PROGRAM nittinian BITEN 87-1

GRAM KRACKER

GENEVE

Innehåll

ISSN 0281-1146

Redaktören . . .

Årsmötet
avhölls den 21 februari i Stockholm. På annan plats i tidningen finns verksamhetsberättelse och ekonomisk redovisning för förra året. Ett protokoll från mötet kommer i nästa nummer. Redan nu kan vi väl avslöja att det blev omval på flertalet poster i styrelsen. Sedan förra året kvarstår Jan Alexandersson som ordförande, Mikael Nordlin som kassör och undertecknad som redaktör. Claes Schibler och Börje Häll kvarstår också i styrelsen. Nya i styrelsen blev Eddy Hedlund och Niels Hovmöller. Göran Nygren gör comeback i styrelsen efter ett par års uppehåll för studier. Själv tror jag den nya styrelsen innebär att föreningen står väl rustad inför det nya året. Personsammansättningen borde borga för att intressen hos olika medlemskategorier blir tillgodosedda.

Checklist
Anders Persson presenterar programmet *Checklist* i det här numret. Som han och Lars Thomasson skriver så kan listningar i tidningen i fortsättningen se ut på det sätt som visas. Det vore bra med synpunkter från medlemmar som brukar knappa in program från tidningen. Har Du någon nytta av en checkbokstav som hjälp? Såvitt jag förstått det hela är nya versioner av programmet på väg, bl a för Mini Memory. Hör av dig med Din åsikt!

Tillbehörslista
har Jan Alexandersson sammanställt en. Som framgår av den finns det en hel del att välja på för den som är intresserad av att expandera sin dator. Även på programsidan finns det mycket godbitar. Speciellt roligt är det med alla freeware-program som kommit senaste året. Föreningen har genom att beställa freeware från olika håll fått flera intressanta kontakter. Bland annat har vi utbytt material med några killar i Australien.

För den som är intresserad av att bygga egna tillbehör har det också kommit en hel del matnyttigt på senare tid. Det finns massor med tips som vi får från olika tidningar. Speciellt verkar tyskar och engelsmän vara på hugget när det gäller "hembyggen". Problemet är bara att materialet behöver en del bearbetning innan vi kan trycka det.

Den som vill byta t ex tangentbord eller kontakt i modulporten hittar det han söker hos Digitalservice som annonserar i detta nummer.

Forth
är med igen i detta nummer. En del tycker nog att det är för lite BASIC-program i tidningen. I det här numret har vi ett spel och några grafiska tips. Det vore bra om vi fick mera synpunkter från medlemmarna på tidningen.

Nya program och artiklar har vi alltid plats för. Är Du osäker på om något Du skrivit är något som vi kan publicera går det bra att höra av sig per telefon eller brev direkt till min adress (se nedan).

Slutligen
hoppas jag att alla förnyar medlemskapet i Föreningen Programbiten och därmed försäkrat sig om att tidningen kommer hem i brevlådan i år också. Jag tror mig kunna utlova många intressanta artiklar.

Peter Odelryd
Järnåldersringen 340
136 65 Handen
☎ 08/777 12 16

I redaktionen:

Redaktör	Peter Odelryd
Utmaningsredaktör	Anders Persson
Forth-redaktör	Lars-Erik Svahn
Programförmedlare	Börje Häll
Allt-i-allo	Claes Schibler

Föreningens och redaktionens adress:
Föreningen Programbiten
c/o Schibler
Wahlbergsgatan 6, 1 tr ned
121 46 Johanneshov

Föreningens telefon:
08/761 51 62 tisdagar kl 18–20. Michael Öhman svarar.

Datainspektionens licensnummer: 82100488

Postgiro: 198300-6
Medlemsavgiften för 1987 är 120:–

Annonser, insatta av enskild medlem (ej företag), som gäller försäljning av moduler eller andra tillbehör i enstaka exemplar är gratis.

Övriga annonser kostar 1 000 SEK per helsida, 500 SEK per halvsida. För lösblad som skickas med tidningen gäller 1 000 SEK per blad.

Föreningens tillbehörsförsäljning

Följande finns att köpa för medlemmar genom att motsvarande belopp insätts på postgiro 198300-6.

