för en ordmängd på 600 ord. Slutsatserna ovan framhävs där ännu tydligare.

2. När jag med min ABC80 med 32kB försöker överföra FASTCAS.32K från kassetten med LOADCAS: stannar överföringen med ERR 11 på rad 29. Denna rad lyder "29 ??? 29 GOLV+". Kopiering till diskett med CASDISK3 fungerar, men när programmet sedan skall köras stannar det vid samma rad med ERR 11 LINE 29 ERR 59. Vid försök att läsa filen med VISA får jag STOP vid rad 29, nu utan frågetecken, och ERR 58 LINE 80. Alltså läst läge!

Jag har för mig att jag stött på de tre ??? någon gång tidigare i ett kassettprogram och då efter en del försök lämnade programmet åt sitt öde, och vidare att jag sett en sådan rad nämnas i någon uppsats i ABC-bladet - men i vilket nr?

Vad betyder den stoppande raden 29 ??? 29 etc och hur skall jag forcera den?

<1308>

Göran Tengner

Programredaktionen svarar:  
Programredaktionen har uppmärksammat felet på kassett 16. Använd FASTCAS från ABC-kassett nr 17 i stället.




**SALJAREN**

- Registrering av order

**FÖR LUXORS ABC-DATORER:**

## LUX-NET

**som administrativt system i äkta fler-användarmiljö.**




**SEKRETERAREN**

- Ordbehandling
- Telex/teletex




**LUX-NET CENTRAL**

- 52 Mbyte winchester
- Bandstreamer
- Floppy

Centrala skrivare




**Lokal skrivare för fakturor**

**EKONOMIÄVD**

- Fakturering m.m.




**LÖVERANSÄVD**

- Orderhantering




**Lokal skrivare för följesedlar m.m.**




**Elektronisk post**




**CHEFEN**

- Statistik
- Grafik
- Kalkyler




**ABC CENTER**

*Vi kan nätverka!*

**T-DX**

Box 227  
191 23 Sollentuna  
Sollentunavägen 225  
T-DX DATORER AB 08-92 03 30

**ABC CENTER**  
Stockholm  
Kungsgatan 79  
08-50 68 75

**ABC CENTE**  
Gävle  
Engelbrektsväg 2  
026-10 53 55

**RADANNONSER****Säljes**

ABC80 med 32 k minne, 80 tecken, smartaid, 2\*60 k flex, bandspelare samt printer U-80 säljes för 6000:- totalt.

Priset inkluderar mängder med program (bl a ordbehandling) samt litteratur. Även delköp kan tänkas.

&lt;1318&gt;

Lennart Thörnell  
Kungsgatan 70  
641 36 Katrineholm  
0150-138 79 efter 19

**Säljes**

Printer för ABC80, Centronics 779 med autostart och stora/små bokstäver. Interfacekort för ABC80 ingår.

Pris: 2000:-

Diskettstation, DataDisc 80, avsedd för ABC80/800.

Pris: 4000:-

Floppy-controller-kort för ABC80/800 diskettstationer. Dubbel density. passar bl DataDisc, Metrics FD.

Pris: 1500:-

Printerinterface-kort för AB80/800, Centronics-snitt. Passar bl a DataDisc- och Metric-diskettstationerna. Innehåller även 2 st EPROM med DOS och printerdrivprogram. Pris: 800:-

&lt;383&gt;

Lars Ericson  
Kummelvägen 32  
161 39 Bromma  
08-80 33 21

**Säljes**

Olivetti skönskrivare ET 121, DY311, DY 450, Praxis 35, HP Plotter 7470A, ABC850 Winchester + övrig utrustning till ABC-datorena till starkt rabatterade priser

Ring för prisuppgift.

Hans-Åke Nilsson  
Bild & Ljudhörnan  
Andra Östergatan 34  
271 00 Ystad  
0411-17075

**Säljes**

Modem DATA WIEW, 9-polig D-kontakt passande till ABC80/800. Televerkets stickpropp medföljer. Oanvänt under ett år.

Pris: 800:-

Anders Bergman  
Nybrogatan 43  
114 39 Stockholm  
08-60 60 67

**Säljes**

ABC80, 32 kB, 80/40 tecken på bildskärm Skivminne FD2 med dataseparator Skrivare Centronics 779, Luxors interface Kassettminne, databuss En stor mängd program bl a Baz80.

&lt;3676&gt;

Bengt Klarström  
Runbovägen 3  
191 48 Sollentuna  
08-754 50 60

**Säljes**

ABC80 inkl bildskärmsstativ, numeriskt tangentbord, minneskort 16 kB. Program på kassett: Miniprint II och Basicinstruktör. Program på skiva DP: Läs/Skriv-rutin Basregister ABC. ABC-klubbens kassetter nr 3 - 17, ABC-bladet 1, 1982 - 4, 1985 samt diverse litteratur.

Pris totalt 4.150:-

&lt;2827&gt;

Lars-Olle Gustavsson  
Snapphanegatan 20  
271 00 Ystad  
0411-175 31

**Säljes**

Matrisskrivare Metric 8300P. Anslutes till ABC80 via parallellinterface. Obetydligt använd. Komplet med manualer.

Pris: 1200:-

&lt;5621&gt;Olle Wiren

0586-826 53, dagtid  
0550-193 86, kvällstid

**Säljes**

A/D-omvandlare till ABC80/800/802/806 och 4680. 32 kanalers upplösning, 12 bitar. Nypris 5600:-, nu 2500:-.

Reläkort 16 utgångar, 100V, 1A. Nypris 2800:-, nu 1800:-.

Reläer, halvledare, opto, triac, styrspänning 3,5 - 15 V DC. Kopplingsström 220-240 V, 2,5 A. Nypris 155:-, nu 85:-. Ombärliga vid datorstyrning av 220 volt.

08-20 63 05 efter 18.00

**Säljes**

ABC80 med TKN80 och 32 k, printer och floppy 2\*160 k, bandspelare säljes omgående till bra pris.

&lt;3513&gt;

Rikard Edström  
08-38 34 35

**Säljes**

ABC800M, bildskärm 815 i skick som ny. Program ORD800, Basregister 800. Disk 2\*320, inkl plats för expansionskort.

&lt;1693&gt;

Christer Olsson  
021-202 82 efter 17.00

**Säljes**

ABC80 med 32 k, TKN80, dubbel flex Metric FD2UD, program för bokföring och ordbehandling. Mycket litet använd. Säljes med använd printer motsvarande Epson MX80F/T.

Priside för hela paketet 12.500:-

Per Hallgren  
0762-515 01

**Säljes**

ABC80 ombyggd med separat tangentbord (Schroff). Originalmonitor med amber skärm, TKN80, 32 k, bandspelare samt ABC-kassetter 1-15 samt manual.

Pris 3900:-

&lt;5153&gt;

Anders Klockars  
018-17 82 47 (arb)  
018-12 43 16

**Säljes**

Skivminne ABC830 kompl m expansionslåda ABC890, kort och kablage

Pris: 5800:-

Skivminne ABC832. Pris: 5000:-  
Kontrollerkort 1000:-. Dito snabbt (Luxor) 2600:-. Modem 300 b 700:-

&lt;2265&gt;

Lennart Holmström  
Ekebo Österstad  
591 90 Motala  
0141-711 05

**Köpes**

Högupplösningsskärm t ex ABC812 och skrivare.

&lt;2265&gt;

Lennart Holmström  
Ekebo Österstad  
591 90 Motala  
0141-711 05

**Säljes**

Mikrorobot, byggsats 4500:-. Lämplig för hobby och utbildning. Kan anpassas till ABC-datorer.

&lt;1484&gt;

Sven-Erik Berggren  
Bäckvägen 12  
682 00 Filipstad  
0590-139 95

**Köpes**

Defekt 80x köpes

&lt;3095&gt;

Lennart Bjurdell  
Rankhusvägen 30  
196 30 Kungsängen  
0758-743 04

**Säljes**

Assembler för ABC80, ASMZ, 51/4", 160 kB med debygger och editor. Nypris 2.300:- säljes för 1.000:-

ICS-mikroprocessor-kurs med interface-del.

Nypris 12.000:- säljes för 5.000:-

&lt;4580&gt;

Hans-W Siebert  
Skravvägen 8  
613 00 Oxelösund  
0155-33140 (dagtid)

**Säljes**

ABC832-floppy 6.000:-

&lt;6535&gt;

Mats Bende  
Bladvägen 21  
541 41 Skövde  
0500-304 56

**Säljes**

ABC802 med tangentbord ABC55 och bandspelare  
Pris 6.000:- i

ABC80, 32 kB med omvänd video-funktion, monitor och bandspelare och litteratur

&lt;5324&gt;

Bertil Söderlund  
08-743 02 75

**Säljes**

ABC80 mmed antireflexbehandlad bildskärm. BASIC II från Liber inbyggd vilket ger samma BASIC i en ABC802. Fullt utbyggd minneskapacitet 32 kB och en 48 kB RAM-floppy. UFD-DOS, printerrutin mm. Myab:s TKN-80 inbyggd vilket ger möjlighet till 80 tecken per rad.

ABC830 skvminne 2\*160 kB och hembyggd expansionslåda med fyra I/O-kortplatser (varav två lediga).

STAR Gemini-10X Printer. Helt Epson-kompatibel plus litet till. Har 4 kB buffer och högupplösningsgrafik.

Kassettdbandspelare ABC80.

Tillbehör Comporian joystick och en speech synthesiser med anpassningskort. Program ABC-ORD, Basregister II, Läs & skriv, assembler/disassembler, samtliga ABC-kassetter och mycket annat godis. Totalt antal disketter ca 80 st (blandat SD/DD varav hälften är tomma) samt diskettförvaringslåda och rengöringsdisketter.

Terminalbord speciellt anpassat för ABC80 ingår om så önskas.

Alltsammans säljs till högstbjudande.

16 kB RAM expansionsminne, typ RAMEX80 till ABC80 för montering i tangentbordet. Enkel montering.

Priside 400:-

<1386>

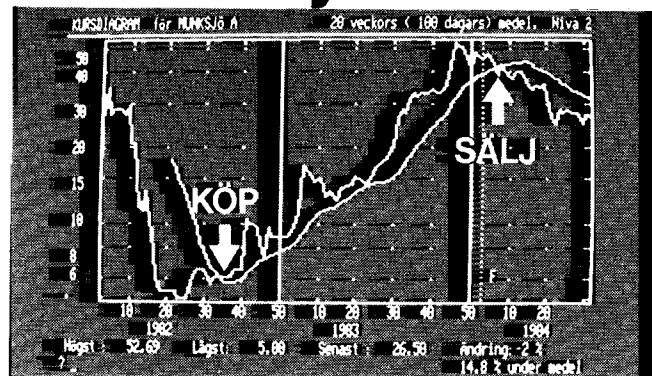
Michael Widell

Höstvädergatan 33

417 33 Göteborg

031-53 92 47 (säkrast kvällstid)

## Chartanalys av aktier



Kursdiagram för Munksjö A från 1982-1984 med 20 veckors glidande medelvärde.

Programpaketet AKTIER för LUXOR ABC 800M/806 eller FACIT DTC hjälper Dig fatta rätt köp- och säljbeslut. Du följer lätt upp

### Kurshistorik

Emissionskorrigerade kursdiagram med olika valfria glidande medelvärden för chartanalys.

### Nyckeltal

P/e-tal, vinst/aktie, direktavkastning m.m.

### Eget innehav

Aktuell vinst/förlust per aktiepost beräknas. Innehavet kan delas upp i max. 19 olika portföljer med kod för sekretess.

Menystyr och självinstruerande program.

Enkel och snabb kursinläsning från tidningarnas börslistor veckovis.

Kurshistorik kan beställas på diskett för samtliga börsaktier.



**ELEKTRA**  
RATIONALISERINGS AB  
Solvingevägen 1, 632 33 ESKILSTUNA

**RING 016-42 09 02**  
för info

## EPROM II

### EPROM-PROGRAMMERARE TILL DIN ABC80 / ABC800

Komplett EPROM-PROGRAMMERARE till följande 5V-minnen: 2758, 2516, 2716, 2532, 2732, 2564, 2764, 27128, 27256.

**NYHET!** Programmeringsspanning 25/21/12,5 V. "Intelligent" programmeringsalgoritm.

EPROM-PROGRAMMERAREN består av:

- Kretskort för ABC80/4680 bussen
- Drivprogram på flexskiva
- Utförlig bruksanvisning på svenska

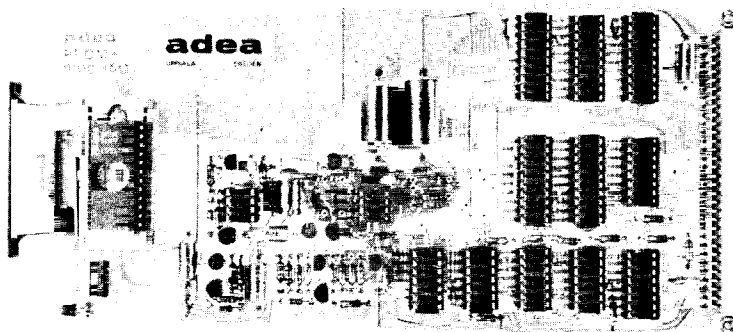
Drivprogrammet har följande funktioner:

- Programmering från RAM-minne
- Kopiering av EPROM till RAM-minne
- Kopiering av RAM-minne till fil
- Inläsning av fil till RAM-minne
- Radertest av EPROM
- Verifiering av EPROM/RAM-minne
- Byte av EPROM-typ

**NYHET!** Bildskärmsredigering av RAM-minne

**NYHET!** Utskrift valbart i Hex, Decimal, Oktal, Binärt format

**NYHET!** • Hardcopy-funktion



Tillbehörskort för programmering av enchipdatorerna 8742, 8748, 8749

**ADEA Elektronik AB**  
Box 16015, 750 16 UPPSALA  
☎ 018-10 06 02

# SKRIVUT

För en tid sedan hade Sven Wickberg en artikel i bladet om programmet RAKHÖGER. Det verkade kul, så jag knapade in det. Till en början var jag mycket nöjd med det, men efter ett tag kom jag på fler och fler finesser som jag ville ha, men som saknades i programmet. Bland annat saknade jag en möjlighet att få utskriften att stanna vid sidbyte, så att jag kunde använda skrivmaskinsark i stället för det vanliga, pyjamasrandiga, datalakanet.

Jag gjorde några ändringar i det ursprungliga programmet, men till sist tyckte jag att det blev för jobbigt att hela tiden behöva ta hänsyn till hur Sven hade använt variabler, så jag började skriva ett helt nytt program, som jag döpte till SKRIVUT.

Raderna 100-150 dimensionerar variabler och definierar standardvärden. Här kan vi ändra om de förprogrammerade värdena inte passar.

Raderna 170-270 skriver en presentations-text, och visar standardvärdena. På rad 270 kan vi godkänna alltihop med en tryckning på <RETURN>.

Om vi inte godkände standardvärdena får raderna 290-430 det mödosamma jobbet att hämta in de värden vi vill ha. Även här finns standard svar till alla frågor.

Koderna på raderna 490 och 840 är till för att min EPSON ska kunna skriva ända längst ner på papperet utan att signalera "NO PAPER".

På rad 510 börjar det egentliga arbetet, med att en rad hämtas in från infilen. Därvid kontrolleras först om det finns någon text i bufferten Q\$, där den överskjutande delen av en för lång rad placeras. Om där fanns text läggs den över i textvariabeln A\$, annars hämtas en ny rad från infilen.

Raderna 520-600 sköter om att utföra de kommandon som kan läggas in i infilen. Kommandona måste alltid ligga först på raden. Följande kommandon accepteras:

- .C gör att de två närmaste raderna skrivs ut utan något linefeed (t ex för understrykning).
- .P ger en sidframmatning.
- .R slår av/på rak högermarginal.
- .Sn sätter sidnumret till n. Detta möjliggör alltså hopp i sidnumreringen.
- .Mn sätter vänstermarginalen till n.
- .Ln sätter radlängden till n.
- .An sätter radavståndet till n/72 inch. Fungerar endast på EPSON MX-80 och kompatibla skrivare.
- .A återställer radavståndet till 1/6 inch, vilket är standard vid nättillslag på de flesta skrivare. Fungerar endast på EPSON och kompatibla.

Rad 610 kollar om den rad som lästes in på rad 510 var för lång. Om den var det kapas raden och den överskjutande delen placeras i Q\$.

Den raka högermarginalen fixas på raderna 660-700. Precis som i Sven Wickbergs program börjar inskrivningen av mellanslag på varannan rad i mitten av raden. Om vi inte vill ha justering av högermarginalen, kommer rad 640 att orsaka att hela rutinen hoppas över. Likaså hoppas rutinen förbi om raden är kortare än 5/6 av vald radlängd.

När raden väl är utskriven ökas radräk-naren R%, varefter programmet kollar om sidan är full. I så fall anropas subrutinen på raderna 790-860 som ombesörjer utskrift av sidnummer, sidbyte och eventuellt stopp.

När utskriften är färdig anropas subrutinen igen, för att skriva ut sidnummer även på den sista sidan. Denna lösning har dock den nackdelen, att om vi valt stopp vid sidbyte kommer datorn att vänta på att vi skall byta sida, även efter den sista sidan. För att undvika detta har jag lagt in flaggan F%, som sätts innan subrutinen anropas sista gången, och orsakar därigenom ett återhopp från subrutinen, före det avsnitt som skriver ut uppmaningen om att byta sida.

Naturligtvis kan programmet förbättras ytterligare, t ex skulle man kunna lägga till en rutin så att det går att välja var en för lång rad skall kapas.

Och därmed skulle genomgången av programmet, och även denna artikel vara slut.

<5169>

Ture Pålsson

```

1 REM <+ SKRIVUT .BAS +>
2 REM Skriver ut en textfil på printer
3 REM med rak högermarginal och divers
  e
4 REM andra finesser.
5 REM Av (c) Ture Pålsson
6 REM +-----+
+
100 DEFFNJ%=INSTR(1%,'JjYy'+CHR$(13%),W$
  )>0%
110 DIM A$=120,D$=120
120 R1%=60% : M%=10% : R2%=-1% : REM Rad
  längd,vmarg,rak hmarg
130 S1%=-1% : S2%=1% : S0%=0% : REM Sidn
  ummer,sidnummer start,stopp vid sidb
  yte
140 S3%=60% : REM sidlängd
150 P$='PR:VSA30C72.5' : REM Printerkod
160 REM kolla om defaultvärden är ok
170 : CHR$(12%);>' Filutskrift <'
180 ; : 'Printerkod : 'P$
190 : 'Radlängd : 'R1%
200 : 'Vänstermarg : 'M%;M%
210 : 'Rak högermarg'
220 : 'Sidlängd : 'S3%
230 : 'Sidnumrering,'
240 : 'start från : 'S2%
250 : 'Ej stopp vid sidbyte'
260 ; : 'OK (J) ' : GET W$ : ; W$
270 IF FNJ% GOTO 440

```

```

280 REM standardvärden dög ej ...
290 ; : 'Printerkod ( 'P$;' )';
300 INPUT W$
310 IF W$<>' ' P$=W$ : GOTO 290
320 ; : 'Radlängd ( 'R1%;' )';
330 ONERRORGOTO 340 : INPUT R1%
340 REM
350 ; : 'Vänstermarg ( 'M%;' )';
360 ONERRORGOTO 370 : INPUT S2%
370 ; : 'Rak högermarg (J)';
380 GET W$ : R2%=FNJ%
390 ; : 'Sidnummer (J)'; : GET W$
  : S1%=FNJ%
400 IF NOT S1% GOTO 420
410 ; : 'Start från ( 'S2%;' )'; : O
  NERRORGOTO 420 : INPUT S2%
420 ; : 'Stopp vid sidbyte (J)'; : GET
  W$ : S0%=FNJ%
430 ONERRORGOTO 440 ; : 'Sidläng
  d ( 'S3%;' )'; : INPUT S3%
440 REM print
450 ; : 'Infil ? ' : INPUTLINE F$
  : F$=LEFT$(F$,LEN(F$)-2%)
460 IF INSTR(1%,F$,'.')=0% : 'TXT' : F$
  =F$+'.'TXT' ELSE ;
470 : 'OK (J) ?' : GET W$ : IF NOT FNJ%
  GOTO 120
480 OPEN F$ ASFILE 1% : OPEN P$ ASFILE 2
  %
490 : $2%,CHR$(27%,56%,13%);
500 ONERRORGOTO 760
510 IF Q$<>' ' A$=Q$ : Q$=' ' : GOTO 610 E
  LSE INPUTLINE $1%,A$ : A$=LEFT$(A$,L
  EN(A$)-2%)
520 IF LEN(A$)>2% B$=LEFT$(A$,2%) : D$=R
  IGH$(A$,3%) ELSE B$=' ' : D$=' '
530 IF A$='.C' F1%=-1% : GOTO 510
540 IF A$='.P' GOSUB 790 : GOTO 510
550 IF A$='.R' IF R2%=0% : GOTO 510
  ELSE R2%=-1% : GOTO 510
560 IF B$='.S' S2%=VAL(D$) : GOTO 510
570 IF B$='.M' M%=VAL(D$) : GOTO 510
580 IF B$='.L' R1%=VAL(D$) : GOTO 510
590 IF A$='.A' : $2%CHR$(27%,65%,12%,27%
  )+CHR$(50%,13%); : GOTO 510
600 IF B$='.A' : $2%CHR$(27%,65%,VAL(D$)
  ,27%)+CHR$(50%,13%); : GOTO 510
610 IF LEN(A$)>R1% Q$=RIGHT$(A$,R1%+1%)
  : A$=LEFT$(A$,R1%)
620 A%=INSTR(1%,A$,'.') : IF A% A$=LEFT$
  (A$,A%-1%)+CHR$(175%)+RIGHT$(A$,A%+1
  %)
630 IF NOT R2% GOTO 710
640 IF LEN(A$)>R1% GOTO 710
650 REM rakhöger
660 IF LEN(A$)<R1%*5%/6% GOTO 710
670 IF R%/2%=R%/2 C%=LEN(A$)/2% ELSE C%=
  1%
680 B%=INSTR(C%,A$,'.') : IF B%=0% IF C%
  =1% GOTO 710 ELSE C%=1% : GOTO 680
690 A$=LEFT$(A$,B%)+ ' ' +RIGHT$(A$,B%+1%)
700 IF LEN(A$)<R1% C%=B%+3% : GOTO 680
710 : $2%,TAB(M%);A$; : IF F1% : $2%,CHR
  $(13%); : F1%=0% : GOTO 740 ELSE : $
  2%
720 R%=R%+1%
730 IF R%=S3% GOSUB 790
740 GOTO 510
750 IF ERRCODE<>34% : CHR$(7%)'$ Error';
  ERRCODE : STOP
760 IF S1% F%=-1% : GOSUB 790
770 ; : '$$$$ KLART $$$'
780 END
790 IF NOT S1% GOTO 830 ELSE : $2%STRING
  $(S3%-R%,10%)
800 : $2%
810 : $2%TAB(M%+(R1%-INT(LOG(S2%))-4%)/2
  %);<' ;S2%;' >'
820 IF F% RETURN
830 IF S0% ; : 'Byt sida, tryck sedan
  på RETURN' : GET W$ ELSE : $2%,CHR$(
  12%);
840 : $2%,CHR$(27%,56%,13%);
850 R%=0% : S2%=S2%+1%
860 RETURN

```

**BLADET DATA**

RING 026- 14 24 64 75/1200  
DATA, PRISER och ANNONSER

Tidskriften Illustrerad vetenskap har några sidor med ovanstående rubrik. Härförleden fanns ett problem av följande innehåll (återges här ut minnet)

"Det är något speciellt med talet 48. Läger man till 1 får man en jämn kvadrat  $49=7*7$ . Tar man hälften och lägger till 1 får man också en jämn kvadrat  $25=5*5$ . Säg ytterligare ETT sådant tal."

Anande att det var ganska mycket jobb skrev jag ett enkelt datorprogram (KLAUS1.BAS). Det visade sig välbetänkt, för nästa tal var stort. Jag gick baklänges. Om  $5*5$  kan ge upphov till 48 så borde man kunna göra en liknande manöver för  $6*6$ ,  $7*7$  osv och kolla ifall talet minus ett gånger två plus ett var jämn kvadrat. Nästa tal som matchar är 29 som ger talet 1681 som motsvarighet till 49.

Du kan själv kolla om det finns ytterligare fler. Man kan inte köra med heltal här, för man kommer för fort upp över den högsta tillåtna heltalsgränsen.

Speciellt i amerikanska publikationer kan man finna exempel på hur outtröttliga räkne-mästare har hittat kluriga kombinationer på egenskaper hos hela tal.

Nuförtiden kan vilken ABC80-ägare som helst klå dem genom att låta datorn göra jobbet. Här bifogas exempelvis SVEN1.BAS som löser följande problem:

"Talet 2 har den egenskapen att om dess kvadrat fördubblas och man därefter lägger till 1 får man en ny jämn kvadrat. Finns det fler sådana tal?"

Från början tänkte jag mig att visa upp fler exempel, men varför skall jag förstöra det roliga för läsarna? Säkert kan någon hitta på någonting ännu mycket lustigare

att visa upp. Jag tar emot alla förslag och lovar att det roligaste skall få ett hedersomnämning i ABC-bladet.

En kort kommentar till programlistningarna. I sådana här fall måste man miss-tänka att det kan dröja ganska länge innan datorn hittar en lösning. För att vara säkra på att programmet fungerar och att datorn jobbar bör man ha en pausfågel. Även om en sådan fördröjer datorns arbete, är det bättre att kunna se vad som händer.

Här har jag valt att skriva värdet av räknaren I på skärmen. Jag har varit så slö att jag inte försöker positionera den vilket enkelt hade kunnat göras med `CUR(23%,0%)I`; - den får helt enkelt skrivas ut rad för rad.

Men hur skall man kunna se att man fått napp? Genom att efter ett lyckat värde sätta den fyllda rutan (ascii 127) efter I - och sedan INTE ta med semikolon.

Effekten kan studeras och varieras. Med `CUR` blir det förstås bättre såtillvida att man endast spar de lyckade talen i en lång kolumn.

En annan metod - om man räknar med lång sökning och inte har tid att sitta vid skärmen hela tiden - är att beställa utskrift på printer av lyckade tal. Då fordras t ex följande tilläggsrader:

```
50 OPEN 'PR:' ASFILE 1%
---
130 IF (lyckat värde) THEN ;$1%,I%;
```

Då kommer printerbufferten att spara ihop en rad lyckade värden innan de skrivs ut. Vill man se dem genast och har gott om papper tar man förstås bort det sista semikolonet.

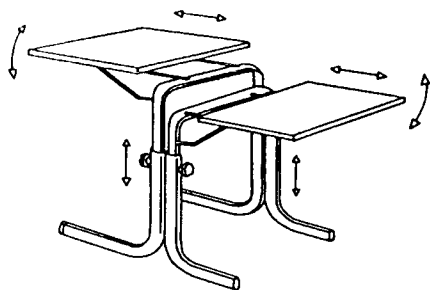
1384  
Sven Wickberg

# KLAUS kryptiska sidor

```
10 REM KLAUS1
11 REM lösning av ett problem på Klaus
    kryptiska sidor
100 FOR I=6 TO 1000
110 ; I;
120 A=(I*I-1)*2+1
130 IF SQR(A)=INT(SQR(A)) THEN ; CHR$(1
    27%)
140 NEXT I
```

```
10 REM sven1
11 REM 2*2 fördubblat plus 1 är jämn k
    vadrat. Fler?
100 FOR I=1 TO 10000
105 ; I;
110 A=I*I*2+1
120 IF SQR(A)=INT(SQR(A)) ; CHR$(127%)
130 NEXT I
```

## PC BORDET



Terminalbord med steglös justering av höjd, djup och vinkel. Stativ i beige lack, skivor i eklaminat. Bordet levereras omonterat.

Tangentbordsskiva 610 x 330 mm  
Monitorskiva 610 x 380 mm  
Sidoskiva (ej bild) 350 x 710 mm

PC-bordet ..... 1225:-/st  
Sidoskiva ..... 475:-/st  
Hjulsats ..... 155:-

## HÄR ÄR ARKMATAREN!

Utnyttja Din skrivare mer effektivt. Se vår annons i ABC-bladet 4/85 sidan 45.

## DATORHUVAR

Skydda Din datorutrustning mot smuts och damm. Beställ våra kvalitetshuvar i smidig galon. Finns i färgerna vit, svart, marinblå och vinröd.

ABC80, ABC55 (tangentbord)..... 63:-/st  
ABC 800, ABC77, ABC99 (tangentbord).... 79:-/st  
ABC830/832 (skivminne)..... 79:-/st  
ABC80, ABC812/815 (bildskärm)..... 128:-/st  
Vi tillverkar huvar till alla typer av datorer, skrivare etc! Kontakta oss.

## Rengöringssats



1 st 5,25"-rengöringsdiskett för 15 rengöringar av diskett-driven. 8 st våtservetter för rengöring av tangentbordet. 1 flaska antistat- och rengörings-spray för bildskärmen.

**149:-/st**

**LC** Lennart Christofferson  
**GRUPPEN**

Alla priser  
exkl. moms

DATATILLBEHÖR  
Besök: Svartbäcksgatan 50 H, Uppsala  
Post: Box 15037, 750 15 Uppsala  
Telefon: 018/14 00 70

# ABC VDP

## Vad?

I ABC-bladet nr 4, 1985 finns en artikel 'Höjoplöselig Grafik med ABC80'. Den inspirerade mig till att skriva några rader om en del experiment jag gjort med grafik. Jag beskriver en tillsats till ABC80 som ger bättre grafik på bildskärmen. I stället för standardupplösningen 80\*72 ger tillsatsen upplösningen 256\*192. Den händige amatören kan bygga den för under 1000:- Den nya grafikbilden visas på den vanliga ABC80-bildskärmen. Man kan växla mellan den vanliga ABC80-bilden och grafikbilden med en enkel instruktion (OUT 2,x går även som kommando).

## Varför?

När jag hade byggt om min ABC80 till 64 kbyte RAM så blev det gamla RAM-minnet över. Man skulle nämligen vid ombyggnaden sätta i nya minneskaplar, det gick inte att komplettera de gamla. Vad skulle jag då göra av dessa gamla minneskaplar? Högupplösningsgrafik är spännande! Men det behövs mer än RAM-minne. Jag studerade kopplingsschemat för ABC800 med grafik. Den grafiktillsatsen innehåller en väldig massa elektronik för att göra bilderna. Det verkade inte lockande att försöka med någon imitation av den.

Skoldatorn Compis var mycket omskriven vid denna tidpunkt. Den ger fin grafik med en speciell mikroprocessor (tex NEC 7220). Detta var en mycket intressant möjlighet ända tills jag fick veta priset på processorn. Den kostade 800 kronor på våren -84. Sedan hittade jag med hjälp av en artikel i den amerikanska tidskriften BYTE en enklare mikroprocessor (Texas TMS 9929) som bara kostade 135:-. Den fick det bli. Denna processor sitter i flera persondatorer t ex Spectravideo, TI 99/4A. Texas kallar den Video Display Processor förkortat VDP.

## Teknik

I vertikal led är upplösningen hos grafik-tillsatser oftast 256 linjer per bild. Det beror på att en standard TV-mottagare som man av prisskäl vill använda ger just 256 linjer per bild och 50 bilder per sekund. Vid Sveriges Radios TV sändningar får man dubbelt så stor upplösning genom att sända en bild till, lite förskjuten från den första. Det betyder att det ska ritas 2 skärmbilder innan man har fått se allt, det blir 25 hela bilder per sekund. Detta system duger inte för högupplösningsgrafik. Titta på TV2 pausbild och se hur 2:an hoppar! Det vore inte bra om ABC80:s tvåor hoppade på det viset. I horisontell led behöver upplösningen inte vara större än i vertikal led alltså 256.

Detta betyder att man på bildskärmen kan rita  $256 \times 256 = 65536$  olika punkter. Om varje punkt ska kunna tändas i någon av 16 olika färger så behövs 4 bit/punkt i bildminnet ( $2^4 = 16$ ). Bildminnets storlek blir då  $65536 \times 4 \text{ bit} = 32768 \text{ byte}$ . Den grafikprocessor jag valde är ursprungligen gjord till amerikansk TV-standard och bara delvis anpassad till europeisk standard. Den ritas därför bara på 192 linjer per bild. För att minska behovet av (på den tiden) dyrbart bildminne så kan processorn dessutom inte alltid ge olika färg till punkter som ligger bredvid varandra. Mer om denna lustighet senare.

## Blockschemat

ABC-VDP byggs på ett labkort i enkelt Europaformat som ansluts till ABC-bussen, kanske i flexskivestationen. Även kabeln mellan tangentbordet och bildskärmen ska passera ABC-VDP. Med omkopplaren i det läge som den är ritad i blockschemat så passerar signalerna från tangentbordet till bildskärmen och ABC80 betar sig helt nor-

malt. Ett program i ABC80 får nu fylla bildminnet, via VDP:n, med bildinformationen. Programmet slår sedan om omkopplaren och högupplösningsskärmen kommer fram på bildskärmen.

Vill man absolut se färgerna så skaffar man en extra färgTV och bygger den anpassningselektronik som kallas MOD i block-schemat. MOD (=modulator) ska vara lite olika beroende på om man använder antenn- eller videoingång på färgTV:en.

## Framtiden

Man kan ju undra vad det finns för anledning att ägna sig åt denna grafik-tillsats när den som beskrevs i ABC-bladet 4, 1985 kan så mycket mer: tex. rita linjer, cirklar och bokstäver fruktsamt snabbt och nästan på egen hand. Låt mig nämna några skäl:

1. ABC-VDP blir säkert billigare.

2. Man behöver ingen extra TV-apparat. Det behöver man inte till den andra tillsatsen heller om man kompletterar den med omkopplaren. För den intresserade bifogar jag schema över omkopplaren. Den består av en CMOS omkopplaren MC 14016 som styrs från D-vippan 74LS74. Transistorerna på utgångarna minskar inverkan av omkopplarens resistans i tillslaget läge.

3. De olika färger som ABC-VDP kan göra blir på ABC80-bildskärm olika gråtoner. 'Sprites' från VDP är en form av grafik som kan vara rolig att prova.

Låt mig också nämna några skäl att man INTE ska bry sig om ABC-VDP:

1. Det är ett ganska avancerat bygge, ingenting för förstagångslöpare.

2. Texas vill nu istället sälja en processor som heter TMS 9129. Det enda som egentligen skiljer är anslutningen av bildminnet.

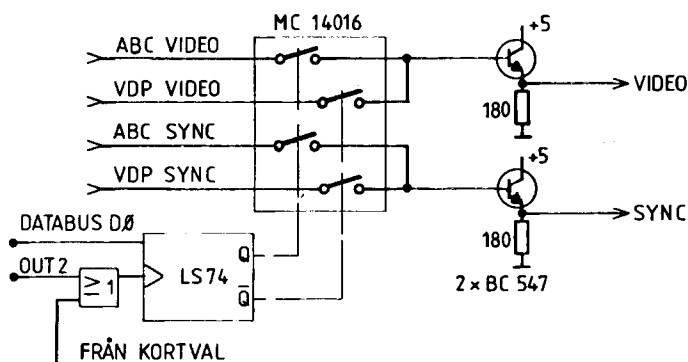
3. Det vore trevligt om man kunde visa både ABC80- och VDP-bilden samtidigt. Då måste båda två göra sina linjer och bilder samtidigt vilket ställer nya krav på elektroniken. Ibland går jag och funderar på hur detta ska lösas. Det kanske blir en ny version av ABC-VDP.

4. Det finns inget fullständigt kopplingschema över ABC-VDP publicerat i ABC-bladet. Den som är så händig med lödkolven att han/hon törs ge sig på det här kan kanske göra ett schema själv. Eller komma på något sätt att skaffa sig ett.

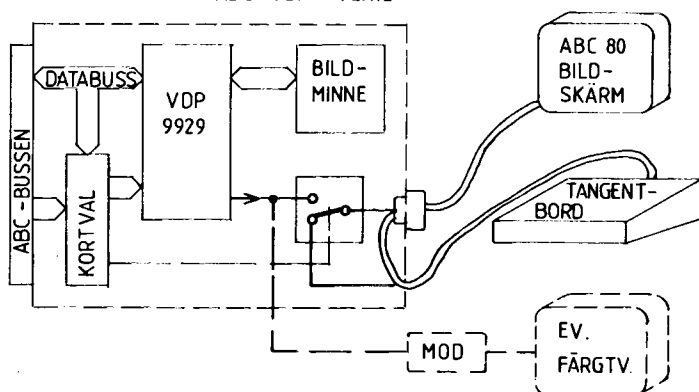
<6555>

Lars Karis

ABC-VDP VER.2  
VIDEO-OMKOPPLARE



BLOCKSHEMA ABC-VDP VER.2



Matematiker är ett lustigt släkte (har jag sagt det förut?). Man förströr sig med alla möjliga högst onödiga och möjligen onyttiga tankar om hur man kan mixtra med de hela talen. Ett sådant (obesvisat, tror jag) teorem är att alla tal som behandlas med CBS-metoden förr eller senare konvergerar till 1.

Sanningen att säga har jag totalt glömt vad CBS står för. Jag utlyser därför en pristävling: Den som kan tala om vad det betyder skall få den absolut oöverträffade äran att vara ABC-klubbens matematiske Besserwisser. (Antalet pris ej begränsat.)

Till saken. Tänk på ett tal!

Om talet är jämnt, halvera det. Om det är udda, multiplicera med 3 och lägg till 1.

Gör om föregående stycke till dess du får 1. Teorin säger att det alltid skall inträffa till slut, men vägen dit kan vara mycket lång, och en matematikers livslängd är för kort för att man skall prova detta empiriskt för hand. Vi låter datorn göra det.

Jag har valt att arbeta med ASCII-räkning. Det ger på ABC80 upp till omkring 25 siffrors noggrannhet (på ABC800 bra mycket mer), medan vanliga flyttal är oanvändbara för heltalsjobb redan vid 1 000 000.

Vid ASCII-räkning görs talen om till teckensträngar, som sedan av datorns inre bearbetas med samma metoder som när man räknar för hand, men mycket fortare förstås.

På 70-80 läggs talen i  $A1\$$  och  $A9\$$ , och sedan görs ett tabellhuvud som skall visa hur många steg man måste gå innan resp tal konvergerat till 1. Det påstående är förresten detsamma som att säga att förr eller senare blir talet av formen  $2^{**}N$ , där  $N$  är ett heltal, alltså ett tal i den

inte helt obekanta serien 2, 4, 8, 16...

På 120 läser vi över värdet A1\$ till A2\$, som vi sedan bearbetar i subrutinen 170-. S% är en räknare som håller reda på antalet steg i serien.

Nu flyttar vi över A2\$ till A\$ och dividerar med 2, dels med noll decimaler, dels med 1 decimal. Bliir resultaten B\$ och C\$ lika är det ett jämnt tal, som skulle divideras med 2. Det har vi redan gjort i B\$, så vi läser över det till A\$ och gör om manövern.

COMPX(B\$,C\$) är en funktion som jämför de numeriska strängarna B\$ och C\$. Om de är lika är COMPX() $\rightarrow$ 0, annars  $\rightarrow$ -1 resp  $\rightarrow$ 1. (Experimentera litet!)

Observera att det går åt pipan i vissa fall om B\$ och C\$ inte har lika många decimaler! Det sammanhänger med sorteringen av strängar. Strängen "0.1" anses som mindre än "0.10".

Men det har ingen betydelse här, så jag har högaktningsfullt struntat i kontrollen. (Varsågod, vän av ordning!)

Rad 220: om värdena inte är lika är talet udda. Då multipliceras först med 3 (MUL3()), sedan adderas 1. Antalet decimaler sätts till noll.

Räknaren uppdateras och så da capo till dess A\$="1" (rad 200). Då skrivs antalet steg ut, man hoppar till 140 och tar nästa tal, ända till dess man nått A9\$.

Jag har hittills inte hittat något motbevis mot CBS-teoremet. Men skam den som ger sig. Det finns många oprövade tal lägre än 99999999999999999999.

Det finns ingen påfågel i programmet.  
Ett allvarligt förbiseende (vän av ordning  
- är du kvar?).

<1384>

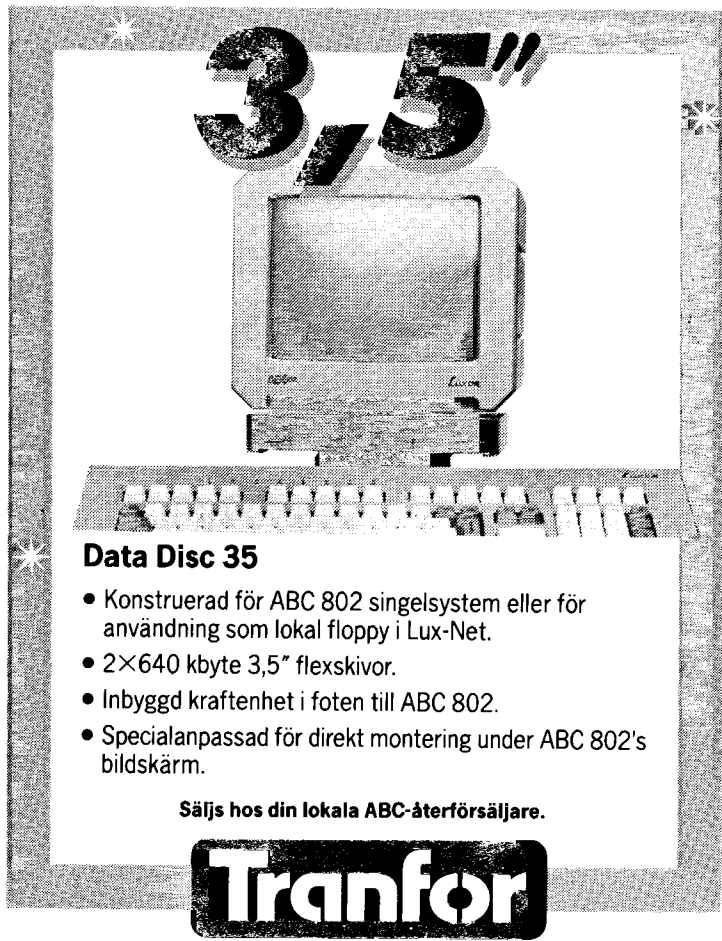
Sven Wickberg

# Mera räknekul med CBS

```

10 REM CBS
20 REM kollar konvergens till 1
30 REM jämna termer halveras
40 REM ojämnna blir 3* +1
50 REM -----
60 ; CHR$(12%)'Undersökning av CSB-kon-
   vergens'
70 ; ; ; 'Fr o m tal: ' ; : INPUT A1$
80 ; ; ' T o m tal: ' ; : INPUT A9$
90 REM -----
100 REM gör tabell
110 ; ; ; 'Tal', 'Antal termer'
120 A2$=A1$
130 GOSUB 170
140 A2$=ADD$(A2$, '1', 0%)
150 IF COMP$(A2$, A9$)>0% THEN END ELSE
   130
160 REM -----
170 REM beräkning
180 S%=0%
190 A$=A2$
200 IF A$='1' THEN ; A2$, S% : RETURN
210 B$=DIV$(A$, '2', 0%) : C$=DIV$(A$, '2',
   1%)
220 IF COMP$(B$, C$)=0% THEN A$=B$ ELSE
   A$=ADD$('1', MUL$(A$, '3', 0%), 0%)
230 S%=S%+1%
240 GOTO 200

```



## ERBJUDANDE TILL ABC-KLUBBEN !!!

Eftersom vi masskopierar ABC-disketterna, är vi nu i tillfälle att erbjuda NYA NASHUA DISKETTER MED KRAFTIG RABATT TILL MEDLEMMARNA. NASHUA tillverkas i USA och kallas "spinning discs" med Black label.

Viken dessutom erbjuder **ETT PARTI MED** "Blue/White" etikett, helt nya, levereras i plastpåse utan ask. Kan levereras i plastask mot ett pristillägg av 3:- per skiva för 5 1/4" och 4:- st för 8".  
**Bäst i satskontorets stora test.**

Bland 17 olika disketter var NASHUA, Storgemaster och Datalife!! de enda disketter som klarade STATKONTORETS opartiska test U:56, utan något fel. VI KAN ERBJUDA DENNA KVALITETSDISKET TILL MYCKET FÖRMÅNLIGT PRIS (delvis med hjälp av en lägre dollarkurs).

**LÅGT NETTO FÖR STORFÖRBRUKARE!!!**

**OBSERVERA ATT DETTA ÄR ETT MYCKET BRA KÖP FÖR SKOLOR OCH INSTITUTIONER samt förstås FÖRETAG OCH ANDRA STORFÖRBRUKARE.**

| TYP kvalité                                                                     | Pris per/st för, | "Black label" |             | "Blue/White" |             |
|---------------------------------------------------------------------------------|------------------|---------------|-------------|--------------|-------------|
|                                                                                 | Ca butikspris    | 10 st         | 100 st      | 10 st        | 100 st      |
| <b>3 1/2" Disketter:</b>                                                        |                  |               |             |              |             |
| <b>MF-1, SS/DD 135 TPI</b><br>(1x80 spår, för bl a DD-32, Macintosh)            | <b>67:90</b>     | <b>42:-</b>   | <b>38:-</b> | <b>-</b>     | <b>-</b>    |
| <b>5 1/4" Disketter:</b>                                                        |                  |               |             |              |             |
| <b>MD-1D, SS/DD 48 TPI</b><br>(1x40 spår, för bl a FD2/ABC-830/DD-82)           | <b>30:90</b>     | <b>-</b>      | <b>-</b>    | <b>16:-</b>  | <b>13:-</b> |
| <b>MD-2D, DS/DD 48 TPI</b><br>(2x40 spår, för bl a FD2D, ABC-830, DD-84)        | <b>42:-</b>      | <b>22:-</b>   | <b>19:-</b> | <b>19:-</b>  | <b>16:-</b> |
| <b>MD-1F, SS/DD 96 TPI</b><br>(1x80 spår, för bl a ABC-832/DD-56)               | <b>40:-</b>      | <b>-</b>      | <b>-</b>    | <b>20:-</b>  | <b>17:-</b> |
| <b>MD-2F, DS/DD 96 TPI</b><br>(2x80 spår, för bl a ABC-832/DD-56)               | <b>49:-</b>      | <b>29:-</b>   | <b>25:-</b> | <b>24:-</b>  | <b>20:-</b> |
| <b>CD, DS/DD 96 TPI "STORAGE MASTER"</b><br>(2x80 spår, för bl a ABC-832/DD-56) | <b>49:-</b>      | <b>29:-</b>   | <b>25:-</b> | <b>-</b>     | <b>-</b>    |
| <b>8" Disketter:</b>                                                            |                  |               |             |              |             |
| <b>FD 2D, DS/DD 26x256 byt sek.</b><br>(2x77 spår, för bl a ABC-838, DD-88)     | <b>51:90</b>     | <b>39:-</b>   | <b>35:-</b> | <b>24:-</b>  | <b>20:-</b> |

Disketterna passar till alla ABC-maskinerna samt dessutom till de flesta PC såsom, IBM, Appel, Philips, Commodore, Compis m fl. Rekiviera vår broschyr om Du är osäker om de passar, då kan Du se hela vårt sortiment av flexskivor, även streamerkassetter, rengöringsdisketter mm.

**Beställ nu !!! Alla priser är med moms, fraktfritt vid minst 50 st. Betalar Du beloppet på POSTGIRO nr: 33 33 99 - 4 så slipper Du dessutom postföretagsskattavfägen.**

**LOVISEBERG AGENTURER AB**

**Postnr 171 71 Solna, telefon: 08-85 50 50**

— Tillverkare av massminnen till ABC-datorerna sedan 7 år! —

TRANFOR DATA AB · SOLLENTUNAVÄGEN 225 · BOX 227 · 191 23 SOLLENTUNA  
TELEFON **08-96 01 80** · TELEX 15332 TRANFOR S

# Från 80 till 800

## Ideer om programkonvertering

Den gamla trojänaren ABC80 tillverkas inte längre. Även om mycket tyder på att den kommer att vara i omlopp länge än, måste man räkna med att många f d ABC80-ägare vill eller måste gå över till ABC800, dvs ABC800/802/806, och att de vill ta sina gamla program med sig. (I fortsättningen menar jag alla tre när jag säger ABC800.)

Det finns också en enorm programbank för ABC80-program som nyttillkomna ABC800-ägare skulle vilja skörda ur.

Slutligen kan vi finna problemställningen att vilja köra i princip samma programkod på båda maskintyperna. Antingen vill eller måste man använda samma disketter eller också är båda maskintyperna anslutna till förslagsvis CAT-NET.

Eftersom jag har erfarenhet av samtliga alternativ har jag hunnit filosofera en del över Luxors s k kompatibilitet. Kanske kan mina erfarenheter vara till nytta också för andra.

## Kompatibla uppåt, sa Luxor

I utvecklingsarbetet för nya datortyper har Luxor redovisat den lovvärda ambitionen att maskinerna skall vara "kompatibla uppåt", dvs att program från en tidigare maskin också skall kunna köras på en senare.

Detta innebär att det nästan alltid är möjligt att läsa in den gamla ABC80-programmen till den nya datorn. Däremot är det inte säkert att det går att köra dem där. Det måste ofta först göras en rad modifikationer. Det finns nämligen vissa hinder som har att göra med dels lagringsmediet (kassett, skiva) och dels programskrivningen.

Det betyder att det ingalunda är säkert att samma program kan köras på båda maskintyperna. Dock är det en avsevärd vinst att man inte behöver knappa in programkoden i dess helhet.

## 1: Kassetterna

Med den första ABC80 följde en bandspelare. Man hade visserligen en del trassel med den och det kom så småningom en ny modell. Samma bandspelare kunde kopplas till ABC800, men det skedde en produktutveckling så att ABC800 kunde spela in och av med mycket större hastighet. Det betyder att ABC80-kassetter fortfarande kan läsas av ABC800-bandspelare, men inte tvärtom.

Det gäller alltså att föra över sina program från kassett till diskett.

## 2: Skivorna

Skivspelarna har utvecklats: från enkel packningstäthet till dubbel, från enkelsidig till dubbelsidig och vidare till kvadrupel täthet. Inför varje steg har det funnits en möjlighet att läsa över informationen från föregående steg till det nya, men efter ett tag kan man vara i den situationen att man inte längre med sin nyaste maskinpark kan läsa sina äldsta disketter. Det gäller att som engångsföreteelse föra över allt man är intresserad av till det nya formatet.

Detta leder till problem för amatörer och skolor med de äldre typerna. Ofta är man hänvisad att behålla det man har - och varför inte? Grejorna kan fås billigt och de fungerar fint länge ännu. Men det kan som sagt bli problem med anpassning till eller snarare från senare varianter.

Ännu värre kan det bli om flera olika skivstorlekar blir populära. 8"-skivorna tycks vara mycket hårt standardiserade och under

UFD-doset i ABC800 finns program som gör det möjligt att koppla ihop 5"- och 8"-apparater till samma dator och spela över från den ena till den andra.

Men det är knappast något för hobbyhackaren som skall bekosta allt ur sin privat-ekonomiska ficka.

Nu står 3.5"-disketterna i farstun, och då kan det bli mera problem. Men låt oss inte göra oss bekymmer i förväg.

## 3: Programmen

Processorn i en dator förstår sig bara på ettor och nollor. Om programmet är skrivet i BASIC måste det först tolkas och överföras till binär form som datorn förstår. Detta tar tid och gör att BASIC-program på de flesta datorer går ganska långsamt.

I Luxormaskinerna har man ett mellanläge. När programmet laddas in i arbetsminnet - vare sig man skriver från tangentbordet eller laddar in det från skiva/kassett - sker en första omvandling (kompilering) så att BASIC-texten förvandlas till sifferkoder som är lättare för maskinen att läsa. Därför går ABC-BASIC ganska fort om man jämför med de flesta andra BASIC-dialekter.

Sparas ett program med SAVE kommer det att lagras i denna kompillerade form. I programlistan står extension .BAC, om man inte "för hand" har valt en annan beteckning. I de flesta fall är det en fördel med .BAC-lagring. Programmen laddas och körs snabbare, även om de vanligen tar mer plats på skivan/kassetten.

ABC80 och ABC800 har olika sätt att kompilera BASICen och kan därför inte läsa varandras .BAC-program. Man får feltexten "fel programformat" om man försöker.

Sparas programmet i stället med LIST lagras det exakt som det skrivits på skärmen och i ASCII-format, dvs alla tecken lagras med siffervärde motsvarande ASCII-koden. I programlistan står extension .BAS.

Denna lagring har några fördelar. Dels tar den vanligen mindre plats än .BAC-formatet, dels kan den läsas som en textfil. En textfil är nämligen ASCII-lagrad på samma sätt. Och eftersom ASCII är en (nästan) gemensam standard för mikro-datorerna, kan program i ASCII-format överföras mellan dem med de vanliga överföringsrutiner som finns.

Skall man köra ett ABC80-program på en ABC800 spar man programmet med LIST <programnamn>, för över skivan till ABC800-systemet och skriver RUN eller LOAD på vanligt sätt. Då laddas programmet in och förvandlas samtidigt till det halvkompilerade format som ABC800 vill jobba med.

SÅ LÅNGT är det enkelt.

## Programanpassning???

Har man TUR går nu programmet att köra. Men det har man nog inte. Redan vid inladdningen anges vanligen några radnummer med olika slag av felmarkering som först måste rättas till. Det finns nämligen skillnader i ABC80-BASIC och BASIC II, som maskinen inte själv kan reda ut. De aktuella programraderna markeras i listan med ett antal frågetecken, så de är relativt lätta att hitta.

## Nödvändiga ändringar

Här är några av de vanligaste rättelser som måste göras.

\* END måste stå ensamt på en rad.

Om man har en rad som t ex

IF <villkor> THEN END

får man antingen byta END mot STOP eller också göra ett hopp till lämplig plats där man kan skriva sin END-rad.

\* Instruktionen ASC(tecken) måste bytas till ASCII(tecken). På de ABC806:or där jag har provat tycks ändringen ske automatiskt vid inmatningen. Men OBSERVERA att om man då spar programmet från ABC800-maskinen, kommer programmet i fortsättningen att innehålla varianten ASCII som ABC80 inte kan läsa.

\* I vissa fall tolereras inte en REM-sats i slutet på en programrad. Detta kan ha med kolonet att göra (se nedan) men jag har märkt det särskilt i rader som börjar med IF:

IF <villkor> THEN <instruktion> : REM <förklaring>

Här måste man antingen ta bort REM-satsen eller skriva med utropstecken:

IF <villkor> THEN <instruktion> ! <förklaring>

Det senare är ingen nackdel om man i fortsättningen bara kör programmet på ABC800.

I en rad andra fall, som jag inte kunnat totalt analysera, krånglar ABC800 om det kommer en kolonsats efter en IF-sats. Någon har påpekat att om man ändrar THEN till GOTO går det bra, men jag rekommenderar att man försöker dela upp raden på flera. Ev får man då upprepa villkoret på flera rader.

## Dolda fel

Somliga fel visar sig inte i förväg utan först när man börjar köra. Några exempel:

\* ONERRORGOTO 0 och BLANK-TECKEN

Det skall heta ON ERROR GOTO (ingen nolla och två mellanslag). Om datorn står i EXTEND-läge, vilket nog är det vanliga, blir blanktecknen i vissa lägen viktiga. Man kan inte skriva LIST130 utan det måste vara LIST 130.

På grund av en bug i ABC80-tolken skrivs ett extra blanktecken på vardera sidan om varje kolon vid LIST. Dessa blanktecken försöker en ABC800 läsa in en mening i, och ibland måste de tas bort för att programmet skall fungera.

Över huvud taget får man se upp med programrader som innehåller flera satser åtskilda av kolon. Jag har inte helt lyckats reda ut varför det ibland hakar upp sig. Ett ON ERROR GOTO <nr> måste ibland flyttas ut och stå i en rad för sig.

Speciellt lurigt kan det vara i satser av typ

CHAIN "DR0:MENY" och DATA DR0:MENY

eftersom ABC80 sätter in mellanslag "DR0 : MENY" som BASIC II inte kan tolka rätt.

## \* Mera ON ERROR

ABC800 fångar upp många fel som inte ABC80 klarade. På det lilla kortet under tangentbordet på ABC80 finns en prick vid de fel som kan fångas upp med ONERRORGOTO. På ABC800 kan alla fel uppfångas, även vissa sådana som på

ABC80 ger programavbrott av mera allvarligt slag. Detta kan ibland göra det svårt att hitta felet.

Jag har stött på följande exempel. I en meny skall man välja att göra CHAIN till ett visst program. Programmet hittas och ABC800-datorn försöker ladda in - men det visar sig vara ett ABC80-program i BAC-format. Då träder felhanteraren i funktion och man återvänder till menyn (i det här fallet) precis som om man inte gjort något val. ON ERROR skulle fånga upp att man t ex skrev en bokstav i st f en siffra vid valet, och man tänker inte genast på den här typen av fel.

Lösning: Ta bort ON ERROR och kör programmet igen. Då blir det programavbrott, man kan se på vilken rad och med ;ERRCODE får man veta vad det var för fel. Sätt sedan dit ON ERROR igen.

#### \* INPUT A\$,B\$

Om ett ABC80-program kräver inmatning av flera variabler i samma INPUT-sats, skall inmatning ske med kommatecken emellan: "Sven","Wickberg". Om man matar in bara det ena värdet skriver datorn ett nytt frågetecken och väntar på nästa.

Så fungerar inte BASIC II. Den SKALL ha BÅDA värdena med komma emellan, annars blir det felmarkering och ev programavbrott. Det gäller alltså att se till att man har ett ON ERROR etc på plats före en sådan inmatning. Och återhoppet måste ske till just den ON ERROR-raden, annars blir det programavbrott om felet upprepas.

#### \* RND och avrundning

Instruktionen RND ger i båda maskintyperna decimaltal mellan noll och ett. Vill man slumpdra ett heltal mellan 1 och 6 (tärningskast) skriver man INT(RND\*6)+1.

RND\*6 ger tal större än noll men mindre än 6. INT() trunkerar (klipper bort) decimalerna och ger heltal 0-5. För att få dem 1-6 lägger man till 1.

Det hela tar mindre plats och går fortare med heltalsvärden och på ABC80 kan man enklast skriva:

```
X%=RND*6%+1%
```

Något INT() behövs inte eftersom decimaltalet på höger sida automatiskt förlorar sina decimaler genom trunkering när det tilldelas heltale X%.

Men i ABC800 spelar oss BASIC II ett spratt. När flyttal förvandlas till heltal på detta sätt, genom tilldelning till heltalsvariabel, sker inte TRUNKERING som i ABC80, utan KORREKT AVRUNDNING.

Man kan alltså räkna ut för att få talet 7 med ovanstående rad!

Lösningen är förstås enkel: ingen smartprogrammering - ta med INT() som det var från början.

#### \* INP(56)

Det säger sig kanske självt att det som inte är BASIC inte kan läsas av en annan maskin. Instruktioner som börjar POKE, PEEK, OUT, INP och CALL fungerar vanligen inte om man inte ändrar argumenten, dvs de siffror som anger adresser och portar som man vill göra något med.

INP(56) i ABC80 avläser tangentbordet och håller reda på dels om en tangent tryckts ned sedan sist och vilken tangent det var. Dessa funktioner ser annorlunda ut i BASIC II.

SYS(5) är tangentbordsflaggan - 0 om ingen tangent tryckts, annars 128. Nollställs med GET.

Z=SYS(6) för över ASCII-tecknet för senast nedtryckta tangent till inmatningsbufferten, där den kan avläsas med GET.

```
Z=SYS(6):GET W$
```

Uträttar alltså samma sak som:

```
W=INP(56):W$=CHR$(W)
```

#### \* CHAIN

I ABC80 betyder CHAIN "" detsamma som END+NEW. Detta fungerar inte i BASIC II, där man i stället får skriva CHAIN "NUL:". Problemet kan dyka upp i en meny där olika menyval länkar till olika program och man vill ha ett val för "AVSLUTA". Man kan naturligtvis också nöja sig med att i det fallet få programavbrott och felmarkering, även om det inte ser snyggt ut och kan irritera den ovane.

#### \* EXTEND : INTEGER

I EXTEND-läge, som vi redan varit inne på, kan BASIC II ta emot långa variabelnamn, vilket för det mesta är en fördel. Nackdelen är att mellanslag i vissa fall blir betydelsefulla på ett sätt som ställer till problem.

Ofta är det också en fördel med INTEGER-läget. Man slipper skriva alla % efter variabelnamnen. Men om man laddar in (resp sparar) ett ABC80-program som är skrivet utan % kan man råka ut för att variablerna tolkas som heltal. Detta kan vara mycket pinsamt om de i verkligheten är flyttal och skall avläsa decimaler. Felet kan vara svårt att se för den ovane, eftersom heltalen i INTEGER-läge ser ut "som vanligt" (medan flyttalen skall ha en punkt efter sig).

Det kan vara klokt att ligga i NO EXTEND:FLOAT-läge när ABC80-program laddas in.

#### \* Spara med SAVE

När alla anpassningarna är gjorda och programmet fungerar är det vist att spara det i sin ABC800 med SAVE. Detta går bra om man i fortsättningen alltid skall köra det i ABC800. Men det skall man kanske inte alltid och då har vi nästa problem:

#### 3: Samtidigt för ABC80/800

Hur gör man om man vill ha program så skrivna att de kan avläsas av och köras i både ABC80 och ABC800?

Ingenjörsfirman CAT har i sitt nätverk alla programrutiner tillgängliga för båda maskintyperna. Flertalet program är lika, men i några fall skiljer man dem åt och kallar dem då .80 resp .800.

Kan de så kan väl vi, i varje fall det som går att göra i BASIC.

För det första måste programmen vara lagrade i .BAS-format. Detta leder till att mycket långa program kommer att kännas en aning trögstartade, i varje fall i ABC800. Programmen skall först läsas in och sedan kompileras. Men det får man stå ut med. Är programmen någorlunda korta märks det inte så mycket.

För det andra gäller det att undvika instruktioner som ser olika ut i de olika maskinerna. Man måste givetvis avstå från BASIC II-finesser av alla slag, och man får sorgfälligt undvika de skrivsätt som resp maskin inte själv kan rätta till.

Man får lov att köra i NO EXTEND:FLOAT. Man kan skriva ONERRORGOTO <n>, vilket kan tolkas av ABC800, men man får avstå från ONERRORGOTO 0. Ett sätt kan vara att i stället för nollan skriva ett radnummer som ligger sist i programmet med bara REM på. Man räknar ju inte med några fel i det läget. (Ännu smartare: i stället för rem-sats skriv ;Oväntat fel!)

Man slipper felet med mellanslag kring kolon genom att skriva in och spara motsvarande program med ABC800-maskinen som inte gör några sådana mellanslag. Annars kan man alltid sätta ut anföringstecken:

```
DATA "DRI:PROGRAM" CHAIN
"SYS:MENY"
```

CHAIN"" kan ersättas med CHAIN "NUL:" som visserligen ger felmeddelande i ABC80, men det gör inte så mycket om meningen är att avsluta körningen.

#### \* INPUT och DATA

I ABC80 tas alla mellanslag bort, medan i ABC800 bara mellanslag i början och slutet av strängarna avlägsnas.

Kan lösas med att sätta citationstecken kring alla data resp att använda INPUTLINE. (Det heter INPUT LINE i 800 och sparas på detta sätt om man skriver LIST från ABC800 - och då kan inte ABC80 läsa det.

#### När det inte går

Om man ändå står där med instruktioner som MÅSTE vara med, men som är inkompatibla: INP(56), PEEK och POKE?

Ett alternativ kan vara att låta programmet pröva vilken maskin som är inkopplad. Om detta har det varit en del diskussion i klubbens MSG-system. Men följande adresser kan vara användbara: 39 och 14.

Address(39) är noll för ABC80 och icke noll för de andra. Det betyder att PEEK(39) är sant om en ABC800 är inkopplad, annars ej. (Detsamma gäller om Address(14) - men jag har ännu inte kommit på vad det annars är för skillnad eller varför det här så här.)

Så man kan skriva:

```
IF PEEK(39) THEN <800-instr> ELSE <80-instr>
```

I värsta fall måste man ha olika program. Då kan man börja körningen med ett program som frågar vilken maskin man har (om det inte kan undersökas automatiskt som ovan) och sedan gör CHAIN till rätt variant.

Allting går om man bara har tid och pengar. Pengar har hackern inte för gott om, men påhittigheten har visat sig enorm inom klubben.

Jag väntar med stort intresse på synpunkter och ytterligare knep.

#### Nästa steg

Nu har (tyvärr) IBM intagit Luxors ledande plats i smådatorförsäljningen. Allt fler inrättar sig efter IBM-marknaden, så även CAT.

I ABC-bladet nummer 4, 1985 fanns inte mindre än TVÅ annonser om konvertering av ABC-program till IBM-standard. En vacrer dag (den kanske är här när detta kommer i ABC-bladet) behöver vi en artikel om anpassning till IBM i CAT-NET med både Luxor och IBM anslutna.

#### Vem skriver den artikeln?

<1384>

Sven Wickberg

# Reaktionstest

Det finns här och där i landet apparater med vilka man kan mäta sin reaktionshastighet. Man lägger i en slant (förr var det 10 öre) och väntar sedan på en signal. Vid signalen börjar myntet falla och man stoppar det genom att trycka på en knapp. Myntet fastnar då längs en skala som talar om hur lång falltiden har varit. Sälunda kan man få reda på sin egen reaktionsförmåga (och bedöma om man bör köra bil mer den dagen).

I Avancerad programmering på ABC80 fanns ett program som utträttade något liknande. Härmdagen hade jag anledning att snabbt smälla ihop ett reaktionsprogram och fann då att det kunde göras bra mycket enklare.

## Programförklaring

Programmet är kanske inte så märkligt i sig, men det innehåller några intressanta delrutiner, nyttiga att lära för novisen - om vi nu längre har några noviser på ABC80 som inte längre tillverkas...

Först gäller det att få en signal att uppträda slumpvis. Med ABC80:s slumpvals-generator kan man komma in på oförutsedda delar av lyckohjulet med RANDOMIZE, som alltså måste stå först (100). Sedan har jag valt att dra slumpalt efter slumpalt och göra signalen om slumpalet är mindre än 0,005. Vid större tal kom signalerna så fort att man knappt hann med, vid mindre är det för långt mellan dem, men det kan vara en smaksak som var och en kan pröva sig fram till.

Om RND-villkoret är uppfyllt sätter vi i gång klockan på rad 200. Eller rättare sagt, vi ställer klockan.

## Klockan

ABC80:s klocka har den egenheten att den räknar ner klockbyrtarna baklänges. Adressen 65008 rör sig fortast och går från 255 ned till 1, ett steg för var femtiondels sekund.

När den räknat ut ett varv, vilket tar 5,12 sekunder, förs "ett i minne" över till 65009 som i sin tur räknar ned från 255 till 0 och behöver ungefär 20 minuter för att gå varvet runt och göra motsvarande manöver på 65010. (I själva verket är det krångligare än så, men detta räcker i vårt sammanhang.)

För de flesta ändamål behöver man bara de två första adresserna. Jag har valt att 255-ställa endast dem i rad 200.

## I väntan på trycket

Nu väntar programmet på e? tangenttryckning som registreras genom att INP(56) får 128-biten ettställd. I de övriga bitarna lagras ascii-värdet för den nedtryckta tangenten, men det använder vi inte oss av här. Så länge INP(56)<128 stannar programmet på rad 210.

När tangenten nedtrycks går programmet till nästa rad och nu gäller det att ta tid. Den metod som visas är ett smart sätt att få fram tiden i sekunder. Den bygger på Booles algebra (dvs logiska variabler)

och har utförligt beskrivits i ett tidigare nummer av ABC-bladet (1983:4 sid 12).

Raderna 500-520 är nog självförklarande, men observera att P och M inte får vara heltal. (Detta är en speciell fälla för ABC800-ägaren som kan ha apparaten i INTEGER-läge, varav mera nedan.)

På rad 600 väljer jag att låta skriva på skärmen tiden för reaktionssnabbheten I, antal försök J och medelvärde M.

M? Så står det inte! Nej, det som står är M korrekt avrundat till två decimaler.

## Avrunda till 2 decimaler

Den kortaste tid man direkt kan mäta genom att avläsa adress 65008 (den som rör sig snabbast) är 1/50 av en sekund, eller 0,02 sek. Men datorn minns I som 0,02000 sek, även om slutnollorna inte skrivs ut. Vid medelvärdesberäkningen, t ex division med 3, går det inte jämnt upp och det kan bli massor av onödiga decimaler, som inte ger någon som helst mening. Beräkningsnoggrannheten kan inte bli större än plus/minus 0,02 sek, alltså två decimaler i detta fall.

Egentligen är även detta för stor noggrannhet, eftersom datorn bara avläser jämna hundraleds sekunder, medan medelvärde även kan bli udda, men låt gå för det.

Om man vill klippa bort decimaler på ABC80 kan man inte göra det direkt. Man måste först se till att alla de siffror man vill spara finns i en heltalsdel. Sedan kan man med INT() klippa bort decimalerna. Därefter får man flytta tillbaka decimalkommat till rätt plats.

Med den metoden sker s k trunkering, dvs man bryr sig inte om hur de bortklippa decimalerna ser ut. Vill man ta hänsyn till att sista siffran skall höjas om den första borttagna siffran är 5 eller mer, måste man göra ett par ytterligare manövrer, och det är vad som skett här.

Först multipliceras M med 1000 vilket innebär att vi flyttar upp TRE decimaler till heltalsdelen. Sedan tilläggs 5, som ökar den sista siffran. Om denna är 5 eller mer kommer även den näst sista siffran (= decimal TVÅ) att höjas. Det var ju detta vi ville.

Nu divideras med 10, vilket gör att den tredje decimalen, som vi alltså inte vill ha, hamnar till höger om decimalkommat och försvinner när vi gör INT(resultatet).

Genom att dividera återstoden med 100 får vi tillbaka decimalkommat på rätt ställe och två, korrekt avkortade, decimaler.

Puh! Det är nog lättare att göra denna operation än att förklara den. Hängde du med? Om inte finns bara ett råd: sälj din ABC80 och köp en ABC800. Där kan du få hela operationen utförd "gratis" med PRINT USING "\$\$. \$\$" M - fast då måste du vara säker på att du inte behöver fler heltals-siffror än du sätter ut "brädgårds-tecken" för till vänster om decimalpunkten.

## Tangentbordsbufferten

Nu återstår bara att förklara rad 120. Jag råkade ut för att trycka för tidigt, alltså innan det kommit något pip. Tangentbordet har en buffert som lagrar ett tecken, och därmed kom denna för tidiga tryckning att "ligga på lager" till nästa pip. Följaktligen fick jag tiden noll, och det blir en aning fuskigt.

Tangentbordsflaggan finns i adress 65013 på ABC80 och genom att nollställa den avlägsnas spåren av den för tidiga tryckningen. (Det går också bra med POKE 65060,0 som har samma verkan, men av en annan anledning.)

Så där, programmet är klart! Man måste avbryta med CTRL-C, såvida man inte utnyttjar de lägre bitarna i INP(56) för att definiera någon stopptangent. (Hemläxa)

Men vänta ett tag! I dessa tider blir en artikel om ABC80 inte komplett utan ett

## Tillägg för ABC800-ägare

På ABC800-serien finns inte INP(56)-möjligheten, utan man får använda något annat. Dessutom fungerar klockan på ett annat sätt, varför programmet måste kompletteras med ett par speciella rutiner.

## SYS(5,6,8)

Enligt handböckerna är SYS(5) en tangentbordsflagga. Det betyder här att SYS(5)<>0 ända till dess en tangent tryckts ned. Vid tangenttryckningen får SYS(5) värdet 128 som står kvar till dess man utför GET eller INPUT.

Om man inte behöver veta vilken tangent som tryckts (och det behöver man inte i programmet REAKTION) kan man alltså ersätta den aktuella raden med:

```
210 IF SYS(5) GET Key$ ELSE 210
```

Om man behöver veta vilken tangent som tryckts - man kanske skulle vilja avsluta med S i stället för med CTRL-C - använder man SYS(6). Den har egenskapen att återföra värdet för senast nedtryckta tangent till tangentbordsbufferten, där den avläses med t ex GET.

```
Z=SYS(6)      ! återför värdet av
                senaste tangent
GET Tangent$   ! avläs tangentbordsbufferten
```

och sedan jobbar man vidare med värdet Tangent\$ t ex

```
IF Tangent$='S' THEN STOP
```

Observera att ordet END måste stå ensamt på en rad i BASIC II. Det är därför jag använder STOP här.

## Klockan

Klockan i ABC800-serien är också litet annorlunda än hos ABC80. För det första ligger är de aktuella adresserna

```
-14 timme
-13 minut
-12 sekund
-11 femtiondels sekunder
```

Det är alltså -11 som rör sig fortast (och bör avläsas först). Den går från 1 till 49 och lämnar "ett i minne" hos -12 när den kommer till 50.

Adressen -12 i sin tur går från 0 till 59 och lämnar ett i minne hos -13 vid 60.

(På samma sätt räknar -13 upp till 59 minuter och -14 upp till 23 timmar - och för den delen -15 till 31 dagar, -16 till 12 månader och -17 till 99 år. Allt det här

får man ut automatiskt med ;TIME\$. I ABC806 finns en batteridrivna klocka som man inte behöver ställa mer än en gång, medan man i ABC800 måste ställa klocka och datum efter varje kallstart. Hur det är med klockan i ABC802 vet jag inte - den tycks kunna gå litet hur som helst om man har otur.)

Om man för reaktionsprogrammet nöjer sig med att notera hela och delar av sekunder kan man skriva raderna 200 och 220 som nedan:

```
120 POKE 65506%,0%
200 POKE -12%,0%,0%
---
220 I=PEEK(-11%)/50+PEEK(-12%)
---
600 ; i, J, : ; USING "$$. $$" M
```

Jag har använt notationen PEEK(-11) i stället för PEEK(65525), eftersom det första verkar vara lättare att komma ihåg. På ABC80 bör man dels använda den positiva versionen och dels skriva den som heltal, annars både går det långsammare och tar mer plats. Jag vet inte om detta gäller även ABC800, men det kan vara ide att undersöka saken.

#### Varning för heltal

ABC800 har fördelen att kunna ställas i INTEGER-läge så att man slipper alla %-tecken när man skriver programmet. Har man otur skriver man alltså heltalsvariabler även där det inte skall vara några. Det hände mig när jag skulle konvertera programmet till ABC800. Jag blev förvånad över att få tiden noll, vad jag än gjorde!

Både I, P och M måste vara flyttal, endast J är heltal.

#### Bufferten, och PRINT USING

Rad 120 nollställer tangentbordsbufferten precis som i ABC80- programmet men det är en annan adress, och PRINT USING omnämndes tidigare i denna artikel. På min apparat kunde instruktionen USING inte användas på samma rad som ;I, J, - jag vet inte varför (har för mig att det har lyckats i andra sammanhang). I varje fall gick det bra med ett kolon emellan. Man kan också skriva PRINT USING på en särskild rad, men jag ville behålla likheten mellan programversionerna.

#### Om inte ABC80 fanns...

I ett program från början skrivet för ABC800 skulle jag troligen använda fler-radiga funktioner för att definiera procedurer, och dessutom långa variabelnamn, men här ville jag bara göra en enkel konvertering av ABC800-programmet. Ofta kan man skriva programmen så att de går att läsa och köra på både ABC80 och ABC800, men det går jag inte inte på nu.

#### Hur fort går det?

En sista något undrande anmärkning. När jag provar ABC800-rutinen får jag kortare reaktionstider än i ABC80-rutinen. Eftersom det är högst tveksamt om en ABC800 kan ha ett sådant förändrande inflytande på min fysik (jag har förresten jobbat med en ABC80 försedd med Gejos RAM-minne och inladdad ABC800-tolk), måste det bero på mindre "spilltid" vid ställning och avläsning av klockan. Hur det hänger ihop med den saken överlämnas i vanlig ordning åt läsekretsen att försöka fundera ut.

<1384>  
Sven Wickberg

```
10 REM reaktion
100 RANDOMIZE
110 IF RND<.005 ; CHR$(7%); ELSE 110
120 POKE 65013%,0% : REM neutralisera f
    eltryck
200 POKE 65008%,255%,255%
210 IF INP(56%)<127% THEN 210
220 I=(PEEK(65008%) XOR 255%)/50+5.12*(
    PEEK(65009%) XOR 255%)
500 J=J+1 : REM antal försök
510 P=P+I : REM totaltid för försöken
520 M=P/J : REM medelvärde
600 ; I,J,INT((M*1000+5)/10)/100
700 GOTO 110
```

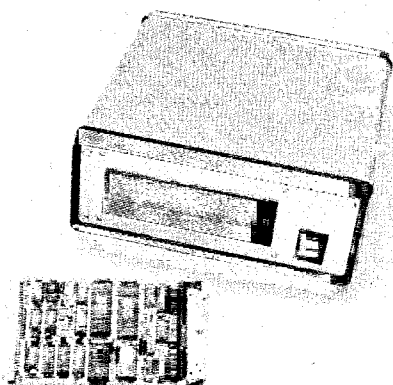
```
10 REM reaktion.800
100 RANDOMIZE
110 IF RND<.005 ; CHR$(7%); ELSE 110
120 POKE 65506%,0% : REM neutralisera f
    eltryck
200 POKE -12%,0%,0%
210 IF SYS(5%) GET W$ ELSE 210
220 I=PEEK(-11%)/50+PEEK(-12%)
500 J=J+1% : REM antal försök
510 P=P+I : REM totaltid för försöken
520 M=P/J : REM medelvärde
600 ; I,J%, : ; USING "$$. $$" M
700 GOTO 110
```

**BILD & DATA**

RING 026- 14 24 64 75/1200  
DATA, PRISER och ANNONSER

**Prisgenombrott!**

**Prisras för Winchestrar:**



**12,5 MB: 12 500:- kr – 0,1 öre/byte**  
**20 MB: 14 500:- kr – 0,0725 öre/byte**

Du kan nu enkelt och billigt förse din ABC 800, 802 eller 806 med en hårddisk på 12,5 Mb. Hårddisken levereras i en separat låda med XEBEC 1410 A kontrollkort och eget 65 W nättaggregat. Du placerar det medföljande sk host-adaptorkortet i din datorbuss och ansluter den 50-poliga flatkabeln till hårddiskenheten. Svårare än så är det inte!

Till BILD & DATAs hårddisk medföljer allt du behöver. Det ingår en färdigformaterad hårddisk BASF 6188 (12,5 Mb) med kontrollkort och strömförsörjningsaggregat i lådan. Hårddisken har halv standardhöjd. Det medföljer också host-adaptorkort, kablage och

installationsmanual. Inom kort kan vi också erbjuda hårddisk BASF 6188R (20 Mb).

Din dator innehåller den nödvändiga programvaran för att köra hårddisken om Du har UFD-DOS.

Moms tillkommer på samtliga priser.

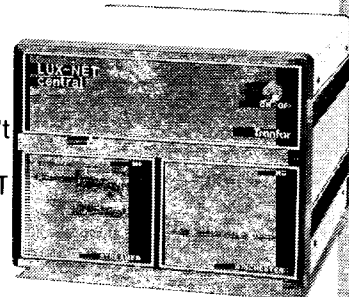
**BILD & DATA**

Oskarsgatan 1  
802 23 GÄVLE  
Telefon 026/14 24 38

## LUX-NET CENTRAL

### DataStore 60/ LUX-NET

- Konstruerad för användning som centralt massminne (20 eller 60 Mbyte) i ett LUX-NET system.
- 20 Mbyte bandbackup (filorienterad).
- 640 k floppy (låsbar) för programladdning.
- 15 positioners bakplan för utbyggnad med tex centrala kommunikationskort.
- Extra fläktkyld för kontinuerlig drift.



Säljs hos din lokala ABC-återförsäljare.

**Tranfor**

— Tillverkare av massminnen till ABC-datorerna sedan 7 år! —

TRANFOR DATA AB · SOLLENTUNAVÄGEN 225 · BOX 227 · 191 23 SOLLENTUNA  
TELEFON 08-96 01 80 · TELEX 15332 TRANFOR S

# Möte ABC800

(Text 1803) Kent Berggren <6019>  
Ärende: ABC806 BILDMINNET  
Hur hanterar den den vanliga bildminnet.  
Ligger det på samma plats som i en  
ABC802?

(Text 1808) Benny Löfgren \* <2615>  
Ärende: Step på FD-55F  
Jodå, man kan läsa 40-tracksskivor på 80-  
tracksskivor, allt man behöver göra är att  
ställa om en switch på kontrollkortet,  
det låter sig göras även på 4112. Vidare  
behöver man ett program som är speciellt  
förberett för ändamålet (man måste t ex  
konvertera clusterstorleken), ett sådant pro-  
gram är COPYLIB, vilket ju alla har.

(Text 1810) Benny Löfgren \* <2615>  
Ärende: ABC806 BILDMINNET  
Det ligger på precis samma plats. Däremot  
så måste man göra lite special för att  
komma åt attributminnet, om du vill det.

(Text 1811) Göran Engelbo <369>  
Ärende: Angående adress 39  
Kommentar till inlägg i ABC-Bladet 3/85,  
sid 36. Det där om ABC80 är helt fel!

Så här är det:  
1 - 800M  
2 - 800C  
3 - 802  
4 - 806  
På gamla tolkar (800M, 800C och faktiskt  
också de allra första 802:orna) ligger i  
denna byte koden 10 som innebär att man  
inte vet vilken maskin det är. (DTC är  
alltså fel slutsats!)  
Ett tips är att titta på 65364. Den är 40  
på 800C och 80 på 800M om den inte  
stälts om från programmet. Den metoden  
täckers dock inte in de allra första 802:orna,  
men de är \*mycket\* få.  
Mvh/Eng

(Text 1812) Sven Wickberg <1384>  
Ärende: CAPS LOCK  
Jag testade OUT 34,8 etc från ABC-bladet  
och fann:

- 1) att det är tvärtom på den 806 jag provade på (PÅ=AV, AV=PÅ)
  - 2) när man slagit av den på detta sätt kunde man inte längre få på den (eller också tvärtom - ursäktat att jag inte skrev upp det och jag kommer inte att att prova nu - prova själva får ni se)
- Det vore kul med kommentarer från allvetarna.

(Text 1814) Benny Löfgren \* <2615>  
Ärende: CAPS LOCK  
Faktum är att denna 'finess' faktiskt står i manualen till datorn (minns inte om det var dator- eller BASIC-manualen).

(Text 1817) Patric Ljung <5455>  
Ärende: Mer om ABCUTE  
Vad syftar du på?  
Att man kan sätta flera prommar på varann går ju lika bra på vilken burk som helst.  
Man kan iofs använda 256 tecken på ABC802 men då måste man ange grafik-mod, vilket kan bli jobbigt. Eftersom bit 7 (värde 128) inte är kopplad till teckenprommet utan till en inverterare som inverterar tecknet på skärmen. Förövrigt genereras inte grafik för att aktuell kod ligger i bildminnet utan för kodens teckenfont har bit 6 och 7 satta på ett sätt som medför grafik. De möjligheter som finns är flash, Hide, Graphic, Normal.  
Detta innebär att man kan sätta en rad i grafik-mod med valfri kod. Förresten, är det någon som kan få Flash på sin 802a??? Jag kan med min, men inte min kompiss. Jag tror det beror på teckenprommet.  
Vill ni ha 802or testa? Skriv i FL5H 'Blinkar det???'  
Moss / Patric

(Text 1821) Patric Ljung <5455>  
Ärende: CAPS LOCK  
Det ligger faktiskt till så här:  
OUT 34,8 sätter på Capslock och gör det därmed omöjligt att stänga av den. När man sedan skriver OUT 34,136 stängs inte Capslock av utan det blir bara möjligt att stänga av den genom att trycka på den.  
Moss / Patric

(Text 1863) Kent Berggren <6019>  
Ärende: Bild-minnet ABC 802  
Jag har labbat lite med port 56 och 57. Men jag får det inte att fungera riktigt.  
out 56,12,57,190  
Det borde flyttat upp minnet till 49152, men jag får med massa skräp från gamla skärmen. Vidare skrivs nu inknäppt text oxo på nya skärmen. Det borde den väl inte göra. Jag har inte ändrat reg 14 o 15.

# Utdrag ur MSG-systemet

Är det någon som kan reda ut vad jag skall göra, för att få det att fungera riktigt.

(Text 1867) Kent Berggren <6019>  
Ärende: Bild-minnet ABC 802  
Är det så att den skall ha ascii värdet ut på porten?  
out 56,12,57,(+3 ascii värden)  
Jag blir inte riktigt klok när jag läser i bit för bit om CRTS 6845  
HJÄLP MEJ!

(Text 1888) Harald Nautsch <4479>  
Ärende: HR-grafik i CPM (?)  
En annan fråga: Skulle det vara möjligt att använda grafiken när man kör CPM på 806?

(Text 1889) Patric Ljung <5455>  
Ärende: HR-grafik (adressering)  
Hur du adresserar HR-minnet 128K på ABC806 står i ABC-bladet Nr 2 1984. Om du inte har den kan jag förklara, annars tycker jag det är onödigt, jag kan kommentera den litet.  
mvh / Patric

(Text 1909) Kent Berggren <6019>  
Ärende: Bild-minnet ABC 802  
Skriv out 56,13,57,190  
i ett prg, där efter gör du ett GET skall du se att minnet hoppar tillbaka igen.

(Text 1910) Patric Ljung <5455>  
Ärende: Bild-minnet ABC 802  
Tja GET är väl inte samma sak som interrupt, prova att skriva:  
10 OUT 56,13,57,190  
20 WHILE INP(5)=0  
30 WEND  
40 FOR I=0 TO 5000 : NEXT I  
35 OUT 34,7 : Piip  
Om du kör denna rutin kommer nog inte skärmen att hoppa tillbaka förrän loopen har körts efter pipet, och du trycker någon tangent.  
Det är nog GET-funktionen som ställer tillbaka minnet, inte interruptet.  
Moss / Patric

(Text 1913) Magnus Bodin <4304>  
Ärende: CTCR: Skillnader ABC800/2/6  
Vilka är de egentliga skillnaderna?  
Jag har bl.a. försökt slå om markörens utseende, men inte fått samma resultat på de olika maskinerna.  
Fråga2: Vilka tecken visas egentligen på skärmen om man lägger ned koder direkt i bildminnet? Hur ser PF-koder ut? Och koder < 32? Även här blir det skillnader på 800/2/6. Finns det några "osynliga" tecken på 806:an?  
Magiska hälsningar Magnus

(Text 1914) Bo Kullmar \* <1789>  
Ärende: Comporian  
Enligt ett brev från en advokat så gick Comporian i konkurs den 17:e dennes! Synd, för de är skyldiga klubben 3 775 för annonser i senaste ABC-Bladet. Vi får väl inget, misstänker jag. Detta skriver jag som en upplysning till er som använder deras program. Kanske tar något annat företag över, det skulle ine förvåna mig. Är det någon Linköpingsbo som vet mera? I så fall skulle vi kanske flytta diskussionen till fritt.

(Text 1919) Patric Ljung <5455>  
Ärende: CTCR: Skillnader ABC800/2/6  
1. På ABC800C är det anslutna, där kan man inte ställa om markören. Det är väl nåt liknande som på ABC80, 802/806/800M ska väl vara samma. Man kan på vissa burkar ange startlinje nedanför slutlinje, då får man markören "delad".  
2. På ABC806 visas alla tecken på skärmen, det vill säga att alla adress ledningar är kopplade => 256 tkn. från 0 till 255. Här ligger de första grafiktecknen på 1-31 (0 är blank), 32-127 ligger de vanliga tecknen. 129-159 ligger grafiktecknens andra del, (128 är blank), 160-255 ligger vanliga tecken igen (används inte) kan användas som alternativ tecken uppsättning. Jag gör det i min 806a.  
På ABC802 är bara 0-127 kopplade 128-255 ger samma tecken fast inverterat. 0 är tom, 1-31 innehåller div styrkoder för grafik, hide, flash, normal. 32-127 är vanliga tecken. Grafik tecknen återfås genom i GWHIT eller nåt dylikt.  
På 800M är jag inte så säker, det är väl bara 0-127 som används där med, men 1-31 innehåller inte några styrkoder, tja, jag vet inte.  
Himn, var det nåt mer?  
Moss / Patric

(Text 1920) Patric Ljung <5455>  
Ärende: Comporian  
Himn, de har väl inte gått i konkurs men måste se över läget. Jag pratade med Örjan häromdagen och han sa att det eventuellt skulle reda upp sig. Nästa vecka (första

hela veckan i November) skulle saken vara avklarad, dvs så skulle man veta då hur det blir.  
Vi får avakta och se. ABC-klubben är inte de enda, men det är inte så kul, jag har jobbat litet åt Comporian.  
Moss / Patric

(Text 1925) Benny Löfgren \* <2615>  
Ärende: CTCR: Skillnader ABC800/2/6  
Ett tillägg: i 800M kan man få fullständig 256 teckens uppsättning om man byglar om videokortet och sätter i en 2732-teckenprom i stället. Då fungerar det som på 806, dvs alla 8 bitarna används för att slå upp rätt tecken i teckengeneratoren. Kan med fördel utnyttjas för att få inverterade tecken eller underlinje.

(Text 1936) John Stagg <5741>  
Ärende: CTCR: Skillnader ABC800/2/6  
1. Nej, ABC802 har inte precis likadant styrning av markören som andra 800 varianter. Jag har tittat på två olika 802 promversioner och både betedde sig något annorlunda jämfört med de inlägg man läser här i monitorn beträffande kursrouteseende-styrning. I senaste promversionen är det väldigt svårt att ha vissa kursrouteseende därför att tolken "gör om" cursorn rätt oft.  
Mvh TAP

(Text 1940) Patric Ljung <5455>  
Va' de' nåt mer? - Ja.  
Man kan då känna av om flera tangenter är nertryckta samtidigt, detta resulterar i att man både kan gasa och svänga höger och skjuta i ett spel till exempel.  
Men man måste alltså hålla reda vilka tangenter som är nertryckta. När man trycker ner en tangent genereras ett interrupt, och ytterligare ett när man släpper. Man måste således vara ganska snabb, om man vill hållas i basic bör man lägga upp en buffert i nån variabel som interruptet lägger upp alla koder i, annars kan man låta interruptet sätta bitar som indikerar att en viss tangent är nertryckt. Om man nu vill kunna shifta en tangent måste man således ha en flagga som indikerar om shift är nere eller inte. OBS. Tidiga versioner av ABC99 klara inte UP/DOWN-mod  
Vidare finns en del finesser hos ABC77 (ABC55)

1. Bell från  
129 Bell till  
2 5 ms klick från (gäller tangent tryck)  
130 5 ms klick till  
4 20 ms klick från  
132 20 ms klick till  
5 Repetering från  
133 Repetering till  
6 Normal ASCII-mod  
134 UP/DOWN-mod  
7 Bell (pip i högtalaren)  
8 CAPS LOCK från (Normal mod)  
136 CAPS LOCK till (Normal mod)  
8 CAPS LOCK indikator från (UP/DOWN-mod)  
136 CAPS LOCK indikator till (UP/DOWN-mod)  
9 ger 5 ms klick  
10 ger 20 ms klick  
Det var nog allt.  
Moss / Patric

(Text 1961) Magnus Bodin <4304>  
Ärende: Nya kommandon i ABC800.  
Fråga: Vad menar alltså, är att om jag skapar ett nytt kommando med namnet "lax" som exekveras direkt, dvs direkt i syntaxchecken, hur kan jag då i "lax-rutinen" känna av om användaren skrivit  
lax  
eller  
1049 'Hejsan' : Lax  
Så att jag kan säga ifrån med felkoden 144 eller liknande.  
Hur kollar man om man är i RADCOMPILE-mode, alltså??  
Magiska hälsningar Magnus

(Text 1962) Kent Berggren <6019>  
Ärende: V24:  
Följande rutin kan läsa av DCD och CTS på V24: och PR:

```

; 8      32
; DCD    CTS
; LÄSER AV PÅ PORT-A
BAUD:   LD A,5
        OUT (33),A
        LD A,226
        OUT (33),A
        LD A,16
        OUT (33),A
        IN A,(33)
        RET
; LÄSER AV PÅ PORT-B
        LD A,5
        OUT (65),A
        LD A,226

```

```

OUT (65),A
LD A,16
OUT (65),A
IN A,(65)
RET

```

De är de enda två bitarna jag vet att man kan läsa av på porten. Hoppas det hjälper någon lite.

(Text 1966) Valter Pettersson <2008>  
Ärende: Överföring V24:  
Ett liknande problem som jag ej lyckats lösa:  
Jag läser indata på V24: men vet ej mängden indata, ej heller filsluttecken (som inte alltid sänds). Programmet vet alltså ej att data från sändaren är slut utan hänger i nån GET eller INPUT.  
Hur kan detta tillstånd brytas av "operatören" från tangentbordet gärna med Ctrl... tryckning?  
Ctrl-C går ju men det bryter ju programmet (Basic) och är oacceptabelt.  
Assembler går väl på nåt sätt men finns inget enklare?

(Text 1970) Gunnar Faith-Ell \* <2733>  
Ärende: Överföring V24:  
Du bör ha optionsprom av lägst version 7-2, annars kan ett byte rekommenderas. Från BASIC kan du sedan känna av om det finns tecken i bufferten med "IF PEEK2(PEEK2(65500)+6)>0 GOTO <hämta-etteckenmedget>". Se "manualen" till optionsprommet, den beskriver det ordentligt med programtempel.

(Text 1996) Kristoffer Eriksson <5357>  
Fråga: Nya kommandon i ABC800.  
Vad är det för mystiskt kommando du hittat på som måste exekveras redan vid syntaxkollen/kompileringen, och dessutom gör skillnad på om det ingår i ett program eller ges direkt??? Räcker det inte att kompilera den som vanligt och kolla vid exekveringen; kommandos utan radnr exekveras ju direkt, och då kan du helt enkelt kolla bit 2 i adress 65315 (INT.DIR i Y.INT, du har väl Bit för Bit...). Annars skulle jag föreslå att du kollar om radnr i den delvis kompilerade raden är 0, vilket betyder avsaknad av radnr. Radnumret ligger i två bytes på adressen PEEK2(65306)-250 (Y.GENEND) tror jag, men jag har inte testat.  
H1 = Pekar till textrad som kompileras (inbuffert)  
DE = Pekar på kompileringsbufferten där internkoden byggs upp (DEstination)

(Text 2005) Gerry Eriksson <4842>  
Ärende: 31250 bit/sek  
SIO (V24) kan du öppna på vanligt sätt (OPEN "V24:,..."). Ändra sedan baudraten för kanalen med CTC:n  
CTC ch0 TxC vid splitt speed, CTC chl RxC och TxC när splitt speed ej bygglad.  
För 31250 Baud så blir det:  
OUT 96,71,96,3 : TxC CTC ch0  
OUT 97,71,97,3 : RxC  
1  
Detta förutsätter att SIO:n jobbar i /16-mod vilket den gör om man öppnar kanalen via OPEN. CTC är i countermode och klockas via External Clock input som är ansluten till en 1,5 MHz signal (CPUClock/2).  
Delningsfaktorn är 3.  
För andra baudrates mellan 366 - 38k Baud räknas delningsfaktorn som skall laddas ut med:  
N=1,5E6/(Baudrate\*16)  
För baudrates under 366 Baud skall CTC:n sättas i TIMER-mode.