För TI-99/4A:	
Användartips med Mini Memory	60:–
PB-Forth kassett (föreningens Forth)	100:–
PB-Forth diskett (föreningens Forth)	100:–
TI-Forth (diskett)	50:–
TI-Forth (manual)	150:–
PB-Forth källkod (3 disketter) [1]	120:–
TI-Forth källkod+demo (2 disketter) [2]	120:–
Game-Forth diskett	50:–
Forth&99 (3 disketter) [1]	120:–
Disassembler (diskett)	50:–
Multiplan/TI-Writer uppgraderingar (diskett)	50:–
Super Debugger (diskett)	50:–
Nittinian T-tröja	40:–
99'er magazine nr 12/82, 1–5, 7–9/83 (per styck)	20:–
Programbiten/Nittinian 1985 (TI 59+99)	100:–
Programbiten/Nittinian 1984 (TI 59+99)	80:–
Nittinian, årgång 1983 (TI-99/4A)	60:–
För TI-59 m m	
Programbiten, årgång 1983 (kalkylatorer)	40:–
Programbiten, årgång 1982 (kalkylatorer)	40:–
Programbiten, årgång 1981 (kalkylatorer)	40:–
Programbiten, årgång 1980 (kalkylatorer)	40:–
Programbiten, årgång 1978/79 (kalkylatorer)	40:–
Programbiten, fem årgångar 1978–83	160:–
Programbiten, sex årgångar 1978–84	220:–
Programbiten, sju årgångar 1978–85	300:–
Katalog med belgiska och engelska program för räknare TI-57, TI-58, TI-59	20:–
Föreningens programmeringsblanketter (TI-59), olika typer, block om 50 blanketter (se PB 83-1 sid 30) per block	11:–
Patenthandlingar TI-59	25:–
Tom magnetkortsplånbok	10:–

[1] Säljes endast till den som tidigare köpt PB-Forth
[2] Säljes endast till den som tidigare köpt TI-Forth

FÖRENINGEN PROGRAMBITEN

STYRELSENS ÅRSBERÄTTELSE 1987-02-20

STYRELSENS ÅRSREDOVISNING FÖR 1986

Verksamheten 1986 har bedrivits i form av tidningsutgivning, medlemsmöten, telefonjour och försäljning av program. Antalet betalande medlemmar har minskat från ca 360 till 259 medlemmar.

STYRELSE OCH FUNKTIONÄRER

Föreningen har haft följande styrelse:
Jan Alexandersson, ordförande
Mikael Nordlin, kassör
Peter Odelryd, redaktör
Börje Häll, programförmedlare
Claes Schibler, registerförare och materielförvaltare
Hans Wickström, sekreterare
Anders Törnkvist
Michael Öhman
Tony Rogvall

Funktionärer med särskilda uppgifter har varit Anders Persson, utmanarredaktör, och Lars-Erik Svahn, FORTH-redaktör.

Under året har 7 protokollförda styrelsemöten hållits varav protokoll nummer 2 saknas.

FÖRENINGENS EKONOMI

Föreningens kostnad för tryckning av medlemsbladet har väsentligt reducerats. Detta har främst skett genom att färre exemplar numera trycks, ca 600 ex mot tidigare ca 1000 ex. Vidare har under 1986 vissa betalningar skett, som rätteligen hänför sig till verksamhetsåret 1985. Det är främst fråga om disketter för programförsäljningen. I verksamhetsberättelsen för 1985 (men inte i bokslutet) uppskattades dessa betalningar till ca 14000 kr. En ytterligare minuspost har varit fördyrade portokostnader genom postens beslut. Det minskade medlemsantalet har vidare försämrat årets resultat med ca 12000 kr. Det oaktat har årets underskott stannat vid ca 7000 kr. Föreningens likviditet uppgår till ca 37000 kr. Årets underskott, liksom eventuellt kommande underskott i motsvarande storleksordning, kan därför bäras.

FÖRENINGENS VERKSAMHET

Fem nummer av tidningen Programbiten har givits ut. En riktigt satt text har införts genom att Peter Odelryd har kunnat överföra flexskivor från TI-99/4A till sättmaskin.

Tre allmänna medlemsmöten, varav ett i samband med årsmötet, har hållits. På dessa möten har en komplett TI-99/4A funnits tillgänglig. Nyheter när det gäller program och hårdvara har visats.

Dessutom har 2 medlemsmöten hållits i Staffanstorp.

15 nya FREEWARE på diskett har skaffats från utlandet. Försäljningspriset för programförsäljning till medlemmarna har sänkts från 60 kr till 50 kr per flexskiva. Numera bifogas i förekommande fall tryckt manual.

Samlingsdiskar för 83, 84 och 85 har plockats ihop.

Ett nytt skrivarhjul har skaffats.

SIG-FORTH har fortsatt med 3 utskick.

Föreningens funktionärer har bidragit genom ideellt arbete. Arvoden har, liksom under 1985, inte utgått.

Stockholm som ovan

För styrelsen

Jan Alexandersson, ordförande

Mikael Nordlin, kassör

Ordföranden . . .