(Text 2015) Kristoffer Eriksson <5357>  
Ärende: FGCTL  
LD C,7  
LD B,Färgregister  
OUT (C),Färgvärde  
Det finns 16 färgregister (0-15) en för varje färgnr som kan lagras i grafikminnet. Färgvärdet bestämmer vad det blir för färg på skärmen och består egentligen (om jag inte minns fel) av två färgkoder om vardera fyra bits, varav det mest värda biten anger förgrunds/bakgrunds-färg och de övriga Röd, Grön och Blå i någon viss ordning.  
LD A,Text+4  
OUT (54),A  
där Text är 0 eller 128 väljer om textskärmen ska visas eller inte.

(Text 2017) Bo Kullmar \* <1789>  
Ärende: Hur ska det gå för Luxor (och oss?)  
Luxor struntar i hobbyisterna. Hade det inte varit så att vi numera har många "företagsmedlemmar" så hade de inte varit intresserade av oss.  
Hållera annars med om att ABC-Klubben nog är borta om 10 år, men några är till bli väl vi och Luxor kvar.

(Text 1943) Bo Kullmar \* <1789>

Ärende: Comporian

Ja, Patric har nu uppmärksammat mig på den exakta lydelser i brevet. Jag har det inte här och skrev bara vad jag kom ihåg. De har alltså inställt betalningarna och uppdragat åt en advokat att utreda förutsättningsarna för fortsatt verksamhet. Alltså inte konkurs.

(Text 1949) Thomas Andersson <3917>

Ärende: Listskydd

Vad är egentligen vitsen med att listskydda systemprogram?

Om man kör t ex Lib och har en 8061a så skulle det ju vara trevligt om man fick se filnamnen i färg och inte som nu i svart-vitt. Detta är ju lätt att ordna om man blev inläst i programmet.

Mvh T Andersson

(Text 2020) Jan Holmberg <3141>

Ärende: Lite info från Luxor:

Som bekant går f.n. inte tangnetbord 99 att använda ihop med programmen Teledatammatning 800, Teledatammatning 800 och VT-102 (från Autocode).

Det hela beror på en bugg i 99 bordet. Luxor kommer nu att gratis byta ut ett prom i bordet så att programmen sedan går. Felet har att göra emd UP/DOWN någonting.

Alltså ni som har 99 bord och vill byta prom kontakta er återförsäljare om några veckor så byter de ut det (garantiärende).

En ny version av Teledatammatning 800 och även av Teledatammatning 800 kommer runt jultiden.

Den nya versionen kommer att innehålla en del nyheter såsom styrning av modem 800 m.m.

De nya programversionerna kommer även att gå att lagra på winchester (vilket de inte går idag).

(Text 2024) Peter Goldmann <5080>

Ärende: Autostart/Reset

ABC806,ABC832,gamla controllerkortet ger med skiva som innehåller ADOPPT.ABS och BASICINI autostart vid tillslag av strömmen. Men om jag ändrar i den \*\*\*\*\*BAC fil som jag sedan skall köra så blir det problem med Autostart/Reset.

Exempelvis så har jag lagt QZ.806 på en skiva i BAC-format. När jag sedan läste om inverterad bild här i monitorn, så la jag in en ny programrad som ger blå bakgrund och vit text. När jag sedan sparar den nya versionen med SAVE så tappar jag Autostart funktionen. Men jag kan tyvärr inte kolla detta direkt utan det verkar som om några bitar finns satta i minnet som inte påverkas av RESET, när väl programmet väl är kört en gång.

Följande händer då vid klappstart: Jag trycker på RESET-knappen upprepade gånger och kanske jag får napp - och det startar - men si - ofta kan då ERROR 43 eller meddelandet SKIVAN SKRIVSKYDDAD komma upp på skärmen.

Lägger jag nu in en liten snutt som "TEXT" eller ett litet program som jag kör, då kan jag gå igång min skiva med RESET utan fel.

Vad är det för liten flagga eller BIT som sätts?

Har det att göra med UFD-DOSET? Jag låter nämligen BASICINI starta DRINI som startar QZ.806.

Bo Kullmar har ju beskrivit hur man gör i senaste numret av ABC-bladet Och det är noga med att man har en nyformaterad skiva. Men jag tycker ju att om man nu har QZ.BAC sist på skivan i ordning räknat borde ju vid SAVE av en ny version av QZ.806 med ev ändrad kodord eller att jag vill ha \* före medlemsnumret - så vill jag ju helst slippa göra om hela proceduren med formatering och kopiering på nytt.

Hälsningar Peter

(Text 2029) Kristoffer Eriksson <5357>

Ärende: Fler frågor om FGCTL

Grafiken visas alltid. När den inte syns beror det på att alla färger är svarta, dvs nollor i alla färgregister.

Som du (väll) vet kan man använda 806:ans grafik i två lägen, en med 256 punkter (eller 240) per linje och 16 färger vilket kräver 4 bits för en färgkod i grafikinnet, och en med 512 punkter och 4 färger där bara 2 bits behövs för färgkoden. Nu är det så att hårdvaran som sköter grafiken alltid arbetar likadant, oavsett vilken upplösning som används. Skillnaden åstadkomms helt mjukvarumässigt.

När en del av en linje ska visas på skärmen hämtas således en halv byte = 4 bits från grafikinnet som får bestämma vilket färgregister som används. Detta register bestämmer färgerna på två pixels på skärmen. Har man valt upplösningen 256 punkter/linje har dessa två pixels samma färg och uppträder som EN punkt, då finns 16 färgregister att välja på med varsin färg. (Får man pixlarna att visa olika färg i stället får man blandfärger som man kan experimen-

tera lite med. Går att göra helt i BASIC). Med 512 punkter/linje och 4 färger används alltså de två pixlarna i varje färgregister var för sig, men eftersom fortfarande 4 bits i stöten hämtas från grafikinnet och dessa ska ge två punkter på linjen och dessa ska kunna vara vilken som helst kombination av de fyra utvalda färgerna blir det lite mer komplicerat. Varje färgregister får representera en kombination av två färger.

Två pixels, 4 färger ==> 4U2=16 kombinationer ==> 16 färgregister ==> Två pixels, 16 färger.(4U2+4 upphöjt till 2)red) Översättningen går alltså från 2-bits-färg till 4-bits-färg (Röd, Grön, Blå, För-/Bakgrund) Jag förmodar att detta arrangemang ger vissa tekniska fördelar. Hoppas du får ut något av beskrivningen, den blev inte så genomtänkt.

(Text 2034) Gunnar Faith-Ell \* <2733>

Ärende: Autostart/Reset

Autostarten av program görs med en kommandorad som skrivs in med hjälp av PREABS.BAC som ligger på systemdisken, du kör igång programmet och lägger in ADDOPT.ABS som maskinkodsfil och CHAIN 'QZ.806' som basiskommando och sedan så ska det väl inte vara några problem under förutsättning att ADDOPT.ABS är rätt initierad. Några ändringar i programmet kan inte påverka autostarten av programmet eftersom den hela tiden ligger lagrad på disken.

(Text 2085) Helene Bodin <5133>

Ärende: ABC expansionslådor

Luxor säljer inte längre expansionslådor ABC890, utan har nu sedan juni 1985 sagt att det 'strax' kommer en ny expansionslåda som då skall heta ABC894, och tydligen bygger på den lilla lådan från ABC834 driven. Den kommer då förmodligen att bara ha 3-4 kortplatser. Ingen på Luxors försäljningsavdelning kan dock lämna några tekniska specifikationer eller leveranstider. Alltså HJÄLP! Vet någon något om exp lådor från Luxor? Vem på Luxor arbetar med att utveckla nya prylar. Alla som jag pratat med skall undersöka saken och sedan ringa upp...

(Text 2093) Bo Kullmar \* <1789>

Ärende: ABC expansionslådor

Vad jag har hört så är den nya expansionslådans redan ut. Björnhem påstår i alla fall att han har sålt sådana lådor. Den är alltså gjord för att hänga på "andra sidan" på ABC802:an. Tranor har dock alltid själv sålt en expansionslåda med 7 kortplatser och den kostar numera 3 600 SEK exkl moms om inte 894:an passar.

(Text 2096) Martin Björkman <3854>

Ärende: ABC expansionslådor

Ett företag i Gbg (Microsupport tel 031-416665) gör en utmärkt låda med 10 platser och ett rejält tilltaget nättag (mkt bättre än det i ABC890)

Mvh Martin i Gbg

(Text 2104) John Stagg <5741>

Ärende: ABC802 MEM: <=> CAS:

Jag borde kanske förtydliga detta. 65502/3 är alltså systempekaren till MEMBUF. Bufferten flyttas till DOSBUF5 med POKÉ 65502,64000,swap% (64000) Sedan borde man skriva DOSBUF5 genom POKÉ 64913,0 (detta enligt ett tidigare inlägg i detta möte av någon annan medlem). Hoppas det är klart nu!

John

(Text 2109) Göran Sundqvist <1255>

Ärende: Släcka och tända skärm

Kan man göra med: OUT 56,0,57,126 ! Släck skärm resp. OUT 56,0,57,127 ! Tänd skärm informationen ligger hela tiden kvar i bildminnet.

(Text 2111) Kristoffer Eriksson <5357>

Ärende: Släcka och tända skärm

CONSI-rutinen använder (älmstone på den maskin jag kollar) OUT 56,1,57,0 för att släcka och OUT 56,1,57,80 för att tända. Släcka och händer är att CRT:n får visa 0 resp 80 tecken per rad. Det tycks fö duga med vilket udda tal som helst för att bilden ska släckas. Man kan också få den att visa 0 rader, med samma resultat. Det är alltså inte någon särskild IC inblandad som Ulf Hedlund påstod. Bygeln kan nämligen avläses av CONSI-rutinen med OUT 67,16:X=INP(67)AND 8 (dvs, motsvarande i maskinkod), då hamnar resultatet i X. Den här bygeln är kopplad till DCD-signalen på SIO:ns kanal B. Vill du ändra tiden innan skärmen släckas, får du ersätta CONSI-rutinen med en egen.

(Text 2122) Petter Stahre <4501>

Ärende: Hur fungerar FGPICTURE?

Det hade jag också mycket problem med först, men nu förstår jag det! Det fungerar så här:

I ABC806:an så finns det ju minne reserverat för 4 grafikbilder. När du startar datorn, så skriver du i bild nr 0. Datorn är alltså startad så här:

FGPICTURE 0,0,4

Där den 1:a nollan betyder att man ritar i bild nr 0.

Den 2:a nollan betyder att bild nr 0 visas (du kan alltså rita i bild nr 2 tex, och visa bild nr 0).

4:an betyder att det är 4 bilder du använder (bild nr 0,1,2,3).

Alltså:

Om du skriver FGPICTURE 0,3,4 så visas bild 3, medans du kan rita i bild 0.

Om du skriver FGPICTURE 2,0,4 så visas bild 0, medans du ritar i bild 2 osv osv...

Den fungerar alltså som 4 papper.

Du visar någon papper nummer 2, där har du ritat en dator. När han/hon tittar på pappret så ritar du något annat på papper nummer 0.

Detta skulle se ut så här på ABC806:an...

FGPICTURE 0,2,4

Hoppas att du förstår!

/ mvh Petter Stahre

(Text 2132) Ulf Sjöstrand <1208>

Ärende: 3270-emulering

Vad behövs för att emulera 3270 på en "naken" 806? Vore trevligt om kostnadsuppskattning ingick i svar(en). 806:an är bara försedd med 815 monokrom men jag vet att 3270 kan ge färg på skärmen. I systemet finns också en 832(DD12), Selicmodem 16. Kan ej användas 832 är väl synkron kommunikation?

Kan man få simultan utskrift på skrivare som med Bok:s QZ.BAC?

Mvh Ulf S

(Text 2135) Ulf Hedlund <4868>

Ärende: Portarna i ABC 800 M

Dvs, ut går det lika bra att sända tecken på bägge portarna, men in är B-porten lämpligare, då det sitter en SIO där och en DART på kanal A.

(Text 2136) Ulf Hedlund <4868>

Ärende: ABC99

Du tänder diod 1-8 med

OUT 34,(diod-1)\*16

och släcker med

OUT 34,(diod-1)\*16+128

(Text 2141) Benny Löfgren \* <2615>

Ärende: 3270-emulering

Det finns flera olika typer av SDLC 3270-emulering. T ex finns RJE, SNA och BSC. Samtliga dessa är (som SDLC antyder) synkrona protokoll, och de finns att få tag i från både Luxor och DIAB, som har gjort dem. Jag skulle tro att de kostar runt en 5.000 :- styck.

(Text 2145) Harald Sjöstedt <75>

Ärende: Bugg i '800-806

Nu har jag hittat senaste buggen i BASICII. Ja bugg och bugg, men det är fel i alla fall. Om man händelsevis skulle knappa in följande program:

10 I=17

20 poke 65329,129 : poke 65293,129

30 chain "x2"

Och sedan, av en annan händelse (p<0.001), skulle knappa in något i den här stilen:

10 !

Observera att variabelerna heter SAMMA ASCII-bokstav ! Spara porgl med SAVE och porg 2 som "x2" med list. Buggen då ? Jo kör porgl som chainar "x2" och vad ser ni ? Ja vad ser ni JO ERROR 2 (Que?) ! Står inte i manualen ! Noggrann som jag är konsulterar jag min lilla röda och min debugger varvid jag finner att:

När COMTOP/BOTTOM pokas upp läses .BAS-filen in på en högre (större) adress än .BAC-filen. När .BAS-filen kompileras tillverkas ett programblock på 20 byte först i filen (Här var den förra buggen, 37 bytes "tappas bort") där bland annat pekaren till vanliga variabler finns. En annan pekare till vanliga variabler finns också, nämligen adress 012eh. Den förra pekaren sätts upp till variabelnas nya adress 08100h, men den senare ändras INTE ! Således kommer BASICII vid förkollan att leta efter .BAS-i .BAC-filens variabellista ! När sedan VARPEK skall räknas ut i rutinen som i min bok kallas VAR FINNS (adress 0219ch) kommer denna bli NEGATIV = CARRY = ERROR ! Detta förutsätter att det är samma variabel.

Testa med heltal (i) => Error 1

Testa med flyttal (.) => Error 2

Testa med sträng (s) => STOP

Testa med sträng (i) => Error 2

Testa med en variabel med annat namn och den hamnar UNDER BOTTOM !

Detta måste alltså vara:

1 .BAC-program med variabel flyttat upp bottom

2 .BAS-program med "samma" variabel qkar ur

Som errcode erhålles en funktion av variabeltypen. Vad sägs om det, mina herrar !

(Text 2147) Gerry Eriksson <4842>

Ärende: Portarna i ABC 800 M

Så länge det inte är frågan om synkron kommunikation så spelar det inte någon roll då SIO:n och DART:en uppför sig lika. Hårdvaran runt omkring sätter i och för sig stopp för split-speed (kanske anledningen till att Terminal är förbehållet V24:)

(Text 2149) Benny Löfgren \* <2615>

Ärende: Portarna i ABC 800 M

Nja, det spelar en betydande roll, nämligen den att SIO:n i ABC:n kan köras med interupt, och det kan så vitt jag vet inte DART:en.

(Text 2153) Bernt Johansson <3384>

Ärende: Portarna i ABC 800 M

Det går bra att köra alla 4 SIO/DART kanalerna med interrupt.

(Text 2170) Benny Löfgren \* <2615>

Ärende: DART, CTC...

DART = Dual Asynchronous Receiver Transmitter

CTC = Counter Timer Circuit

CPU = Central Processing Unit

PIO = Parallel Input/Output

SIO = Serial Input/Output

CIO = Counter Input/Output

SCC = Serial Communications Controller

DMA = Direct Memory Access

Några fler kommer jag inte på för tillfället...

(Text 2173) Lars Michael Jogbäck <5862>

Ärende: ABC99

Lysdiod

OUT 34,

|   |     |
|---|-----|
| 1 | 128 |
| 2 | 144 |
| 3 | 160 |
| 4 | 176 |
| 5 | 192 |
| 6 | 208 |
| 7 | 224 |
| 8 | 240 |

Hoppas du blev nöjd med detta

(Text 2175) Benny Löfgren \* <2615>

Ärende: Portarna i ABC 800 M

Blev det rätt det där??? Kanal A är ju PR, dvs DART:en och kanal B är V24, dvs SIO:n. På kanal B kan man köra split speed, dock ej på kanal A, och det innebär att kanal B har separat Receive och Transmitt Clock, medan kanal A har gemensam.

(Text 2222) Sven Wickberg <1384>

Ärende: 80 -> 800

Jag stöter på mystiska fel när jag försöker konvertera en rad ABC80-program till ABC800-maskiner.

I princip lagrar jag programmet i .BAS-format med LIST <programnamn>. Sedan hämtar jag .BAS programmet till en ABC806 med LOAD. Då hämtas den in och kompileras till 806:ans specialkod. Men när jag gör RUN visar sig en rad fel. Självklart vet jag vissa förändringar i BASIC:en (jag har läst Bit för bit), men här uppträder saker som jag inte förstår.

Exempel:

I många fall godkänns inte rader som avslutas med: REM xxx Tar jag bort REM-satsen eller sätter ! i stället för: REM går det bra. Men det FINNS också rader där REM-satsen accepteras! ONERRORGOTO 0 tycks IBLAND läsas in allright och göras om till ON ERROR GOTO, ibland inte.

I några fall har jag måst dela upp en rad med ett stort antal instruktioner i flera rader. Har ABC800 svårt att svälja många semikolon på en rad.

Lustigt nog tycks maskinen acceptera ASC(x) och göra om den till ASCII(x), i varje fall ibland.

En annan varning. Har man INTEGER-läge kommer alla variabler att bli heltal även utan %. Skulle man av någon anledning vilja göra en ändring så att programmet kan köras för både 80 och 800 (t ex i ett CAT-NET) så kan man inte utan vidare spara programmet med LIST igen och sedan köra på 80. Då försvinner nämligen alla heltals%. Man måste ha FLOAT-läge innan man spar. Det är inte så lätt som det verkar att köra .BAS program i både 800 och 80. Närmast tycks kompatibiliteten innebära att man inte behöver knappa om hela programmet, utan kan slippa undan med en del mindre modifierationer.

(Text 2233) Benny Löfgren \* <2615>

Ärende: 80 -> 800

Det första man måste tänka på är givetvis att ha FLOAT och NO EXTEND inkopplat. Att du får fel i slutet på raden med: REM bla bla kan bero på att 800 kräver att vissa instruktioner står ensamma på en rad, t ex END.

ONERRORGOTO 0 görs INTE om till ON ERROR GOTO, utan det får man göra själv.



(Text 2410) Kristoffer Eriksson <5357>  
Ärende: ASM-rutin för printerstyrning  
Jag föreslår att du utnyttjar filsystemet, det ger större flexibilitet. Så här kan man göra:

```
CALL USEUTLU
; = 105
LD A,0
; 0 = Normal OPEN
LD HL,FILNAMN
CALL OPNCMD
; = 93
LD A,4
; 4 = Operationen PRINT
LD HL,TEXT
LD BC,TEXTEND-TEXT
CALL IO
; = 90
```

RET  
FILNAMN: DEFM "PR;"  
DEFB 13  
TEXT: DEFM "Denna text skrivs på skrivaren"

TEXTEND: \*  
Denna variant passar bäst för korta utskrift. USEUTLU åstadkommer vid RET i ovanstående rutin ett tvångsanrop till en filstängningsrutin. Vill man undvika det kan man POP:a den adress som ligger överst på stacken efter USEUTLU-anropet. En mer generell variant är:

```
LD IX,LUBLOCK
LD HL,FILNAMN
LD A,0
CALL OPNCMD
... osv
LUBLOCK: DEFS 23
CLOSE: LD IX,LUBLOCK
LD A,2
; 2 = Operationen CLOSE
CALL IO ...
```

På det här sättet öppnar man först filen, använder den då och då, och stänger den till sist. Varje gång filen används måste man se till att IX innehåller adressen till LUBLOCK.

(Text 2430) Patric Ljung <5455>  
Ärende: Lagring av maskinskrut  
Om maskinkoden är bestämd till att ligga på ett bestämt ställe så är fjäljande rutin MYCKET snabb och lätt.  
OPEN "MFL.ASM" AS FILE 1  
DIM Maskinkod\$=0  
GET \$1,Q\$ COUNT 2 : Storlek=CVT\$(Q\$)  
POKE VAROOT(Maskinkod\$),Storlek,SWAP%-(Storlek),Adress,SWAP%(Adress),0,0  
GET \$1,Maskinkod\$ COUNT Storlek  
CLOSE 1

Då kanske man skulle förklara alla stegen. Först öppnas filen. Filen är uppbyggd som så att först ligger två bytes som anger längden (storleken) på maskinkoden. Men först DIMar vi variabel till noll så den inte tar upp onödig plats i minnet. Därefter läses storleken in i Q\$ och konverteras till Storlek. Följande steg är att Ändra på variabel roten så att DATA arean börjar där maskinkoden skall börja och längden på variabel är den längs (storlek) som maskinkoden har. Sedan läses maskinkoden in och filen stängs.

Notera en sak! Allra först måste pekarna ändras så att datorn inte lägger in BASIC-program på minnes arean som maskinkoden skall ligga på. Man går nästan tillväga på samma sätt när du skall spara maskinkodebn om den ligger i minnet.  
Storlek=(storlek på maskinkoden)  
DIM Maskinkod\$=0  
POKE VAROOT(Maskinkod\$),Storlek,SWAP%-(Storlek),Adress,SWAP%(Adress), Storlek,SWAP%(Storlek)  
PREPARE "MFL.ASM" AS FILE 1  
PUT \$1,CVT\$(Storlek)  
PUT \$1,Maskinkod\$  
CLOSE 1

Vidare kanske en del då förstår andra smarta lösningar på andra problem.

DIM Screen\$=0  
POKE VAROOT(Screen\$),2000,SWAP%(2000),0,120,2000,SWAP%(2000)  
Variabeln Screen\$ 'innehåller' således skärmen. Ex: Gör ovanstående tryck CTRL+L (rensa skärmen) hoppa ner ett par rader och skriv Screen\$="Hej".  
Hej skrivs ut överst på skärmen.  
MID(Screen\$,Rad\*80+Kolumn,Längd)=Text\$  
Alternativt PRINT.  
Moss / Patric

(Text 2438) Torsten Frank <5452>  
Ärende: PLC med ASMZ  
Om man gör en utlistning (CON:PR) av en assembler-fil med assemblern ASMZ skrivs det på varje sida PLC (och en hel del annat). Det står tydligen för "Current location counter". Problemet är att jag inte kan få PLC:n att ändra sig. Det står inte någon ting alls i detta fält på utskriften, hur jag än bär mig åt. Finns det någon instruktion som ändrar PLC:n (?), eller ändras den automatiskt vid nytt ORG, utan att det syns på listan?

(Text 2439) Jan Lundmark <2765>  
Ärende: PLC med ASMZ  
PLC ändrar man med 'PLC <nr>' där <nr> är ett heltal i intervallet 0-31. När du sedan länkar och ska orga de olika PLC:erna till olika adresser så skriver du:  
exvis:  
PLCNR 3  
ORG 8000H  
PLCNR 4  
ORG 0C000H

i din kommandofil till länkaren.  
Jo just det ja, det är mycket trevligare att orga vid länkningen än att skriva in orgar i källkoden.

(Text 2463) Curt Rehnberg <1121>  
Ärende: Rad 25 och inverterad text på ABC80X

Är det någon som har kläm på hur man på ABC802 och ABC806 kan fixa till inverterad text på någon rad(er).  
Hur man får till rad 25 vet jag, men hur fixa inverterad text där? Helst ska man kunna köra inverterat ibland och inte ibland. Någon som vet, hör av er .....  
Gott Nytt År ....  
Hälsning Curt Rehnberg

(Text 2464) Kent Berggren <6019>  
Ärende: Rad 25 och inverterad text på ABC80X  
Bit 7 hög. Funkar inte det?

(Text 2466) Gerry Eriksson <4842>  
Ärende: Rad 25 och inverterad text på ABC80X  
Att sätta bit 7 hög går bara på ABC802. På 806 så kan du sätta en bakgrunds färg och skriva text med svart (BLK).

(Text 2467) Gerry Eriksson <4842>  
Ärende: V24:ans interrupt-vektorer.  
Vektorerna innehåller adressen till interrupt-rutinerna.  
Vektorerna för V24: (SIO'n chA) är som följer:

|        |                                                                          |
|--------|--------------------------------------------------------------------------|
| Adress | Interruptfunktion                                                        |
| 65480  | Transmitter buffer empty                                                 |
| 65482  | External/Status Change (DCD, CTS, SYNC and Break sequence)               |
| 65484  | Receive Character Available                                              |
| 65486  | Special Receive Condition (Parity error,RX overrun error, Framing error) |

Du får då ändra vektorn i 65482 till adressen för en rutin som undersöker om det var DCD som orsakade interruptet. Givetvis måste även SIO'n vara programmerad för interrupt (WriteReg 1).

(Text 2472) Einar Eriksson <1720>  
Ärende: Överfört från KOM - Printerkabel V24 till RS 232C  
Har skrivaren RS 232 C ingång så ska kabeln se ut så här.

|             |      |                   |
|-------------|------|-------------------|
| V24 (9-pol) |      | RS 232 C (25-pol) |
| 1 DTR       | till | 6                 |
| 2 TxD       | till | 3                 |
| 3 RxD       | till | 2                 |
| 4 RTS       | till | 8                 |
| 5 CTS       | till | 20                |
| 6 + 12 V    |      | NC                |
| 7 GND       | till | 7                 |
| 8 DCD       |      | NC                |
| 9 - 12 V    |      | NC                |

NC = Not connected  
Så här är originalkabeln kopplad som Luxor levererar med skrivaren Epson.  
Glöm inte att ställa in rätt hastighet på skrivaren samt att ställa in rätt paritetbiter. Dessutom måste man ställa in om skrivaren ska använda XON/XOFF eller DTR som handskakning.

(Text 2485) Stefan Westman <4969>  
Ärende: Program lagrade som strängar.  
Tack för alla tips om lagring av program på strängar.

Jag fick mitt program att fungera.  
Då jag nu har upptäckt att det går snabbare att läsa in ett program på detta sätt tänkte jag lagra fler program som strängar.  
Problemet är dock följande.  
Ett program som har lyft golvet till 237\*256 och använder grafikminnet skall sparas som sträng.  
Jag gjode då följande.

1. Sparade minnescell 0-237\*256 på en sträng.
2. Sparade minnescell 134+241\*256 - 65535 på en sträng.
3. Sparade grafikminnets alla celler på disk med hsave.

Nu händer följande: Om jag läser in alla sparade delar i ordning 3-2-1 utan att resätta kan jag fortsätta och köra programmet. Sker inläsningen efter reset så kommer jag in och kan köra delar av programmet men vissa rutiner gör att maskinen auto-startar om.

Finns det någon som kan ge några hintar om var problemet kan sökas?  
Jag kör på ABC 806  
mvh Stefan

(Text 2489) Kristoffer Eriksson <5357>  
Ärende: Rad 25 och inverterad text på ABC80X  
På 806 är det ganska enkelt, eftersom man kan skriva t.ex PRINT CUR(24,0) YEL NWBG BLK "Detta är rad 25";  
På 802 däremot är man tvungen att POKE:a i texten eller använda andra knep, varvid bit 7 ska vara ett-ställd. Dock finns det en del 802:or med större teckenprom som har andra tecken än inverterade normala tecken på de koderna. Med "andra knep" menar jag exempelvis:

```
DIM Line25$=0
POKE VAROOT(Line25$),80,0,32640,SWAP%
(32640),80,0
Text$="Detta blir inverterat på rad 25"
FOR I=1 TO LEN(Text$)
MID$(Line25$,I,1)=CHR$(ASC(MID$(Text$,I,1))
OR 128)
NEXT I
fast när man ändå måste behandla varje tecken för sig, går det ju lika bra med ren POKE.
```

(Text 2493) Torsten Frank <5452>  
Ärende: Clustersize mm  
Finns det något enkelt sätt att ta reda på clusterstorleken för en viss enhet och ta reda på om DOS = ABC 6-1X (då ska ju tydligen fysiska sektornummer adderas med 32) ?

(Text 2495) Gunnar Faith-Ell \* <2733>  
Ärende: Clustersize mm  
Det finns ett program på systemskivorna till UFD-dos som kollar vilket dos som sitter i burken.

(Text 2505) Kristoffer Eriksson <5357>  
Ärende: Rad 25 och inverterad text på ABC80X

För att invertera endast en bit av en rad på 806 skriver man i stil med detta exempel:  
PRINT CUR(10,30) YEL NWBG BLK "Inverterat" BLBG " "  
BLBG på slutet återställer resten av raden till svart bakgrund och blanktecknet behövs för att den nya färgen verkligen ska skrivas ut i bildminnet, i alla fall när man har ATTRIBUTE 1. Ska man genast fortsätta med ej inverterad text får man förstärka ställa in en förgrundfärg också, annars blir det ju svart på svart. Om det redan står något mera i slutet av raden behövs inte "BLBG" " " eftersom den gamla texten ändå behåller sin färg, det är bara på helt o-skrivna rader som bakgrunds färgen sträcks ut över hela raden.

Tänk på att om print-satsen avslutas med "t" fortsätter de inställda färgerna och övriga attribut att gälla även vid nästa print oavsett om man har ATTRIBUTE 0 eller 4.  
Grafik- och färghanteringen skiljer ju ganska mycket mellan 806 och 802. På 802 skriver man i förväg ut styrtecken på lämpliga ställen i bilden, medan det i 806 hela tiden är senast använt styrtecken som gäller.

(Text 2507) Kristoffer Eriksson <5357>  
Ärende: Program lagrade som strängar.  
Det verkar klart överdrivet att även spara PROM-delen av minnet på filen! (Adress 0-30719 är prom, om man inte mappat in annat minne. 30720-32767 är bildminne för text.)

Att sedan spara alla systemvariabler inklusive stacken måste vara rena självmordet! (Varför inte spara CPU-registren också, eller åtminstone stackpekaren?) När stacken vid inläsningen byts ut kommer ju BASIC-tolken att spåra ur. Eventuella omlänkningar av vissa systemvariabler bör göras en i taget, inte med strängmetoden.

Är det verkligen nödvändigt att spara grafikminnet också?

(Text 2563) Nils Hansson <519>  
Ärende: RAM-floppy på ABC806  
Jag har fått anledning att titta på hur jag kan använd ramfloppyn i olika egna tillämpningar i kombination med grafikminnet. Vad jag inte har förstått är vad det innebär att RAM: är spärrat.  
Finns det något sätt att göra reset utan att ramfloppyrutinen rammlar ur minnet? Det vore praktiskt för alla filer ligger ju i alla fall kvar i RAM:.  
Mvh Nils H

(Text 2565) Gunnar Faith-Ell \* <2733>  
Ärende: RAM-floppy på ABC806  
Om du har den äldre drivrutinen för RAM-floppyn så stannar minnet kvar men då finns det risk för att det blir fel på innehållet. Eventuellt kan man nog byta RAM-kapslar till några som håller sig längre utan refresh och lösa problemet med minnesförluster på det sättet.

(Text 2597) Mikael Liden <5651>  
Ärende: GRAFIK & RAMFLOPPY samtidigt  
Här är några nyttiga adresser:  
65267 - max antal tillätna HR-bilder  
65268 - max antal HR-bilder  
gäller ABC806

(Text 2608) Mikael Liden <5651>  
Ärende: Formattering  
Kan någon beskriva vilka kommandon och data som behöver skickas till kontrollern för att formatera disketten. Antar att det är olika för varje diskett-storlek. Är i första hand intresserad av 640Kbyte-disketter. (Orkar inte dissassemblera DOSGEN)  
Tänkte göra ett snabbt helkopieringsprogram av disketter, som dels formatterar och dels kopierar snabbt de sektorer som är använda på disketten (dvs de som är markerade i bitmappen). Har tidigare gjort ett snabbt helkopieringsprogram för 160K som sköter kopieringen genom att först läsa upp sektorerna i kontrollern och sedan skriva ut dem på den andra disken (max 3 sektor åt gången, bufferten i kontrollern är inte större), det man vinner i hastighet på att förfara på detta sätt är att slipper läsa in sektorerna till datorn. Detta program formatterar dock inte disketterna, det vore naturligtvis bra att få till ett snabbt kopieringsprogram som både formatterar och tar kopia om intresse finns för ett dyligt program så skickar jag naturligtvis in det.

(Text 2609) Jan Lundmark <2765>  
Ärende: Formattering  
Är det den gamla kontrollern? I så fall blir det knepigt för att formatera ett spår fixt och färdigt med data i sektorerna så måste kontrollern ha hela spåret i sin buffert inklusive adressmärken. till detta kommer sedan kodnutten som ska göras. Innan man ger formatkommandot till kontrollern så måste bufferten fyllas med info om hur adressmärkena ska se ut. Sedan så sköter kontrollern om att fylla sektorerna med 'E5'. Detta gäller alltså den gamla luxor/diabkontrollern med 1k minne på kortet. Hur luxor/diabs nya / myabs gör har jag ingen aaaaaning om.

(Text 2610) Gerry Eriksson <4842>  
Ärende: Formattering  
I ABC blad nr 2/85 sid 32 hittar du en text av Göran Altius <123> (= f.d. text 259 i detta möte) som beskriver kommandon för MYABS UNIDISK-kort bl.a. formattering. Gjorde en testsnutt och provade med en DD12 (spec.) och det fungerade fint. (Formattering av bara ett spår) 12an utrustad med DIABS ny kontrollern. Har du inte nr. så hör av dig så får vi ordna det på annat sätt.

(Text 2612) Bo Michaelsson <913>  
Ärende: Rad 25 på 806 och 802  
Följande lilla program ger en ny rad, rad 25, på ABC 800 M.  
Raden påverkas inte av CHR\$(12).  
DEF FNRad25  
OUT 56,6,57,25  
POKE 65362,0,24  
; SPACES(80);  
; "På denna rad skrivs en text";  
RETURN 0  
FNEND

Rutinen fungerar utmärkt på 800 M men inte på 806. Har någon en ide som fungerar på båda?

(Text 2614) Gunnar Faith-Ell \* <2733>  
Ärende: Rad 25 på 806 och 802  
På ABC800M kan man om man vill lägga in TVÅ nya rader men då får man flytta upp BASIC:n lite.

(Text 2617) Göran Sundqvist <1255>  
Ärende: Rad 25 på 806 och 802  
genom att programmera om CRTc-kretsen, kan man få 30 rader om 64 tecken.

(Text 2630) Kristoffer Eriksson <5357>  
Ärende: Rad 25 på 806 och 802  
Om man betänkar att man på 806:an kan nå den 25:e raden med vanlig PRINT och att den enda skillnaden mot övriga rader på skärmen är att den inte scrollar, ser man lätt vad som är felet med din lilla snutt. När SPACES(80) har skrivits ut så görs ett radbyte till den 24:e raden som är skärmen sista rad enligt scroll-rutinen, och skärmen scrolla. Därför hamnar den efterföljande texten inte på rätt rad. Använd TS="På denna rad skrivs en text"  
; TS SPACES(80-LEN(TS));  
istället. Då fungerar det på alla maskiner. F.ö. är det en ganska smart metod att POKE:a radrännaren, jag kom själv inte på den för mer än några månader sedan. Tilläggas bör att på 806:an tömmer CHR\$(12) ALLA rader, även den 25:e.

(Text 2661) Henrik Ahlen <5133>  
Ärende: Skärmflimmer  
Min ABC800 lider svårt av att skärmen ofta plötsligt börjar flimra kraftigt. Avhjälps genom att slå på datorn eller vänta en stund. Är värst den första timmen efter påslag, v arför jag numera låter den vara på dygnet runt. Datorn har varit på TDX för service, men kom tillbaka likadan som förut. Vad kan göras åt detta? Tacksam för tips.

(Text 2664) Gunnar Faith-Ell \* <2733>  
Ärende: Skärmflimmer  
Lyft på kortet på bildröret i bildskärmen och spraya lite. Dra ut videokortet i datorn och spraya kontaktarna lyft på kretsarna och spraya, kolla kontaktarna på sladden, så ska den nog fungera. Kontaktspray hjälper alltid...

(Text 2711) Patric Ljung <5455>  
Ärende: Rad 25 på 806 och 802  
Nej, nej!  
; FNR25\$ (10) 'Detta är rad 25';  
Här används rutinen precis som ett vanligt CUR. Observera att ett semikolon måste stå sist för att skärmen inte skall scrolla. DEF FNR25\$(Kolumn) LOCAL Kolumn  
POKE 65362,Kolumn,24  
RETURN "  
FNEND

Om man först vill rensa rad 25 får man skriva:  
; FNR25\$(0) SPACE\$(80) FNR25\$(0) Text\$(Jämför med CUR)  
; CUR(24,0) SPACE\$(80) CUR(24,0) Text\$(Vill man så kan man...  
DEF FNCS(Rad,Kolumn) LOCAL Rad,Kolumn  
POKE 65362,Kolumn,Rad  
RETURN "  
FNEND  
Så är funktionen helt utbytbar mot CUR.  
\_OBS\_ att den 25 raden måste initieras först.  
OUT 56,6,57,25 ! Sätt 25 rader  
MOSS / Patric  
PS. Hoppas alla problem är utredda

(Text 2739) Robert Svedjehammar <2350>  
Ärende: Kablar & skatbon  
Visst gör det någonting om man börjar plöcka bort de skydd som finns för att förhindra utstrålning, kanske inte för dig, men för omgivningen som försöker använda någon form av radiomottagare. Det finns en orsak till varför de sitter där. Tillverkaren skulle knappast förordat konstruktionen om det inte behövdes.  
Hälsning från en radioamatör med mycket störningar.

(Text 2761) Curt Rehnberg <1121>  
Ärende: Kablar & skatbon  
De kåpor och skärmar m.m. som sitter monterat på Luxormaskinerna sitter inte bara för att förhindra utstrålning av önskad störningar. Det är även för att hindra att störningar av olika slag kommer IN, kanske mer det sista än det första.  
Vad som är viktigt i sammanhanget är att kedjan är s.a.s. obruten. Vissa kontakter från Luxor är dock tyvärr "slarvigt" monterade varför just kortslutningar inte är ovanligt. Speciellt en variant av kabel mellan 806 och 815 (Den korta)  
H-ning CR

(Text 2767) Lars Hedberg <2497>  
Ärende: Programstatus  
På adress SYS(10)+38 ligger informationen (=65318d):  
b0 - Det har varit förköll  
b1 - LISTSKYDDAT program  
b2 - Double på flyttal  
b3 - Flyttal finnes  
b4 - COMMONflyttal finnes  
b5 - Programmet ligger högt i minnet  
b6 - HR-Grafik användes  
b7 - RESUME användes  
På adress SYS(10)+29 ligger lite annat av intresse (=65309d):  
b0 - Integer mod  
b1 - Extend mod  
b2 - Mellanslag signifikant  
b3 - Variabler finns ej (?)  
b4 - COMMON bort (?)  
b5 - Läger in COMMON (?)  
b6  
b7  
En lustig variant är att ändra SYS(10)+29 så att man befinner sig i extend mod men UTAN signifikanta mellanslag.  
Hoppas det svara på eventuella ?  
Hälsningar TFP Lars Hedberg

## Möte ABC80

(Text 929) Guido Petz <1098>  
Ärende: basicii till abc80  
Enligt Börje Larsson på Sala Teknik finns nu Basic2-kortet på marknaden. Priset är 2500+moms!  
På kortet finns följande faciliteter:  
1 anpassad ABC 802-Basic  
2 totalt 80k ram, varav 48k ramflopoy  
3 UFD-dos  
4 Seriell printerrutin  
5 förberedd för CP/M  
6 förberedd för nätverk

Enl. Börje är man intresserad att förhandla om rabatter för ABC-medlemmar. Jag undrar nu om någon vet om klubben har börjat förhandla med Sala Teknik, som tydligen har övertagit basic2-projektet från Liber? Vidare vore det bra att veta hur många ABC80-ägare vill hotta upp sin gamla kärra! För något är sen antynde Gunnar Tidner för mig att klubben var i kontakt med Liber för att förhandla fram en rabatt för medlemmarna!  
Hör av er ni som vet!!

(Text 930) Bo Kullmar \* <1789>  
Ärende: basicii till abc80  
Nej, det är nog ingen som har varit i kontakt med Sala Teknik för att förhandla med rabatter. Det som vi har gjort är bara så att vi för några år sedan gav lite synpunkter till en konsult som Luxor hade anlitat för projektet.  
Kan du skriva det brev till sylvemötet här och ange telefonnummer till Sala Teknik, så hoppas jag att vi kan få någon ABC80 fantast i styrelsen att ta kontakt med Sala Teknik för att få fram någon form av rabattavtal.  
Vi har ganska ont om tid, eftersom material till ABC-Bladet nr 4 skall vara inne snart.

(Text 940) Jörgen Gustavsson <3528>  
Ärende: CMDINT.SYS  
I våras fördes en diskussion i MSG om man behövde detta till annat än formattering. Jag kommenterade då att det fanns ett program i basic som klarade formattering och att jag skulle försöka få författaren till programmet att sända in det. Börje Gustavsson har nu sänt in sitt FORM.BAS. Nu skulle jag vilja återuppliva diskussionen: Har man någon nytta av CMDINT.SYS?

(Text 941) Bo Kullmar \* <1789>  
Ärende: CMDINT.SYS  
Svaret är Nej, om man frågar Luxor. Se bara på Meny0 där allt ligger i BASIC-program. Svaret är Ja, om man frågar DAIB. DAIB gör mest under DOS och brukar till och med se till så att man kan anropa ett BASIC program från DOS.  
Jag tror det är en vana. Det är nog svårare att köra från DOS, eftersom man använder switcher. För övrigt av samma typ som används i Monitors. Något som t ex Meny0 blir väldigt enkelt att använda.  
Själv använder jag ofta ett formateringsprogram i BASIC som fanns med på någon skiva till BASIC II boken för ABC806. Jag har fått programmet av någon. Jag tror dock inte att det är fritt att lägga in här. Fördelen med detta program är att man slipper anropa DOSGEN varje gång på nytt. För min egen del så skulle jag nog vilja svara nej på frågan. På CAT-NET kan man t ex överhuvudtaget inte köra program under DOS! Kan man anropa DOS program från BASIC så räcker det.

(Text 944) Anders Franzen <5258>  
Ärende: CMDINT.SYS  
Det enda skälet till att använda CMDINT.SYS är väl möjligheten att lagra maskinkod i ABS-filer som kan läsas av både ABC80 och ABC800. Man kan ju förstås fråga sig vilka program som går att köra på maskinerna. Nej, det finns nog ingen anledning att använda CMDINT. Program skrivna i BASIC kan göras mycket mer användarvänliga, t ex finare menyer.

(Text 945) Benny Löfgren \* <2615>  
Ärende: CMDINT.SYS  
Varför säger du så? Man kan göra precis samma användarvänliga menyer om man kör program under DOS som om man skriver programmen i BASIC. DOS är inte bara assembler, om nu någon tycks tro det.  
Jag tycker att hela diskussionen är löjlig, ställ samma fråga till en IBM PC-ägare: Behöver du verkligen ditt PC-DOS, klarar du dig inte med BASICen? och vad tror ni man skulle få för svar?

Användarvänlighet är nog bra - så länge man är nybörjare. När man börjar lära känna sitt system så börjar man tycka att det är bekvämt att kunna ange alla parametrar till det program man ska köra på kommandoraden, så man slipper vänta på att programmet ska laddas in, och sen svara på ett antal frågor och sen vänta på att programmet är klart.

(Text 946) Jörgen Gustavsson <3528>  
Ärende: CMDINT.SYS  
Det kan hända att diskussionen förefaller löjlig för en del.  
Bakgrunden till att jag ställde frågan är följande:  
- Jag kan inte assembler eller annan maskinnära programmering. Jag använder min dator flera timmar om dagen och då till 95% färdiga program. Min egen programmering inkränker sig till en del basic-program samt anpassning av andra program till mina behov.  
- På systemskivan som följde med datorn har jag kopierat in en hel del andra program. En del av dessa förutsätter CMDINT. Det ena efter det andra av dessa program har jag hittat basicvarianter till. Detta ser jag som en fördel: Jag kan läsa och ev ändra programmen + endast detta program behövs på den skiva jag använder. Det enda programmet jag till slut använde under CMDINT var DOSGEN. När sedan FORM.BAS kom ställde jag denna fråga.

(Text 953) Henrik Schyffert <2314>  
Ärende: CP/M och KERMIT  
Som glad amatör så har jag fått veta att Kermit inte går att använda på ABC-80 och CP/M för V24-porten stämmer inte. Är det rätt och i så fall vad skall man använda under CP/M för att kunna köra kom ??  
För det andra så förstär jag att det finns massor av program i CP/M på 8'skivor i IBM-format 3740 (tror jag). Dessa program har jag möjlighet att använda men jag har 51/4 skivor i format ABC-831 (som är samma som 832 men bara på ena sidan). Kan någon kopiera ??? Skulle det bli flera kopior så vet ju inte jag om det.  
Ang ABC-ord så finns det 'debuggade versioner' - brevlådan -  
Är någon intresserad av en 'deal' ???  
Henrik och Stig

(Text 957) Ola Arwidsson <5275>  
Ärende: EJ CP/M och KERMIT  
KERMIT finns till ABC800 (basic), skulle inte den kunna gå att använda ? (eller är jag ute och cyklar).

(Text 958) Bo Kullmar \* <1789>  
Ärende: EJ CP/M och KERMIT  
Ja, du är ute och cyklar. KERMIT:en till ABC800 är skriven i BASIC II och kan ej köras på ABC80 varken under CPM eller under BASIC I. Sedan använder sig nämnda KERMIT av den sk inbyggda terminalrutinen i ABC800-serien och en sådan finns inte i ABC80. ABC80:s hårdvara saknar ju sådant som normalt behövs för kommunikation. På ABC80 måste man köra bitbang och sinumera en SIO. Jmf David Andersson terminalrutin SFTSIO till ABC80. Jag har för mig att Myab säljer någon terminalrutin som man kan köra på ABC80 under CPM. Kolla med MYAB.

(Text 959) Anders Franzen <5258>  
Ärende: EJ CP/M och KERMIT  
Vad är bitbang?

(Text 963) Karl Lindström <837>  
Ärende: EJ CP/M och KERMIT  
Bit-bagning är att du har en byte (= 8 bitar), slänger ut den (t ex) den högraste biten på en tråd, roterar en bit åt höger, slänger ut den högraste igen roter är höger en gång till osv tills du har slängt ut 8 bitar. Då har man slängt ut en byte seriellt på en tråd.

(Text 970) Anders Olsson <1019>  
Ärende: 64K a la Microdatorn  
Jag har sett en del rättselser till artikeln i senare nummer. Kärnan är dock att just 8-bitars refreshen inte finns med, och det har vad jag vet inte publicerats någon rättelse om det. (Har jag fel?)  
Jag håller på att skriva på en beskrivning på den minnesombyggnad som jag själv har gjort. Om jag får den färdig är det inte otänkbart att jag skickar den till bladet. Kruket är att minnesombyggnaden förutsätter ett specialskjutet bipolarit prom för minnesavkodningen. Så jag kanske inleder med en byggbeskrivning på en promskjutare för :do, vem vet? Den som har tålamod får se!

(Text 978) Kjell Svensson <5318>  
Ärende:RAM-kort  
Vet någon vad som skiljer mellan de olika expansionslådorna när det gäller minnesbussen? Jag har ett CMOS ramminne som fungerar utmärkt i min expansionslåda (ABC), men när jag provade i klubblokalen (DATA-DISK) ville det inte längre. Jag misstänker (i min låda):

stift A4= XMEMFL

stift A5= XMEMW  
Hoppas på hjälp.  
Kjell.

(Text 979) Kent Berggren <6019>  
Ärende:RAM-kort  
Det kan inte vara så enkelt att du satt in den på fel sida i Datadysken. Den har en I/O och en mem sida.

(Text 980) Bo Kullmar \* <1789>  
Ärende:RAM-kort  
Så enkelt kan det inte vara för även ABC-exp har IO och MEM på olika sidor. Jag vet att det skiljer sig också, för det funkar inte att köra CAT-NET kort i en Tranfor IO låda. Däremot får det utmärkt i en ABC-låda. Jag vet inte vad som skiljer, men LG vet i alla fall.

(Text 1004) Arne Hartelius \* <2731>  
Ärende: Gejo's 80-tecken och printer  
Nej, jag har Gejo's 80-tecken och är nöjd med det. Men OK, det är betydligt krångligare än Myab's innan man har fått fason på det hela. Vad gäller krocken mellan bildminne och printerrutin så har det bara betydelse vid hardcopy. Har man däremot Ramdator som jag har så måste man ju välja bort printerrutinen. Men den kan man ju då lägga på den plats där Myabs 80-tecken ligger. Det går väl på ett ut, men är som sagt krångligare i början med Gejo's. Gejo's var vid det tillfälle jag köpte den ca 500 kr billigare än Myab's. Den stora fördelen med Gejo's är att när jag laddar in 800-tolken så kan jag köra både 40 och 80-tecken. Har man vant sig vid 80-tecken från ABC80 är det inget lajans att köra 40 på 800:an. Enligt uppgift har dock Gejo slutat sälja sin 80-teckenstillsats.

(Text 1007) Hans Johansson <1678>  
ÄRENDE: EJ CP/M och KERMIT  
Trots att man måste fuska lite med metoden för överföringen så tycker jag att det borde gå att fixa en KERMIT till ABC80. Jobbar för tillfället med KERMIT i ett program från Autocode och som host en UNIX dator, funkar fint, men visst vore det kul att även kunna labba lite med ABC80'an. Så skicka gärna ett brev till mig om det är någon som vet mer eller om det är någon som har lust att sätta sig och skriva ett KERMIT prog till 80'an. Jag har en del dokumentation och protokollbeskrivningar till KERMIT.

(Text 1017) Johan Struwe <3338>  
Ärende: Konstigt Z80-register?  
Jag har en skiss över Z80-CPU:ns uppbyggnad framför mig, och har till min stora förvåning upptäckt ett register med namnet WZ! Vad används detta register till? Det går inte att komma åt ifrån maskinkod. Från vissa håll har jag hört att registernamnen HL,BC och DE skulle stå för en förkortning: Hi-Low,Byte-Counter & Destination Är detta en efterkonstruktion männe? Varma kalsingar !!!!

(Text 1022) Peter Ohlen <4659>  
Ärende: Konstigt Z80-register?  
WZ-registret har den funktionen att vid instruktioner när man behöver mellanlagra ett värde.  
T ex: LD A,(nn)  
Först hämtas opoden, två följande bytes hämtas och läggs i WZ-reg.  
I nästa steg läggs WZ ut på adr.bussen och läsning beordras, varefter det som finns på databussen läggs i A.  
HL heter så för att det används som adr.reg., och då var det ju logiskt med High-Low.  
Mr Qwerty

(Text 1038) Sven Wickberg <1384>  
Ärende: Formaterar m m  
När man formaterar en skiva med DOSGEN får man en "blank" skiva, vilket innebär att man fr o m lägsta minnesadressen (32768) får FF och 00 i varannan byte. Litet högre upp (möjligtvis vid starten på det gamla 16K minnet) blir det plötsligt 00 i vareda byte.  
Varför?  
När letade efter en adress i basictolken upptäckte jag att ;CALL(292) på min ABC80 med checksumma 9913 (8881 vid 80 tkn) blänker hela minnet med 00 i varje byte. Någon som vet varför?

(Text 1043) Arne Hartelius \* <2731>  
Ärende: Formatering m m  
Vad menar du? Resultatet av DOSGEN.F är en skiva med idel E (E-FNUIT) i varje sektor utom vissa biblioteks-sektorer. Resultatet av DOSGEN utan påföljande F är att biblioteksspår nyinitieras, men att de gamla programmen ligger kvar på skivan och går att leta upp och rätta med i ex Börje Gustavssons förtärliga RESCUE.BAS (ABC80/DISKHANT-biblioteket).

Eftersom .ABSprogram inte kan förutsätta annat än grundutrustning dvs att man bara har minne från 49152 -> så arbetar dessa program bara i den övre delen av minnet. ABS betyder ju ABSolute-adress. Formatering eller dosgenerering av en skiva har inte med den undre delen av minnet att skaffa. Den enda del av minnet som hamnar på skiva är dosbuffert 0 slumpmässiga innehåll som inte töms på skräp innan den kopieras till sektor noll på skivan (den sektor där skivnamnet står).

Jag är säker på att du kan allt det här sedan länge. Men varför blandar du då ihop skivformatering med minneskartan?

(Text 1048) Sven Wickberg <1384>  
Ärende: RAM-minnet efter uppstart  
Rätt, Arne, jag skrev förmodligen i natt-mössan. Jag menade som det står i rubriken nu, att RAM-minnet i min ABC80 efter uppstart innehåller FF resp 00 omväxlande från 32768 och uppåt något på 40000 där det plötsligt blir 00 i varje.  
Jag undrade varför det är så. Det har kanske ingen praktisk betydelse, men man tycker det vore naturligt att ett jmfylligt blankt minne innehåller idel nollor (eller idel ettor) och inte ibland noll och ibland 1.

(Text 1049) Arne Hartelius \* <2731>  
Ärende: RAM-minnet efter kallstart  
Jag undrar också över det där. Helt oviktigt kanske det inte är eftersom den enda byte som skiljer checksumman 10042 från 9913 är en som har till uppgift att ge 'de sista oprögraderade byten' ett värde.  
Varför besvara sig med ett ometändigt checksummebyte för en sån detalj om det saknar vikt.

(Text 1054) Sven Wickberg <1384>  
Ärende: Varning för tillsatser  
Min ABC80 har jag liksom många andra försatt med extra minne och 80 tkn. Det har inte givit några problem. Men det finns mer att välja på: RAM-floppy, UNIDISK, CP/M. Jag har satt in Gejos RAM-floppy för att komma åt BASIC II på ABC80, och då har jag haft alla möjliga konstiga fel: programbortfall, underliga felmeddelanden, låsningar. Eftersom jag körde den "nya" ABC80 med separat kraftpaket och extra monitor trodde jag det var fel på dem, men Håkan hos Björnhems förklarade att de var ok.

Det har nu påpekats för mig att jag antagligen har för mycket grejer i min burk. Likriktaaggagaten till ABC80 tappar orken med tiden, och marginalerna är så små att man lätt råkar ut för att ström och spänning inte räcker till för alla extragrejer. Detta skulle förklara varför vi tidvis haft besvär med alla handikapptillsatser, som också skall dra nytta av spänning och ström från ordinarie strömförsörjning.

Är det någon som rätt ut detta och vet exakt besked?  
Vad jag vet i dag är att t ex CAT-NETS kretskort där nästan dubbelt så mycket som tillsatser skall få göra totalt - det har mättits upp och påpekats för mig. Men det tycks gå bra om man inte har något annat än 32 K RAM extra.

(Text 1055) Gunnar Faith-Ell \* <2733>  
Ärende: Varning för tillsatser  
Jo, det stämmer att nättagregatet till "nya" ABC80:n endast är dimensionerat för dator-delen, prova att byta likriktaarbrygga och säkringar (transformatorn är överdimensionerad). Ev kan spänningsregulatorn behöva bytas även det beror på hur mycket ström man behöver. Jag har hört talas om att vissa nättagregat inte ens orkade driva dator-delen, men efter ovastående åtgärder så gick de bra.

(Text 1062) Mikael Liden <5651>  
Ärende: 16K RAM extra ABC80  
Vilket sätt är enklast och billigast att bygga ut ABC80:s minne med 16K RAM?

(Text 1063) Peter Ohlen <4659>  
Ärende: 16K RAM extra ABC80  
Enklast är att köpa ett kort som du sätter i disken. Ett sådant kort kostar runt en tusenlapp tror jag. Det billigaste är en sats som ABC-Data säljer och den kostar 595 kr+moms. Det krävs dock att man gör fyra lödningar och plockar bort några kretsar och sätta två sm åvårlånga kort i stället för de DRAM som satt där innan. Jag har för mig att ABC-Data kan utföra det om

man tycker att det verkar vara för svårt. Det billigaste är förstås om man kommer över ett kort begagnat.

(Text 1064) Kent Berggren <6019>  
Ärende: 16K RAM extra ABC80  
Det finns ytterligare ett sätt och det är att köpa på den nya kapslarna på de gamla. "pigi back". Beskrivning har funnits i någon gammal Microdatorn. Vilken vet jag ej. Doch finns beskrivningen nere på klubben.

(Text 1065) Bo Michaelsson <913>  
Ärende: Standardbandspelare till ABC 80  
Finns någon tillsatt för att kunna koppla en vanlig bandspelare till ABC 80? Det vore mycket trevligt att kunna spara på kassetter eller rullband nu när man inte kan köpa det vanliga kassetminnet längre. Rullband skulle också kunna användas som en billig förvaringsplats för filer då banden rymmer mycket.

(Text 1066) Sven Wickberg <1384>  
Ärende: Standardbandspelare till ABC 80  
Såvitt jag förstått går en standardbandspelare utmärkt till ABC80 om man sätter på och stänger av den för hand. Den saknar således ett relä som startar och stoppar motorn resp inkopplingen av kapstan-rullen i takt med programmet, annars är det som vanligt.

(Text 1067) Bernt Johansson <3384>  
Ärende: Standardbandspelare till ABC 80  
Det går aldrig att läsa tillbaka filer från en audiobandspelare. Tror att det beoror på att den inte har DC-utgång. Kassetminnet skickar ut en TTL-kompatibel signal med fast DC-nivå.

(Text 1070) Stefan Berg <216>  
Ärende: Standardbandspelare till ABC 80  
Är du säker på det? Jag lyckades efter diverse justeringar både spela in och av en vanlig kassetbandspelare, utan några fel alls. Bergis nåt elektroniskt eller så...

(Text 1071) Kent Berggren <6019>  
Ärende: Standardbandspelare till ABC 80  
Va då skulle det inte kunna gå med en standard bandspelare. Man kan väl ha en nivå omvandlare och en pulsformare så måste det kunna fungera. I alla fall på dessa låga farter.

(Text 1072) Bernt Johansson <3384>  
Ärende: Standardbandspelare till ABC 80  
Ja, det är just det man måste ha. En flankdetektor och pulsformare och det finns i kassetminnet. Dom allra allra första ABC80 gick att köra med en audiospelare. Det följde faktiskt med en kassetradio som standard. Men säkerheten blev för dålig.

(Text 1080) Bo Engborg <2369>  
Ärende: Interrupt och Chr\$(5)  
Vad händer (före), under och efter en interrupt?

Ex...  
Rutin xyz  
xyz  
EXX  
CALL NÄGONTING ; Tar lång tid  
EXX  
FORTS...

Hur ska man förhindra att interrupt sker under Call någonting (för att kunna rädda registren).  
Kanske med rätt kombination av DI, EI ...??  
Vad sker när en Videotexbas begär ENQUIRE, Chr\$(5)??  
Vad ska datorn svara för att därefter kunna ge kodord??  
Hälsningar Bos!!

(Text 1083) Benny Löfgren \* <2615>  
Ärende: Interrupt och Chr\$(5)  
En interruptrutin FÅR INTE förstöra något register, varken de ordinarie eller prim-registren. För att stänga av alla interrupt utom NMI gör man DI, för att slå på dem igen gör man EI. Sedan kan vissa av periferienheterna, t ex PIO- och SIO-kretsarna fås att stänga av sina egna interrupt, men det är olika beroende på vad man pratar med.

(Text 1114) Arne Hartelius \* <2731>  
Ärende: Processorns odelade uppmärksamhet  
Jag förstär inte heller vad jag pratar om. Det är därför jag vill att du ska skriva en artikel om det. En disassemblering av mina inlägg! Nå allvarligt talat. Om man använder sig av en hjälpare typ SuperSmartAid, SuperBasic, Löparen eller varför inte din egen SLAV så kommer dessa att sluta fungera om man använder sig av programmet VÄXLAPGM. Jag antar att detta beror på att den avbrottsrutin som finns i VÄXLAPGM inte gör retur till den avbrottsrutin som finns i ens hjälpare utan istället går direkt tillbaka till Basicloken, där 799, eller var det nu var.

Vad jag efterlyser är alltså en generell rutin som undersöker om det finns någon annan rutin som använder sig av tangentbordsinterrupt och i så fall lagrar startadressen till den i slutet av sig själv. Om inte så gäller adressen för återhopp till token. Rutinen ska alltså bara användas vid initieringen av en ny rutin. Den ska bara lämna ifrån sig en adress som den egentliga rutinen ska använda sig av vid återhopp. Sen kan den utplåna sig själv om den vill.

Tänk dig att du har den här snutten i början av Basicprogrammet VÄXLAPGM. Den undersöker om det finns en hjälpare med i bilden (minneskartan). Denna hjälparens avbrottsrutin börjar på en viss adress. Just denna adress grabbar vår lilla snutt tag på och spar. Fanns det ingen hjälpare är adressen 799. När VÄXLAPGM sen hunnit till in-pokandet av själva assemblerkoden läggs så denna adress in på därför avsedd plats. Förstår du vad jag är ute efter?

(Text 1119) Anders Franzen <5258>  
Ärende: Processorns odelade uppmärksamhet  
Jo, jag anar vad du menar. Det bästa vore kanske att göra som fadern till TERM100, att låta varje program som ställer till det med olika interruptrutiner ståda upp efter sig när man lämnar programmet. Det finns dock många gamla program som inte gör detta och det kan vara mycket jobbigt, kanske onödigt, att skriva om dem. Jag kan inte så här på rak arm säga om man kan spara adressen till en interruptrutin INNAN den lagts in i minnet. Man kan tänka sig ett program som man exekverar först då man lagt in t ex VÄXLA och som på någotvis...

Nej, jag vet inte. Måste tänka, återkommer.

(Text 1133) Arne Hartelius \* <2731>  
Ärende: Mjuk respektive hård exit från program  
Nej, nej. Det är ju just det jag (vi?) vill undvika. Massor av ABC80-ägare har inte printrutinen i prom utan måste ladda in detta vid uppstart. Det gäller alla som har bandspelare eller enkel densitet på på sin drive. Massor av medlemmar använder sig av en ramladad hjälpare. En del har rammaskin och eller ramfloppy.  
Ingen som tillhör någon av de uppräknade grupperna är speciellt förtjust i att behöva initiera systemet efter varje programkörning, så det vore en katastrof om det blev kutym att lämna program med CALL(0). Idealet som jag ser det är som följer:

Att det program man kör återställer BOFA och STACK till de värden de hade innan och dessutom försäkras sig om att enhetslistan inte kommer att förstöras av ny programinladdning.  
Att information om varmstartsadress lämnas om sådan kan vara aktuell i sammanhanget.  
Att man när det gäller assemblerrutiner i möjligaste mån lägger dessa under taket istället för ovanför golvet. Vid ett ev misöde har man då större möjligheter att initiera rutinen på nytt utan ny inladdning.  
Att man inte använder sig av DOS-buffertar utan att undersöka om de är lediga, eller själv använder sig av dem utan vederbörlig uppmärksamhet.  
Jag betraktar RESET-knappen som en nöd-utgång och tycker att det är ett onsyggt sätt att (behöva) avsluta en programkörning på.

(Text 1136) David Andersson <5201>  
Ärende: Processorns odelade uppmärksamhet  
I första-hand kollar initieringsrutinen vad värdet är på I-reg, innan det ställs om. Sedan förutsätter den att låga byten av vektorn (lagrad i PION) är 52 och tittar i minnet (adress I\*256+52, I\*256+53) var tidigare interruptrutin ligger sedan avslutar den inte sin egen interruptrutin med EI.

RETI  
utan med  
JP adress  
där adress är uträknad sålunda:

LD A,I  
LD (SPARA),A ;spara gamla I-reg  
LD H,A  
LD L,52  
LD E,(HL) ;DE=adress till int-rutin  
INC HL  
LD D,(HL)  
LD (address),DE ;självmodifierande kod  
Med denna hantering får man automatiskt JP 798 om I=0.  
Man bör inte glömma att ha en adress till casset-interrupten när man ändrar I.  
Om alla program som använder int gör så här som måste de termineras i motsatt den ordningen de initieras.  
Om PION används för annat än tangent interrupt så blir det genast mycket krångligare.  
(assembler rutinen ovan är inte testad)

(Text 1152) Sven Wickberg <1384>  
Ärende: CALL(292)  
På min ABC80 med cs 9913 innebär ;CALL(292) att HELA interminnet NOLLSTÄLLS. Rubb och stubb inte bara FÖRSVINNAR - varenda adress blir NOLL. Detta sker inte ens vid RESET eller CALL(0). Jag tycker mig ha märkt att CALL(0) ibland inte ens tar bort program ur interminnet utan bara sätter första adressen=1 vilket datorn tolkar som att minnet är ledigt.

Jag har Superbasic och med rutinen OLD kan jag alltså få tillbaka program efter reset (liksom efter NEW som också ettställer första adressen) samt vid program som är preparerade med den gamla run-only-rutinen (som ju verkar så att första byten blir 1 efter körning så att man inte kan lista eller köra igen utan ny inladdning från skivan).

Är det någon som kan förklara verkan av CALL(292) eller varför denna finns över huvud taget - när den inte används vid uppstart av systemet. Då är nämligen varannan adress = 0 och varannan = 255 och en hel del andra oregelbundenheter förekommer. (Jag har tittat efter med Superbasics SCAN.)

För övrigt kan man ta sig ur BYE-mod med \$ABC om man har Magnus Lundbergs Superbasic + diverse andra program, utan att nolla alla andra rutiner i minnet. Vet inte hur det fungerar.

(Text 1153) Benny Löfgren \* <2615>  
Ärende: CALL(292)  
Det är med största sannolikhet så att det inte är någon MEDVETEN rutin som ligger på adress 292. Det är säkert bara en konsekvens av att man hoppar in mitt i en rutin, eller till och med mitt i en instruktion.

(Text 1156) Gunnar Forssell <1631>  
Ärende: CALL(292)  
Var det inte Halmstads-David som hade gått djupare in i det där?

Jag har för mig att han upptäckte att ABC80'n försökte att kompiera en rad, men då man inte hade några ingångsvärden satta så satte den igång med att friskt kompiera ickeexisterande saker. Och då uppstår visst effekten med blankning av minnet...  
(En snabbtitt i dissen av basinen visar att det mycket riktigt är i det området kompileringen sker. Inhoppt sker straxt efter sättning av vitala parametrar...)

(Text 1160) Magnus Bodin <4304>  
Ärende: CALL(292)  
Jo, CALL(0) gör endast NEW genom att sätta BOFA=EOFA, (BOFA)=1.  
CALL(292) är ett inhoppt mitt i en rutin, resultatet blir som det blir pga att man kommer från BASICen. Till skillnad från assemblernivå kan man ju inte här kontrollera vad som ligger i varje register när man anropar en maskinkodsrutin. (Vilken man förresten kan i FORTH!) Resultatet blir ett slags LDIR-FILL. Helt oavsiktligt, men sen Kalle Lindström skrev ett brev till Bladet där det varit allmänt känt, trevligt utlopp för "diskreta" program...

(Text 1168) Gerry Eriksson <4842>  
Ärende: Utslitet tangentbord  
En kondensator är i princip två plattor med ett isolerande skikt emellan t.ex luft, polykarbonat m.m.  
Det är även så som tangenterna i ABC80 fungerar. De kan ses som en variabel kondensator där kapacitansen ändras när du trycker ned dem. Det är denna ändring man känner av mellan de två halvorna och INTE om de är kortslutna (=MYCKRE dålig kondensator). Det blir ett mindre luftgap och därmed ökad kapacitans d.v.s. tangent nedtryckt. Jämför även en radio där ratten för frekvensinställning är kopplad till en vridkondensator som består av ett antal fasta plattor och en axel med plattor på som vrids in olika mycket imellan de fasta = frekvensen ändras. På samma sätt är det med aluminiumplattan på tangenten.

(Text 1178) Kent Berggren <6019>  
Ärende: Utslitet tangentbord  
Om du moterar ner en knapp ser du den utan som skall tvätt på krets kortet. Ser ut som 2 halv av en cirkel. Förtämnad. Det bästa medlet att tvätta bort smuts är isopropanol efter som det inte lämnar några rester.  
Inlägg 1179 är raderat.

(Text 1191) Anders Franzen <5258>  
Ärende: VXLPGM  
Programmet VÄXLAPGM.UTL innehåller några allvarliga buggar. Det går inte att läsa filer från kasset och man får inte skriva in för långa rader när VÄXLAPGM är aktivt.  
ABC80 dyker när man läser från kasset därför att avbrottsvektorn för kassetinterrupt saknas. Om en BASIC-rad skrivs in

och som kompilerad upptar mer än 72 bytes dyker datorn därför att avbrottsvektorn för tangenttryck placeras i kompilieringsbufferten! Man kan åtgärda det första felet på ett enkelt sätt.

Det andra är mer svårslutligt. Det bästa vore om maskinkoden till VAXLAPGM placerades på något annat ställe än i POKE-arean högst upp i minnet. Man kan emellertid flytta avbrottsvektorn till annan plats. Dessa rader gör VAXLAPGM lite bättre:

151 IF PEEK(65064)=251 POKE 65064,250

(sänk stack)  
540 POKE 64052,148,255,148,5  
(interruptvektorn)  
541 POKE -112,250  
(rätt värde till I-reg)  
Rad 540 ersätter gammal rad, de övriga är tillägg. Nu kan man läsa från kassett utan problem. Denna ändring ökar maxlängden på BASIC-rader till 199 bytes vilket dock är för litet ibland.

(Text 1193) Olov Lindström <6411>  
Ärende: Hur adresseras filerna på floppyn  
Track 1 och 2 verkar innehålla pekare till det block (T\*256+S\*32) där ytterligare inf finns.  
Men vad betyder innehållet i den sektorn som pekas på. Det vore fint om någon kunde svara på det.  
O.Lindström

(Text 1195) Bengt Österholm <2077>  
Ärende: Hur adresseras filerna på floppyn  
Det är en headersektor som i sin tur pekar på olika sektorer där programmet finns. Där kan man räkna ut hur många sektorer ett program upptar. Man kan även utläsa på vilken plats i biblioteket filnamnet står på. Det här har beskrivits i ABC-bladet tidigare men jag kommer inte ihåg vilket nummer det var.  
2-82(RED)

(Text 1200) Mikael Hovmöller <4688>  
Ärende: String\$  
Varför klarar inte string\$(x,y) större tal på x än antal bytes som finns kvar i minnet? Varför har man inte gjort en loop, som gör Call 11 x ggr? (Det kanske fungerar så, men då borde det inte bli ERR 3 när x är för stort.)  
M.v.h. Micke

(Text 1202) Anders Franzen <5258>  
Ärende: String\$  
När du skriver "; STRING\$(x,y)" placeras den skapade strängen på stacken. Då måste det ju naturligtvis finnas tillräckligt med utrymme på stacken. Notera att SPACE\$ fungerar på samma sätt. Att göra en loop som gör CALL 11 för varje tecken i strängen vore dumt, men säger att vi nödvändigtvis ska skriva ut strängen? Tänk om vi vill göra "A\$=x+SPACE\$(100)!" Genom att lägga den skapade strängen på stacken i samma format som övriga strängar kan den behandlas som en vanlig sträng också.  
Här kommer ett tips i samband med detta: Skriv ej så här: 10 ; A\$+space\$(x)+b\$+string\$(x,y)  
Skriv så här istället: 10 ; a\$ space\$(x) b\$ string\$(x,y)  
Använd alltså inte "+" i printsatser! Summan av allt läggs på stacken och man kan räkna ut för ERR 3 om programmet är stort!

(Text 1227) Stefan Persson <1980>  
Ärende: Generera BREAK från TERM100  
Kan man på något listigt sätt generera koden för BREAK, en tangent som finns på vissa terminaler och som används i bl a LUNET (Lund University Network). Jag hittar inget tips i handledningen för Term-100. Jag vet inte ens vad BREAK-tangenten skickar ut för kod på nätet.  
Tacksam för hjälp.  
<1980> Stefan P.

(Text 1228) Stefan Berg <216>  
Ärende: Generera BREAK från TERM100  
Jodå. Om du lägger MENY-funktionen som finns på CTRL-E (E-FNUTT) på någon annan tangent (görs genom att gå in i MENY med CTRL-E (E-FNUTT) men att gå därför genom att trycka på t ex CTRL-Q) så kan du generera BREAK med CTRL-E (E-FNUTT). (Här för mig om att jag gjorde så.) Om jag inte missminner mig så skickas då ASCII 0 (NUL) ut på linan, d v s signalen går låg. (Rätta mig, den som vet bättre!)

(Text 1230) Kristoffer Eriksson <5357>  
Ärende: Generera BREAK från TERM100  
Break brukar vara längre än det NULL-tecken, och ingår inte i ASCII-alfabetet. Man får inte break genom att trycka flera gånger på CTRL-E (E-FNUTT) heller, eftersom start- och stopp-bitarna mellan NULL-tecknen skiljer det från break. Men försöka duger ju.

(Text 1275) Kjell Svensson <5318>  
Ärende: ASS  
En medlem (icke modemägare) ville att jag skulle påpeka ett fel i ASS. Han påstod att instruktionen RLC B gav resultatet 203D istället för 203,0.  
Han saknade också möjlighet att ändra en rad som gett syntaxfel. Jag har själv inte provat programmet i fråga.  
Kjell.

(Text 1286) Anders Franzen <5258>  
Ärende: ASS  
Medlemmen utan modem har helt rätt, ASS klarar inte av assemblerinstruktionen RLC B. Hela familjen RLC utom RLC A blir felaktigt. Det beror på ett fatalt fel. Data för alla assemblerinstruktioner ligger i en tabell och datavärdena för RLC-instruktionen är helt riktiga, det är bara det att programmet som läser tabellen tolkar värdet fel! Sorry, jag får väl försöka rätta felet.  
Det går inte att med ASS ändra en rad som gett syntaxfel. Använd istället editorn TED (eller någon annan) för att skriva källkod med, spara texten på fil och kör sedan programmet ASS för att assemblera filen. För att rätta syntaxfel använder man sin editor igen.  
Tills felet med RLC är rättat, var god undvik RLC s där s inte är A.

(Text 1293) Kjell Svensson <5318>  
Ärende: JP 197 resp JP 799  
JP 197 = NEW  
JP 799 = T-bordsinterruptrutinen  
Kjell.

Inlägg 1294 är raderat.

(Text 1295) Kent Berggren <6019>  
Ärende: BASICERR.SYS  
Jag har inte hittat någon som kan förklara för mig hur error texten kommer fram automatiskt vid fel. Igentligen skulle jag vilja använda detta till att åter starta mitt prg istället. Jag har satt någon som lyckats med det, men jag har inte riktigt fattat hur. Igentligen skulle det räkna med att ABC80 gjorde Z=call(0). Är det någon som har grävt ner sig i BASICERR.SYS? Mycket tacksam om någon kan hjälpa mig.

(Text 1298) Anders Franzen <5258>  
Ärende: BASICERR.SYS  
BASIC-tolken är så finurligt skriven att den känner av om BASICERR.SYS finns på dina skivor. Det görs när du trycker RESET. När du får ett fel i BASIC kontrollerar tolken om BASICERR.SYS finns (kollar den flagga som satts av det-som-görs-vid-reset-rutinen) och anropar såfallet en subrutin i DOS. Vad den gör vet jag inte men man kan ju gissa! Den läser väl in aktuell felmeddelande beroende på felkod och visar på skärmen...

(Text 1327) Kjell Svensson <5318>  
Ärende: TKN 40/80 - jag blir galen!  
Det finns många filer ställen man kan kolla radbredd på, PEEK(590) eller PEEK(472) bör vara de mest korrekta ställena att titta på. Där får du ju 40/80 som svar.  
PEEK(884) kan variera beroende på vilken 80tkn-ombyggnad man har.  
Kjell S.

(Text 1332) Bengt Österholm <2077>  
Ärende: TKN 40/80 - jag blir galen!  
Jag tycker att peek (900) är mycket bra för detta ändamål. Det är ju lätt att komma ihåg och resultatet är också 40 respektive 80 beroende på vilket mode skärmen är i.

(Text 1350) Peter Ohlen <4659>  
Ärende: Odokumenterad Z80-instruktion  
Instruktionen vars kod är CB,37 hex är inte dokumenterad men finns i Z80n. Den ska ha följande funktion, "Shift left, insert 1". Är det någon som vet varför den inte är dokumenterad och om det finns fler sådana instruktioner.  
/nyfiken

(Text 1360) Kjell Svensson <5318>  
Ärende: SEKTOR 1-5 PÅ SKIVAN  
Använder sektor 1-5 på skivorna till något? mvh Kjell S.

(Text 1362) Peter Ohlen <4659>  
Ärende: SEKTOR 1-5 PÅ SKIVAN  
Nej!

(Text 1365) Kristoffer Eriksson <5357>  
Ärende: Odokumenterad Z80-instruktion  
Om instruktionen inte finns dokumenterad, hur kan då veta att den finns? Jag har för mig att någon för en tid sedan skrev att de har "gluggarna" i instruktionsuppsättningen berodde på instruktionerna inte fungerade som de skulle.

(Text 1369) Kent Berggren <6019>  
Ärende: Odokumenterad Z80-instruktion  
Det finns någon eller några instruktioner som inte finns dokumenterade. Jag minns att det var någon på Metric (troligen Björn Damm) som skrev ett prg där han utnyttjade sig av en av dessa rutiner. Sedan märkte man att prg inte fungerade på alla Z80 och

han fick ringa Zilog i staterna och kolla. Det fanns då flera "masker" av Z80 och det skilde sig mella olika fabrikat. (Masker = layout på kretsen)  
Jag tror att ni skall fråga Mats Petterson 1082 om han minns det hela. Han brukar logga in i bland här.

(Text 1374) Sven Wickberg <1384>  
Ärende: Mina filer sabbas  
Jag har en skiva med autostart och TERM100 som alltid fixar till en massa saker vid inkoppling (fixar printerformat och ger viss valfrihet till TED, TERM100 m m)  
Plötsligt började det uppträda ERR 35, 37 och framför allt 48 på diverse filer.  
Jag tog Magnus Lundbergs DDISP för att titta på skivan: biblioteket på spår 2 såg rätt ut, men när jag tittade på en av de aktuella filerna fann jag att headern var bortat! Sabbad av en annan fil som skrivit över den sektorn.  
Jag har hört att det felet kan uppkomma, men jag har också MLIS justerade DOS som lär ska ha rättat till det felet. Nu är min fråga: är detta vanligt förekommande? Kan man göra något åt det? Jag hade BACKUP /duktig gosse! så det gäller inte filerna, utan själva systemet.)  
I annat fall är det tydligen lugnast att aldrig ta bort en fil som en gång kommit in på skivan. Det tycks gå bra så länge man lägger till alla filerna bak, men när man börjar leta efter lediga sektorer inne bland de andra händer det här.

(Text 1376) Magnus Stråle <2504>  
Ärende: Odokumenterad Z80-instruktion  
De icke-dokumenterade instruktioner som jag känner till berör IX & IY registerna. Man kan tex addera A med låg eller hög del av IX resp. IY. Jag har tyvärr inte mina papper tillgängliga nu, men det skall nog inte vara så svårt att lista ut objekt-koden för dessa instruktioner.  
M.v.h MS

(Text 1377) Bengt Österholm <2077>  
Ärende: Mina filer sabbas  
Jo, jag har varit med om det där också. Det är tydligen så att headern aldrig kan ligga ensam, utan när den skrivs in så skrivs även nästa sektor in oavsett om den sektorn är ledig eller ej.  
Det enda botmedlet jag kunde komma på var att när man har brukat en diskett en längre tid så att den är ordentligt segmenterad så kan det vara lämpligt att kopiera den fil för fil så att alla lediga sektorer hamnar längst bak tillsammans.

(Text 1382) Kenneth Borg <1738>  
Ärende: Likriktare  
Efter att ha servat ca 25 abc80 i 5 år vill jag säga att det i särklass vanligaste felet på abc80 är att den stora likriktarbryggan till 5v lämnar in. Skumma fel som rensningar av minnet och svajig bild o.dyl. kan bero på likriktaren. Kontroll göres med oscilloskop, och man mäter spänningen över den stora elektrolytkondensatorn. Om frekvensen på "ripple" är 50 Hz är likriktaren trasig.

(Text 1411) Sven Wickberg <1384>  
Ärende: Sändfil med TERM100 ?  
Jag har TERM100 (ett fantastiskt bra program !!) som har fungerat till belåtenhet nästan ett år. Men nu har jag fått problem: När jag försöker sända en fil, huggs de 3-4 vänstersta tecknen bort, dvs i stort sett radnumret.  
Vad kan det bero på? Jag försökte skicka filen till Norrköpingsmonitorn, och det blev likadant varje gång.

(Text 1413) Arne Hellström <4838>  
Ärende: TERM100 Ver 1.4  
Kan någon hjälpa mig att ändra på antalet sektorer som reserveras på disketten med Y1 i TERM100. Antalet sektorer som nu reserveras (preparerar?) är 32. När dom är använda så reserveras 32 nya. Det är i den skarven jag får skräp i mina filer. Om man kunde välja antal sektorer själv så skulle det inte behöva hända. Detta kanske inte händer på en modern drive Men jag har en gammal FD2.

(Text 1418) David Andersson <5201>  
Ärende: Sändfil med TERM100 ?  
TERM100 sänder nog alla tecken korrekt, men Norrköpingsmonitorn kan nog inte ta emot tecken som kommer direkt efter en carriage-return, alltså de första teckna på en rad.

(Text 1419) David Andersson <5201>  
Ärende: TERM100 och diskallokering  
TERM100 reserverar inga sektorer på disketten. Det gör DOSet när man skapar en fil och skriver till filen.  
Doset ska deallokera överflödiga sektorer när filen stängs.

(Text 1421) Johan Struwe <3338>  
Ärende: BASIC-problem ?!  
Jag satt och höll på med ett BASIC-program till min ABC80 där jag använde listor och matriser. I detta fall strängvariabler A\$(x) och A\$(y,z). I början av programmet hade jag en DIM-sats:  
10 DIM A\$(100)  
och lite längre ner hade använde jag sträng-listan med samma variabelnamn.  
100 ; A\$(1,4)  
Detta ville inte datorn gå med på. Jag fick ERR 1 LINE 100, "Fel antal index". Inte förrän jag bytte variabelnamnet på rad 100 till 100 ; B\$(1,4) så fungerade det. Ska det vara så här? Jag har hittills haft för mig att ABC80 kan använda alla variabeltyper samtidigt och oberoende av varandra.

Har jag haft fel?  
(Hoppas jag har presenterat frågan korrekt - det var en tid sedan problemet uppstod - minnet kanske sviker...)

(Text 1423) Bengt Österholm <2077>  
Ärende: TERM100 Ver 1.4  
Det låter som om du inte har satt protokoll till YES i keypad-mode R4. När din dator har fyllt 32 sektorer så lägger den upp 32 sektorer till. Detta tar lite tid och om du då har satt protokollet så sänder din dator XOFF när den inte hinner med att lagra alla tecken som kommer. Då slutar monitorn att sända tills du sänder XON igen. Det görs automatiskt när din FD2:a har skrivit klart i header-sektorn och är redo att ta emot mer data.

(Text 1436) Anders Franzen <5258>  
Ärende: BASIC-problem ?!  
Pga en bugg i tolken kan man inte ens skriva 10 DIM A\$(1),A\$(1,1)=1  
När man skriver RUN får man ERR 1 (fel antal index). Det är alltså inte möjligt att ha en vektor och en matris med samma namn. Fast det går ju att kalla dem för A1\$ och A2\$. Samma underlighet gäller heltals- och flyttalsvariabler.

(Text 1437) Gerry Eriksson <4842>  
Ärende: BASIC-problem ?!  
Som jag ser det är det inte frågan om en bugg i tolken. Det finns två huvudtyper av variabler enkla resp. indexerade variabler (skalära o matrisvar.). Och du bara kan använda samma variabelnamn en gång för resp. variabeltyp. Hur många index den innehåller spelar då ingen roll utan namnet är upptaget för typen indexerade variabler. Däremot kan man ha samma namn men olika antal index för heltal, flyttal och strängar. T.ex. DIM A%(20),A\$(10),A\$(4,4) Samma regler gäller för övrigt även 800:an. Mvh Gerry

(Text 1438) Arne Hartelius \* <2731>  
Ärende: Matriser på ABC80  
A\$(4,4) var väl ändå att ta i ?

(Text 1448) Anders Olsson <1019>  
Ärende: Reparationer av flexskivheter  
Om det kostar mellan 1000 o 2000 kronor att få en FD reparerad borde väl den lite händigare hellre än att lämna FD:n på rep köpa en ny drive o sätta in. Det går att få nya slimline drivar för under 1500 kronor. De är dessutom mycket driftsäkrare än de gamla fullhöjdarna. (Direktrikt utan rem, stegmotor med stålband i stället för skruv m m).

Man måste bli mycket misstänksam om man får tillbaka sin floppy efter rep och "det enda som behövdes var justering". Justeringen \*ändrar\* sig normalt inte (om inte den som justerat sist glömt att dra åt skruvarna), men däremot kan olika detaljer slitas ner så att justeringen inte stämmer. Då måste man först byta den slitna detaljen och först därefter justera.

(Text 1455) Kent Berggren <6019>  
Ärende: RTS  
Anders F. frågade mig varför det är så viktigt att kunna hantera det. Jo det använder man när man t ex "pratar" med en printer. Den talar då om att den är klar att ta emot data. RTS o CTS. Kan man använda sig av när man inte har XON/XOFF, och inte kör med modem. Ta till ex om du vill föra över en file från en anna dator som inte har det protokollet. Du tappar ut datan ur dess printer utgång. Och för att stoppa data använder du dig då av RTS o CTS. Mycket enkelt. Tag Metrics prg T80RTS som finns som är det enda prg till ABC80 som jag vet som hanterar dessa signaler okey. Jag kan oxo tänka mig en plotter som man kan köra 2vägs. Både skriva till och hämta från. Att där syslar man inte med XON/XOFF. I alla fall finns det massa olika uppfattningar om hur man skall hantera data vi v24:

(Text 1466) Bengt Holgersson <559>  
Ärende: Extraminne till ABC80 med 64k enligt MIKRODATORN  
Kort beskrivning hur man kan bygga ut sin ABC80 med 64k till multiplar av 64k. (128k, 192k, 256k o.s.v.)  
Beskrivning av grundutbyggnaden finns i tidningen MIKRODATORN. (Nr 9, 10 -83 och Nr 1 -84)  
OBS !!! DETTA ÄR INGET FÖR PERSONER UTAN PRAKTISKA KUNSKAPER OM HUR EN MIKROPROCESSOR ARBETAR. Principen för expanderingen är att löda på fler RAM-kapslar ovanpå de redan existerande. (Piggybacking)  
De enda stift på RAM-kapslarna som skall kopplas annorlunda är:

RAS (Row Address Strobe)  
Klockar in ena halvan av adressen

CAS (Column Adress Strobe)  
Klockar in andra halvan av adressen

För varje ny bank med 64k behövs:  
1st 74LS10  
8st 4164 (Obs, måste vara varianten med 7-bit refreshadress)

Se bild längre ned.

Adressavkodningen av de nya bankerna görs lämpligen av ett bipolärt prom i likhet med tidigare. Man får då stor frihet att skräddarsy vissa banker för speciella funktioner. (CP/M, BASIC II, RAMDISK el.dyl) Att bygga en programmeringsenhet för bipolära prom kan man göra ganska enkelt med hjälp av databladet över prommen. I övrigt är det kanske inte lämpligt att använda OUI7 som utport för bankswitchningen eftersom denna ej är fullständigt avkodad.  
M.v.h Bengt H

(Text 1472) Anders Olsson <1019>  
Ärende: Extraminne till ABC80 med 64k enligt MIKRODATORN

Tack, Bengt! Du har utan att veta om det gett mig lösningen på något som jag har grunnat på i ett par år, nämligen: 4164 MÅSTE VARA VARIANTEN MED 7-BIT REFRESH  
Jag har klurat o klurat o klurat på den där beskrivningen i Mikrodatorn och det enda jag har kommit på är att den rent teoretiskt aldrig skulle kunna fungera. Mina 4164or har 8-bits refresh. Jag gjorde en egen konstruktion med ett litet kretskort som lade till den 8:e biten.  
Vad har 4164:or med 7-bits refresh för fabrikat / beteckning?

(Text 1476) Leif Andersson <155>  
Ärende: SNABBFORMATTERING  
Kan någon tipsa om ett prg som formatterar disketter utan att man varje gång behöver svara på alla frågor. Har ett svagt minne att det skall finnas någonstans, vad heter det i så fall?

(Text 1479) Jörgen Gustavsson <3528>  
Ärende: SNABBFORMATTERING  
Det finns ett program på sista eller näst sista kassetten som heter FORM (tror jag). Visserligen frågar programmet, men om du inte vill svara på frågor är det lätt att ändra programmet. Att programmet är lätt att ändra beror på att det är gjort i basic och inte kör under CMDINT. Börje Gustavsson har visst svängt ihop programmet.

(Text 1484) Gunnar Forssell <1631>  
Ärende: Ram-disk.  
För alla er glada som har byggt ut ABC'n med 64k med hjälp av Mikrodatorn så har jag modifierat den ram-diskrutin som har funnits här på monitorn lite. Den funkar som ABC80's ram-disk.  
Jag ska lägga in den nu, få se om jag klarar av det med TERM100...

(Text 1489) Gunnar Forssell <1631>  
Ärende: Extraminne till ABC80 med 64k enligt MIKRODATORN  
Bra beskrivning! Jag har ett par frågor dock.  
Kan du göra prommar enligt specar? jag skulle vilja ha bankar om 32k vardera där det är de nedre 32k'na som berörs.  
Hur fixar man CS-signalen? Via prommet? Kan inte du skriva lite mer utförligt hur man gör? Visa hur man gör en komplett 64k extra sats alltså?

(Text 1494) Bengt Holgersson <559>  
Ärende: Extraminne till ABC80 med 64k enligt MIKRODATORN  
De flesta japanska varianter av 4164 har 7 bit refreshadress. Det enda jag har stött på med 8-bit adress är Siemens. Men det finns säkert flera. Det säkraste är att fråga INNAN man köper dem. Man kan få 150ns 4164:or för ca 10 kr. st:  
M.v.h Bengt H

(Text 1496) Bengt Holgersson <559>  
Ärende: ADRESSAVKODNING MED BIPOLÄRA PROM  
Vi kan ta ett exempel som gäller alla nämligen ABC80's ursprungliga adressavkodning. Detta görs i ett bipolärt PROM som sitter i position E7 och har beteckningen ABC80/13

Man bör nog ha ett schema över datorn för att hänga med riktigt. (Sno er och skaffa en servicemanual innan de tar slut.) Nåväl, i denna kan man se att detta prom har storleken 256 x 4 vilket betyder att man har 256 celler om 4 bitar vardera. För att kunna adressera dessa 256 celler behövs 8 ledningar varav 6 stycken är kopplade till CPU:ns 6 övre adressbitar. Detta betyder att man kan få ut en valfri 4-bit kod för varje 1k-block av processorns adressområde.

De resterande 2 inbitarna är kopplade till två bygglar för att kunna välja in av 4 minneskarter. (0-3) Normalt har dessa värdena 1 resp. 0 vilket medför att minneskarta 1 är vald. De 4 utbitarna används sedan för att välja 1 av 4 typer av minne:

|       |              |           |
|-------|--------------|-----------|
| BIT 0 | Extern minne | Aktiv låg |
| BIT 1 | Intern ROM   | Aktiv låg |
| BIT 2 | Bildminne    | Aktiv hög |
| BIT 3 | Intern RAM   | Aktiv låg |

Eftersom man bara vill ha 1 typ av minne inkopplat åt gången så får vi följande koder för de olika typerna av minne.

|              |          |   |   |
|--------------|----------|---|---|
| Extern minne | 1111 = F | - | - |
| Extern minne | 1010 = A | - | - |
| Intern ROM   | 1001 = 9 | - | - |

Om vi sedan tittar på innehållet i prommet (ABC80/13) så kan vi försöka gissa vad konstruktören har haft i tankarna.

|     |                  |                |
|-----|------------------|----------------|
| 000 | AAAAAAAAAAAAAAAA | !MINNESKARTA 0 |
| 001 | AAAAAAAAAAAAAAAA | !              |
| 020 | AAAAAAAAAAAAAAAA | !              |
| 030 | 3333333333333333 | !              |
| 040 | 9999999999999999 | !MINNESKARTA 1 |
| 050 | AAAAAAAAAAAAAAAA | !              |
| 060 | AAAAAAAAAAAAAAAA | !              |
| 070 | 3333333333333333 | !              |
| 080 | 9999999999999999 | !MINNESKARTA 2 |
| 090 | 9999999999999999 | !              |
| 0A0 | AAAAAAAAAAAAAAAA | !              |
| 0B0 | 3333333333333333 | !              |
| 0C0 | 9999999999999999 | !MINNESKARTA 3 |
| 0D0 | AAAAAAAAAAAAAAAA | !              |
| 0E0 | AAAAAAAAAAAAAAAA | !              |
| 0F0 | 3333333333333333 | !              |

M.v.h Bengt H

(Text 1507) Lennart Einarsson <6554>  
Ärende: FÖR LÅNG RAD....  
Jag har ett program som är sparat på kassett, det är inkört med 'LIST' kommando. Och det är en rad i programmet som har mer än 120tkn. Jag är väldigt intresserad av att få igång den filen igen men jag lyckas inget vidare. Det enda jag kan åstadkomma är ERR 20. Men jag har för mig att man kan poka maskinen till att inte bry sig om sådana bagateller. Problemet är, vilken adress är det.  
H E L P ... M E !!

(Text 1509) Gunnar Forssell <1631>  
Ärende: FÖR LÅNG RAD....  
Jag har för mig att det finns ett prg, "ERR20FIX" eller något dittat, som gör det du vill. Det är inte bara att poka!

(Text 1511) Kent Berggren <6019>  
Ärende: LÅNGRAD.BAC  
Ligger i KAS/K17  
Den räddar för långa rader.

(Text 1517) Arne Hartelius \* <2731>  
Ärende: FÖR LÅNG RAD....  
Använd FILOMV .BAC - CAS 2 eller LÅNGRAD .BAC - CAS 17 eller TED .BAC - ABC/EDITORER

(Text 1542) Bo Kullmar \* <1789>  
Ärende: Tangenter till ABC80  
Enligt Kjell Svensson (som in saknar modem) så säljer Björnheims (återförsäljare i Sthlm) lösa tangenter till ABC80. Luxor säljer bara hela tangentbord och det blir dyrare.

(Text 1555) Anders Johnson <4001>  
Ärende: 16K extra RAM som självbygge  
Det finns en beskrivning för utbyggnad av ABC80:s RAM till 64K i Mikrodatorn nr 9 1983. Finns det någonstans en motsvarande beskrivning för utbyggnad med bara 16K extra, dvs motsvarande div. tillsatser som finns till salu hos olika firmor? Jag tycker det är för dyrt med ca 600:- plus moms för en inbyggdssats för 16K extra. Det är, tror jag, ungefär vad den billigaste kostar.

(Text 1561) Gunnar Faith-Ell \* <2733>  
Ärende: 16K extra RAM som självbygge  
Ja, det finns det i nummer 7, 1982, den beskrivningen lär faktiskt fungera utan ett antal uppdateringar i senare nummer.

(Text 1562) Kent Berggren <6019>  
Ärende: 16K extra RAM som självbygge  
Jag tror att det är den jag har byggt efter Funkar ganska bra, men inte om du har GEJE:s ramfloppa.

Inlägg 1563 är raderat.  
Inlägg 1564 är raderat.

(Text 1568) Anders Johnson <4001>  
Ärende: Bildskärm/TV/TKN80  
Skall skaffa en bildskärm för 80-teckens ABC80.  
Duger en svartvit TV till detta? Jag syftar på billiga 12-tumsapparater som finns till salu lite varstans. Om videobandbredden är för liten, brukar det då vara enkelt att öka den genom något smärre ingrepp (såsom om fallet med originalskärmen för ABC80)? Det vore intressant att se åsikten hos någon som har personlig erfarenhet.

(Text 1569) Arne Hartelius \* <2731>  
Ärende: Bildskärm/TV/TKN80  
Jag borde vara den sista att yttra mig om hårdvarusidan, men vad jag vet ligger det till så här:  
En vanlig TV ritar 625 linjer och börjar sen om med första linjen. Detta görs under en viss tidsenhet. (25 ggr / sek) ?  
En monitor ritar 312 linjer under samma tidsenhet, men gör det i gengäld 2 ggr. Detta för att få en flimmerfri bild.  
Så en monitor borde väl alltid vara att föredra. Men jag kanske har missuppfattat det hela.

## Möte Medforum

(Text 1239) Stefan Persson <1980>  
Ärende: VAD KÖR NI PÅ?!!  
För dem som har möjlighet och intresse/fallenhet att utveckla program till ABC80 med 32 K och 40/80 tecken, så får man väl utgå från att en större del av klubbmedlemmarna fortfarande nyttjar denna utrustning? Sven Wickberg har ju t ex sjungit trojkanarens lov flera gånger, och jag har ingen tanke på att byta. Hur är det med de andra i klubben? Av inlägg i bladet att döma finns det fortfarande många "entusiaster" kvar.

(Text 1240) Göran Westlund <5976>  
Ärende: VAD KÖR NI PÅ?!!  
Kanske kan ABC80/802/806 så småningom bli "hobbymaskin"? Det finns ju väldigt många maskiner ute (jag har hört om kring 30 000, kan det vara sant?). De flest av dessa har väl anskaffats för "yrkesmässigt" bruk. Då kommer de också antagligen att fungera länge än efter det att företaget bytt. Jag vet t ex att ett stort antal statliga myndigheter är på väg att byta till större maskiner. Så det borde bli en marknad för billiga begagnade 800-system. Kanske kunde klubben med hjälp av Luxor bevaka detta? Hellre borde ju dessa dyrgripar spridas till glada amatörer (t ex f d ABC80-ägare) än hamna i sopen.

Sen är man ju väldigt spänd på hur 800an ska vidareutvecklas. "Den får väl själv" hävdade en "Nokia närtstående person" som jag talade med nyligen. Han menade att Nokia inte kunde ha plats för två olika PC-system inom sin produktion. Luxor kanske i framtiden får ägna sig åt att tillverka Nokias PC? Men Nokia-PC:n är ju en vanlig, charmlos MS-DOS-maskin. Vad jag ser fram emot är några nya 800-modeller, t ex "804" - en prisbillig 802-variant med 3" disketter (paketpris 8000 med ORDIII, Kalkyl och registerprogram i paketet, ässå en hyfsad skrivare för 1500 därtill, om man vill ha), samt en "808", en 806a med monokrom grafik samt 20 Mbyte halvhöjds winchester inbyggd, i kompakt utförande. Förlagsvis till ett pris av ca 12000:- Och en "810", lika 808 men med nätmöjlighet integrerad. Är det någon som vet Luxors policy? Tar 1600 och 9000 allt intresse nu, eller kommer 800an att leva vidare?

(Text 1347) Patric Ljung <5455>  
Ärende: Kopieringskydd  
I Bladet nr 3 1985 står det på sid 50 om kopieringskydd. Man skulle alltså sticka hål på skivan och sedan kolla om det går att läsa sabbad sektor och skriva på den. Detta är inte allt för bra, då läshuvudet kan skadas om skivan är deformation. Vidare kan man väl sätta på skyddstejpen vilket innebär att man inte kan skriva, eller? Om man då sätter på en skyddstejp, kan man inte skriva och skyddet blir meningslöst om man sätter tejpa på skivan (även original). Stämmer inte detta???

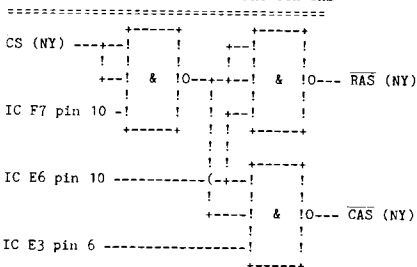
(Text 1348) Kent Berggren <6019>  
Ärende: Kopieringskydd  
Du har missuppfattat det hela. Det är frågan om illegalkopiering.

(Text 1350) Benny Löfgren \* <2615>  
Ärende: Kopieringskydd  
Jag tror inte att förslaget innebär någon skrivning, det räcker väl att konstatera ifall skivan går att läsa, eller hur? Vad beträffar skador på läshuvudet så finns det bot för det också, dock inte med så enkla medel som det nästskönt. Det har nämligen i USA utvecklats utrustning för att "brännmärka" skivor med laser, man bränner ett litet, litet hål i skivan, och detta håll ger samma effekt som ovan nämnda nålstick.

(Text 1353) Patric Ljung <5455>  
Ärende: Kopieringskydd  
Att man brännmärker skivor med laser visste jag, och att nälmärkning är litet riskabelt med tanke på läshuvudet. Dessutom syns det ju, borde i alla fall synas, att man petat hål. Så det borde ju inte vara en omöjlighet att "pricka" in samma sektor, eller? Om man är litet exakt av sig så.  
Moss / Patric

(Text 1379) Bo Kullmar \* <1789>  
Ärende: Försäljning av adresser ur medlemsregistret  
Vi säljer i bland adresser ur registret och enligt styrelsebeslut 1984 så är kostnaden 500 kronor i engångsavgift och 2 kronor per etikett. Detta har Berit Gustavii själv rätt att avtala med "kunder" om. I tveksamma fall hänvisar hon till mig eller styrelsen enligt nämnd beslut.

SCHEMA ÖVER INKOPPLINGEN AV RAS OCH CAS



Naturligtvis så blir det inga etiketter till de som har meddelat oss att de inte vill ha direktreklam. Sen är det ju självklart att vi säljer bara för direktreklam för data-produkter. Vi har inte sålt så många adress-etiketter hittills.

Nu finns det dock en framställan från P-Data. De var inte nöjd med priset ovan, så de har kontaktat Stig Löfgren. De skall få etiketter men jag vet inte vad Stig har gjort för överenskommelse vad gäller priset. P-Data vill visst erbjuda medlemmarna att köpa program till 50% rabatt. Jag känner dock inte till några närmare detaljer, så fråga mig inte!

Annars vill vi helst att de skall annonsera i ABC-Bladet i stället. Datainspektionen har inget emot att vi säljer adresser. Vi får göra vad vi vill med vårt register. Vi har licens. Däremot är det ett brott mot datalagen om någon medlem skriver av matrikeln och upprättar ett dataregister. Det spelar då ingen roll hur den tekniska överföringen sker.

(Text 1428) Benny Löfgren \* <2615>  
Ärende: Kopieringskydd  
Ja, du får för det första CRC-fel när du försöker läsa in sektorn. Dock är det då så att sektorn ligger kvar i kontrollern, och kan enkelt läsas därifrån med ett enkelt controllerkommando.

(Text 1462) Gunnar Faith-Ell \* <2733>  
Ärende: Kopieringskydd  
Vissa datorer kan få driven att strunta i tapen (driven måste då naturligtvis vara gjord för det oxi). Anledningen är att de, avsiktligt, kan skriva sönder en piratkopia.

(Text 1466) Stefan Berg <216>  
Ärende: Kopieringskydd  
Hm, det är väl bergis inte så bra för företagets image? Vad blir nästa steg - att slå ut själva burken om det ligger en piratkopia i minnet? (Borde gå om man använder en gammal PET, har jag hört nåstans...)

(Text 1473) Gunnar Faith-Ell \* <2733>  
Ärende: Kopieringskydd  
Har inte Apples Macintosh vissa möjligheter till att kvadda diskar med skyddsremsan på?

(Text 1481) Arne Hartelius \* <2731>  
Ärende: Styrelse presentation.  
Mitt telefonnummer står varken i gamla matrikeln eller på något annat ställe i ABC-sammanhang. Ändå blir jag dagligen uppringd av medlemmar som vill ställa frågor och få tjänster utförda. Att publicera våra telefonnummer på framträdande plats vore ju en uppmuntran till medlemmar att ringa i tid och otid.

Kent, du måste förstå att vi redan lägger ner större delen av vår fritid på ett obetalt arbete, och att vi vill ha en privat del av tillvaron också.  
Våra telefonnummer kommer att finnas i den nya matrikeln, om än inte speciellt duplicerade på baksidan.

## Möte MONITOR

(Text 2449) Bo Kullmar \* <1789>  
Ärende: 020-linjen  
Som jag skriver i ABC-Bladet som kommer ut nu är det in i det mest möjligt att införa 020-tjänsten, tyvärr. För det första så tilltar Lurverket bara det för Datavissnssystem och då skall det vara temporärt fram till att man kopplar ihop sig med Televerkets Datavissnssystem.  
För det andra så är intresset för det från medlemmarna för litet. Den förfrågan som jag riktade i ABC-Bladet nr 2 har gett ett mycket litet antal svar. Något 10 tal, bara.  
Detta tycks beror på att många som kör riks, kör det på ett företags bekostnad. Sen är det ju många som riner från 08-området, vi har ju 1/4 av medlemmarna här uppe.  
Televerket skall starta ett brevsystem och till det kommer man att använda 020. Kanske kan Televerket mjukna med tiden och då kanske kan vi skaffa oss det. 02 är i samma situation som vi. Vanlig 020 tjänst som erbjuds till vanliga abonnenter är för dyrt för där får man betala \*\*varje\*\* samtal till högsta rikstaxan.

(Text 2470) Bo Kullmar \* <1789>  
Ärende: I år!  
Nu om några timmar är det snart ett år sedan vi öppnade sysemet. Oj vilken skillnad det är nu på kapacitet vad gäller antalet linjer. Vi har ju nu dessutom två nya 802:or som bara väntar på att kopplas in. Winchester till LUX-NET kommer nu att leveras vilken dag som helst. Klubben

blir den första kund som köper den. Den finns avbildad i Transfors annonsbilaga på bild nr 9. Den innehåller en 52 MB winchester, en 640 KB floppy och en streamer för bandbackup samt 15 kortplatser. Jag kommer ihåg när jag tog ner den gamla monitorn och startade upp den nya för alla medlemmarna! Vi hade problem med CAT-NET:s centrala ISAM då. Det vara bara en prerelease, dvs förutgåva. Den klarade inte index på svensk sortering, men när vi körde med ASCII sortering så gick det. Detta innebär faktiskt att vi fortfarande kör med ASCII sortering och inte svensk ordning på ÅÄÖ! Fast det har väl ingen märkt!  
PS annonsbilaga i ABC-Bladet nr 3, som just nu är på väg ut med posten.

(Text 2522) Sven Wickberg <1384>  
Ärende: Har det hänt något med PB?  
Ursäkta att jag stör i ABC-monitorn, men hur ska man klagas i PB när man inte kommer ut? Jag har ringt 20 samtal i dag på eftermiddagen. Försökt alla numren varit för sig. Antingen blir det inget svar, eller är det LINE FAILURE (jag ha automatmodem) eller kommer jag fram men får inte inloggningssekvensen. EN GÅNG kom jag in, men blev efter en stund abrupt utkastad. Någon som vet eller kan ge beskedet vidare?

(Text 2523) Bo Kullmar \* <1789>  
Ärende: Har det hänt något med PB?  
Ja, jag brukar köra på PB någon gång i veckan bara. Det händer att jag får problem att komma in också. Fast du verkar ha ovanligt mycket problem.  
Ett tips kan vara att du kanske skall felanmäla din telefon och säga att det går dåligt när du ringer Åkersberga. Jag hade liknande problem när jag ringde för 80 64 40. Ibland kom jag in på en dålig linje som gjorde att jag \*\*alltid\*\* tappade bär-vägen efter några minuter. Televerket sa att jag skulle felanmäla min egen telefon i stället, vilket jag har gjort. Får se om problemen upphör.

(Text 2528) Kent Berggren <6019>  
Ärende: XOFF  
Jag har kommit under fund med att det efter ett Xoff kommer en del tecken innan det hinner stoppa på andra sida. Ca 7 st i ABC-monitorfallet. Hur är det med andra stora monitor behöver skriva av sig fler tecken innan de stoppar. Jag skall göra en buffert där de tecken som kommer efter ett XOFF skall hamna och då vill jag veta hur stor man måste göra den.

(Text 2542) Benny Löfgren \* <2615>  
Ärende: XOFF  
Monitorn reagerar blixtnabbt på XOFF, så jag tror inte på att 40 tecken inte räcker. Är det inte ditt BASIC-program som inte hinner med? Längst tid tar det i 1200/75, då XOFF-tecknet givetvis skickas med 75 baud, dvs 1/16-del av den mottagande hastigheten. Då får man ett teoretiskt glapp på 16 tecken innan XOFF ger effekt.

(Text 2570) Bo Kullmar \* <1789>  
Ärende: Sökning i MSG  
MSG programmet i CAT-NET är så stort som det kan vara. Det är länk squeeze, eftersom man inte kan ta in ett så stort program i minnet. I LUX-NET består MSG av två program, varav nr 2 är lite mindre. Som sagt är det inte programstorleken som hindrar det för det går att göra ett 3:e MSG program för sökning. Nej, det är alltså det faktum att disken skulle bli alltför slös som hindrar det. Vi får räkna med att vi måste bygga ut systemet med fler noder i stället!

(Text 2648) Bo Larsson <5630>  
Ärende: Koppla V.24/RS-232  
BOK att REKOMENDERAR. Igår köpte jag boken Koppla V.24/RS-232 av Joe Campbell, en mycket lärorik och faktiskt smält underhållande bok.  
Boken beskriver de problem man kan stöta på när man skall försöka koppla ihop två seriella enheter, hur man lurar gränssnittet samt mycket mera. Detta är en bok som borde ha funnits hemma hos de datorpular som blivit gråhåriga vid tretton års ålder när de i sitt anletes svett försökt få sin nyinköpta printer handskaka med sin dator. För att underlätta interfacing beskriver boken vidstående koppling, som är enkelt testverktyg för att analysera spänningar och polaritet i sitt gränssnitt. För att rätt lysdiöd skall reagera enligt ovanstående har jag märkt den flata sidan (och eller) det korta benet med (o. Resistansen på motståndet på kan variera med 20%. Med denna koppling i bakhuvudet kan man ju för en billig penning lätt tillverka sin egen Break-out Box, som i handeln kostar skojtan.

(Text 2652) Kent Berggren <6019>  
Ärende: KERMIT  
Hur går det med att implementera den till monitorn? Jag har för mig att jag någon stans har sett att man hade för avsikt att göra det. Det ligger ett program som heter Rkermitt under ABC800/kommunik. Vad jag förstår så är den ansed för att användas i monitor system.

(Text 2655) Bo Kullmar \* <1789>  
Ärende: KERMIT  
Inget är gjort ännu. LG skall göra, men nu skall vi i första hand köra igång LUX-NET systemet. Vi får se när LG har tid annars får jag väl göra en insats sen när jag har tid.  
Det som heter Rkermitt måste köras under en REMOTE programvara och det kan vi inte använda.

(Text 2656) Bo Kullmar \* <1789>  
Ärende: 1200/1200  
Ingen 802a kommer att finnas ute i lokalen och det är inte meningen att någon 802a skall tas i anspråk för lokala arbeten normalt. Måste det ändå ske, så skall det vara en på gruppnumret. För lokala körningar mm kommer den 806a som finns i lokalen att användas.  
Min \*\*ambition\*\* är att vi skall kunna byta ut modemen till modem som klara såväl 1200/1200, 1200/75 och 300/300 och i samband med detta köra allt på gruppnumret. Detta kan tänkas ske någon gång under 1986.

## Möte Lokalavd

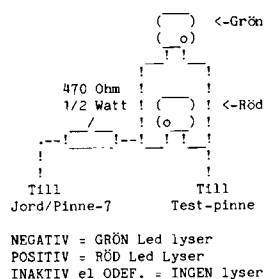
(Text 46) Linus Thrybom <1848>  
Ärende: Göteborgsmonitorn  
Va bra med en Göteborgsmonitor tänkte man när senaste bladet kom!  
Men icke!  
I luren hörs bara en evig upptaggetton. Vadan detta? Har det kraschat (eller är det bara så att alla de ca 25 gånger jag ringt dit har någon varit där före mig?) Någon som vet?  
Mvh Linus

(Text 47) Bo Kullmar \* <1789>  
Ärende: Göteborgsmonitorn  
Ring Anders Lundberg och fråga. Telefonnumret finns i en fil i TEXT/DIV. Den innehåller adresser och telefonnummre till alla lokalavdelningarna.  
Jag härde visst att det vara några problem, bl a med telefonledningarna.

(Text 67) Bo Kullmar \* <1789>  
Ärende: Lokalavd  
Nja, intresset för detta möte är inte så stort från Stlm-are ihellär så det lönar sig knappast att starta ett nytt möte. Stöd utgår från ABC-Klubben i form av pengar, 5 kronor per medlem inom regionen och år. Vi kan ordna med pengar till möteslokal och sådant. Dessutom har jag en del lappar från ett 20-tal personer i Skåne som vill vara med i en lokalavdelning i Skåne. Naturligtvis kan vi också köra ut etiketter från medlemsregistret på alla som har 2 som första siffra i postnummret och bekosta utskicket.

(Text 68) Bengt Holgersson <559>  
Ärende: Lokalavd  
Jag undrar även om det kanske kan utgå någon form av bidrag från tillverkaren av datorerna. Rabatter och dylikt.  
Om man skulle starta något så krävs det ju lite hårdvara även om man kanske inte behöver nå 40Mbyte monitor med en gång. Vidare kanske man behöver lite tips om hur allting bör organiseras. Mötet jag föreslog kanske kunde hjälpa till att sammanföra de som är intresserade så att något händer. M.v.h Bengt H

Bo Larsson <5630>



(Text 69) Gunnar Faith-Ell \* <2733>  
Ärende: Lokalavd  
Varför inte göra ett försök med ett utskick inom området?

(Text 70) Gunnar Faith-Ell \* <2733>  
Ärende: Lokalavd  
Jag tror nog att Luxor m.fl intressenter gärna vill se att en förening kommit igång innan de beslutar sig för att bidra med utrustning. ABC-Klubben har viss rabatt som kanske kan utnyttjas. För det första måste ni ha ett möte där en styrelse väljs, lämpligen skickas en kallelse till alla medlemmar inom regionen. Och då ser ni även hur stort gensvar ni får.

(Text 71) Bengt Holgersson <559>  
Ärende: Lokalavd  
Jag vet att Mats Knuts kom med en liten trevare för några år sedan men det rann väl ut i sanden.  
Det kanske kunde hjälpa om moderklubben organiserade någon form av enkät för att se hur många som är inresserade att jobba aktivt.

(Text 73) Gunnar Faith-Ell \* <2733>  
Ärende: Lokalavd  
Det bästa är väl om du fick fram en förlaga och sedan bad klubben om bidrag till porto och kopieringskostnader.

LOKALAVD.

## Möte MSG

(Text 9) Sven Wickberg <1384>  
Ärende: INMATNINGSMINNE  
Vad är det som gör att ABC och PB kan ta emot en lång rad tecken från tangentbordet i sitt minne, medan många andra system bara minns ett enda tecken. (Jag tänker på att man kan köra sin inloggningssekvens med datorn blador på med Vålkommen (2 gånger) och en massa annan spaltfylldnad.)

## Möte FORTH

(Text 63) Ulf Dahlen <2422>  
Ärende: UFORTH  
Jag har knäpat ihop en FORTH som jag fyndigt döpt till UFORTH (Ulf's FORTH är en vettig gissning). Den hanterar textfiler istället för screener och går således att köra även på ett kassettsystem. Den har ord som understödjer grafiken, random access till skiva, många strukturerad etc. Är någon intresserad, så ta kontakt med mig! Antagligen kommer jag att ge alltihop till ABC-klubben.

(Text 66) Magnus Bodin <4304>  
UFORTH: Intresse!  
Källkoden är av stort intresse, lägger du in den i FORTH-biblioteket?  
Magiska hälsningar Magnus :~)

(Text 67) Magnus Bodin <4304>  
UFORTH: Intresse!  
Ja, källkoden vore av intresse, liksom källkoden till FORTH2, men UFORTH kanske ersätter denna? Har du föresten bytt FORGET enligt FORGET.FIG, i kärnan alltså?  
Random-access? Vad lägger du för betydelse i det uttrycket? Kan man spara token med nya definitioner a la FORTH18?  
Ja, fler frågor finnes, och dessa skulle besvaras å det rappaste om du "la in" källkoden på FORTHbiblioteket.  
Förresten, har du en CROSScompiler? Håller på att skriva en till ABC-FORTHen, kan kanske vara av intresse till klubben också???

Har man en CROSScompiler, så kan man ju skriva program i FORTH som blir körbara utan att man behöver ladda in kärnan, skärmfiler etc..  
Magiska hälsningar Magnus

(Text 68) Ulf Dahlen <2422>  
UFORTH: Intresse!  
FORTH2 är ju bara den gamla vanliga FORTHen med några extraord. Jag skickade in den för ganska länge sedan och visste inte att den hade lagts in i programbanken. Extraorden är för grafik (SETDOT etc) och strukturerad (CASE-typ). Min dokumentation borde också finnas. Annars ska jag försöka lägga in den (tar ett tag, nu bor jag i Linköping och ABC:n finns i Båstad). UFORTHen är en alldeles egen skapelse gjord helt från scratch. Den gör allt som BASICen gör och lite där till.  
Med random access menade jag helt enkelt motsvarande CALL(28666) osv i BASICen. UFORTH hanterar textfiler precis som

BASICen. Du kan skriva ditt UFORTHprogram med en vanlig texteditor (TV i ex) och sedan ladda in den i UFORTHen eller skriva den direkt i UFORTHen. Ord för hantering av textfiler (OPEN, PREPARE etc) finns naturligtvis. UFORTHen klarar att göra lite grafikanimation om än inte så snabbt som SPRITE-paketet som kom på någon kassett. Om jag ska lägga in UFORTHen vill jag också lägga in hela dokumentationen och dessa filer är inte riktigt klara. En manual (25 sidor) finns dock.

(Text 76) Magnus Bodin <4304>  
Ärende: FORTH-83  
Sven Östberg gick tyvärr bort i Augusti, vilket innebär att en av våra mesta FORTH-programmerare i klubben även lämnade oss. Som alltid måste man ta nya tag då något sorgligt händer, och trots det låta ens vanliga liv gå sin gilla gång. FORTH-83 var ju ett av Svens stora projekt, att implementera, göra om skärmar (anpassning) till FIG-FORTH. Jag skulle gärna vilja återuppta Svens arbete där han slutade, då på en ABC-maskin, helst ABC806. Anpassning skulle då ske till '800-doset så att man inte skulle vara bunden till UFD-dos. Man skulle då även ha möjligheten att köra på nätverk typ CATNET el ("LYX-NET"). Är det någon som har tillgång till en FORTH-83, eller vet var Svens arbete tog vägen? Tacksam för svar här, inom kort.  
PS. FORTH är något som angår oss alla! Magiska hälsningar Magnus

(Text 84) Ulf Dahlen <2422>  
Ärende: FORTH med kassett? Jodå, "on it's way"!  
Mitt lilla skötebarn UFORTH går hur bra som helst att köra på ett kassettsystem eftersom begreppet "screener" inte finns (ställer annars till problem då det kräver direktåtkomst av enskilda block på "lagringsmediet"). Förhoppningsvis lagom till jul eller nyår ska jag ta mig samman och skicka in eländet så får vi se vad du tycker om den. Hälsningar Ulf

(Text 86) Gerry Eriksson <4842>  
Ärende: CASE definition  
Är det någon som vet var man kan få tag på CASE screenen som används i editorn som finns i FORTH-bibl. Den har varit publicerad i ABC-bladet, men saknar det nummret. Finns den på någon kasset eller så?  
Hälsningar Gerry

(Text 87) Magnus Bodin <4304>  
Ärende: CASE definition  
Nej, inte på någon kassett tyvärr.  
: CASE ?COMP CSP 'ICSP 4 ; IMMEDIATE  
: OF 4 ?PAIRS COMPIL OVER COMPIL =  
COMPIL OBRANCH HERE 0 ; COMPIL  
DROP 5 ; IMMEDIATE  
: ENDOP 5 ?PAIRS COMPIL BRANCH HERE 0 ;  
SWAP 2 ?ACMPIL THEN 4 ;  
IMMEDIATE  
: ENDCASE 4 ?PAIRS COMPIL DROP BEGIN  
SP 'CSP ' = 0 ; WHILE 2  
ACMPIL THEN REPEAT CSP ! ;  
IMMEDIATE

Denna rutin hittad i Discover FORTH

(Text 89) Magnus Bodin <4304>  
Ärende: CASE, igen...  
Oooppss, igen...  
Denna gången lite rejälare...  
(Det var en stund sedan jag använde ON:)  
Snackade helt i nattmössan för en kvart sen!  
Så här används ON som förresten heter ON: (med kolon).  
ON: <namn> <ord0> <ord1> <ord2> .. <ordx>;  
Då definieras ordet <namn>.  
När man sedan utför <namn> så här:  
<n> <namn>  
så utförs <ord0> om n=0, <ord1> om n=1 osv.  
Så ja, denna gången var det helt rätt, jag kollade just.  
Skillnaden mellan CASE och ON: blev nu ytterligare en: ON: används för att definiera ett ord som i sin tur är "vektorisat". CASE väljer inne i ett ord vad som skall ske. Föredrar CASE själv, faktiskt.  
Magiska hälsningar Magnus :-)

(Text 90) Bo Kullmar \* <1789>  
Fråga: MVPFORTH.DOC's ursprung  
Det är Robert Claesson, Bodin. Han sökte och fick projektbidrag för att utveckla en MVPFORTH till ABC80. Själva FORTHen var inte i textformat så jag lade inte in den. Sen har inget hänt med programvaran alls. Sven Östberg skulle låna en ABC80 och titta på den, men han gick som bekant bort och sedan dess är det ingen som har intresserat sig för den eller FORTH som sådant.

(Text 93) Magnus Bodin <4304>  
Ärende: multi-WHILE  
Ibland kan det vara praktiskt att använda flera WHILE inom samma 'BEGIN ... REPEAT'-slinga. Om nu inte Din FORTH klarar detta (Den FORTH som klubben distribuerat klarar det INTE) så kommer här en omdefiniering av REPEAT och WHILE. OBS! REPEAT måste omdefinieras före WHILE, eftersom REPEAT använder sig av WHILE vid omdefinieringen.

```
( MultiWHILE 3 Dec 85 Magnus Bodin )
: REPEAT ACMPILEA AGAIN DUP 4
  ?PAIRS BEGIN DUP 4 =
  WHILE 2 - ACMPILEA ENDIF
  REPEAT ; IMMEDIATE
: WHILE ACMPILEA IF 2+ 2SWAP ;
  IMMEDIATE
```

Så här använder man flera WHILE:

```
BEGIN
<chuvudfö1jd>
<flagga$1 lämnas på stacken>
WHILE
<fö1jd$1>
<flagga$2 lämnas på stacken>
WHILE
<fö1jd$2>
.
<flagga$3 lämnas på stacken>
WHILE
<fö1jd$3>
REPEAT
Först utförs <chuvudfö1d>.
```

Om flagga\$1 är sann, utförs <fö1jd\$1>, annars fortsätter exekvering efter REPEAT. Om flagga\$2 är sann, utförs <fö1jd\$2>, annars fortsätter exekvering efter REPEAT. Om flagga\$3 är sann, utförs <fö1jd\$3>, annars fortsätter exekvering efter REPEAT. Lycka till, FORTH är något som angår oss alla!  
Magiska hälsningar Magnus

(Text 98) Magnus Bodin <4304>  
Ärende: DO .. WHILE .. LOOP  
WHILE kunde behövas i en loop också, eftersom man ofta gör test i loopar, typ <flagga> IF LEAVE THEN etc.  
Med denna omdefiniering av LOOP, kan WHILE användas inom DO .. LOOP, och kommer då att låta "LOOP" motsvara "REPEAT", se här:

```
10 0 DO
  ." Tryck <CR>"
  KEY 13 = 0=
  WHILE
  ." ... fel!"
  LOOP
  ." Tio felslag, eller rätt."
  Ovanstående (om det kompileras i en kolon-definition) kommer att skriva ut texten "Tryck <CR>", testa en tangent, om det är ett <CR>, skrivs texten "Tio .." ut, annars skrivs "... fel!" ut. Denna loop genomlöps ånyo om sann flagg lämnats före WHILE, tills loopindexet=10 förstås.  
Här kommer omdefinieringen:  
( OBS! Omdefinieringen av REPEAT & WHILE i inlägg 93, krävs. Givetvis medför detta att man även kan använda flera WHILE inom samma DO .. LOOP, snurrigt? )
```

```
( DO .. WHILE .. LOOP 4 Dec 85 )
: (LOOP2) R> R> R> 2DROP DROP ;
: LOOP
  ACMPILEA LOOP
  BEGIN DUP 4 =
  WHILE 2 - ACMPILEA ENDIF
  REPEAT COMPIL (LOOP2) ;
  IMMEDIATE
```

Magiska hälsningar Magnus

(Text 107) Magnus Bodin <4304>  
Ärende: multi-WHILE  
Nu kan man ju snabbt och enkelt definiera två ord till, nu när loop är omdefinierad. IFLOOP & LOOPUNTIL  
IFLOOP används sålunda:  
10 0 DO  
 <fö1jd>  
 .  
 <flagga lämnas på stacken>  
 IFLOOP  
Om flaggan=true, och loopindex<10 så genomlöps loopen igen.  
LOOPUNTIL gör motsatsen:  
10 0 DO  
 <fö1jd>  
 .  
 <flagga lämnas på stacken>  
 LOOPUNTIL  
Om flaggan=false, och loopindex<10 så genomlöps loopen igen.  
Simpla ord, som alltså kräver att WHILE och LOOP omdefinierats, enligt tidigare inlägg:  
: IFLOOP ACMPILEA WHILE  
 ACMPILEA LOOP ; IMMEDIATE  
: LOOPUNTIL COMPIL 0=  
 ACMPILEA IFLOOP ;  
 IMMEDIATE  
(OBS. FIG-FORTH)  
Magiska hälsningar Magnus :-)

(Text 122) Magnus Bodin <4304>  
Ärende: FORTH vs ALGOL-språk  
Man kan se skillnaden i den här liknelsen: Om vi ser språken som restauranter så kan följande skillnader visas:  
När vi befinner oss i Pascal-restauranten, behandlas vi som en gäst. Det finns ett antal reserverade (bör) som vi ABSOLUT INTE får sätta oss vid eller ändra på. Vi får däremot lägga till nya bord, göra långbord, om vi skulle ha med oss ett stort sällskap (program). Menyn (kommandolistan) är bestämd av köksmästare och tillika restauranthefen, Niklaus Wirth. Det finns givetvis a la carte, där man kan välja egna rätter (procedurer), men dock inte hela menyer (kontrollstrukturer, typ WHILE, REPEAT, FOR, osv)...  
I FORTH-restauranten är vi själva restauranthefer. Bjuder vi in ett stort sällskap så kan vi givetvis anpassa de reserverade (borden) efter detta. Skulle vi vilja komponera någon rätt med nytt recept, så att förändringar måste ske under själva tillagningen (kompileringen) kan vi givetvis göra detta. Här har vi ingen köksmästare som styr oss. Charles Moore, mannen som grundade restauranter-kedjan lämnade bara en grund att stå på. Om sedan restauranten skall kunna servera fisk, skaldjur, sniglar (flyttal, strängar etc.) skall kunnas läggas till samt dras ifrån allt efter restauranthefens behov. Om restauranthefen tycker att han vill utvidga sin restauranter, så att den kan ta emot flera sällskap på en gång, även fast Moore inte tänkte på det i början, (multitasking) så kan han givetvis göra detta. Vad lär vi oss av detta?

Jo att FORTH-restauranterna borde vara mer utbredda än Pascal-dito. Varför åter inte en vanlig programmerare på FORTH-kedjans restauranger vanligtvis?  
\* FORTH-kedjans restauranger verkar ha tidigare entredörar, och lata programmerare stannar ofta i vestibulen eftersom entredörren trötter ut dem.  
\* Man måste tänka på ett annat sätt när man äter hos FORTH's restauranter, (postfix, omvänd polsk not. (RPN)) detta skrämmar flera programmerare. - Va? Skall jag behöva tänka annorlunda för att kunna äta?  
\* Inte tillräckligt många FORTH-gourmeer har funnits som kan visa upp vad FORTH

Så: egentligen har att erbjuda.  
Nästa gång du är hungrig, testa en FORTH-restauranter i stället för en Pascal-dito!  
En nöjd köksmästare

(Text 123) Magnus Bodin <4304>  
Ärende: Mikrodatorn 5/85 s.53  
Citerar här en artikel av Lars-Erik Svahn, för att ytterligare klargöra vad FORTH är för något och varför man bör ta sig en extra titt på språket.  
"Forth är komplett som system. Det ger full kontroll över datorn och dess periferenhet, samtidigt som det rymmer alla de möjligheter till strukturerad programmering som krävs av ett modern högnivå-språk."  
"FORTH är generellt. Det som gör FORTH så generellt, är dess många och lättanvända utbyggnadsmekanismer."  
"Man kan t o m ändra syntaxen i redan färdiga konstruktioner"  
"Som framgått är FORTH ett kompilerande språk, men också ett tolkande."  
"Även kompilationsprocessen är dialogbetonad och helt avdramatiserad."  
"... kompilerad Pascalkod är åtminstone dubbelt så långsam."  
"Har då medaljen ingen baksida?"  
"Sant är att FORTH kan vara motsträvt i början."  
"Efter en rimlig tids träning inser man att 'allt' kan göras i FORTH, och i regel på ett elegant sätt."  
"Datormiljö är det tvärtom postfix notation som är enkel och naturlig."  
"Dessutom är postfix notation smart, man behöver t ex aldrig använda några parenteser."  
"Den som vill lära sig FORTH skall absolut läsa de böcker som Leo Brodie på FORTH Inc. har skrivit:  
"Starting FORTH" och "Thinking FORTH."  
"Programspråket FORTH tillhör inte något av de enkla språken att lära sig, men det är resurssnålt och har den fördelen att när man väl kan det så går det mycket snabbt att programmera med FORTH."  
"FORTH är framtidens språk, så GO FORTH!"  
En nöjd köksmästare lagade ännu en måltid..

## ABC-datorer o dyl.

|                             |             |
|-----------------------------|-------------|
| 2 st beg. ABC 806 datorer   | 7.800:- /st |
| 3 st beg. ABC 800/M -"      | 4.500:- /st |
| 1 st beg. ABC 802 -"        | 5.500:- /st |
| 5 st beg. ABC 811 f.monitor | 1.000:- /st |
| 2 st beg. ABC 55 T.bord     | 1.600:- /st |
| 1 st beg. ABC 77 T.bord     | 2.600:- /st |
| 1 st NY ABC 830 m. kont.    | 5.200:- /st |
| 1 st beg. ABC 832 m. kont.  | 7.000:- /st |
| 1 st beg. BIDA 55/96 -"     | 7.000:- /st |

|                       |         |
|-----------------------|---------|
| 128K Ram kort         | 1.500:- |
| 80 Tkn kort till 800  | 1.000:- |
| HR kort till d:o      | 1.200:- |
| Floppykort gamla mod. | 1.300:- |
| Floppykort Nya mod.   | 2.200:- |
| CAT/Net kort (1st)    | 1.000:- |
| Winchester ST 506 5Mb | 1.200:- |
| Winchester ST 406 5Mb | 1.500:- |

## BILD & DATA

Alla priser exkl. moms.

Oskarsgatan 1  
803 23 GÄVLE  
Tel 026-14 24 38

# Möte CP/M

(Text 274) Sven Östberg <5246>  
Ärende: Att programmera portarna i cp/m  
Hur gör man detta på ett hårdvaruoberoende sätt. Det gäller alltså att programmera tty-porten med paritet och stoppbitar. Såvitt jag kan se, kan man bara ställa in den för baudrate och 'nulls outputted after cr'.

(Lo.f.s. kan jag göra inställningarna direkt i Forth, men då måste jag veta vilken port som är tty. Kanske det går att hämta den informationen från cpm-systemet? Genom att avläsa någon byte någonstans?)Ihe

(Text 275) Gunnar Faith-Ell <2733>  
Ärende: Att programmera portarna i cp/m  
Jag tror knappast att det går att göra på ett hårdvaruoberoende sätt! Adresserna till portarna och övrig utformning av hårdvaran kring portarna skiljer sig ganska kraftigt mellan olika datorer.

(Text 276) Sven Östberg <5246>  
Ärende: Att programmera portarna i cp/m  
Men hur fungerar då tex. Xmodem?  
Eller måste man då programmera porten med ett särskilt program innan Xmodem startas upp?

Samma sak med PIP, förresten. Men utan att tänka på det särskilt har jag alltid initierat modemporten först genom att starta upp Kermit eller något annat modemprogram. Onödigt?

(Text 277) Bo Larsson <5630>  
Ärende: Att programmera portarna i cp/m  
Att det inte finns något generellt sätt att programmera cpm's dataportar är ju ganska givet men att använda dem på ett generellt sätt i program fungerar ju.  
I Xmodem får man i källkoden manuellt ange vad sina portar har för nummer och även skriva rutiner som initierar sina modemportar. De enda portar man inte behöver ange är dom som har med konsolen att göra, de initieras automatiskt enligt rutiner som jag senare ger exempel på.  
Bra exempel på program som använder alla portarna (även modem) på ett generellt sätt är just PIP och den generella varianten på Kermit.

I cp/m's Bios (system-beroende) måste man ange vad portarna till sin dator har för nummer och hur dom maskas. För att man sedan skall kunna använda den informationen måste det finnas ett pek till dom rutiner som använder dessa data.  
För att klara detta startar Bios'en en hopp-tabell till dessa rutiner, rutinerna i den tabellen måste komma i en speciell ordning som är standard i cp/m. Saknar man en rutin i sitt system måste det i alla fall finnas ett virtuellt hopp till denna rutin som endast innehåller ett RET.  
De rutiner som är intressanta här är rutinerna 2 -> 7 i tabellen:

```
BIOS:  jp wboot      ; varm-boot
        ; (Bios+ 0)
->      jp const      ; console status
        ; (Bios+ 3)
->      jp conin       ; console input
        ; (Bios+ 6)
->      jp conout      ; console output
        ; (Bios+ 9)
->      jp list        ; list output
        ; (Bios+12)
->      jp punch       ; punch dev.
        ; (Bios+15)
->      jp reader      ; reader dev.
        ; (Bios+18)
```

Bios'en ligger på olika ställen beroende på minnesstorlek och dator men det är inget man behöver tänka på, peket till själva Bios'en ligger alltid på adress 0001h.  
Adress 0000h innehåller en hoppinstruktion, så ett hopp till adress 0000h gör alltså ett hopp till Bios'en som direkt går till en varmboot-rutin.  
För att initiera egna rutiner som använder ovanstående kan man göra följande:

```
INITADRS: ld hl,(0001)
           ; hitta bios-adressen
           de,3
           ; öka med 3
           add hl,de
           ; (skippa varmboot)
           ld (STAT+1),hl
           ; spara const-adr
           add hl,de
           ld (CONI+1),hl
           ; spara conin-adr
           add hl,de
           ld (CONO+1),hl
           ; spara conout-adr
           add hl,de
           ld (LST+1),hl
           ; spara list-adr
```

```
add hl,de
ld (PUN+1),hl
; spara punch-adr
add hl,de
ld (RDR+1),hl
; spara reader-adr
ret
; Kontroll av console-status anropar denna
; rutin och återvänder ; med A=0ffh om
; console ready, A=00h om not ready.
```

```
VSTAT: push BC ! push DE
        ! push HL
STAT:   jp $-5
        pop HL ! pop DE ! pop BC
ret
; Denna rutin gör console-input och åter-
; vänder med tecken i A.
```

```
VCONI: push BC ! push DE
        ! push HL
CONI:   jp $-5
        pop HL ! pop DE ! pop BC
ret
; Denna rutin gör console-output, ingång
; är output-tecken i A.
```

```
VCONO: push AF ! push BC ! push DE
        ! push HL ld c,a
VLST:   jp $-5
        pop HL ! pop DE ! pop BC
        ! pop AF ret
; Denna rutin gör lst-output, ingång är
; output-tecken i A.
```

```
VLST:   jp $-5
        pop HL ! pop DE ! pop BC
        ! pop AF ret
; Denna rutin gör pun-output, ingång är
; output-tecken i A.
```

```
VPUN:   push AF ! push BC ! push DE
        ! push HL ld c,a
PUN:    jp $-5
        pop HL ! pop DE ! pop BC !
        pop AF
ret
; Denna rutin gör rdr-input och återvänder
; med tecken i A.
```

```
VRDR:   push BC ! push DE ! push HL
RDR:    jp $-5
        pop HL ! pop DE ! pop BC
        ret
mhv Bosse Larsson
```

(Text 278) Sven Östberg <5246>  
Ärende: Att programmera portarna i cp/m  
Tack för svaret. Jo lite har jag tittat på det där med bios-anrop. Anledningen till att jag ställde frågan om porten är just att jag skulle skriva ett modemprogram som kan spridas i HEX-form, d.v.s. det ska kunna användas utan en massa programmering av slutanvändaren.  
Jag är inte riktigt nöjd med svaren jag fått. I åtminstone mitt cp/m-system finns ett program som heter CONFIGUR.COM där man ändra bland inställningarna mellan logiska-fysiska periferenheter. Det programmet borde kunna göras hur hårdvaruoberoende som helst, och om hårdvaran tillåter att serieinterfacet kan programmeras, varför då inte ge möjlighet att göra detta? (Eller börjar cp/m bli alltför gammalt?)  
Dom modemprogram som jag har (Kermit... e.t.c.) är alla specialhack för just mitt datormärke, så uppstarten är hårdvaruoberoende.

Om jag förstätt saken rätt så kan man kommunicera med modemprogram genom att skriva till PUN: och läsa från RDR: och att det lätt låter sig göras genom BIOS-anrop. Men det hjälper ju föga om sen inte kontakten med hårdvaran fungerar.??

(Text 285) Bosse Gidmark <455>  
Ärende: Spridning?  
Jag har en massa källkoder som kom med ett CP/M till ABC80. Äfter som man inte kan använda dem utan tillsatsskort undrar jag om man kan sprida dem. Ett smakprov följer.

```
;
; SYSTEM EQUATES
;
; Version 3.3 BIOS for ABC80 - UNI80
;
; This file is assembled together with the
; three sourcefiles
; BOOTXXX.ASM,BIOSXXX.ASM and BIOSXXB
; .ASM. The equates in this
; file selects different features by conditional
; assembly.
;
; msize equ 64
; cp/m version memory size
; in kilobytes
biosver equ 33
; BIOS version number
```

```
;
abc80 equ 1
; BIOS for ABC80
abc800 equ 0
; BIOS for ABC800
;
; device addresses
;
keybrd equ 16
; keyboard i/o-port
v24 equ 38
; i/o-port
sound equ 6
; soundgenerator
vram40 equ 07c00h
; standard videoram address
vram80 equ 05800h
; TKN80 videoram address
stdram equ 0c000h
; start of ram in standard
ABC80 equ 0efh
; mapping i/o-port
; constants
;
delay1 equ 10000
; delay for repeat for 1:st
char
delay2 equ 500
; delay for repeat for fol-
; lowing char.
bellen equ 30000
; length of bel
idelay equ 23
; gives 2400 baud
usrmap equ 1
; normal cp/m-mode
sysmap equ 2
; system mode
maxram equ 31
;
;
; Z80-instructions
;
ldir equ 0b0edh
ldrr equ 0b8edh
dlnz equ 10h
dlnz equ 0a3edh
otir equ 0b3edh
inir equ 0a2edh
jrc equ 030h
jrnz equ 038h
jrz equ 020h
jr equ 028h
jrl equ 018h
slae equ 023cbh
; sla e
; rrrc equ 01bbeh
;
; sequates for databoard-bus
;
data equ 0
; data in and out
sel equ 1
; scard select
stat equ 1
; status from card
c1 equ 2
; command 1
c2 equ 3
c3 equ 4
c4 equ 5
;
; control characters for screen
rs equ 1eh
; home
clear equ 1ah
; clear screen
esc equ 1bh
; escape
bs equ 08h
; backspace
vt equ 0bh
; cursor up
ff equ 0ch
; forward space
lf equ 0ah
; linefeed
rtn equ 0dh
; return
;
;
;
; (msize-20)*1024
ccp equ 3400h+bias
bdos equ ccp+806h
; base of bdos
bios equ ccp+1600h
; base of bios
cdisk equ 4
; current disk
iobyt equ 3
; device assignment
;
; Om jag brytit mot något ta bort inlägget
```

(Text 293) Bo Michaelsson <913>  
Ärende: CP/M 3.0 och ABCDISK samt CPMCOPY  
Programmet ABCDISK som kopierar från ABC-format till CP/M 3.0 finns nu hos MYAB. Även programmet ABCDISK som kopierar i den andra riktningen finns. Dessa program skulle enligt löfte vid försäljningen av CP/M 3.0 ingå men nu tar MYAB betalt för dem. Det kostar 150 kronor för två skivor 8 tum.

(Text 295) Gunnar Faith-Ell <2733>  
Ärende: CP/M 3.0 och ABCDISK samt CPMCOPY  
Skulle de inte klara andra versioner av CP/M oxo ? Varför är det två diskar (?) räcker det inte med en ?

(Text 299) Bo Michaelsson <913>  
Ärende: CP/M 3.0 och ABCDISK samt CPMCOPY  
Jag vet inte om de går på andra versioner. Att programmen ligger på två skivor beror på att det ena, ABCDISK.COM, arbetar under CP/M, det andra, CPMCOPY, arbetar under ABCDOS.

(Text 300) Kjell Brealt <283>  
Ärende: CP/M 3.0 och ABCDISK samt CPMCOPY  
150 Kr kallar MYAB för mediakostnad. Jag har blivit lovad programen gratis om jag kommer dit med ett par skivor.

(Text 301) Bosse Gidmark <455>  
Ärende: pipa?  
Jag som är ganska ny när det gäller CP/M undrar hur man hanterar filer under CP/M mellan 2 datorer. T ex en 802 och någon anna burk. Kan man hantera RTS eller blir det XON/XOFF eller vad.  
Problemet är att en god vän har en massa filer i sin burk som jag vill ha över till ABC802 CP/M (DIAB). Hur gör jag??

(Text 303) Gunnar Faith-Ell <2733>  
Ärende: pipa?  
Ska du köra via modem eller ?  
I så fall bör båda använda tex MODEM7 eller något liknande program.

(Text 309) Bo Michaelsson <913>  
Ärende: 2.2 -> 3.0  
Luxor har slutat att leverera CP/M till sina datorer har man sagt mig. MYAB verkar försöka anpassa sina program så de går att köra på alla i 800-serien oberoende av korten. Vad jag har förstätt - men jag har inte provat - så ska de fungera på ABC 80 också.

(Text 318) Bosse Gidmark <455>  
Ärende: CP/M 3.0  
Jag fattar inte riktigt. Varför kan inte MYAB lägga in de olika drivrarna. CP/M är ju tänkt att användas till flera diskar samtidigt. Är det någon som förstår hur det fungerar att den inte har mer än 64k i ABC802. Det måste ju vara en av vitsen med att gå över till 3.0. Det enda jag vill vara att få tag på programmet COPYDISK till CP/M. För att kunna använda detta prg måste jag byta till 3.0. Vilken jag inte behöver. Bedrövligt.  
Är det någon som har testat JET-80 den finns i v 3.0 och klarar av 6 diskar samtidigt, 3 olika typer, + en Hard disk m.m. Vad är karen CP/M eller vad det heter. Tydligt finns det ett prg som gör att man under CP/M-86 kan köra CP/M 2.2. Finns det ett till CP/M 3.0?

(Text 323) Bosse Gidmark <455>  
Ärende: CP/M 3.0  
Vid frågan till MYAB så visade sig att det finns en 3.0 som går att flytta, det är bara byta till den om man behöver någon. Den är lite bakom A: står för Winc. M: för ramdisk o s v

(Text 324) Per Holmgren <5213>  
Ärende: Concurrent CP/M 86 och Concurrent PC/DOS  
Förvirringen är total. Vad jag vet finns Digital Research's Concurrent CP/M 86 för maskiner med iapx86/88/186. Den lär skola vara kompatibel med Microsoft's MSDOS, och tillåta viss form av multitasking. Flera fönster med springade job, foreground/background etc...  
Sedan har det i några sammanhang talats om Concurrent PC/DOS, som alltså inte skulle vara helt synonymt med DR's produkt. C PC/DOS skulle alltså vara framtagna av Microsoft eller IBM själva. C PC/DOS skulle dessutom lösa samtliga problem med PC:n, alltifrån type-ahead buffering till multitasking, och dessutom kunde den nog koka kaffe själv (med 16 sockerbits upplösning !!)  
Helt klart är att ingen av dessa går att köra på en 8-bit karta utan en co-processor. Tips: Glöm Concurrent CP/M !!  
/Papa

# En billig talsyntes

Jag tänker berätta lite hur man i ABC80 kan digitalisera ljud med en mycket enkel och billig hårdvara. Återgivning av ljudet kan sedan ske helt utan extra hårdvara med denna metod.

## Bakgrund

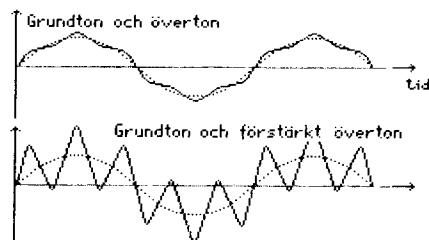
I begynnelsen upptäcktes att om man spelar en vanlig ljudkassett i ABC80s bandspelare så hördes ljudet på bandet, visserligen med stark distortion, men hörbart. Då det är möjligt att från programvara känna av kassettporten samt styra högtalaren så infann sig tanken att lagra ljudet digitalt och efteråt spela upp det. Försök visade att det inte gick så bra. På kassettporten kan programmet bara känna om signalen skiftar från plus till minus eller tvärtom (flankdetektering). Polariteten, om signalen ligger på plus eller minus, kan inte kännas av. V24 porten däremot kan känna polariteten.

I början av projektet tänkte jag att det skulle krävas lite hårdvara både för att digitalisera ljudet och för att bilda en analog signal när ljudet återges, men när jag läst en artikel i BYTE (februari 1981 sid 36-43) så tänkte jag att det borde vara möjligt att återge ljudet i ABC80s inbyggda högtalare. Det visade sig vara riktigt, hårdvara behövs bara vid 'inspelningen'. Att återge ljudet utan extra hårdvara ger möjlighet att distribuera ljudet till alla som har en ABC80. (t ex RULLE.BAS på kassett §17).

## Digitalisera ljudet

Ett ljud (t ex ett vokal-ljud eller ett instrument) kan beskrivas som ett antal överlagrade sinusvågor (grundton och övertoner). För att tal ska kunna återges så måste både grundtoner och övertoner i ljudet återskapas, annars blir inte talet förståeligt.

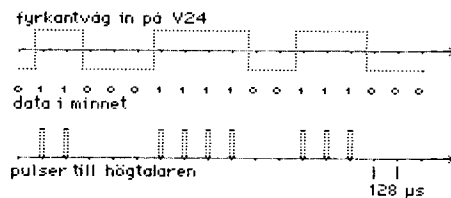
För att digitalisera ljudet så samplar man (känner av) ljudet, i det här fallet, 7812 gånger per sekund. Vid varje sampel lagras bara en bit information; 1 om signalen är positiv och 0 om den är negativ. Den här metoden gör i praktiken fyrkantvåg av ljudet. Grundtonen i ljudet ger den huvudsakliga formen på fyrkantvågen och information om övertonerna försvinner. Detta åtgärdas genom att förstärka övertonerna mer än grundtonen och låta övertonerna göra att signalen växlar polaritet flera gånger under en period. Ett filter får derivata (eller differentiera) den ursprungliga signalen.



## Återge ljudet

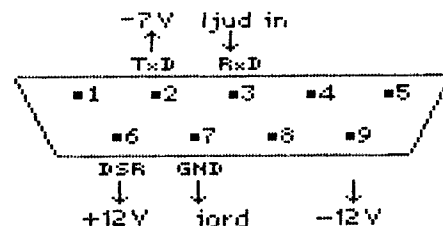
När ljudet ska återges är det viktigt att bitarna som representerar ljudet matas ut i precis samma takt som de samlats, även nu 7812 gånger/s. Om man låter en etta ge en fast positiv spänning och noll en negativ så borde utsignalen, som nu liknar fyrkantvåg, integreras i ett filter innan den går till högtalaren (för att kompensera deriveringen vid digitaliseringen). Detta är dock inte nödvändigt för att förstå talet och vi kommer inte heller göra det. På grund av egenheter i ABC80s ljudkrets så kan man inte mata ut den fyrkantvåg som bitarna representerar utan ljudet måste huggas upp i korta pulser, egentligen en puls (eller spik) per bit som är ettställd. I nyare ABC80 med högtalaren i separat nät del blir ljudet ganska svagt och dåligt, ev. kan det förbättras med utmatning av längre pulser.

Anledningen till att jag valt samplingsfrekvensen 7812 är att PIO:n kan programmeras att ge interrupt så ofta, det ger en tillräckligt exakt tidbas.



## Hårdvara

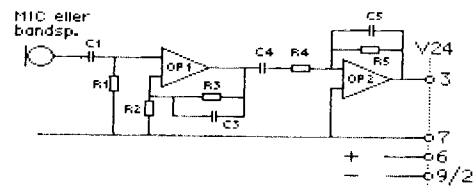
Filtret som används vid 'inspelning' består av två Op-förstärkare och några motstånd och kondensatorer. Op-förstärkarna tar matningsspänning från V24 kontakten pinne 6 och 9 (eller 8) och den förstärkta och filterade signalen går in på V24 kontakten pinne 3. OBS! För att matningsspänningen ska bli rätt måste man först göra OUT 58, INP(58) OR 8 (sätta pinne 2 låg=-7V) innan man kopplar in förstärkaren/filtret. V24an ger med denna koppling -7V och +12V, inte direkt balanserat men det gör nog inget. Alternativt kan minusspänning tas från pinne 9 (-12V) vilket borde minst lika bra (men jag har ingen V24 sladd med pinne 9 ansluten).



Förutom att derivata signalen så måste filtret ta bort frekvenskomponenter över ca 3500 Hz annars kommer höga frekvenser att återges felaktigt. De skulle samplas mindre än 2 gånger per period och data kan misstolkas som en annan, lägre, frekvens.

Kondensatorn före första steget fungerar både som isolator för likström och som filter för lågfrekvent brum.

Förstärkningen i filtret kan få vara så stor att förstärkarna bottenar. Efter förstärkaren/filtret är vi bara intresserade av om signalen är positiv eller negativ, inte amplituden på signalen. ABC80 tillåter att signaler till V24an är +/- 30V (volt). En signal som är över 3V registreras som en nolla (0) och signaler under 0V registreras som etta (1). Mellan 0V och 3V är hysteres, dvs. en nolla eller etta ligger kvar tills signalen går över endera gränsen. V24an blir alltså en nollgenomgångsdetektor och hysteresen gör att bakgrundsbrus inte behövs höras, små variationer kring 0V ger inget utslag.



- |             |           |
|-------------|-----------|
| R1=150 kohm |           |
| R2=1.2 kohm |           |
| R3=56 kohm  | C3=1.0 nF |
| R4=1.8 kohm | C4=47 nF  |
| R5=39 kohm  | C5=1.0 nF |

# LJUD2.TXT

```

;-----
;86-02-11
;DAVID ANDERSSON <5201>
;ASSEMBLER RUTINER FÖR ATT SPELA IN
;OCH ÅTERGE LJUD.
;-----
PRGADR: EQU 61440 ;PROGRAMMET PLACERAS UNDER STACKEN -
;DET MÅSTE VARA PÅ EN JÄMN PAGE-ADRESS
HI: EQU .HIGH.PRADR ;HÖGA BYTEN AV ADRESSEN
SADDR: EQU 51200 ;STARTADRESS FÖR LJUD-DATA (16K RAM)
EADDR: EQU PRGADR-1 ;SISTA ADRESS FÖR LJUD-DATA
;-----
; PROGRAMMET BÖRJAR HÄR
ORG PRGADR
JP START ;CALL(PRADR) SPELA IN LJUD
JP PLAY ;CALL(PRADR+3) SPELA UP LJUD
; VARIABLER
BYTE: DEFB 0 ;SHIFTA IN/UT EN BYTE
COUNT: DEFB 8 ;RÄKNAR 8 BITAR I EN BYTE
ADDR: DEFW 0 ;ADRESS TILL LJUD-DATA
; INTERRUPT VEKTORER
ORG PRGADR+52 ;NORMAL I-VEKTOR I PIO:N
SINT: DEFW SINT1 ;STROBE INTERRUPT (SÄTTS VID INITIERING)
DEFW 1428 ;CASSETT INTERRUPT
;-----
; RUTINER SOM ANROPAS FRÅN BASIC
; SPELA IN LJUD
; ANROPAS FRÅN BASIC, AVBRYTS MED <RETURN>
; <MELLANSLAG> ÅTERSTARTAR INSPELNING
START: LD A,8
LD (COUNT),A ;INITIERA VARIABEL
LD HL,EADDR
LD (ADDR),HL ;SÄTT SLUTADRESS FÖR INSPELNING -
;INSPELNING SKER DÅ INTE FÖRENS -
; <SPACE> TRYCKS NER
LD HL,SINT1 ;INTERUPT FÖR INSPELNING
LD (SINT),HL ;SÄTT INTERUPTRUTIN 1
CALL INITI ;INITIERA PIO-INTERUPT
RECORD: IN A,(56) ;TESTA TANGENTBORDET
CP 13+128 ;<RETURN>
JR Z,EXITI ;TERMINERA INTERRUPT OM RETURN
CP 32+128 ;<SPACE>
JR NZ,RECORD ;SPELA IN LJUD OM EJ SPACE, LOOPA
LD HL,SADDR ;SPACE ÄR INTRYCKT -
LD (ADDR),HL ;SÄTT STARTADRESS FÖR DATA
JR RECORD ;LOOPA
; SPELA UPP LJUD
; ANROPAS FRÅN BASIC, AVBRYTS MED <RETURN>
PLAY: LD A,1
LD (COUNT),A ;INITIERA VARIABEL
LD HL,SADDR
LD (ADDR),HL ;SÄTT STARTADRESS FÖR DATA
LD HL,SINT2 ;INTERUPT FÖR ÅTEGIVNING
LD (SINT),HL ;SÄTT INTERUPTRUTIN 2
CALL INITI ;INITIERA PIO-INTERUPT
PLAY1: LD HL,(ADDR) ;TESTA OM EFTER SLUTADRESS
LD A,H
CP HI
JP NC,EXITI ;OM SÅ, TERMINERA INTERRUPT
IN A,(56) ;TESTA TANGENTBORDET
CP 13+128 ;<RETURN>
JR NZ,PLAY1 ;LOOPA OM INTE RETURN
JP EXITI ;TERMINERA INTERRUPT

```

```

;-----
; INITIERINGS RUTINER
; INITIERA STROB INTERRUPT
INITI: LD A,I ;TESTA OM I-REGISTRET REDAN ÄR SPARAD
CP HI
JR Z,INITI3
LD (SAVEI),A ;OM INTE, SPARA SIST I LISTAN
INITI3: LD HL,ILIST ;PIO DATA FÖR STROBE INTERRUPT
JR SETPIO
; TERMINERA STROB INTERRUPT
EXITI: LD HL,ULIST ;PIO DATA FÖR NORMAL INTERRUPT
JR SETPIO
; PROGRAMERA PIO OCH ENABLE INTERRUPT
; HL SKA PEKA PÅ LISTA ,5 BYTES LÅNG
SETPIO: PUSH BC
LD BC,4*256+57;B=4 BYTES, C=PORT 57
DI
OTIR
LD A,(HL) ;A=SISTA BYTEN I LISTAN -
LD I,A ;TILL I-REGISTRET
EI
POP BC
RET
; DATA FÖR PROGRAMERING AV PIO PORT A (KEYBOARD)
ILIST: DEFB 04FH ;MODE 1 (ALL INPUTS)
DEFB 087H ;ENABLE INTERRUPT
DEFB 087H ; -"
DEFB 087H ; -"
DEFB HI ;HIGH VEKTOR
ULIST: DEFB 0CFH ;MODE 3 (CONTROL)
DEFB 0FFH ;DIRECTION ALL INPUT
DEFB 0B7H ;ENABLE,OR BITS,GO_HIGH, MASK_FOLLOWS
DEFB 07FH ;BIT 7 IS MASKED
SAVEI: DEFB 0 ;OLD HIGH VEKTOR (SÄTTS VID INITIERING)
;-----
; INTERRUPT RUTINER
; STROB INTERRUPT, 7812 PERIODER/S
; INTERRUPT RUTIN FÖR INSPELNING AV LJUD
SINT1: PUSH AF
PUSH HL
IN A,(58) ;KÄNN AV V24-PORTEN
AND 1 ;TESTA OM LJUDSIGNALEN, CLEAR CARRY
JR NZ,NOOUT ;HOPPA OM 1 (AKTIV LAG)
XOR A ;GE PULS I HÖGTALAREN
OUT (6),A
LD A,121
OUT (6),A
SCF ;SÄTT CARRY CARRY ÄR SAMPLAD BIT
NOOUT: LD HL,BYTE
RL (HL) ;CARRY TILL BIT 0 AV (BYTE)
LD A,(HL) ;A=BYTE
INC HL ;HL=ADRESS TILL COUNT
DEC (HL) ;RÄKNA NER COUNT
JR NZ,NONEX2 ;HOPPA OM COUNT>0
LD (HL),8 ;COUNT=8, FÖR NÄSTA BYTE
LD HL,(ADDR) ;ADRESS TILL LJUD-DATA
LD (HL),A ;SPARA BYTE I LJUD-DATA
JP NXTADR ;SÄTT NY ADRESS, RETURN
;FROM INTERRUPT
; INTERRUPT RUTIN FÖR ÅTERGIVNING AV LJUD
SINT2: PUSH AF
PUSH HL
LD HL,BYTE
RL (HL) ;BIT 7 AV (BYTE) TILL CARRY
JR NC,NOOUT2 ;HOPPA OM BIT=0
XOR A ;BIT=1, GE PULS I HÖGTALAREN
OUT (6),A
LD A,121
OUT (6),A
NOOUT2: INC HL ;HL=ADRESS TILL COUNT
DEC (HL) ;RÄKNA NER COUNT
JR NZ,NONEX2 ;HOPPA OM COUNT>0
LD (HL),8 ;COUNT=8 FÖR NÄSTA BYTE
LD HL,(ADDR) ;ADRESS TILL LJUD-DATA
LD A,(HL) ;A=LJUD-DATA BYTE
LD (BYTE),A ;SPARA BYTE FÖR ATT SHIFTA UT
NXTADR: INC HL ;NÄSTA BYTE
LD A,H ;TESTA OM EFTER SISTA ADRESS
CP HI
JR NC,NONEX2 ;JA, SPARA INTE NY ADRESS
LD (ADDR),HL ;SPARA NÄSTA ADRESS
NONEX2: POP HL ;RETURN FROM INTERRUPT
POP AF
EI
RETI
;-----
END

```

Vad Op-förstärkarna heter vet jag inte men användningen är inte krävande och det går nog lika bra med selleri (alltså vanliga 741:or). Uppkopplingen har jag gjort på några IC-hållare och virat trådar mellan stiften.

Ungefärlig totala förstärkningen ges av R3/R2\*R5/R4. Derivering sker i C4 och R4 (egentligen ett högpasfilter för 1900 Hz). C3,R3 samt C5,R5 står för lågpasfiltren (2800 resp. 4000 Hz). Experimentera gärna med andra värden på komponenterna för att testa om ljudet kan bli bättre.

#### Programmet

När en bit ska lagras känner programmet av signalen på port 58 bit 0. Då 7812 bitar för varje sekunden ljud ska sparas så måste minnet utnyttjas effektivt genom att lagra 8 bitar i varje byte. Ändå så kommer en sekund att ta cirka 1 Kbyte i minnet. Programmen utnyttjar 10K minne som räcker till 10 sekunder ljud.

Programmet fungerar så att PIO:n initieras att ge interrupt med 128 us intervall (7812 per sekund) och en av två interruptrutiner aktiveras. Interruptrutinen för inspelning, SINT1, sparar en bit i minnet och ekar ljudet i högtalaren. Interruptrutinen för återgivning, SINT2, hämtar bitar från minnet och skickar bitar till högtalaren.

Med BASIC programmet som listas sist behöver man inte skriva ;CALL(adrs) för hand medans man testat och spelat in olika ljud. Bakåtpil börjar digitalisera ljud och mellanslag startar inspelning av ljudet. Framåtpil spelar upp ljudet, och RETURN avbryter allting. Med knappen P visas minnet bit för bit och med R får man tillfälle att räkna ut gränsfrekvenser för RC-filtre.

```

1 POKE 61440%,195%,56%,240%,195%,94%,24
  0%
2 POKE 61492%,172%,240%,148%,5%,62%,8%,
  50%,7%,240%,33%,255%,239%,34%,8%,240%
  ,33%
3 POKE 61508%,172%,240%,34%,52%,240%,20
  5%,132%,240%,219%,56%,254%,141%,40%,6
  4%,254%,160%
4 POKE 61524%,32%,246%,33%,0%,200%,34%,
  8%,240%,24%,238%,62%,1%,50%,7%,240%,3
  3%
5 POKE 61540%,0%,200%,34%,8%,240%,33%,2
  07%,240%,34%,52%,240%,205%,132%,240%,
  42%,8%
6 POKE 61556%,240%,124%,254%,240%,210%,
  146%,240%,219%,56%,254%,141%,32%,241%
  ,195%,146%,240%
7 POKE 61572%,237%,87%,254%,240%,40%,3%
  ,50%,171%,240%,33%,162%,240%,24%,3%,3
  3%,167%
8 POKE 61588%,240%,197%,1%,57%,4%,243%,
  237%,179%,126%,237%,71%,251%,193%,201
  %,79%,135%
9 POKE 61604%,135%,135%,240%,207%,255%,
  183%,127%,0%,245%,229%,219%,58%,230%,
  1%,32%,8%
10 POKE 61620%,175%,211%,6%,62%,121%,211
  %,6%,55%,33%,6%,240%,203%,22%,126%,35
  %,53%
11 POKE 61636%,32%,47%,54%,8%,42%,8%,240
  %,119%,195%,236%,240%,245%,229%,33%,6
  %,240%
12 POKE 61652%,203%,22%,48%,7%,175%,211%
  ,6%,62%,121%,211%,6%,35%,53%,32%,18%,
  54%
13 POKE 61668%,8%,42%,8%,240%,126%,50%,6
  %,240%,35%,124%,254%,240%,48%,3%,34%,
  8%
14 POKE 61684%,240%,225%,241%,251%,237%,
  77%

```

```

20 REM -----
30 REM 86-02-24
40 REM DAVID ANDERSSON <5201>
50 REM
60 REM PROGRAMMET ANROPAR MASKINKODS-
70 REM RUTINER FÖR ATT SPELA IN OCH
80 REM ÅTERGE LJUD.
90 REM -----

```

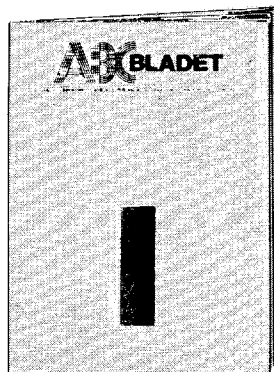
```

100 REM
110 A1%=51200% : REM ADRESS FÖR LJUD-DA
  TA
120 OUT 58%,INP(58%) OR 8% : REM PINNE
  2=-7V
130 ;
140 ; 'Spela in med BAKÅTPIL + MELLANSL
  AG'
150 ; 'Återge med FRAMÅTPIL'
160 ; 'Titta på lagrade bitar med P'
170 ; 'Beräkna filterfrekvenser med R'
180 ; 'Avbryt funktionerna med RETURN'
190 REM MAIN LOOP
200 GET A$
210 IF ASC(A$)=8% GOSUB 1000
220 IF ASC(A$)=9% GOSUB 2000
230 IF A$='P' OR A$='p' GOSUB 3000
240 IF A$='R' OR A$='r' GOSUB 4000
250 GOTO 190
1000 REM SPELA IN
1010 ; '<-- In';
1020 Z%=CALL(61440%)
1030 ; CHR$(13%)SPACE$(10%)CHR$(13%);
1040 RETURN
2000 REM ÅTERGE
2010 ; '--> Ut';
2020 Z%=CALL(61443%)
2030 ; CHR$(13%)SPACE$(10%)CHR$(13%);
2040 RETURN
3000 REM BITPEEK
3020 FOR A%=A1% TO 61439%
3030 FOR I%=7% TO 0% STEP -1%
3040 IF PEEK(A%) AND 2%*I% ; '1'; ELSE ;
  '0';
3050 NEXT I%
3060 IF INP(56%)<128% GET A$
3070 IF A$<>CHR$(13%) NEXT A%
3080 ;
3090 RETURN
4000 REM BERÄKNA RCNÄT
4010 ONERRORGOTO 4080
4020 ; 'C (nF) =' ; : INPUT C
4030 ; 'R (kohm) =' ; : INPUT R
4040 C=C*1E-9
4050 R=R*1000
4060 ; 'f (Hz) =' ; 1/R/C/2/PI : ;
4070 GOTO 4010
4080 RETURN

```

## UTGIVNINGSPLAN 1986

Nr 1, 1986  
Manusstopp : 3 februari  
Annonsbokning : 7 februari  
Materialdag : 17 februari  
Till tryck : 3 mars  
Medlemmarna : 1 april



ABC BLADET

2

Nr 2, 1986  
Manusstopp : 21 april  
Annonsbokning : 28 april  
Materialdag : 5 maj  
Till tryck : 26 maj  
Medlemmarna : 18 juni

Nr 3, 1986  
Manusstopp : 11 augusti  
Annonsbokning : 18 augusti  
Materialdag : 25 augusti  
Till tryck : 15 september  
Medlemmarna : 8 oktober

ABC BLADET

3

ABC BLADET

4

Nr 4, 1986  
Manusstopp : 20 oktober  
Annonsbokning : 27 oktober  
Materialdag : 3 november  
Till tryck : 17 november  
Medlemmarna : före jul

# ALFALIB

## Masterbiblioteket

ALFALIB.TXT 860206 218  
CROSSLIB.TXT 860206 303  
LIB.TXT 860206 219  
LÄSMEJ.TXT 851109 7  
TEST.FIL 860201 3

## ABC80/ASM/KOD

CALL.ASM 850912 3  
FYLLCRT.ASM 840920 7  
HELP.ASM 841120 123  
HEX.ASM 850519 31  
HSCROLL.ASM 850518 7  
JOY5458E.ASM 860215 2  
JOYINFO.TXT 860215 6  
JP.ASM 850912 3  
KOMMANDO.ASM 851001 67  
LIFE.Z80 850924 15  
LOAD.ASM 850912 3  
LANGRAD.ASM 860120 17  
MOD64K.ASM 850518 15  
MODABC.INF 860215 17  
TED1.ASM 851007 19  
TED2.ASM 851007 31  
TED3.ASM 851007 39  
TED4.ASM 851007 35  
TED5.ASM 851007 27  
TED6.ASM 851007 23  
TED7.ASM 851007 35  
TED8.ASM 851007 31  
TED8FDB.ASM 851216 30  
TEDLOGIK.TXT 850509 19

## ABC80/BERÄKN

A1.BAS 850517 3  
A108.BAS 850517 7  
A130.BAS 850517 3  
A133.BAS 850517 7  
A147.BAS 850517 11  
A154.BAS 850517 7  
A156.BAS 850517 7  
A3.BAS 850517 3  
A52.BAS 850517 3  
A54.BAS 850517 7  
A58.BAS 850517 7  
A60.BAS 850517 3  
A72.BAS 850517 11  
A81.BAS 850517 3  
A93.BAS 850517 3  
A95.BAS 850517 7  
A97.BAS 850517 7  
ACKER.BAS 840919 7  
ADMED1.BAS 850519 3  
BINARY2.BAS 850518 3  
BINÄR.BAS 850517 7  
BTGANLYS.BAS 840919 19  
BTGSANA.BAS 840919 19  
BUDGET.BAS 840919 7  
BUDGET.REM 840919 7  
BUDGETRE.BAS 840919 43  
BUDGETVA.BAS 840919 7  
CALC.BOK 841128 11  
CALC.INF 841128 15  
CALC1.BER 850517 27  
CALC1.REM 850517 15  
CALC2.BAS 850517 31  
CALC2.REM 850517 15  
CALC3.BOK 841128 43  
COUNT.BAS 850517 15  
DAGAR.BAS 840919 11  
DATEM.BAS 840919 11  
DELECT.BAS 840919 15  
DEKLARAT.BAS 850517 31  
DIFFEKV.BAC 860216 12  
DISTRLAG.BAS 850517 39  
EXEMPEL.NÄT 851024 3  
EXEMPEL.SIM 851024 3  
FACULIST.BAS 850517 7  
FAKTOR.BAS 850517 19  
FUNKTION.BAS 840919 23  
HEADFIT.BAS 840919 23  
INFO.BAS 850519 7  
INSTURKT.BAS 851024 71  
KALENDER.2 850518 51  
KALENDER.3 851024 15  
KALENDER.BAS 840919 11  
KALENDNY.BAS 850517 35  
KALENDR.BAS 840919 7  
KALENPR.BAS 851028 19  
KONTO.BAS 850517 7  
KRETSMAT.BAS 851024 43  
KYL.BAC 841119 27  
KYLINST.TXT 841119 11  
LINSER.BAS 850517 51  
LISTNING.BAS 851024 43  
LOGSIM.BAS 851024 11  
LOTTBOK.BAS 851029 7  
MATTE1.BAS 850519 87  
MATTE2.BAS 850519 43  
MULTI.BAS 840919 11  
MULTI2.BAS 840919 7  
MULTSTOR.BAS 850517 27  
MANEP05.BAS 850517 23  
OPAMP301.BAS 850517 15  
P16B.BAC 840919 23  
PARASCAN.BAS 840919 11  
PARITET.NÄT 851024 3  
PERSNR.BAS 840919 15

PLANTOPT.BAS 840919 23  
POKOMOBS.BAS 851024 27  
POTAVTVA.BAS 840919 7  
PRIMFAST.BAS 850519 7  
PRIMTAL.BAS 850629 7  
PROG1B.BAC 840919 31  
RABATT.BAS 840919 11  
RESULTAT.BAS 840919 35  
REX.BAS 840919 23  
ROTENUR.BAS 860207 10  
RYMDGEOM.BAS 850517 19  
RÄKNARE.BAS 840919 23  
RÄKNARE.REM 840919 11  
SÄLDSEK.BAS 850517 7  
SIGNALMA.BAS 851024 31  
SIMULERI.BAS 851024 51  
SINCOS.BAS 840919 15  
SJKUPENG.BAS 850517 7  
SKATT.BAS 840919 27  
SKATT84.BAS 850517 15  
SKATT85.BAS 850517 11  
SQRRROOT.BAS 840919 7  
STÄLTEMP.BAS 840919 7  
TELRN.BAS 840919 7  
TRIAREA.BAS 851104 11  
ÖREJÄMN.BAS 840919 7

## ABC80/BOKFÖR

BOKFÖR.BAS 850205 11  
BOKFÖR.MAN 850205 7  
BOKSLUT.BAS 841126 15  
BOKSLUT1.BAS 850205 27  
CMWKONTO.BAS 851024 11  
CMWKONTO.REM 851024 27  
DAGBOK.BAS 850205 23  
HUVUDBOK.BAS 850205 23  
INBALANS.BAS 850205 19  
KASSABOK.FIL 841126 11  
KONTO.ABC 850518 59  
KONTO.BAS 850518 7  
KONTOABC.MAN 850518 11  
KONTOBCK.BAS 851024 11  
KONTOBOK.BAS 851024 11  
KONTOBUD.BAS 851024 27  
KONTOINM.BAS 851024 15  
KONTOKOR.ABC 850513 23  
KONTOLIS.BAS 851024 19  
KONTOMEN.ABC 850513 7  
KONTOPRP.ABC 850518 27  
KONTOUTD.BAS 851024 11  
KTÖVARD.BAS 850205 35  
LÄSKONTO.FIL 841126 7  
MENYPROG.BAS 841126 11  
PRINTDAT.FIL 841126 15  
PRINTER.GIR 841126 15  
PRINTER.MOT 841126 15  
PRINTER.SÖK 841126 19  
RABALANS.BAS 850205 23  
TKONTO.BAS 850205 19  
TRANSKT.FIL 841126 19

## ABC80/DIDACTOS

BITTE1A.DID 850517 15  
BITTE1B.DID 850517 11  
BITTE2.DID 850517 15  
BITÖVN.DID 850517 3  
CASDISC.BAS 850517 11  
KOLON.BAS 850517 15  
CMDINT.SYS 850517 3  
DIDACTOS.BAS 850517 27  
DIDCOPY.BAS 850517 15  
DIDEDIT.BAS 850517 23  
DIDINFO.BAS 850517 35  
DIDMERGE.BAS 850517 23  
DIDOSGEN.BAS 850517 15  
DIDOSGEN.LD 850517 19  
DIDPRINT.FJÄ 850517 19  
DIDQUIZ.BAS 850517 27  
DIDSLICE.BAS 850517 35  
DIDSORT.BAS 850517 15  
ENAI.DID 850517 15  
GALLOWAY.DID 850517 7  
GOTTIT10.DID 850517 11  
HÖRENV81.DID 850517 11  
INSTALL.BAS 850517 7  
LÄSENV81.DID 850517 15  
ORBSFJÄ.TXT 850517 3  
OBSSYS.TXT 850517 3  
PRO.BAS 850517 55  
TATENT10.DID 850517 15  
TBTENT10.DID 850517 15  
TYTENT1.DID 850517 15  
TYTENT11.DID 850517 11  
TYTENT12.DID 850517 7  
TYTENT13.DID 850517 11  
TYTENT3.DID 850517 11  
TYTENT4.DID 850517 7  
TYTENT5.DID 850517 11  
TYTENT6.DID 850517 7  
TYTENT7.DID 850517 7  
TYTENT8.DID 850517 7  
TYTENT9.DID 850517 7  
TYUPPS1.DID 850517 7  
TYZAR1.DID 850517 7

## ABC80/DISKHANT

ASTART.BAS 850517 15  
BAKLIB.BAS 850517 15

BASTOCAS.BAS 841023 11  
BYECHAIN.BAS 840919 19  
CAT.BAS 840919 15  
COMPARE.BAS 840919 7  
COPYDTC.BAS 850517 15  
COPYFAST.BAS 840919 11  
COPYLIB.BAS 840919 15  
COPYTEXT.BAS 840919 7  
DELETE.BAS 850517 11  
DIR.BAS 841128 15  
DISKDDP2.BAS 840919 7  
DISKED.BAS 841127 23  
DISKEDIT.BAS 850517 23  
DISKSTAT.BAS 840919 23  
DISKTITT.BAS 840919 11  
DYKDÖD.O80 850517 3  
DÖP.BAS 840919 11  
ERR3940.BAS 840919 11  
ERRORSYS.BAS 840919 15  
EXSCREEN.BAS 840919 11  
EXSLIB.BAS 840919 11  
FILEPRO.BAS 850517 7  
FILEPRO.REM 850517 3  
FILTITT.BAS 840919 7  
FLEXLIST.BAS 850517 11  
FORM.BAS 851024 23  
IDMÄRK.BAS 840919 11  
INUTFIL.BAS 840919 11  
JKTEST35.BAS 840919 15  
JÄMFÖR.BAS 850924 7  
LIB.BAS 840919 15  
LIBPOKE.BAS 840919 15  
LIBRUN.BAS 851219 18  
LISTSKYD.BAS 840919 17  
LSTSKYDD.BAC 840919 19  
LSTSKYDD.REM 840919 7  
LÄSDISK1.BAS 850517 35  
LÄSFIL.BAS 850517 31  
MANAGER.BAS 840919 19  
MENY.BAS 850924 11  
MENYLIST.BAS 840919 15  
MINIDOS.BAS 850517 7  
NYTLB.BAS 840919 11  
NYUNSAVE.BAS 840919 11  
ONLY.BAS 850517 7  
PRLISTA.BAS 840919 19  
PROCENT.BAS 840919 15  
PROGRAMS.BAS 850517 23  
PROGRAMS.TXT 850517 19  
PROTALL.BAS 840919 11  
RENSA.BAS 850517 7  
RESCUE.BAS 851024 23  
SC.BAS 841119 15  
SC.INF 841119 7  
SHORT.BAS 841128 7  
SIZE.BAS 841119 11  
SKIIVREG.BAS 850924 27  
SKIIVREG.INF 850924 7  
SKIIVRENS.BAS 840919 7  
SKIIVSERV.BAS 850924 23  
SKRIVRAD.BAS 840919 11  
SORTFILE.BAS 850924 19  
TKNOLIB.BAS 841128 11  
TYPE.BAS 840919 19  
UNSAVE.BAS 840919 11  
VARPGM.BAS 840919 15

## ABC80/EDITORER

MANUAL.SED 850518 95  
PALINDRO.BAS 850902 15  
PED.BAS 850518 10  
RAK.BAS 850517 11  
RAKHÖGER.TXT 850517 31  
SCRSKAP.BAS 850518 3  
SED.BAS 850518 7  
SED.HLP 850518 15  
SED.XXX 850518 99  
TED.BAC 851007 35  
TED.FDB 851216 33  
TED.TXT 851007 63  
TED16.BAC 851007 35  
TEDFDB.BAS 851216 2  
TEDFDB.INF 851228 4  
TV.BAS 850111 3  
TVMAIN.BAC 850111 39  
TVSUB.D88 850111 19  
V2.32K 841121 35  
VDO.16K 850517 55  
VDO.BAC 841121 3  
VDO.REM 841121 39  
VDO2.16K 850517 19  
VDO2.32K 841121 35  
VVC.BAC 841121 11

## ABC80/GRAFIK

ABSTRAKT.BAS 850518 11  
ANSIKTE.BAS 840919 11  
BASPLOT.BAS 840919 15  
BESÖK.BAS 840919 15  
DEMO.TXT 851024 15  
DRAWINGS.BAC 851024 39  
FLAGGA.BAC 840919 19  
FÖRSTORA.BAS 851018 41  
GRAFIK.BAS 840919 7  
GRAFIKUR.BAS 850517 19  
HR1.BAS 860217 23  
HRPRINT.ASM 860215 16  
HRPRINT.BAS 860215 3

HRPRINT.INF 860215 5  
HÄLSNING.BAS 840919 3  
IGELKOTT.BAC 840919 19  
JULSTAR.BAS 850901 11  
KLOCKA.BAS 840919 15  
KLÖTSKUG.BAS 850518 11  
KULGRAF.BAS 840919 7  
LABYRINT.BAS 841128 11  
LIFE.BAC 850924 7  
LIFERIT.BAS 850924 15  
LINDA.BAS 850901 15  
LINDA.DAT 850901 55  
LÖJLIGT.BAS 850519 7  
MAZETTI.BAS 850518 11  
MUNKAR.BAS 841127 19  
MÖNSTER.BAC 840919 19  
MÖNSTER1.BAS 840919 7  
RITA.BAS 840919 15  
RITA2.BAS 850517 11  
RXPL0T.BAS 860215 8  
STKOMP80.BAC 851024 35  
SYMETRI.BAS 840919 7  
SYNKRON.BAC 840919 7  
TAKDROPP.BAS 840919 15  
TRIANGEL.BAS 840919 7  
ÄNDRABIG.BAS 851018 11

## ABC80/KASSHANT

CASBLOCK.BAS 840920 11  
CASDISK3.BAS 841119 15  
CASTEST.BAS 840920 27  
FASTCAS.16K 841119 19  
FASTCAS.32K 841119 23  
FIXERR20.BAS 850517 11  
GÖRCA80.BAS 841128 23  
GÖRCA80.INF 841128 7  
KAS800.90B 840920 7  
KAS800.BAS 840920 7  
KASMENY.BAS 840920 43  
LIBCASPR.BAS 840920 31  
LASCAS.BAS 841120 11  
TESTLOAD.BAS 840920 11

## ABC80/KOMMUNIK

ABCMINI.BAS 840919 7  
ABCTRN4.BAS 840919 31  
ABCTRN52.BAS 840919 19  
ABCTRN56.BAS 850602 27  
ABCV24.EKO 850111 7  
BASTERM.BAS 851015 35  
CASMINI.BAS 840919 11  
CASSEND2.BAS 840919 11  
CASSEND3.BAS 840919 11  
CURSORTS.BAS 850517 55  
EPSON.BAS 850725 7  
FILTR80K.BAS 840919 31  
LOGAIN2.BAS 850517 23  
LOGBOOK.BAS 851210 35  
MOD5124.BAS 850518 47  
MODEMTID.BAS 860131 11  
MONITOR.BAS 841128 27  
NYHETER.TXT 841111 3  
PATCH100.TXT 851106 11  
RINGABC.BAS 850630 23  
RINGWON.BAS 850519 23  
SAFT.BAS 840919 23  
SCROLL1.TXT 851001 7  
SCROLL2.TXT 851001 7  
SCROLL3.TXT 851001 7  
SENDKOM.BAS 841128 11  
SENDKOM.INF 841128 11  
SETUP.BAS 841111 51  
SDMPKOM.BAS 841127 11  
SDMPKOM.INF 841127 11  
SOFTSIO.16K 851015 27  
SOFTSIO.32K 851015 27  
SOFTSIO.INF 851015 27  
TEDTERM.BAC 860208 18  
TEDTERM.TXT 860205 32  
TELELOG.BAS 850518 63  
TELELOG.INF 850518 3  
TELELOG.PRP 850518 23  
TERM100.ABS 851001 31  
TERM100.BAC 850523 23  
TERM100.BAS 841111 7  
TERM100.DOC 841111 83  
TERM100.ITH 841111 71  
TERM100.LAT 841111 23  
TERM100B.BAC 850523 27  
TERM100C.BAC 850523 7  
TERM100L.BAS 851006 7  
TESTSIO.BAS 851015 7  
TRMHLPI.TXT 841111 3  
TRMHLPI.TXT 841111 3  
TRMHLPU.TXT 841111 7

## ABC80/SORTERA

HVKSORT2.BAS 850517 11  
QSORTER.BAS 850517 7  
SORTDEMO.BAS 850518 23  
SORTERA.BAS 850517 27  
STRSORT.BAS 850517 15  
SWESORT.BAS 850517 11  
ABC80/SPEL  
ABGSTAR.BAS 850517 79  
ABGSTAR.INF 850517 23  
ADV2.BAC 851006 19  
ADVENTUR.BAS 850719 43  
ADVENTUR.ABS 851006 19  
ADVENTUR.INF 850601 7  
AERODROM.BAS 850917 43  
ALLEN.BAS 840919 15  
AMAZING.BAS 840919 27  
ARK.BAS 841128 7  
ARKEN.BAS 841128 19  
ARKEN.INF 841128 7  
ARKEOLOG.BAS 850901 43  
ARKEOLOG.HLP 850901 43  
ASTROZIP.BAS 840919 15  
BANDITNY.BAS 860105 11  
BIL16K.BAS 840919 31  
BIL32K.BAS 840919 47  
BILFACE.BAS 840919 15  
BILSPEL.BAS 850517 35  
BLACKJAK.BAS 840919 47

## ABC80/MUSIK

BERGS.BAS 851024 7  
LUFFACE.BAS 851024 11  
MUSTNT1.BAS 850902 35  
PIANO.BAC 840919 15

RIVANLAT.BAS 850517 11  
SYNTIZE.MUS 850517 19  
SYNTIZER.BAS 840919 19  
TICOTICO.BAC 840919 11

## ABC80/OKLASSIF

BAKLÄNGS.BAS 840920 7  
BLDINDEX.TXT 850518 107  
DKTER.BAS 850517 7  
KRYPTO.BAS 850517 7  
LISTROM.BAS 840920 7  
ORDER.BAS 840920 11  
RÖDLUVA.SAG 850518 27  
SKRM.BAS 840920 7  
TYSKAVER.B 840920 55

## ABC80/PRGRAFIK

DONALD.BAS 841119 15  
KENNEDY.BAS 841119 23  
MADONNA.BAS 841119 31  
MUSSE.BAS 841119 27  
SNORREN.BAS 841119 15  
SMOOPY.BAS 841119 19  
TYPO16K.BAS 850412 59  
TYPO16K.TXT 850412 15  
TYPO32K.BAS 850412 75

## ABC80/RADIO

ASCTOP.BAS 850917 15  
CWTÄRNA.BAS 851106 11  
DXGUIDE.BAS 840920 31  
EDITERA.BAS 850917 19  
INIT.BAS 850917 7  
LCBERÄK.BAS 850517 7  
LOGGBOK.BAS 850517 27  
MORSE.BAS 840920 23  
MORSETEK.BAS 850517 63  
MORSEÖVN.BAS 851210 59  
OSKARTIO.BAS 850517 31  
RAMSA.BAS 850917 7  
RYTHM.BAS 850917 7  
SKRIV.BAS 850917 15  
SNABBLEK.BAS 850917 11  
START.BAS 850917 7  
SÖK.BAS 850917 7  
TGTV24.BAS 850917 39  
ÖVERFÖR.BAS 850917 7

## ABC80/REGISTER

BILREG.BAS 850517 11  
DATAREGI.BAS 850517 11  
INFO.PRG 841126 11  
INFOREV.ADR 841126 11  
INFOFAR.ADR 841126 39  
INFOVAR.ADR 841126 19  
KASXADD.BAS 841127 15  
KASXINX.BAS 850924 15  
KASXLIS.BAS 841127 11  
KASXMENU.BAS 841127 7  
KASXMENU.NY 850924 7  
KASXRAP.BAS 841127 23  
KASXSÖK.BAS 841127 7  
KASXSÖK.BAS 841127 19  
KOMPL.ADR 841126 31  
PRINTER.ADR 841126 15  
REGPROG.BAS 850517 19  
REGSORT.BAS 850518 23  
SERIE.BAS 841126 11  
SKRIV.ADR 841126 15  
SKÄRM.ADR 841126 11  
SÖK.ADR 841126 15  
TEXT.ADR 841126 7  
VALPR.ADR 841126 51  
ÄNDRA.ADR 841126 27

# 1986-02-22

|                |        |    |                 |        |    |               |        |    |                |        |     |               |        |    |
|----------------|--------|----|-----------------|--------|----|---------------|--------|----|----------------|--------|-----|---------------|--------|----|
| BLOCKAD .BAS   | 840919 | 19 | RYMDRESA .BAS   | 841119 | 27 | ABC80/TIPS    |        |    | PRLIST .BAS    | 850917 | 11  | EBER .BAS     | 840920 | 87 |
| BLOCKAD .BAS   | 850517 | 19 | SCORESUB .BAS   | 850519 | 15 | -----         |        |    | RAD25COL .BAS  | 850517 | 7   | EGETPROG .BAS | 840920 | 27 |
| BOLLJAKT .BAS  | 850517 | 19 | SCORESUB .INF   | 850519 | 11 | L8NADER .BAS  | 840920 | 23 | RAM .BAS       | 860208 | 5   | ELECTON2 .BAS | 840920 | 27 |
| ROLLSPEL .BAS  | 851024 | 23 | SEABAT .BAS     | 850517 | 67 | LOPOFLEX .BAS | 840920 | 11 | RAM .TXT       | 860205 | 16  | ENORM .BAS    | 840920 | 87 |
| BOMB .BAS      | 840919 | 23 | SJUSLAG1 .BAS   | 850517 | 87 | LOTT0576 .BAS | 840920 | 31 | RAMTEST .BAS   | 841128 | 27  | FLEXFIL .BAS  | 840920 | 7  |
| BOMBING .BAS   | 851024 | 31 | SJUSLAG1 .BAS   | 850517 | 7  | LOTT0576 .REM | 840920 | 7  | RAMTOFIL .BAS  | 860205 | 6   | FLUNSAN .BAS  | 850419 | 63 |
| BREAKOUT .2    | 850709 | 27 | SKATT1 .BAS     | 850518 | 39 | LOTT0NR2 .BAS | 840920 | 19 | REMERASE .BAS  | 850517 | 11  | FLUNSI .BAS   | 850419 | 23 |
| BREAKOUT .BAS  | 850517 | 27 | SKATT2 .BAS     | 850518 | 14 | LOTTLOTT .80T | 841127 | 15 | REMHODE .BAS   | 840919 | 15  | HEXADR .BAS   | 840920 | 7  |
| BRIDGE .BAS    | 840919 | 23 | SKIPIST .BAS    | 851219 | 35 | STRYK12X .BAS | 840920 | 11 | RESETFIX .BAS  | 840919 | 7   | INTEGRAL .BAS | 840920 | 7  |
| BULDERDA .BAS  | 850917 | 15 | SLALOM .BAS     | 850919 | 27 | STRYK12X .REM | 840920 | 35 | RUNONLY .BAS   | 840919 | 7   | KALENDER .BAS | 850725 | 11 |
| BYÅKRAT .BAS   | 840919 | 15 | SOLVALLA .BAS   | 850517 | 31 | TIPSRAD .BAS  | 841119 | 7  | SHOW .BAS      | 841128 | 15  | KALKTL .FN    | 850725 | 15 |
| BÖRSEN .BAS    | 850517 | 27 | SPIDER .BAS     | 840919 | 19 | TIPSSYS .BAS  | 840920 | 35 | SHOW .INF      | 841128 | 7   | KASFIL .BAS   | 840920 | 7  |
| BÖRSEN .INF    | 850517 | 15 | SQUASH .BAS     | 840919 | 47 | TIPSSYS .TPS  | 840920 | 35 | SKARM .BAS     | 840919 | 11  | LOTT080K .BAS | 851029 | 7  |
| BÄTHESAN .BAS  | 850517 | 27 | STARTRK2 .BAS   | 840919 | 75 | UMSYSGAR .BAS | 840920 | 19 | SLAV .BAS      | 851010 | 23  | LAN .BAS      | 850725 | 35 |
| CHEMIST .BAS   | 840919 | 11 | STARTRK2 .REM   | 840919 | 7  | UMSYSTEM .BAS | 840920 | 23 | SLAV .TXT      | 851010 | 15  | MATTE800 .TXT | 840920 | 43 |
| CHOPPER .BAS   | 841216 | 47 | STENKUL .BAS    | 850517 | 59 | -----         |        |    | SLAV16 .BAS    | 851010 | 23  | MATTEAD .BAS  | 850903 | 19 |
| CHOPPER .REV   | 841216 | 47 | STJÄRNA .BAS    | 850517 | 19 | ABC80/UTILITY |        |    | SPECIFIK .BAS  | 841119 | 7   | MAME .BAS     | 850903 | 19 |
| DEFENCE .BAS   | 850517 | 23 | STROKE .BAS     | 851024 | 22 | -----         |        |    | STILMARG .BAS  | 850924 | 27  | MAME .INF     | 850903 | 19 |
| DEFENDER .BAS  | 850517 | 35 | STRKULA .BAS    | 851216 | 22 | ABDATA .REM   | 850517 | 7  | STILMARG .INF  | 850924 | 23  | NOLLSÖK .BAS  | 840920 | 7  |
| DJUR .BAS      | 850517 | 23 | SUPERMSK .BAS   | 850519 | 23 | ABSMKE .BAS   | 850917 | 15 | SWAPER .BAS    | 850517 | 15  | NYPREC .BAS   | 840920 | 11 |
| DJUR .REM      | 850517 | 7  | SURROUND .BAS   | 840919 | 27 | ADATA .BAS    | 850517 | 7  | SÖKAREN .BAS   | 840919 | 7   | PROG1 .BAS    | 840920 | 55 |
| DOGFIGHT .BAS  | 850517 | 31 | SÄSER .2        | 840919 | 43 | ASM .MRG      | 851020 | 7  | TECKEN80 .BAS  | 841127 | 3   | PROG1 .ORG    | 840920 | 55 |
| DOMINO .BAS    | 840919 | 15 | SÄSER .BAS      | 840919 | 55 | ASMCNO .NEW   | 851020 | 7  | TESTVARL .BAS  | 860127 | 11  | PROG1 .RAM    | 840920 | 19 |
| DRAGRACE .BAS  | 850518 | 19 | SÄSER16K .BAS   | 840919 | 55 | ASMDATA .BAS  | 850517 | 11 | UTSKRIFT .2    | 850518 | 11  | RÄKNARE .BAS  | 840920 | 19 |
| DROPOUT .BAS   | 850517 | 15 | TANGFEL .BAS    | 851024 | 11 | ASS .BAS      | 851010 | 39 | UTSKRIFT .BAS  | 850517 | 7   | RÄKNÖVN .BAS  | 850725 | 15 |
| ENBANDIT .BAS  | 850517 | 31 | TANKELAS .BAS   | 850518 | 43 | ASS .TXT      | 851010 | 39 | VAKNA .BAS     | 850901 | 11  | RÄKNÖVN .LÄG  | 850725 | 15 |
| EPEDEMI .BAS   | 851018 | 11 | TIME .BAS       | 850917 | 43 | BDATA .BAS    | 850517 | 23 | VAKNA .TXT     | 850901 | 23  | SEKVITIN .BAS | 840920 | 11 |
| FARTYG .BAS    | 840919 | 15 | TJUGOETT .BAS   | 851024 | 27 | BR .BAS       | 850713 | 3  | VARIABEL .BAS  | 850917 | 7   | START .BAS    | 840920 | 15 |
| FENKAMP .BAS   | 840919 | 51 | TUBEN .BAS      | 840919 | 15 | BR .INF       | 850602 | 3  | VARLIST .BAS   | 850917 | 7   | VÄRD .BAS     | 840920 | 15 |
| FLUGAN .BAS    | 850517 | 31 | TUPP .BAS       | 851024 | 19 | CALL .BAS     | 850713 | 3  | VARLIST .DOC   | 860127 | 18  |               |        |    |
| FLUGORNA .BAS  | 850517 | 31 | TURNABLK .BAS   | 840919 | 27 | CALL .INF     | 850602 | 3  | VISA .BAS      | 840919 | 11  |               |        |    |
| FMG .BAS       | 840919 | 43 | TWSEPL .BAS     | 840919 | 7  | CONCAT .BAS   | 840919 | 7  | VISALAP .BAS   | 860105 | 7   |               |        |    |
| FRAGOR10 .BAS  | 850517 | 11 | UBAT .BAS       | 850709 | 43 | CONVEYOR .BAS | 850917 | 43 | VIXALAPGM .BAS | 850517 | 11  |               |        |    |
| FRÖSTREN .BAS  | 850518 | 39 | UBAT1281 .BAS   | 840919 | 27 | CPMABC80 .BAS | 860215 | 14 | -----          |        |     |               |        |    |
| GLIPSTY .BAS   | 841120 | 27 | UPPSJ0 .BAS     | 840919 | 43 | CSUMTEST .BAS | 840919 | 11 | ABC80/UTLÄDA   |        |     |               |        |    |
| GOBLIN .BAS    | 841119 | 23 | UPPSJ0 .INF     | 840919 | 11 | DELETT .BAS   | 840919 | 11 | BANDITEN .BAS  | 860105 | 7   |               |        |    |
| GROTTAN .BAS   | 840919 | 43 | UPPSJ02 .PRO    | 840919 | 3  | DELMEN .BAS   | 840919 | 11 | BASTICER2 .SYS | 860103 | 11  |               |        |    |
| HAFJISK .BAS   | 850519 | 27 | UPPSJ03 .PRO    | 840919 | 3  | DEVELOP .BAS  | 840919 | 7  | CPR .BAS       | 860105 | 3   |               |        |    |
| HANOTTRE .BAS  | 850518 | 15 | UPPSJ04 .PRO    | 840919 | 3  | DMFIMX .BAS   | 850517 | 11 | CPR .TXT       | 860105 | 23  |               |        |    |
| HIO .BAS       | 840919 | 23 | UPPSJ05 .PRO    | 840919 | 3  | DIS .BAS      | 860127 | 10 | CPR3280 .BAS   | 860105 | 7   |               |        |    |
| HOPPEHÄST .GAM | 850517 | 35 | UPPSJ06 .PRO    | 840919 | 3  | DISABS .BAS   | 851001 | 39 | CW .BAS        | 851014 | 11  |               |        |    |
| HÖG .BAS       | 850518 | 11 | WARFISH .BAS    | 840919 | 19 | DISKREG .BAS  | 840919 | 27 | CW .INF        | 851014 | 7   |               |        |    |
| INTÄ .DAT      | 841128 | 3  | WORLDPOW .BAS   | 840919 | 27 | DISP .BAS     | 850713 | 3  | DIS .XYZ       | 860107 | 43  |               |        |    |
| INSTANGD .BAS  | 840919 | 15 | YATZY .BAS      | 850903 | 23 | DISP .INF     | 850602 | 3  | FILTER .BAS    | 851104 | 27  |               |        |    |
| INWADER .BAS   | 840919 | 39 | YATZY .REM      | 850903 | 7  | EMHMAKE .BAS  | 850917 | 15 | HR1 .OLD       | 860215 | 15  |               |        |    |
| INWADERS .BAS  | 850709 | 35 | YELSUB .BAS     | 850917 | 23 | ENHET .BAS    | 841127 | 3  | SPACE .BAS     | 860212 | 19  |               |        |    |
| JAKT .BAS      | 840919 | 23 | ZBOINFO .BAS    | 850917 | 11 | ETIKETT2 .BAS | 850517 | 11 | START .BAS     | 860104 | 19  |               |        |    |
| JAKTPLAN .BAS  | 840919 | 23 | ZBOJAKT .BAS    | 850917 | 59 | EXTDEL .BAS   | 840919 | 11 | STEREO .BAS    | 851105 | 27  |               |        |    |
| JUKORAN .BAS   | 840919 | 15 | ÅTA .BAS        | 850517 | 11 | FILSUM .BAS   | 840919 | 7  | STEREO .REM    | 851105 | 27  |               |        |    |
| KANOK .BAS     | 850709 | 23 | ABC80/STYROMKÄT |        |    | FILTOHEX .BAS | 850602 | 11 | STEXT .BAS     | 860128 | 35  |               |        |    |
| KATTRÄTT .BAS  | 840919 | 15 | -----           |        |    | FILTORAM .BAS | 860205 | 5  | STEXT .INF     | 860128 | 11  |               |        |    |
| KLIPPY .BAS    | 850917 | 23 | PREKVEN .BAS    | 841127 | 3  | FORMAT .BAS   | 850517 | 11 | TESTVARL .XYZ  | 860107 | 5   |               |        |    |
| KOCK .BAS      | 860220 | 25 | PIREG .BAS      | 850927 | 15 | GRAFFRIN .BAS | 840919 | 7  | TIPSREDA .BAS  | 851219 | 60  |               |        |    |
| KOCK .INF      | 860220 | 9  | PIREG .BAS      | 850517 | 11 | GRAFRUB .BAS  | 840919 | 7  | TIPSREDA .MRG  | 851219 | 4   |               |        |    |
| KOKT .BAS      | 840919 | 19 | PIREG .INF      | 850517 | 7  | GURABS .BAS   | 841128 | 7  | VARLIST .XYZ   | 860107 | 11  |               |        |    |
| KRYSS .BAS     | 860205 | 22 | ABC80/SUBROUTIN |        |    | GURITH .BAS   | 841128 | 7  | XYZKOD .BAS    | 860106 | 8   |               |        |    |
| KRYSS .INF     | 860205 | 19 | -----           |        |    | HEX .BAS      | 850713 | 11 | XYZKOD .BAS    | 860106 | 8   |               |        |    |
| KULFANG .BAS   | 850517 | 19 | BINARY .ASM     | 850517 | 7  | HEXTOFIL .BAS | 850602 | 23 | -----          |        |     |               |        |    |
| KULFANG .BAS   | 851024 | 15 | CAPEDIT .BAS    | 840920 | 7  | HJALP16K .BAS | 840919 | 35 | ABC80/ASMKOD   |        |     |               |        |    |
| LABJAKT .BAS   | 840919 | 15 | DATKOLL .BAS    | 850517 | 7  | HJALP32K .BAS | 840919 | 31 | BASTIC2 .ASM   | 840920 | 35  |               |        |    |
| LANDING .BAS   | 850517 | 27 | DOWNSCRL .BAS   | 850412 | 7  | HJALPARE .REM | 840919 | 11 | BASIC6 .ASM    | 840920 | 35  |               |        |    |
| LANDING .REM   | 850517 | 3  | ENTER80 .BAS    | 850412 | 23 | HLP32 .80T    | 850517 | 11 | GRAFDUMP .ASM  | 841127 | 11  |               |        |    |
| LIFE .BAS      | 850517 | 39 | FILENAME .BAS   | 850412 | 7  | HUVUD .BAS    | 840919 | 11 | HRINOUT .ASM   | 840920 | 7   |               |        |    |
| LISA .BAS      | 841128 | 43 | FNK .BAS        | 840920 | 7  | ITH .BAS      | 850906 | 7  | HRTILLPR .ASM  | 850517 | 59  |               |        |    |
| LISA .INF      | 841128 | 3  | GET .BAS        | 850517 | 15 | JOYSTYRN .BAS | 860215 | 5  | OMVAL .ASM     | 850517 | 67  |               |        |    |
| LOGIKTST .BAS  | 850517 | 55 | GETSKRIV .BAS   | 840920 | 11 | JP .BAS       | 850713 | 3  | RAMBA5 .ASM    | 840920 | 7   |               |        |    |
| MASKLIPP .BAS  | 840919 | 23 | GETSKRIV .REM   | 840920 | 7  | JP .INF       | 850602 | 3  | RANBLK .ASM    | 840920 | 7   |               |        |    |
| MASTER .BAS    | 850517 | 27 | GRPHCHAR .BAS   | 850412 | 11 | KASSLIST .BAS | 850517 | 15 | RAMPRG .806    | 841127 | 27  |               |        |    |
| MASTER2 .BAS   | 850517 | 27 | INGRESS .BAS    | 850517 | 7  | KASSLIST .INF | 850517 | 7  | REMPTE .ASM    | 850709 | 31  |               |        |    |
| MASTIND .BAS   | 851024 | 15 | INPUT .BAS      | 850412 | 7  | KRYPTOSW .BAS | 850725 | 23 | -----          |        |     |               |        |    |
| MATSPGM .BAS   | 850518 | 19 | LEFTSCRL .BAS   | 850412 | 7  | KRYPTOSW .TXT | 850725 | 23 | ABC80/ASTRONOM |        |     |               |        |    |
| MEMORERA .BAS  | 840919 | 11 | MENY .BAS       | 850412 | 7  | LADDAABS .BAS | 850517 | 19 | ECLIP .BAS     | 840920 | 15  |               |        |    |
| METOR .BAS     | 851010 | 43 | MENY2 .BAS      | 850517 | 11 | LARMKLOC .BAS | 850901 | 19 | GALAXY .BAS    | 840920 | 23  |               |        |    |
| MINDLOAD .BAS  | 840919 | 11 | MULT100 .ASM    | 850517 | 7  | LAYOUT .BAS   | 840919 | 31 | JSATS .BAS     | 840920 | 23  |               |        |    |
| MINDPLAY .BAS  | 840919 | 35 | NYTTIG .BAS     | 841127 | 7  | LIB .BAS      | 840919 | 15 | PLANET .BAS    | 840920 | 47  |               |        |    |
| MINTSLA .BAS   | 850709 | 59 | OUTSTR1 .BAS    | 850517 | 7  | LITBLIST .BAS | 840919 | 11 | RADCE .BAS     | 840920 | 23  |               |        |    |
| MONKOLP .BAS   | 851004 | 51 | RITA .BAS       | 840920 | 11 | LISTAREN .BAS | 850519 | 59 | SKYPLOT .BAS   | 840920 | 51  |               |        |    |
| MONSTER .BAS   | 841127 | 7  | RUNTIME .BAS    | 850412 | 7  | LISTAREN .INF | 850519 | 7  | -----          |        |     |               |        |    |
| MOON .BAS      | 840919 | 35 | SATSEXE .BAS    | 860219 | 9  | LISTBART .BAS | 840919 | 7  | ABC80/BASREG   |        |     |               |        |    |
| MOONTRAV .BAS  | 840919 | 27 | SCROLLER .BAS   | 850517 | 11 | LISTVARI .BAS | 840919 | 23 | BAS800 .INF    | 860211 | 10  |               |        |    |
| NORSK .BAS     | 840919 | 27 | SKRM .BAS       | 840920 | 15 | LOAD .BAS     | 850713 | 3  | RAMBLK .806    | 840920 | 7   |               |        |    |
| NYL .BAS       | 860105 | 9  | SKRMPR .BAS     | 840920 | 7  | LOAD .INF     | 850602 | 3  | RYMDSKP .BAS   | 850725 | 11  |               |        |    |
| OLYMPIA1 .BAS  | 850518 | 7  | SKRMPR .REM     | 840920 | 7  | LOOK .BAS     | 840919 | 15 | SETDOT .806    | 850131 | 7   |               |        |    |
| OLYMPIA2 .BAS  | 850518 | 3  | SORTSUB .BAS    | 850517 | 3  | LVARIRAD .BAS | 840919 | 27 | SNOOPY .BAS    | 850725 | 19  |               |        |    |
| OLYMPIA3 .BAS  | 850518 | 35 | STANDBY .BAS    | 850412 | 11 | LXDISK2 .BAS  | 840919 | 43 | SNÖ .BAS       | 840920 | 7   |               |        |    |
| ORDSPRK2 .BAS  | 841128 | 35 | TESTSORT .BAS   | 851029 | 15 | LXDISK2 .REM  | 840919 | 11 | STAPEL .BAS    | 850725 | 7   |               |        |    |
| ORDSPRK1 .BAS  | 850517 | 39 | TV .BAS         | 841126 | 3  | LÖK1 .BAS     | 860215 | 3  | TEST .PTC      | 840920 | 131 |               |        |    |
| OTHELLO .BAS   | 851005 | 31 | TV1 .BAS        | 841126 | 15 | LÖK2 .BAS     | 860215 | 48 | VIRRE .BAS     | 850501 | 35  |               |        |    |
| PADDEL .BAS    | 840919 | 19 | TV2 .BAS        | 841126 | 7  | LÖKINFO .TXT  | 860215 | 10 | WALKKUB .BAS   | 850725 | 3   |               |        |    |
| PLAYTILL .BAS  | 840919 | 11 | TV3TILL .BAS    | 841126 | 7  | LÖKMERGE .BAS | 860215 | 6  | XXXPAC .BAS    | 860211 | 48  |               |        |    |
| QUEST .BAS     | 840919 | 43 | TV3TILL .INF    | 841126 | 7  | LANGRAD .BAS  | 860105 | 7  | -----          |        |     |               |        |    |
| QWERTY .BAS    | 840919 | 27 | TV3TILL .BAS    | 841126 | 7  | MEMCHECK .BAS | 840919 | 11 | ABC80/KASSHANT |        |     |               |        |    |
| RACING .BAS    | 850518 | 11 | TV3TILL .BAS    | 841126 | 7  | MEMSTAT .BAS  | 840919 | 11 | CASDISK1 .BAS  | 840920 | 31  |               |        |    |
| REAKTION .BAS  | 840919 | 11 | TV3TILL .BAS    | 841126 | 7  | MENY .BAS     | 860215 |    |                |        |     |               |        |    |

|              |        |     |                  |        |     |               |        |     |               |        |     |                            |        |     |
|--------------|--------|-----|------------------|--------|-----|---------------|--------|-----|---------------|--------|-----|----------------------------|--------|-----|
| FILTRANS.00M | 841120 | 59  | ABC800/SUBROUTIN |        |     | LISRAM .BAS   | 840920 | 19  | CPM/UTILITY   |        |     | UFOLDMAN.TXT               | 860203 | 213 |
| FILTRANS.BAS | 851025 | 47  | =====            |        |     | LISUPP .BAS   | 850910 | 7   | =====         |        |     | UFORTH .ABS                | 860203 | 41  |
| FILTRANS.RUT | 851025 | 11  | BACKGRUND.BAS    | 850629 | 7   | MENY .BAS     | 851109 | 15  | ALGINTRO.TXT  | 850310 | 31  | UFORTH .BAC                | 860203 | 55  |
| GUTSNY .BAS  | 851019 | 23  | CHANGE .BAS      | 850917 | 7   | MENY2 .BAS    | 850306 | 47  | ALGOLM .HEX   | 850310 | 151 | UFORTH .DAT                | 860203 | 82  |
| INUSERS.BAS  | 850429 | 7   | DATE .BAS        | 850603 | 3   | MINIMENY.BAS  | 851223 | 13  | ALGSTART.TXT  | 850310 | 15  | UFORTH .INF                | 860203 | 6   |
| KERMIT .BAS  | 850417 | 107 | DAYTIME .BAS     | 851029 | 11  | NTTERR .BAS   | 851006 | 23  | ARRAY .ALG    | 850310 | 15  | VINDRUTA.UF                | 860203 | 3   |
| KERMIT .MAN  | 850105 | 23  | ENTER806.BAS     | 850412 | 23  | NTTERR .INF   | 851006 | 7   | BANNER .ASM   | 850514 | 43  |                            |        |     |
| KERMIT .TXT  | 850105 | 19  | FNDATUM .BAS     | 860121 | 17  | OPCODE .DAT   | 840920 | 15  | BANNER .HEX   | 850514 | 19  | INLÅDA                     |        |     |
| LOGIN .BAS   | 840920 | 51  | FNFAHG .BAS      | 841120 | 11  | PARBAS .BAS   | 850917 | 7   | BITMAP .ASM   | 850514 | 27  | =====                      |        |     |
| MINIMON .IDM | 850525 | 19  | FNFAHG .REM      | 841120 | 11  | PEEK .BAS     | 840920 | 19  | BLK .INP      | 850310 | 3   | LÅSMIG .TXT                | 860113 | 4   |
| MSGHMT .BAS  | 860212 | 20  | FNGET2 .BAS      | 850709 | 15  | POSTGRD .BAS  | 850505 | 11  | BLKTEST .ALG  | 850310 | 7   |                            |        |     |
| NTHLPH .INF  | 840920 | 18  | FNGET2 .INF      | 850709 | 15  | PROCKEY .BAS  | 850917 | 19  | BOOKBIO3.HEX  | 860216 | 64  | Observera att filerna      |        |     |
| NYMON .BAS   | 840920 | 91  | FNMSIK .BAS      | 841120 | 11  | PROCKEY .HEX  | 850917 | 11  | BOOLINT .ALG  | 850310 | 7   | ligger bara i inläddorna   |        |     |
| PARPR .BAS   | 850106 | 43  | FNMSORT .BAS     | 850902 | 15  | PROCKEY .REM  | 850917 | 11  | CASETWO .ALG  | 850310 | 3   | en kort tid. De filer som  |        |     |
| PARV24 .BAS  | 850106 | 63  | FORMAT .SUB      | 850629 | 11  | PROCKEY .ASM  | 851019 | 15  | CAT .ASM      | 850514 | 23  | finns i inläddorna just nu |        |     |
| QZ .806      | 860126 | 54  | HCTELED .BAS     | 850629 | 7   | PROCKEY .HEX  | 851019 | 31  | CAT .DOC      | 850514 | 11  | när detta skrivs finns     |        |     |
| REMOASH.BAS  | 850429 | 19  | INLINE .BAS      | 850127 | 11  | RAMBAS .806   | 840920 | 7   | CATLOG .HLP   | 850514 | 23  | troligen inte där när      |        |     |
| REMOE .BAS   | 850429 | 3   | INPLIN .DEF      | 850917 | 11  | RANDP .806    | 840920 | 39  | COMERR .TXT   | 850310 | 11  | ABC-Bladet är tryckt.      |        |     |
| REMOE .DOC   | 850430 | 15  | INPLIN .REM      | 850917 | 11  | REGIST .BAS   | 851109 | 39  | DATABAS .DOC  | 850310 | 27  |                            |        |     |
| RKERMIT .BAS | 850901 | 55  | KOMMANDO.BAS     | 850505 | 15  | REGIST .INF   | 851109 | 11  | DBENTRY .BAS  | 850310 | 27  | INLÅDA/ABC80               |        |     |
| RLOGIN .BAS  | 850429 | 7   | LISKOD .BAS      | 850502 | 11  | REGISTER.BAS  | 850912 | 107 | DEQUERY .BAS  | 850310 | 35  | =====                      |        |     |
| RTERM .BAS   | 850430 | 7   | RENUMBER.BAS     | 850505 | 7   | REGISTER.INF  | 850903 | 7   | DBSETUP .BAS  | 850310 | 31  | ASMCONEK .                 | 851221 | 7   |
| SETUP .BAS   | 840920 | 59  | SCROLL .BAS      | 851006 | 7   | REGISTER.TXT  | 851226 | 7   | DU .ASM       | 850514 | 179 | ASMEYX .BAS                | 851217 | 50  |
| START .BAS   | 850106 | 19  | SETPERM .BAS     | 850529 | 11  | RELOCATE.BAS  | 850917 | 11  | DU .DOC       | 850514 | 23  | BIL2ST .BAC                | 860204 | 56  |
| T .BAS       | 860126 | 54  | SUBRUT1 .BAS     | 850902 | 15  | RELOCATE.REM  | 850917 | 11  | DU .HLP       | 850514 | 23  | COPYFORM.BAC               | 860221 | 21  |
| UPPRINGN.BAS | 860216 | 8   | TESTSORT.BAS     | 851029 | 15  | REMBORT .BAS  | 840920 | 27  | FILE1 .INP    | 850514 | 127 | DEM01 .COM                 | 860222 | 2   |
|              |        |     | TEXTED .BAS      | 860211 | 6   | REKÄNDRA .BAS | 840920 | 11  | FILE2 .INP    | 850514 | 3   | DEM02 .COM                 | 860222 | 2   |
|              |        |     | TYP .BAS         | 850313 | 7   | RESCUE .BAS   | 860121 | 30  | FIN2 .INP     | 850514 | 39  | DEM03 .COM                 | 860222 | 2   |
|              |        |     | VIRTUELL.BAS     | 850902 | 7   | RUNONLY .BAS  | 850902 | 19  | FIN2 .ASM     | 850514 | 39  | EPROM .ASM                 | 851217 | 85  |
|              |        |     |                  |        |     | RUNRAM .806   | 840920 | 7   | FIXTEX .HEX   | 850514 | 19  | EPROM .ASS                 | 851217 | 14  |
|              |        |     | ABC800/SYSTEMPR  |        |     | SCROLL .BAS   | 850424 | 7   | FLYTEST .ALG  | 850310 | 7   | EPROM .TXT                 | 851217 | 7   |
|              |        |     | =====            |        |     | SDIO .BAS     | 850505 | 59  | GENCPM .DAT   | 860216 | 14  | EPROM .BAC                 | 860109 | 4   |
|              |        |     | BACKUP .ABS      | 851010 | 35  | SETCAL2 .806  | 840920 | 19  | GENCPM .INF   | 860216 | 13  | EPYRAHAD.BAS               | 860204 | 34  |
|              |        |     | BACKUP .BAS      | 850210 | 3   | SIH802 .ABS   | 860125 | 6   | GOTOTEST .ALG | 850310 | 7   | GRAFIK .BAS                | 860220 | 20  |
|              |        |     | CATSTART.2       | 850726 | 7   | SIH802 .INF   | 860125 | 6   | HELP .ASM     | 850514 | 159 | LABYRINT.BAC               | 860204 | 28  |
|              |        |     | CATSTART.BAS     | 850726 | 11  | SKRIVLIB.BAS  | 840920 | 23  | HELP .HLP     | 850514 | 7   | LARMCLOC.BAC               | 860131 | 18  |
|              |        |     | DISKCHK.BAS      | 850210 | 3   | SKRT .BAS     | 850901 | 15  | HELP2 .HLP    | 850514 | 79  | LARMCLOC.INF               | 860204 | 4   |
|              |        |     | LIBABS .BAS      | 850210 | 3   | STATUS .BAS   | 851226 | 15  | INV .HLP      | 850310 | 3   | MATRIS .BAC                | 860204 | 69  |
|              |        |     | LIBOLD .BAS      | 850417 | 11  | STILLPAR.BAS  | 851025 | 27  | INV .IND      | 850310 | 3   | PROGRAM .INF               | 860221 | 5   |
|              |        |     | LIBRUSTAT.BAS    | 850517 | 11  | SYSTIME .BAS  | 840920 | 7   | LASH .DOC     | 850514 | 19  | README .SER                | 860222 | 7   |
|              |        |     | OPTLOAD .BAS     | 860101 | 3   | SYSTIME .BAS  | 840920 | 11  | LASH .HEX     | 850514 | 63  | REKLAM .TXT                | 851217 | 3   |
|              |        |     | OPTLOAD .INF     | 860101 | 22  | TESTDR .BAS   | 851019 | 11  | MAILLIST.DOC  | 850310 | 91  | REKLAM2.BAS                | 851217 | 32  |
|              |        |     | OPTROSH .REL     | 851010 | 15  | TGB99 .BAS    | 860222 | 20  | MAINT .BAS    | 850310 | 55  | REKLAM3.BAS                | 851217 | 33  |
|              |        |     | OPTROSL .SYS     | 851010 | 15  | TIME .806     | 840920 | 7   | ONEND2 .ALG   | 850310 | 7   | RUNDIDAC.BAS               | 851221 | 29  |
|              |        |     | PROTALL .802     | 850517 | 11  | TIME .BAS     | 850806 | 7   | PERM .ALG     | 850310 | 7   | SCHACKVL.BAS               | 860204 | 7   |
|              |        |     | PROTALL .BAS     | 850517 | 11  | TYTPRINT.BAS  | 850111 | 7   | PRINT .ASM    | 850514 | 159 | SERV16 .ITH                | 860222 | 79  |
|              |        |     | SUPER .BAS       | 850417 | 19  | VARLIST .BAS  | 840920 | 23  | READWO .ALG   | 850310 | 11  | SERVANT .16K               | 860222 | 3   |
|              |        |     | UPD .BAS         | 850210 | 3   | VARLIST .INF  | 840920 | 3   | REPORT .BAS   | 850310 | 55  | SERVANT .2K                | 860222 | 37  |
|              |        |     | ABC800/TIPS      |        |     | VECKA .BAS    | 851019 | 11  | REFILE .INP   | 850310 | 3   | SERVANT .32K               | 860222 | 38  |
|              |        |     | =====            |        |     | VÄLJEN .BAS   | 860208 | 15  | RUNALG .HEX   | 850310 | 159 | SERVANT .INF               | 860222 | 47  |
|              |        |     | V65 .BAS         | 840920 | 51  | WFL .BAS      | 850427 | 7   | RUNERR .TXT   | 850310 | 11  | SERVANT2.16K               | 860222 | 3   |
|              |        |     | ABC800/UTILITY   |        |     | X80080 .BAS   | 850306 | 7   | RWINT .ALG    | 850310 | 7   | SERVERR .SYS               | 860222 | 9   |
|              |        |     | =====            |        |     | ATERTEXT.BAS  | 840920 | 11  | SHOW .ASM     | 850514 | 51  | SIDSPARK.BAS               | 860203 | 12  |
|              |        |     | ABCTIBM .BAS     | 850725 | 19  | ABC800/UTLÅDA |        |     | SHOW .DOC     | 850514 | 3   | SNABBKOP.BAS               | 860221 | 13  |
|              |        |     | ABCTIBM .INF     | 850725 | 3   | =====         |        |     | SHOW .HEX     | 850514 | 11  | STEREO .INF                | 851219 | 14  |
|              |        |     | ASSEMBLE.BAS     | 850917 | 103 | ABCTIBM .BAS  | 860222 | 9   | SISTR .ALG    | 850310 | 3   | SUMMA .BAC                 | 860222 | 15  |
|              |        |     | ASSEMBLE.DOC     | 850917 | 11  | ADVB800 .HEX  | 851019 | 79  | SSOURCE.DOC   | 850427 | 103 | VIEWDATA.BAS               | 860221 | 24  |
|              |        |     | BASCOMP .BAS     | 850202 | 11  | ASCBAB .BAS   | 860121 | 55  | STRING .ALG   | 850310 | 7   |                            |        |     |
|              |        |     | BASIC12.BAS      | 841127 | 15  | ASCBAB .INF   | 860121 | 7   | STRST .ALG    | 850310 | 3   | INLÅDA/ABC800              |        |     |
|              |        |     | BASIC5 .BAS      | 840920 | 27  | ASCBAB .BAS   | 851029 | 27  | SUB .ASM      | 850514 | 111 | =====                      |        |     |
|              |        |     | BASIC5 .REM      | 840920 | 11  | CONFIC .806   | 851027 | 11  | SUPSUB .ASM   | 850514 | 83  | FÄRKOMB.BAS                | 860221 | 11  |
|              |        |     | BASIC6 .BAS      | 840920 | 27  | FÄRGA2.BAS    | 851223 | 13  | SUPSUB .DOC   | 850514 | 19  | FÄRGNR1 .BAC               | 860221 | 3   |
|              |        |     | BASIC6 .REM      | 840920 | 11  | GENEPRAT2.BAS | 851223 | 13  | UNERA .ASM    | 850514 | 55  | FÄRGTES1.BAC               | 860221 | 7   |
|              |        |     | BASICFIL.BAS     | 850127 | 27  | KERMIT .BAS   | 851223 | 47  | UNERA .HEX    | 850514 | 15  | FÄRGTES2.BAC               | 860221 | 7   |
|              |        |     | CHAINARE.BAC     | 850725 | 3   | KERMIT .TXT   | 851223 | 47  | UNLOAD .ASM   | 850514 | 15  | FÄRGTES3.BAC               | 860221 | 7   |
|              |        |     | CHAINARE.INF     | 850725 | 3   | LISTSKYD.BAS  | 850912 | 12  | UNLOAD .HEX   | 850514 | 15  | FÄRGTES4.BAC               | 860221 | 7   |
|              |        |     | CLUSTER .BAS     | 860121 | 6   | MUS99 .TXT    | 860222 | 4   | UNSPOL .ASM   | 850514 | 55  | LOADPTC .BAC               | 860221 | 7   |
|              |        |     | CONCAT .BAS      | 850417 | 11  | PLATTA99.TXT  | 860104 | 19  | UNSPOL .DOC   | 850514 | 87  | LOADPTC .BAC               | 860221 | 11  |
|              |        |     | CONFIG .UPD      | 851027 | 15  | SVTSKAL1.BAS  | 851223 | 12  | USMAN .TXT    | 850310 | 91  | TAXI .BAS                  | 860203 | 11  |
|              |        |     | COPYDISK.BAS     | 840920 | 7   | SVTSKAL2.BAS  | 851223 | 12  | WFILE .OUT    | 850310 | 3   | TAXKALK .BAS               | 860203 | 69  |
|              |        |     | COPYFAST.BAS     | 840920 | 15  | SVARSKAL.BAS  | 851223 | 12  | WINPUT .INP   | 850310 | 3   | TAXKALK .INF               | 860203 | 17  |
|              |        |     | CPMCPY .BAS      | 841120 | 43  |               |        |     | WZSORC .ASM   | 850514 | 207 | TAXKALK .MEN               | 860203 | 8   |
|              |        |     | CSTEST .BAS      | 841120 | 7   | ABCTENIX      |        |     |               |        |     | TAXKALK .REM               | 860203 | 3   |
|              |        |     | DIR .BAS         | 851019 | 11  | =====         |        |     |               |        |     | TAXKALK1.BAS               | 860203 | 51  |
|              |        |     | DIRINF .TXT      | 851028 | 11  | UPDMKFS .BAS  | 851026 | 15  |               |        |     | TAXKALK2.BAS               | 860203 | 38  |
|              |        |     | DIRLIB .BAS      | 840920 | 11  |               |        |     |               |        |     | TAXMENY .BAS               | 860203 | 21  |
|              |        |     | DISASM .BAS      | 840920 | 91  | CPM           |        |     |               |        |     | TED .BAS                   | 860218 | 26  |
|              |        |     | DISASM .TXT      | 840920 | 15  | =====         |        |     |               |        |     | TED .INF                   | 860218 | 15  |
|              |        |     | DISASM2.BAS      | 850424 | 71  | QZDIR .TXT    | 850430 | 95  |               |        |     | TEDTERM .BAS               | 860218 | 7   |
|              |        |     | DISKREG .BAS     | 841120 | 47  | READ .ME      | 850105 | 91  |               |        |     | TEDTERM .INF               | 860218 | 13  |
|              |        |     | DISKREG .INF     | 841120 | 7   | CPM/C         |        |     |               |        |     | TEST128K.32                | 860221 | 7   |
|              |        |     | DISKSTAT.BAS     | 840920 | 31  | =====         |        |     |               |        |     | TEST128K.32K               | 860221 | 11  |
|              |        |     | DISKTRAN.802     | 850709 | 19  | RM .C         | 850516 | 3   |               |        |     | TEST128K.BAC               | 860221 | 7   |
|              |        |     | DISKTRAN.INF     | 850709 | 3   | RMF .C        | 850516 | 7   |               |        |     | TEST128K.BAS               | 860221 | 11  |
|              |        |     | DOS800 .INF      | 860124 | 5   | SSED .C       | 850516 | 35  |               |        |     | TESTCMOS.BAS               | 860221 | 11  |
|              |        |     | ERRTXT .BAS      | 860124 | 6   | CPM/CPMUG     |        |     |               |        |     | TESTHRO2.BAS               | 860221 | 7   |
|              |        |     | EXTBASIC.BAS     | 850903 | 11  | =====         |        |     |               |        |     | TESTMAP .BAS               | 860221 | 11  |
|              |        |     | FALSKUSU.EEZ     | 850428 | 15  | CATALOG .28   | 850310 | 15  |               |        |     |                            |        |     |
|              |        |     | FILSTAT .BAS     | 840920 | 27  | CMG78090.CAT  | 850124 | 107 |               |        |     | INLÅDA/EJKLARA             |        |     |
|              |        |     | FILSTAT3.BAS     | 860208 | 38  | SIG1050 .CAT  | 850124 | 183 |               |        |     | =====                      |        |     |
|              |        |     | FILSUM .BAS      | 840920 | 7   | SIG10507.CAT  | 850124 | 227 |               |        |     | BLIB .BAS                  | 851223 | 18  |
|              |        |     | FILTOWEX.BAS     | 850709 | 11  |               |        |     |               |        |     | DEPMUSIK .INF              | 851229 | 6   |
|              |        |     | FORMKOLL.BAS     | 851026 | 7   | CPM/KOMMUNIK  |        |     |               |        |     | KALEND .BAS                | 860205 | 26  |
|              |        |     | FORMKOLL.INF     | 850725 | 7   | =====         |        |     |               |        |     | MASTER .806                | 851027 | 11  |
|              |        |     | FÄLTOMP .BAS     | 851226 | 31  | APXND .ASM    | 850514 | 119 |               |        |     | RESCUE .BAS                | 860126 | 32  |
|              |        |     | GRAFDUMP.EPS     | 841127 | 7   | CPKRK .DOC    | 850514 | 199 |               |        |     | SCHACK .TXT                | 851207 | 7   |
|              |        |     | HARDCOPY.BAS     | 840920 | 7   | KERMIT .HEX   | 850516 | 155 |               |        |     | SEGELTUR.FEL               | 860126 | 16  |
|              |        |     | HETOFIL.BAS      | 850918 | 11  | MOD800 .HEX   | 850709 | 83  |               |        |     | SEGELTUR.HEX               | 851223 | 35  |
|              |        |     | HUVUD .BAS       | 851015 | 27  | MOD800 .INF   | 850709 | 7   |               |        |     | SEGLING .BAC               | 860126 | 18  |
|              |        |     | INDKOMP .BAS     | 851226 | 23  | MODEM7 .ASM   | 850528 | 191 |               |        |     | SPINDEL .806               | 8602   |     |

|                |        |    |                |        |    |                |        |    |               |        |    |               |        |     |
|----------------|--------|----|----------------|--------|----|----------------|--------|----|---------------|--------|----|---------------|--------|-----|
| KASTAOM .BAS   | 840920 | 11 | MUSIK .BAS     | 840920 | 11 | SKYTTE .BAS    | 840920 | 11 | MEMORY .ABS   | 840920 | 3  | SCHACK6 .Z80  | 851206 | 23  |
| KILLER .BAS    | 840920 | 7  | OPCODE1 .TXT   | 840920 | 11 | SOLITAR .BAS   | 840920 | 23 | PARASCAN .BER | 840920 | 11 | SCHACK80 .SRP | 851206 | 15  |
| KONVERT .BAS   | 840920 | 7  | OPCODE2 .TXT   | 840920 | 7  | VÄRD .BAS      | 840920 | 15 | TICOTICO .BAC | 840920 | 11 | SORTERA .BAS  | 851206 | 27  |
| LOGIN .BAS     | 840920 | 19 | OPLIST .BAC    | 840920 | 3  | VÄRD16 .BAS    | 840920 | 15 | TIPS .TXT     | 840920 | 19 | SPIDER .BAS   | 851206 | 31  |
| MAKETEXT .BAS  | 840920 | 3  | RADJUST .BAS   | 840920 | 11 |                |        |    | UBAT1281 .GAM | 840920 | 27 | SWAPER .UTL   | 851206 | 15  |
| MEMDUMP .BAS   | 840920 | 31 | REGCASPR .BAS  | 840920 | 35 |                |        |    | VISA .UTL     | 840920 | 11 | USERS .SYS    | 851206 | 3   |
| MYSCO .BAS     | 840920 | 11 | TV80 .BAC      | 840920 | 11 | KAS/K8         |        |    |               |        |    | VISA .BAS     | 851216 | 3   |
| SLOOP .BAS     | 840920 | 23 | TVCAS .BAC     | 840920 | 3  | ASSTRANG .BAC  | 840920 | 7  | KAS/K13       |        |    | VÄXLAPGM .UTL | 851206 | 11  |
| SORTCORE .BAS  | 840920 | 7  | TVEDIT .REM    | 840920 | 23 | ASSTRANH .BAC  | 840920 | 11 | =====         |        |    |               |        |     |
| VIRRE .BAS     | 840920 | 35 | TVMAIN2 .BAC   | 840920 | 39 | ASSTRANL .BAC  | 840920 | 11 | CASDEF .802   | 840920 | 7  | KAS/K17       |        |     |
| VISA .BAS      | 840920 | 3  | TVSUB .BAC     | 840920 | 19 | BYECHAIN .BAS  | 840920 | 11 | CASDISK3 .BAS | 841002 | 15 | =====         |        |     |
|                |        |    | UBAT .BAS      | 840920 | 31 | CAS80 .BAS     | 840920 | 11 | CAT .DSK      | 840920 | 15 | BATRESAN .BAS | 851219 | 27  |
| KAS/K00        |        |    |                |        |    | CASDISK2 .BAS  | 840920 | 15 | CONFIG .UDF   | 840920 | 15 | CASDISK2 .800 | 851219 | 35  |
| =====          |        |    | KAS/K4         |        |    | ERRORSIS .BAS  | 840920 | 15 | DXGUIDE .CW   | 840920 | 31 | CASDISK4 .BAC | 851219 | 15  |
| ABCMINI .BAS   | 840920 | 7  | =====          |        |    | GENESIS .BAS   | 840920 | 35 | ELIZA .800    | 840920 | 39 | CHAINARE .BAC | 851219 | 3   |
| ABCTRANS .BAS  | 840920 | 15 | ABCTRANS .80K  | 840920 | 19 | GODJUL2 .BAS   | 840920 | 11 | FEMKAMP .GAM  | 840920 | 51 | CHAINARE .INF | 851219 | 7   |
| BASTOCAS .BAS  | 840920 | 11 | CASDISK .BAS   | 840920 | 11 | GOLV .BAS      | 840920 | 3  | FLAGGA .BAC   | 840920 | 19 | FASTCAS .16K  | 851219 | 19  |
| CASDISK .BAC   | 840920 | 11 | DELETE2 .ASM   | 840920 | 7  | GOLVIAK .ASM   | 840920 | 15 | GETSKRIV .REM | 840920 | 7  | FASTCAS .32K  | 851219 | 23  |
| CASMINI .BAS   | 840920 | 11 | DOSCREEN .BAS  | 840920 | 15 | HJÄLPARE .REM  | 840920 | 15 | GETSKRIV .SUB | 840920 | 11 | FLUGORNA .BAS | 851219 | 31  |
| CASODSK .BAS   | 840920 | 11 | FILDIS .BAS    | 840920 | 19 | HLPA16 .BAC    | 840920 | 15 | MILJONÄR .800 | 840920 | 27 | FORM .BAS     | 851219 | 27  |
| CHECK .BAS     | 840920 | 43 | FILDIS .REM    | 840920 | 3  | HLPA32 .BAC    | 840920 | 15 | PLANET .800   | 840920 | 47 | JULSTAR .BAS  | 851219 | 11  |
| CKOLON .BAS    | 840920 | 11 | FORTH16 .ABS   | 840920 | 51 | HLPB16 .BAC    | 840920 | 15 | REMHODE .UTL  | 840920 | 15 | KETMINI .BAC  | 851219 | 7   |
| CONCAT .BAS    | 840920 | 3  | FVISA .BAS     | 840920 | 3  | HLPB32 .BAC    | 840920 | 15 | SÄSER16K .BAC | 840920 | 55 | KETMINI .INF  | 851219 | 27  |
| COPYTEXT .BAS  | 840920 | 7  | GAPHALS1 .BAS  | 840920 | 43 | INFO .BAS      | 840920 | 7  | TVREPLAC .BAS | 840920 | 7  | LÄNGRAD .BAC  | 851219 | 7   |
| EXTLIB .BAS    | 840920 | 11 | GLIPP .STY     | 840920 | 19 | MASK .BAS      | 840920 | 55 | TVREPLAC .INF | 840920 | 7  | MASTER2 .BAS  | 851219 | 27  |
| EXTRACT .BAS   | 840920 | 7  | GLIPP .XXX     | 840920 | 43 | ROADRACE .BAC  | 840920 | 63 | TVSEARCH .BAS | 840920 | 7  | MINTMAN .BAS  | 851219 | 59  |
| FILCOM .BAS    | 840920 | 7  | GRAFFPRIN .PRT | 840920 | 7  | RPN .BAC       | 840920 | 47 | TVSEARCH .INF | 840920 | 11 | MSINTAN .BAS  | 851219 | 35  |
| JKTST .BAS     | 840920 | 15 | HJÄLPARE .16K  | 840920 | 31 | SCREEN3 .TXT   | 840920 | 43 | TYPE .DSK     | 840920 | 19 | MX80PR .BAS   | 851219 | 11  |
| KILLER .BAS    | 840920 | 7  | HJÄLPARE .32K  | 840920 | 31 | SÄNK .BAC      | 840920 | 3  | VARGPM .DSK   | 840920 | 15 | PANIC .BAC    | 851219 | 7   |
| KONVERT .BAS   | 840920 | 7  | HJÄLPARE .REM  | 840920 | 11 | SÄNKIF .BAC    | 840920 | 3  |               |        |    | PANIC .STY    | 851219 | 23  |
| LOGIN .BAS     | 840920 | 19 | INUTFIL .BAS   | 840920 | 11 |                |        |    | KAS/K14       |        |    | PANIC40 .BAS  | 851219 | 23  |
| MAKETEXT .BAS  | 840920 | 3  | RESETFIX .ASM  | 840920 | 7  |                |        |    | =====         |        |    | PANIC40 .SRP  | 851219 | 23  |
| MODEMKOL .BAS  | 840920 | 7  | RESETFIX .BAS  | 840920 | 7  | KAS/K9         |        |    | CASDISK3 .BAS | 841127 | 15 | PANIC80 .BAS  | 851219 | 23  |
| MYSCO .BAS     | 840920 | 11 | SCREEN .TXT    | 840920 | 51 | =====          |        |    | DISASM .BAS   | 841127 | 91 | PANIC80 .SRP  | 851219 | 23  |
| NYMEMDMP .BAS  | 840920 | 35 | TV .BAS        | 840920 | 11 | BRIDGE .GAM    | 840920 | 23 | DISASM .TXT   | 841127 | 91 | PANIC80 .SHG  | 851219 | 11  |
| SORTCORE .BAS  | 840920 | 7  | TVLIB .MRG     | 840920 | 7  | BUDGET .BER    | 840920 | 7  | OPCODE .DAT   | 841127 | 15 | PANIC80 .FPG  | 851219 | 23  |
| VIRRE .BAS     | 840920 | 35 | TVMAIN3 .BAC   | 840920 | 39 | BUDGETRE .BER  | 840920 | 43 | SETUP .BAS    | 841127 | 51 | RESCUE .BAS   | 851219 | 23  |
| VISA .BAS      | 840920 | 3  | TVSUBR .ABS    | 840920 | 7  | BUDGETVA .BER  | 840920 | 7  | TERM100 .BAC  | 841127 | 39 | RULLER .BAS   | 851219 | 39  |
|                |        |    |                |        |    | CASDISK .800   | 840920 | 39 | TERM100 .DOC  | 841127 | 83 | STALLPAR .BAS | 851219 | 27  |
| KAS/K1         |        |    | KAS/K5         |        |    | CASDISK2 .BAS  | 840920 | 15 | TERM100 .LAT  | 841127 | 23 | VISA .BAS     | 851219 | 3   |
| =====          |        |    | =====          |        |    | DACAR .BER     | 840920 | 11 | TRMHL .TXT    | 841127 | 3  | TEXT/DIV      |        |     |
| ABCMINI .BAS   | 840920 | 7  | BIGTEXT .BAS   | 840920 | 7  | FILTRANS .800  | 840920 | 59 | TRMHLPI .TXT  | 841127 | 3  | =====         |        |     |
| ABCTRANS .BAS  | 840920 | 15 | BIGTXT12 .BAS  | 840920 | 51 | LAYOUT .UTL    | 840920 | 31 | TRMHLPU .TXT  | 841127 | 7  | ABCR800 .PRG  | 850817 | 11  |
| CASMINI .BAS   | 840920 | 11 | BIGTXT57 .BAS  | 840920 | 15 | MASTMIND .800  | 840920 | 19 | VDO .BAC      | 841127 | 3  | ABCR800 .85   | 850410 | 19  |
| CASODSK .BAS   | 840920 | 11 | BIGTXT69 .BAS  | 840920 | 19 | MEMSTAT .UTL   | 840920 | 11 | VDO .REM      | 841127 | 39 | IBMRTPC .TXT  | 860204 | 110 |
| CKOLON .BAS    | 840920 | 11 | CASDISK .BAS   | 840920 | 11 | MX80PR .UTL    | 840920 | 7  | VDO2 .32K     | 841127 | 35 | LOKALADR .TXT | 850705 | 3   |
| CONCAT .BAS    | 840920 | 3  | CASMON .BAS    | 840920 | 15 | RADANNED .UTL  | 840920 | 19 |               |        |    | REAL .DOC     | 850111 | 99  |
| DOP .BAS       | 840920 | 7  | CMOINT .CAS    | 840920 | 7  | RESULTAT .BER  | 840920 | 35 |               |        |    | ZORK .MAP     | 860205 | 45  |
| EURFRAG .BAS   | 840920 | 15 | CRTOS .BAS     | 840920 | 11 | STUDS .GAM     | 840920 | 47 | KAS/K15       |        |    | TEXT/MONITOR  |        |     |
| FILCOM .BAS    | 840920 | 7  | CRTOS .INF     | 840920 | 15 | TV800 .800     | 840920 | 7  | =====         |        |    | =====         |        |     |
| FILSUM .BAS    | 840920 | 3  | DEFENDER .BAS  | 840920 | 19 | TV800 .REM     | 840920 | 15 | BAMBI .BAS    | 850607 | 11 | ABCFIL .MAN   | 851031 | 39  |
| GUBBEN .BAS    | 840920 | 27 | FILTRANS .BAS  | 840920 | 27 | TVMAIN .800    | 840920 | 15 | CALC .80K     | 850607 | 15 | ALLM .MAN     | 851031 | 39  |
| HÄMARK .BAS    | 840920 | 7  | FILTRANS .REM  | 840920 | 7  | TVSUBR800 .ABS | 840920 | 7  | CALC1 .80K    | 850607 | 7  | BYTE .TXT     | 860126 | 26  |
| KONVERT .BAS   | 840920 | 7  | FORTH32 .ABS   | 840920 | 35 | UMSYSTEM .TPS  | 840920 | 23 | CALC3 .80K    | 850607 | 47 | CPM .TXT      | 840920 | 31  |
| KRIG .BAS      | 840920 | 15 | FYLLE .BAS     | 840920 | 15 |                |        |    | CALCEX .CA3   | 850607 | 7  | DATABAS .TXT  | 860216 | 24  |
| KUGLGRAF .BAS  | 840920 | 7  | NOCTRLC .BAS   | 840920 | 7  | KAS/K10        |        |    | CALCOM .DAT   | 850607 | 3  | EDITOR .MAN   | 851031 | 15  |
| LOGAIN .BAS    | 840920 | 23 | NOCTRLC .TXT   | 840920 | 7  | =====          |        |    | CASDISK3 .BAS | 850607 | 15 | FIODEBAS .TXT | 860105 | 14  |
| LUFFAR .BAS    | 840920 | 23 | ROCK .BAS      | 840920 | 11 | BASICERR .SYS  | 840920 | 11 | DEMO3D .DAT   | 850607 | 27 | FILES .BVT    | 860126 | 70  |
| LAW .BAS       | 840920 | 27 | SAFT .BAS      | 840920 | 23 | BESÖK .GRF     | 840920 | 15 | DRAW .INF     | 850607 | 11 | HUVUD .MAN    | 851031 | 7   |
| MAKETEXT .BAS  | 840920 | 3  | SCREEN2 .TXT   | 840920 | 59 | CASDISK .800   | 840920 | 39 | DRAW16K .BAC  | 850607 | 7  | JACKIE .TXT   | 850915 | 19  |
| MATTEM .BAS    | 840920 | 15 | SD10 .BAS      | 840920 | 39 | CASDISK3 .BAS  | 840920 | 15 | DRAW32K .BAC  | 850607 | 11 | KERMIT .TXT   | 850929 | 19  |
| MÄKEN .BAS     | 840920 | 3  | TOM1 .BAS      | 840920 | 3  | COPYFAST .800  | 840920 | 15 | DRAWBANA .BAS | 850607 | 11 | MODEM .TXT    | 850510 | 15  |
| MYSCO .BAS     | 840920 | 11 | TON1 .BAS      | 840920 | 11 | COPYFAST .DSK  | 840920 | 11 | DRAWBOFA .CAS | 850607 | 3  | MODEM2 .PRT   | 840920 | 31  |
| MÖNSTER .BAS   | 840920 | 7  | TON2 .BAS      | 840920 | 11 | DXGUIDE .CW    | 840920 | 11 | DRAWDJUP .BAS | 850607 | 19 | MONITOR .MAN  | 851031 | 55  |
| OHMDATA .BAS   | 840920 | 7  | TONTEXT .BAS   | 840920 | 19 | FÄRGKART .800  | 840920 | 11 | DRAWJUP .BAS  | 850607 | 19 | MSG .MAN      | 851031 | 51  |
| PRIM .BAS      | 840920 | 11 | TVSPEL .BAS    | 840920 | 11 | GÖRABS .DSK    | 840920 | 7  | DRAWSPIR .BAS | 850607 | 51 | VIDEOTEX .KOD | 850922 | 7   |
| REAKTION .BAS  | 840920 | 15 |                |        |    | HARDCOPY .800  | 840920 | 7  | FROG .STY     | 850607 | 7  |               |        |     |
| ROTEN .BAS     | 840920 | 11 | KAS/K6         |        |    | KALENDER .BER  | 840920 | 11 | FROGGER .BAC  | 850607 | 7  |               |        |     |
| SALOM .BAS     | 840920 | 7  | =====          |        |    | LISTSKYD .DSK  | 840920 | 11 | INVASION .BAC | 850607 | 3  |               |        |     |
| SLOOP .BAS     | 840920 | 23 | ABC80 .BAS     | 840920 | 7  | LOTTOS76 .TPS  | 840920 | 31 | INVASION .BAS | 850629 | 7  |               |        |     |
| SORTCORE .BAS  | 840920 | 7  | ASM .BAS       | 840920 | 7  | LÄSDISK .800   | 840920 | 19 | INVFG .ABS    | 850607 | 19 |               |        |     |
| STALTEMP .BAS  | 840920 | 7  | ASM2 .BAS      | 840920 | 47 | PERSNR .BER    | 840920 | 15 | INVFG .BAC    | 850607 | 15 |               |        |     |
| TIME .BAS      | 840920 | 7  | ASMCON .BAS    | 840920 | 51 | REX .BER       | 840920 | 23 | INVLR .ABS    | 850607 | 15 |               |        |     |
| VIRRE .BAS     | 840920 | 35 | BAGGINS .LSP   | 840920 | 7  | STARTREK .GAM  | 840920 | 47 | INVLR .BAC    | 850607 | 19 |               |        |     |
| VISA .BAS      | 840920 | 3  | BAGLOD .LSP    | 840920 | 7  | SYNTIZER .MUS  | 840920 | 19 | INVMONO .ABS  | 850607 | 15 |               |        |     |
|                |        |    | CASCOPY .BAS   | 840920 | 7  | UPPSJÖ .GAM    | 840920 | 39 | INVMONO .BAC  | 850607 | 15 |               |        |     |
| KAS/K2         |        |    | CASCOPY .REM   | 840920 | 7  | UPPSJÖ .INF    | 840920 | 11 | LADDA .SYS    | 850607 | 23 |               |        |     |
| =====          |        |    | CASCOPY1 .BAS  | 840920 | 3  | UPPSJÖ1 .PRO   | 840920 | 3  | SPRITE .BAS   | 850607 | 23 |               |        |     |
| BLINKA .BAS    | 840920 | 7  | CASDISK .BAS   | 840920 | 11 | UPPSJÖ2 .PRO   | 840920 | 3  | SPRITE .INF   | 850607 | 7  |               |        |     |
| CASDISK .BAC   | 840920 | 11 | DOSCREEN .MRG  | 840920 | 3  | UPPSJÖ3 .PRO   | 840920 | 3  | TERM100 .16K  | 850607 | 23 |               |        |     |
| CASLIB .BAS    | 840920 | 11 | EDITF .LSP     | 840920 | 15 | UPPSJÖ4 .PRO   | 840920 | 3  | TERM100B .BAC | 850607 | 15 |               |        |     |
| COPY .BAC      | 840920 | 3  | EMBLEM .PIC    | 840920 | 3  | UPPSJÖ5 .PRO   | 840920 | 3  | TERM100C .BAC | 850607 | 3  |               |        |     |
| COPYLIB .BAS   | 840920 | 15 | FAC .LSP       | 840920 | 3  | UPPSJÖ6 .PRO   | 840920 | 3  | VDO .16K      | 850607 | 7  |               |        |     |
| DECROM .BAS    | 840920 | 7  | GENEALOG .LSP  | 840920 | 67 | YATZY .800     | 840920 | 35 | VDO2 .16K     | 850607 | 27 |               |        |     |
| DIFRFX .BAS    | 840920 | 11 | GRAFEDIT .BAS  | 840920 | 43 |                |        |    | VISA .BAS     | 850607 | 3  |               |        |     |
| DISKST .BAS    | 840920 | 27 | GRAFEDIT .HLP  | 840920 | 15 | KAS/K11        |        |    | VTINIDAT .ITH | 850607 | 3  |               |        |     |
| EXTDEL .BAS    | 840920 | 11 | HUSET .BAS     | 840920 | 7  | =====          |        |    |               |        |    |               |        |     |
| FILOMV .BAS    | 840920 | 15 | LISP .BAS      | 840920 | 27 | BIL16K .GAM    | 840920 | 31 | KAS/K16       |        |    |               |        |     |
| GOMOKU .BAS    | 840920 | 23 | LISP .BAS      | 840920 | 27 | CASDISK3 .BAS  | 840920 | 15 | =====         |        |    |               |        |     |
| GUBBENOVA .BAS | 840920 | 7  | LOAD .BAS      | 840920 | 3  | DATUM .BER     | 840920 | 11 | ASTART .BAS   | 851206 | 15 |               |        |     |
| HINDLOPP .BAS  | 840920 | 11 | MORSEÖVN .BAS  | 840920 | 39 | DCELECT .BER   | 840920 | 15 | CASDISK2 .800 | 851216 | 35 |               |        |     |
| HUVUD .BAS     | 840920 | 11 | PAUSEÖVN .BAS  | 840920 | 27 | DISKSTAT .800  | 840920 | 47 | CASDISK4 .BAC | 851216 | 15 |               |        |     |
| INVASION .BAS  | 840920 | 35 | SÄV .BAS       | 840920 | 7  | GROTTAN .GAM   |        |    |               |        |    |               |        |     |

# ABC-Östs årsberättelse för verksamhets- året 1985

Klubbens styrelse har bestått av:

|                 |                    |
|-----------------|--------------------|
| Ordförande      | Nils Larsson       |
| Vice ordförande | Lennart Holmström  |
| Sekreterare     | Lars Källner       |
| Kassör          | Tommy Drotz        |
| Ledamöter       | Mats Larsson       |
|                 | Christer Lindehoff |
|                 | Heikki Lehto       |
| Suppleanter     | John Jacobsson     |
|                 | Kari Laaksonen     |

Klubben har under året haft livlig verksamhet med flera studieverksamhet och demonstrationer.

Vi har besökt:

SVT i Norrköping och studerat datorisering i TV:s verksamhet

DATA & KONTORSKONSULT AB, ABC-programmet med LUX-net, CAT-net, ABC1600 och ABC9000

NORRKÖPINGS DATORBUTIK AB där vi fick studera CAD/CAM på ABC-datorer med praktiska exempel från företag

LINKÖPINGS DATACENTRAL med rundvandring samt information om vad centralen användes till. Kvällen avslutades med diskussioner och egna övningar under sakkunnet ledning

STRÅLFORS AB filial i Linköping där vi fick en intressant information om företaget samt en rundvandring på lagret och affären.

Vi har även haft flera välbesökta föredrag och demonstrationer

ARADO AB som visade sina diskettstationer bl a RD 312 2.4 Mbyte och winchester RD 302 5 - 30 Mbyte

MATS LARSSON, användning av diskarean som RAM-minne

COMPORIAN AB som visade TOOLBOX och COMP-REGINA

## Rabatter:

ABC-Öst har för ABC-klubben förhandlat fram rabatter hos följande företag mot uppvisande av giltigt medlemskort:

DATA & KONTORSKONSULT AB i Linköping 10 %

MATERIAL CLEARING AB i Norrköping 10 %

Styrelsen vill passa på tillfället att framföra klubbens tack till alla företag och personer som så bredvilligt ställt upp för klubbens verksamhet.

Vi vill också tacka alla våra medlemmar för stort intresse och mycket välbesökta möten

|                    |                   |
|--------------------|-------------------|
| Lars Källner       | Nils Larsson      |
| sekr               | ordf              |
| Christer Lindehoff | Lennart Holmström |
| Tommy Drotz        | Heikki Letho      |
| Mats Larsson       |                   |
| Johan Jacobsson    | Kari Laaksonen    |

## ABC ÖST

**Från årsmötet 1986 kan förtälas**  
Styrelsen omvaldes för 1986. Det innebär att har du några frågor kan du höra av dig till

|                  |                   |
|------------------|-------------------|
| <b>Ordf</b>      | 2Nils Larsson     |
|                  | 013-13 45 65      |
| <b>Vice ordf</b> | Lennart Holmström |
|                  | 0141-711 05       |
| <b>Sekr</b>      | Lars Källner      |
|                  | 013-662 97        |
| <b>Kassör</b>    | Tommy Drotz       |
|                  | 013-15 15 36      |

Vi har möte onsdagar 5 mars, 9 april, 14 maj och 4 juni kl 19.30 i Ånestadgården i Linköping om inget annat meddelas. ABC-Östs medlemmar meddelas genom kallelse. Olika program, aktiviteter och företagsbesök planeras.

För ABC-Öst/Lars Källner

## ABC STOCKHOLM

Här kommer lite information om vår verksamhet:

### Tisdagskvällar (öppet för alla!)

Varje tisdag har vi öppet hus i klubblokalen Vidängsvägen 1 Alvik. Tider 18.30 - ca. 22.00. Här finns några datorer, lite böcker och verktyg, samt oftast en hel del folk att prata med eller fråga om du har några problem.

### Informationsmöten

Vi kommer att ha ett informationsmöte till under våren. För mötet gäller klubb- rum A i Medborgarhuset i Alvik. Börjar 19.00, fika till självkostnads pris. OBS! ABC-Stockholmare fritt, övriga 10 kr.

Program som följer:

Onsdag 14/5 86 Ämne: Kryptering

### Studiebesök

Något eller några studiebesök brukar det bli, tex QZ, Postel eller Metric.

### Studiecirkel

Vi hoppas få igång en studiecirkel i modembygge kanske i höst. Vill du vara med så skriv en rad till oss med namn och adress så hör vi av oss.

### Årsmöte

Hölls i år 12 mars. Sammandrag kommer senare.

### VIKTIGT:

Vi hör väldigt lite om vad ni vill att vi ska göra så har du några ideer så hör av dig till någon i styrelsen. Tack på förhand.

Det var allt för den här gången från

Styrelsen i ABC-Stockholm

## ABC STOCKHOLM

### REFERAT FRÅN 1986-ÅRS STORA DATORSNACK

ABC-Stockholm inbjöd alla intresserade till en paneldebatt under mottot:

DATAPROGRAM - SKYDD, SÄKERHET, HANTERBARHET

Dabatten ägde rum den 20 januari 1986. Plats: Medborgarhuset i Alvik.

Mötesledare var Kjell Stensson och debattörerna var Herbert Söderström, Lennart Grobgeld, Roland Halvorsen, Gunilla Forsell, Walter Holmer och Gert Persson.

Kjell Stensson hälsade både den talrika församlingen på ca 100 personer och debattörerna välkomna. Debatten började med att kvällens gäster presenterade sig.

Roland Halvorsen är jurist och fd sekreterare i Justitiedepartementets upphovsrättsutredning. RH berättade om utredningen och talade om att den tillsattes år 1976. Utredningen är parlamentariskt tillsatt och består dessutom av ca 25 experter. Utredningen har inte enbart ägnat sig åt datafrågor utan även åt upphovsrättsfrågor gällande foto i undervisning, skärpta regler gällande piratkopiering och uthyrning av grammofonskivor. År 1985 kom det av utredningens förslag som väckt det största intresset, nämligen utredningens förslag SOU 1985:51 som behandlar upphovsrättsfrågorna vad gäller data.

Utredningen hade att ta ställning till hur dataprogram skall skyddas och vad upphovsrätt och dataprogram hade gemensamt. Utredningen kom bland annat att jämföra dataprogram och litterära verk.

Lennard Grobgeld som är sekreterare i Justitiedepartementet berättade om en statlig utredning som ägnat sig åt frågor som rör skydd för företagshemligheter. På grund av modern teknik har det varit nödvändigt att ägna uppmärksamhet åt dessa frågor. Sverige har sedan 1931 haft en lag om illojal konkurrens. Denna lag kan tillämpas när det gäller skydd av företagshemlighet - yrkeshemlighet. Men är ett dataprogram en yrkeshemlighet? Vi saknar en definition av yrkeshemlighet. 1931-års lag skyddar inte formen på samma sätt som lagen om upphovsrätt, men ideer och intellektuella tankar måste skyddas. Ett företag är skyddat så länge en anställd är formellt anställd, sedan blir det knivigare. En anställd får inte ta med sig dokumenterat material hur som helst men hur ska man betrakta det som man sas har med sig i minnet. 1931-års lag kan förlängas med allmän straffrätt och dessutom med 1980-års sekretesslag. Frågan är nu, hur klassificera ett dataprogram?

**Gert Persson** är teknisk direktör vid Data-inspektionen. GP gav sin syn på datautvecklingen enligt följande:  
1950-talet handlade om teknik och 1954 byggdes Besk (binär elektronisk sekvenskalkylator).  
1960-talet var systembyggarnas tid.  
1970-talet handlade om förvirring och miss- tro mot 60-talet och småskalighet.  
1980-talet är smådatorernas tid.

Smådatorerna har vänt upp och ned på allting. Inget är sig likt. Nu handlar det mest om integritet, kontroll av medborgare, 100 000-tals personregister och demografiska databaser.

**Herbert Söderström** talade om Sverige som ett Öppenhetssamhälle. Redan på 1600-talet instiftade Karl XI en kanslistadga som påbjöd att ett ex. av alla tryckta böcker skulle finnas på biblioteket. Härifrån stammar begreppet pliktexemplar. Polhelm byggde inte för att tjäna pengar på hemligheter. I England däremot skyddades personlig integritet av domstol. Sverige har haft en generös attityd men om man så vill kan man nu se anglosachsiska attacker mot vår öppenhet.

**Walter Holmer** är fd datasäkerhetschef i Stockholms läns landsting, numera konsult i ADB-frågor. WH visade en bild som belyste hur viktigt det är att saker hamnar i rätt sammanhang. Om man har olika register för hinkar-spaddblad-handtag och skaft gäller det att sakerna hamnar på rätt plats när de olika registren samkörs. WH redogjorde även för vad som krävs för att bygga upp ett datasystem. Datorer är mycket värdefulla i sjukvården tex i intensivvård, men utan kunnig personal är de inte till så stor nytta för patienterna.

**Gunilla Forsell** är civ.ing och patentombud. GF berättade mycket om hur hårdvara och mjukvara vad gäller data hittills behandlats i patentsammanhang. Hårdvara har hittills varit lättare att ta ställning till. Nu söks patent alltmer på mjukvara och patent kan även sökas på ideer till dataprogram.

Debatten som sedan följde kom mycket att handla om vilkas intresse som vilken lag skyddade. Herbert Söderström hävdade att upphovsrättsutredningens förslag får till följd att tex vilken skolelev som helst som kopierar ett program för sitt eget behov gör en kriminell handling och kan hamna inför svensk domstol. Roland Halvorsen sade att program som krävt stora utvecklingskostnader måste skyddas. Herbert Söderström hävdade att program som är offentliga ska kunna spridas.

Juridiken är uppbyggd på ett materiellt samhälle, dvs våra regelsystem är bättre anpassade till materia än till det informationssamhälle vi har idag.

En annan fråga som väckte publikens intresse var hur överföringen av ideer och kunskap mellan företag och tex universitet a la Silicon valley kunde komma att påverkas av de olika lagarna.

Efter fyra intressanta timmar avslutade Kjell Stensson 1986-års stora datasnack.

Vid tangentbordet  
<1500>  
Marianne Forsman

(483628) 86-02-02 18.12 /86 rader/ Bernt Andersson STOCC (SMOJMN - OP/SESAM-1)  
Mottagare: Förteckning (över Svenska) Databaser  
Kommentar till: (Text 483363)  
Markerad av 9 personer.  
Ärende: Nr 3 / 1986 Årgång 2

\*\*\*\*\*  
\* FÖRTECKNING ÖVER SVENSKA DATABASER 1986-02-02 \*  
\*\*\*\*\*

| TYP  | FÖRKLARING                        | NOT | FÖRKLARING                      |
|------|-----------------------------------|-----|---------------------------------|
| ABC  | Körs med ABC-datorer              | *   | öppen dygnet runt               |
| AMIS | Körs med ATARI-datorer            | .   | öppen endast helger             |
| AMXL | Körs med ATARI-datorer            | &   | öppen kvällar och helger        |
| BBS  | Mycket enkel bas (anslagstavla)   | !   | öppen endast kvällar.           |
| C-64 | Körs med CBM-64 datorer           | 1   | 300/300 baud endast             |
| CLUB | Främst avsedd för medlemmar       | 2   | 75/1200 baud endast             |
| CGBS | CBM-64-bas som körs med C/G BBS   | 3   | 75/1200 baud videotex           |
|      | Kan endast köras med C/G TERM!    | 4   | 1200/1200 baud endast           |
| FIDO | Ingår i FIDO-net (MS/DOS)         | 5   | 300/300 + 75/1200 baud          |
| KOM  | Textsystem med flera möten        | 6   | 300/300 + 1200/1200 baud        |
| REKL | Bas med reklam för visst företag  | 7   | 300/300+75/1200+1200/1200 baud. |
| SPEL | Bas att främst spela spel i       | --> | Tillfälligt stängd!             |
| TLX  | INFO-bas med tillgång till telex. | ?   | Uppgift saknas.                 |
| VTEX | Bas av Videotex/Datavisions-typ   |     |                                 |
| INFO | Bas för informationssökning.      |     |                                 |
| ???? | Uppgift saknas.                   |     |                                 |

| RIKT-NUMMER | TYP.    | NAMN                     | NOT   | RIKT-NUMMER | TYP.    | NAMN                   | NOT    |
|-------------|---------|--------------------------|-------|-------------|---------|------------------------|--------|
| 08          | 143500  | KOM Q.Z.(679280 info)    | *23   | 08          | 151375  | INFO DATA-ARKIV        | *2     |
| 08          | 165240  | INFO DATA-ARKIV          | *1    | 08          | 165255  | INFO DATA-ARKIV        | *3     |
| 08          | 165270  | INFO DATA-ARKIV          | *4    | 08          | 194852  | FIDO DJUNGELTRUMMAN    | *1     |
| 08          | 224420  | VTEX DATAVISION EMILIA   | *3    | 08          | 226500  | VTEX DATAVISION OSQUAR | *3     |
| 08          | 228130  | KOM Q.Z.                 | *23   | 08          | 238660  | KOM Q.Z.               | *1     |
| 08          | 246120  | KOM Q.Z.                 | *4    | 08          | 339200  | VTEX ANNONSMARKNADEN   | *23    |
| 08          | 463528  | ???? BUGS DATABAS        | ?1    | 08          | 608091  | INFO DATAFORUM         | *1     |
| 08          | 622190  | TLX SVENSKA TELEDATA     | *7    | 08          | 622480  | TLX SVENSKA TELEDATA   | *7     |
| 08          | 646676  | SPEL PROGGY CRACKERS...  | *7    | 08          | 647922  | ???? VIXEN-DEMO        | ?1     |
| 08          | 801155  | CLUB ABC-MONITOR         | *4    | 08          | 804900  | VTEX POSTEL            | *3     |
| 08          | 806440  | CLUB ABC-MONITOR         | *5    | 08          | 832173  | KOM P.S.               | &4     |
| 08          | 873220  | KOM EPSILON-1            | *1    | 08          | 882235  | ???? D.A.L.            | !1     |
| 08          | 895214  | KOM HOTLINE III          | *1    | 08          | 929432  | CLUB 1:A ATARI-KLUBBEN | *1     |
| 08          | 944376  | KOM W.O.P.R.             | *1    | 08          | 977266  | BBS KONJAK             | !1     |
| 08          | 998851  | ???? AK-4                | ?1    | 08          | 999260  | CLUB CRAY-3/HOTLINE    | *1     |
| 08          | 7127960 | SPEL FASA (adventure)--> | *1    | 08          | 7145260 | VTEX VARMANS DATA      | &3     |
| 08          | 7300706 | ABC ELFA RADIO & TV      | &1    | 08          | 7402041 | BBS TELEBUTIKEN SKÄRH. | *1     |
| 08          | 7506941 | CLUB STOCC/CCS.(C64)---  | *1    | 08          | 7603312 | FIDO MINI-FIDO         | *1     |
| 08          | 7710280 | AMXL THE STRAPPER BASE   | *1    | 08          | 7715662 | KOM PIONEER            | --->*1 |
| 08          | 7740070 | KOM COMPUTEXT            | *5    | 08          | 7772928 | KOM ZEKES (Även spel)  | *1     |
| 011         | 64354   | KOM NORRKÖPINGSMONITOR   | *1    | 011         | 182710  | TLX SVENSKA TELEDATA   | *7     |
| 013         | 104500  | VTEX ANNONSMARKNADEN     | *23   | 013         | 177620  | KOM COMSOFT/LB         | --->*1 |
| 016         | 126717  | KOM ESKILSTUNA-MONITOR   | &1    | 018         | 155640  | VTEX STATSKONTORET SIS | *3     |
| 019         | 149373  | ???? ABC ÖREBRO          | ?1    | 019         | 149762  | C-64 ÖREBRO-MONITORN   | *1     |
| 021         | 135030  | TLX SVENSKA TELEDATA     | *7    | 021         | 352346  | C-64 VÄSTERÅS-MONITORN | *1     |
| 026         | 111377  | ???? OK BILRADIO         | &1    | 026         | 142464  | VTEX BILD & DATA       | ?3     |
| 026         | 184750  | TLX SVENSKA TELEDATA     | *7    | 026         | 189655  | CLUB 1:A ATARI-KLUBBEN | *1     |
| 031         | 112134  | REKL WETTERGRENS BOKH.   | &1    | 031         | 202223  | ???? DRAGONFIRE        | ?1     |
| 031         | 239196  | ABC ABC-GÖTEBORG         | ?1    | 031         | 292160  | TLX SVENSKA TELEDATA   | *7     |
| 031         | 305060  | AMIS S.A.K.              | *1    | 031         | 305733  | BBS GÖTEBORGS-MONITORN | &1     |
| 031         | 488000  | BBS FLEXBASE             | *1    | 031         | 543853  | BBS BREVBOXEN          | &1     |
| 031         | 547585  | ABC ABC-KLUBBEN          | *1    | 031         | 690470  | TLX SVENSKA TELEDATA   | *7     |
| 031         | 690754  | ???? CBBS                | ?1    | 033         | 129110  | C-64 ÖREBRO-MONITORN   | *1     |
| 033         | 132445  | TLX SVENSKA TELEDATA     | *7    | 035         | 110771  | ABC ABC-BANKEN/DISCO91 | *1     |
| 036         | 121650  | TLX SVENSKA TELEDATA     | *7    | 040         | 111420  | TLX SVENSKA TELEDATA   | *7     |
| 040         | 111495  | TLX SVENSKA TELEDATA     | *7    | 040         | 185754  | ???? MPC DATABAS       | ?1     |
| 040         | 946739  | ???? SINTRAN III         | ?1    | 042         | 113010  | TLX SVENSKA TELEDATA   | *7     |
| 046         | 151093  | KOM LUNMAIL              | *1    | 054         | 100343  | FIDO FIDONET           | ?1     |
| 054         | 133170  | FIDO DAY RAINBOW         | *1    | 054         | 158665  | TLX SVENSKA TELEDATA   | *7     |
| 054         | 166988  | FIDO SUNCITY FIDO        | *1    | 054         | 189675  | FIDO FIDONET           | ?1     |
| 060         | ??????  | C-64 SESAM-2             | --->1 | 060         | 171520  | TLX SVENSKA TELEDATA   | *7     |
| 0457        | 15467   | ???? AMU-CENTER          | ?1    | 0470        | 40680   | TLX SVENSKA TELEDATA   | *7     |
| 0480        | 86060   | TLX SVENSKA TELEDATA     | *7    | 0491        | 13050   | TLX SVENSKA TELEDATA   | *7     |
| 0510        | 20409   | FIDO A.T.L.              | *1    | 0521        | 60689   | CGBS GDF/CCS (CBM-64)  | *1     |
| 0586        | 57058   | ???? ÄPPLER              | *1    | 0586        | 57400   | TLX SVENSKA TELEDATA   | *7     |
| 0620        | 11012   | KOM ABC-SOLLEFTEA        | *1    | 0660        | 82575   | ABC ABC ÖRNSKÖLDSESVIK | ?1     |
| 0750        | 10162   | KOM ESSDATA              | *1    | 0750        | 31282   | KOM TUNGELSTA-BASEN    | *1     |
| 0753        | 36050   | C-64 S E S A M - 1       | *1    | 0755        | 14407   | CLUB S.D.F.            | *1     |
| 0755        | 14440   | CLUB S.D.F.              | *1    | 0760        | 14365   | KOM APPLE-MON          | *1     |
| 0760        | 36034   | KOM UPPTAGETBASEN        | *1    | 0760        | 93957   | AMIS T.U.G.            | *1     |
| 0762        | 10677   | KOM CIRCUS (ATARI)       | *1    | 0766        | 33203   | KOM G.A.M.             | *1     |
| 0910        | 59191   | SPEL RATHOLE (adv.)      | *1    | 0910        | 78228   | KOM NYBASEN            | *1     |
| 0920        | 48351   | ???? T.L.B.              | &1    |             |         |                        |        |

\*\*\*\*\*  
\* Alla bidrag till listan Är välkomna!! \*  
\*\*\*\*\*

# DIV:

## BASIC II/PC

Hargot Lindmark och Ulf Sandberg från DIAB avsåg på årsmötet att BASIC II till MSDOS (IBM PC och datorer kompatibla med IBM PC) beräknas bli klar den 1 april. Det är alltså samma BASIC som finns till ABC800-serien.

BASIC:en skall bli så kompatibel som möjligt med BASIC II. Vill man föra över filer från en ABC-maskin måste man göra det via V24-kommunikation. DIAB efterlyser folk som har erfarenhet av att under MSDOS läsa ABC DOS skivor och skriva dessa filer på standard IBM PC skivor.

I en första etapp kommer inte grafik och ISAM med, utan detta kommer senare. Kostnad för BASIC II/PC uppgavs bli 1950 SEK exkl moms. 950 SEK exkl moms kommer Squez, ISAM och grafiken att kosta enligt andra källor.

## Nya produkter

Dessutom meddelade DIAB att de inte arbetar med någon MSDOS maskin, utan de fortsätter att vidareutveckla maskiner som baserar sig på operativsystemet UNIX (D-Nix). Nya UNIX-baserade produkter kommer ut under året från DIAB. Nokia-Luxor kunde inte svara på om Nokia-Luxor skall sälja DIAB:s nya UNIX-produkter.

Nokia-Luxor avvisar också nya produkter under året. Vad det är för produkter avslöjades inte.

<1789>

Bo Kullmar

## ISAM tips

När man använder ISAM så har en del råkat ut för att indexfilen blir felaktig. Detta beror på att indexfilen och datafilen uppdateras var för sig eftersom det är två filer. Man får alltså inte rycka ur någon skiva från disken utan att CLOSE har gjorts för då riskerar man att indexfilen blir felaktig.

Som bot mot detta finns det reparationer i både Meny0 och REG800. Dessutom finns det ett program i programbanken som Kristoffer Eriksson har gjort som heter ISAMCHECK. Med detta program kan du ta fram information om en ISAM-databas, för att sedan på annat sätt rätta till felen.

<1789>

Bo Kullmar

# RADANNONS

Säljes.

ABC806 med 512k RAM floppy, ABC815, ABC77, ABC890 och 3 x 640kB diskdrive med nya LUXOR Kontrollerkortet. Manualer, servicemanual, scheman, Bit för bit med ABC800, Bygg ut ABC800 med DataBoard plus annan litteratur - ratur Pris: 34.500:-. Dessutom ingår 30 talet disketter med diverse program, bl.a ORD800.

# Angående lagerhanteringsprogram i ABC-bladet nr 4, 1985

Det finns anledning att nämna det förhållandet att P-data har gjort flera olika varianter av Läs & Skriv-rutinerna. Bland annat finns det varianter för ABC80, ABC800 och FACIT DTC. Det finns varianter som suportar utbyggt minne och varianter som klarar både BASREGISTER 800 och MULTIBAS. Enligt P-data kan man använda Läs & Skriv för ABC800 till FACIT DTC, men inte Läs & Skriv för FACIT DTC till ABC800. Mitt Lagerprogram använder den första varianten av Läs & Skriv till ABC800 (tror jag)?

Jag har fått en del frågor angående mitt lagerhanteringsprogram och frågeställarna klagade på att vissa radnummer som jag använder inte fanns i P-datas Läs & Skrivrutiner.

Vid närmare granskning fann jag att det stämde.

Det visade sig att jag modifierat Läs & Skrivrutinerna lite för att de skulle passa bättre till mina program.

Vissa rader i Läs & Skriv skall ersättas med nedanstående rader.

(Dessa är avsedda för ABC800 ock liknande maskiner.)

```

11 POKE -749%,248% : Z%=CALL(63337%,32%
) : POKE -749%,245% : IF Z% : 'DATA
SKIVAN ÄR INTE OK' : STOP
12 REM
16 GOSUB 56000
17 REM
18 REM
20 GOSUB 100 : GOSUB 310 : GOSUB 56100
22 REM
110 REM
111 GOSUB 190
155 Y%=(A%(B%) AND 32767%) : RETURN
181 IF Z% Y%=Z% : Z%=0% : RETURN
190 POKE 63494%,A% : Ø%=CALL(63271%)
192 M%=PEEK2(Ø%+10%) : N%=PEEK2(Ø%+12%)
: Q%=PEEK(Ø%+20%)
194 F%=PEEK2(Ø%+14%) : G%=PEEK2(Ø%+16%)
: H%=PEEK(Ø%+18%)
196 RETURN
238 IF Å%>31% AND Å%<128% AND LEN(I$)<L$
: Å$ SPACE$(L%-LEN(I$)) STRING$(L%-
LEN(I$),8%); : I$=I$+Å$ ELSE : CHR$(
7%);
239 IF LEN(I$)=1% : SPACE$(L%-LEN(I$))
STRING$(L%-LEN(I$),8%);
241 GOTO 231
305 REM
326 REM
328 REM
330 REM
340 REM

```

För ABC800-varianten skall dessutom följande rader bytas ut.

I Läs & Skriv-rutinen bytes rad 192 och 194 ut mot dessa:

```

192 M%=PEEK(Ø%+10%)+SWAP%(PEEK(Ø%+11%)):
N%=PEEK(Ø%+12%)+SWAP%(PEEK(Ø%+13%)):
Q%=PEEK(Ø%+20%)
194 F%=PEEK(Ø%+14%)+SWAP%(PEEK(Ø%+15%)):
G%=PEEK(Ø%+16%)+SWAP%(PEEK(Ø%+17%)):
H%=PEEK(Ø%+18%)

```

I programmet HUVMENY bytes rad 502 ut mot följande rad.

```

502 IF PEEK(32768%)+SWAP%(PEEK(32769%))
=32768% GOTO 600

```

Samt raderna 518-520 bytes mot nedanstående rader.

```

518 D$=A3$+M3$+D3$
520 ON ERROR GOTO 528
522 PREPARE "DATUM.DAT" AS FILE 1% : ;
$1$,D$ : CLOSE 1%
524 POKE 32768%,32768%,SWAP%(32768%)
526 GOTO 530
528 : ERRCODE : STOP
530 ON ERROR GOTO

```

Subrutinen 36000-36004 bytes i programmen INLAGER, UTLAGER, LISTA1, LISTA2, LISTA3, LISTA4 och LISTA5 ut mot nedanstående subrutin.

```

36000 IF PEEK(32768%)+SWAP%(PEEK(32769%
))=32768% CHAIN "HUVMENY"
36002 ON ERROR GOTO 36008
36004 OPEN "DATUM.DAT" AS FILE 1% :
INPUT $1$,D$ : CLOSE 1%
36006 GOTO 36010
36008 : ERRCODE : STOP
36010 ON ERROR GOTO
36012 RETURN

```

I programmen HUVMENY, INLAGER, UTLAGER, LISTA1, LISTA2, LISTA3, LISTA4 och LISTA5 ersättes raderna 36900-36910 med följande programrader:

```

36900 Z%=1% : IF LEN(I$)<>6% GOTO 36910
36902 A3$=LEFT$(I$,2%) : IF VAL(A3$)<85%
OR VAL(A3$)>99% GOTO 36910
36904 M3$=MID$(I$,3%,2%) : IF VAL(M3$)<1
% OR VAL(M3$)>12% GOTO 36910
36906 D3$=RIGHT$(I$,5%) : IF VAL(D3$)<1
% OR VAL(D3$)>31% GOTO 36910
36908 Z%=0%
36910 RETURN

```

Med förhoppning att det skall fungera sedan sänder jag mina hälsningar

<1422>

Sten Åsberg  
Firma MODULSERVICE  
Statistgatan 20  
214 84 MALMÖ  
040-133295

Printerbuffert PBU60. Pris: 3.000:-.  
Assembler 800. Pris: 1.000:-. GPIB - IEEE488-  
interface med programvara. Pris: 1.800:-.  
Serieinterface, 8184, till Epson FX och  
RX printrar (med 2k buffert) Pris: 800:-.

Host adapter till winchestercontroller.  
Pris: 1.800:-. Det kortet som sitter mellan  
ABC-bussen och Winchester-controller-kortet  
SA1410 eller SA1410A.

<4528>

Sören Björnman  
031-87 22 97 eller 87 88 97 (bost)  
031-67 15 31 (arb)

# NYHET, ABC-klubbens PUBLIKATIONER PÅ DISKETT!

## PRISLISTA OLIKA DATAFORMAT >>>

|                     |                                                   | .K                 | .E                  | .D                  | .Q                | .8                |
|---------------------|---------------------------------------------------|--------------------|---------------------|---------------------|-------------------|-------------------|
| <b>SN1.</b> (kod)   | Samlingsnummer 1980-81 inkl disk/kassett Nr 1-2   | <b>125:-</b><br>2  | <b>215:-</b><br>4   | <b>185:-</b><br>2   | <b>160:-</b><br>1 | <b>180:-</b><br>1 |
| <b>SN1</b>          | Samlingsnummer 1980-81 endast tidningar.....      |                    |                     |                     |                   | <b>100:-</b>      |
| <b>SN2.</b> (kod)   | Samlingsnummer 1982 inkl diskett/kassett Nr 3-8   | <b>150:-</b><br>6  | <b>420:-</b><br>12  | <b>330:-</b><br>6   | <b>220:-</b><br>2 | <b>205:-</b><br>1 |
| <b>SN2</b>          | Samlingsnummer 1982 endast tidningar.....         |                    |                     |                     |                   | <b>100:-</b>      |
| <b>SN3.</b> (kod)   | Samlingsnummer 1983 inkl diskett/kassett Nr 9-11  | <b>125:-</b><br>3  | <b>260:-</b><br>6   | <b>215:-</b><br>3   | <b>160:-</b><br>1 | <b>180:-</b><br>1 |
| <b>SN3</b>          | Samlingsnummer 1983 endast tidningar.....         |                    |                     |                     |                   | <b>100:-</b>      |
| <b>SN4.</b> (kod)   | Samlingsnummer 1984 inkl diskett/kassett Nr 12-13 | <b>125:-</b><br>2  | <b>215:-</b><br>4   | <b>185:-</b><br>2   | <b>160:-</b><br>1 | <b>180:-</b><br>1 |
| <b>SN4</b>          | Samlingsnummer 1984 endast tidningar.....         |                    |                     |                     |                   | <b>100:-</b>      |
| <b>SN5.</b> (kod)   | Samlingsnummer 1985 inkl disk/kassett nr 14-17    | <b>140:-</b><br>4  | <b>320:-</b><br>8   | <b>260:-</b><br>4   | <b>175:-</b><br>1 | <b>195:-</b><br>1 |
| <b>SN5</b>          | Samlingsnummer 1985 endast tidningar .....        |                    |                     |                     |                   | <b>100:-</b>      |
| <b>SN1-5.</b> (kod) | Samlingsnummer 1980-85 med disk/kassett nr 1-17   | <b>640:-</b><br>17 | <b>1405:-</b><br>34 | <b>1150:-</b><br>17 | <b>885:-</b><br>6 | <b>915:-</b><br>5 |

Siffran under priset avser antal kassetter/disketter som ingår.

## KODER OCH PRISER

**K = Kassett** (Den vanliga ABC-kassetten ingår i årsavgiften).

**E = Enkel density** 40 spår för FD-2, DD-80 m fl (priset avser 2 skivor per kassettnr)..... **45:-**

**D = Dubbel density** 40 spår för FD-2D, DD-82/84, ABC-830 m fl ..... **30:-**

**Q = Quad density** 80 spår per sida, för DD-56, ABC-832 m fl ..... **30:-**

**8 = 8" Stor floppy** 26/256 sekt. för DD-88, ABC-838 m fl ..... **50:-**

Det går att beställa enbart flexskivor för samlingsnummer om Du varit medlem det året.

## RAPPORTER

ABC-Rapport 1, disassemblering ABC-80..... **100:-**

ABC-Rapport 2, inst.manual för ABC-80 Fig-FORTH..... **60:-**

ABC-Rapport 3, Starting FORTH inkl diskett/kassett..... **220:-**

ABC-Rapport 3, Starting FORTH, enbart boken..... **175:-**

ABC-Rapport 3, FORTH 79, enbart programvaran..... **65:-**

Q-ZENTRALEN, inträdesavgift ..... **50:-**

ABC-MONITOR manual ..... **30:-**

PROGRAM KRONSTAT ver 2.1 för ABC-800, 802, 806 .... **100:-**

(Kan levereras efter 1986-01-01)

Kronstat är ett avancerat statistikprogram; se ABC-bl 85/3, 85/4 (går ej på kassett).

**OBS!** Vid beställning av program, ange dator, format (K, E, D, Q el 8).

Beställning sker enklast genom att sätta in rätt belopp på vårt **POSTGIRO 62 93 00 - 5**. Skriv ditt (medlnr), namn och adress på talongen och glöm ej att ange produkt samt vilket dataformat Du önskar.

## PRENUMERERA PÅ ABC-DISKETT!

Du kan nu få programmen på diskett i stället för kassett, i vilket dataformat Du önskar. Så här går det till:

Om Du har en flexskivenhet, t ex DataDisc 82, (SS/DD 5 1/4") kallar vi detta format för "D". Enligt prislistan ovan ser Du att priset för en sådan diskett är 30 kr.

Sätt in beloppet på vårt **POSTGIRO NR 62 93 00 - 5** och ange "för ABC-diskett", så får Du i fortsättningen diskett i stället för kassett.

I priset ingår kopiering, liblista och porto emballage och den kommer direkt hem till Din brevlåda. Följande disketter betalas i efterskott. Vi använder singleside 48 TPI-disketter som är testade och av känt fabrikat.



Kom ihåg att anmäla adressändring i tid

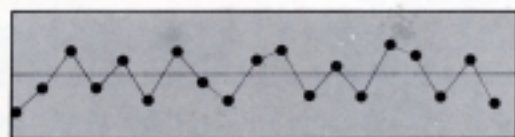


Vad har en  
amerikansk ekonom  
med kvaliteten  
på Nashuas  
disketter att göra?



# ALLT.

För ett antal år sedan utvecklade Dr. W.E. Deming – Amerikansk ekonom med passion för statistik – ett unikt system för kvalitets kontroll baserat på ett antal enkla statistiska regler. Enkelt uttryckt går teorin ut på att få signaler om ev. fel under hela produktionsprocessen, för att på så sätt spåra orsaken, till felen och kunna rätta till dem.



Med andra ord bygger man in en fortlöpande kontrollstruktur i produktionsprocessen varför slutkontroll elimineras och hög jämn kvalitetsnivå erhålles.

Denna statistiska kvalitets kontroll infördes redan 1979 i Nashuas produktion och marknadsavdelning. Resultatet talar tydligt för sig självt både i kvalitet och pris på Nashuas produkter.

Den nya generationen disketter har en utomordentligt hög kvalitet kopplat med en ovärderlig stabilitet. Genom att fel i produktionen undviks kan också priset på produkterna hållas lågt.

DATA 2010

Vill Du veta mera, ta kontakt med oss per telefon eller genom talongen nedan.

## Kupong

Ja, jag vill veta mer om Nashuas disketter.  
☐ Sänd oss information  
☐ Kontakta oss per telefon

Firma: \_\_\_\_\_

Kontaktperson: \_\_\_\_\_

Adress: \_\_\_\_\_

Tel: \_\_\_\_\_

Sändes till TM Data Sverige AB - Box 81073 - 104 61 STOCKHOLM

**TM Data**

Member of Terminal Mart International

TM Data Sverige AB Hammarby Fabriksväg 21A  
S-104 81 STOCKHOLM - Tel. 08-714 07 70