Vi har nu som framgår av PB 86-5 skaffat många nya FAIRWARE (FREEWARE) genom direktkontaktakt med programskrivarna i USA, Kanada och Australien. Totalt finns nu 16 FAIRWARE, 3 TI-PROGRAM som släppts till användarföreningar samt 4 avancerade program skrivna av medlemmar i PROGRAMBITEN som liknar FREEWARE när det gäller innehållet. Utöver dessa finns den vanliga programbanken. För att kunna använda FREEWARE-programmen måste du i allmänhet ha 32 kb expansionsminne. Du som har diskdrive men inget expansionsminne bör nog skaffa ett så snart som möjligt.

Namnet FREEWARE kan vara något missvisande eftersom programmen inte är gratis utan du har möjlighet att prova programmet innan du sänder ett bidrag till författaren. Den avgift på 50:- per skiva som du betalar till föreningen är endast avsett att bekosta diskett, kopiering, utprintad manual och porto. Utöver detta bör du sända den summa som författaren begär. Ett vanligt belopp är 5–10 dollar. Du kan betala via postgirot eller via bank men det tillkommer kostnad på c:a 24 kr om du sänder under 100 kr. Ett billigare sätt kan vara att köpa sedlar och sända i brev med flygpost. Samtidigt som du sänder ditt bidrag bör du skriva och komma med synpunkter på hur programmet fungerar. Skriv gärna att du fått programmet via den svenska användarföreningen PROGRAMBITEN. Genom att gynna författaren kommer han att skriva nya förbättrade versioner av programmet och även sända ut information om att nya versioner finns. Vissa författare sänder ut sådana meddelanden till alla som sänder bidrag medan andra endast sänder det till större användarföreningar.

Du vore bra om du som har grunddatorn TI-99/4A med kassettbandspelare hörde av dig och berättade om vad du vill läsa om i tidningen och kom även med andra förslag till aktivitet. Jag har haft vissa tankar på att organisera tätare medlemsmöten i en något mindre lokal kanske en gång per månad. Det har hittils stupat på bristande tid. Om det finns någon medlem som kan ordna sådana möten genom att fixa lokal och köra dit dator och TV så hör av dig till föreningen.

Det har nu även kommit igång medlemsmöten i Göteborg och Staffanstorp. Även du som bor i andra delar av landet kan försöka ordna lokala möten. Du har medlemsmatrikeln sorterad efter postnummer. Använd denna för att kontakta övriga på samma ort.

Det har kommit en ny bärbar kalkylator och dator TI-74 som har en Basic som i stort sett är identisk med TI-99 Extended Basic med hänsyn till att den endast har ett fönster på 31 tecken. Den kostar c:a 1500 kr och har 8 kb RAM med batteri. Vi hoppas kunna komma med mer information om denna i kommande nummer av tidningen.

Själv har jag skaffat en Horizon 192 kb RAM-disk som jag beställde direkt från tillverkaren. Den kom färdigmonterad med flygpost efter en månad. Den har fungerat mycket bra och är c:a 10 gånger snabbare än en fysisk diskdrive.

Jan Alexandersson
Springarvägen 5, 3 tr
142 00 TRÅNGSUND
☎ 08-771 05 69

Annons

Säljes:

TI-99/4A m expansionsbox, RS232, 32k, diskkontrollkort, 1 dubbelsidig drive, speech synt, Extended BASIC, Editor/Assembler, Disk Manager samt 50-tal disketter med program säljes. 4 500:–

Experimentkort för modulporten 20:– + porto

Ett tiotal kvarvarande videotexmoduler säljes för 200:–/st. Michael Öhman, tel 761 52 62

Utmaningen

Redaktörens adress:
Anders Persson
Kämnärsvägen 4: 1078
222 45 Lund

Automatisk avlusning

I nummer 86-3 beskrev jag ett program, kallat BUGOUT, som genererade kontrollbokstäver till varje rad i ett BASIC-program. Problemet var att programmet krävde skivminne för att kunna fungera.

Jag ansåg då att det borde gå att göra en version som klarade sig med X-BASIC och minnesexpansion, alternativt BASIC och Mini Memory. Nu har *Lars Thomasson,* en annan av Skånes stora begåvningar, åstadkommit en variant för Extended BASIC och minnesexpansion.

Då programmet i skrivande stund varken är helt färdigt eller genomtestat, har jag valt att inte visa det den här gången. Eventuellt kommer en beskrivning i nästa nummer, då – förhoppningsvis – en version för Mini Memory också finns klar.

Däremot finns det ett exempel på hur det ser ut när Lars program listat ett annat program. Utseende på listan är alltid ett kärt diskussionsämne. Den som har synpunkter är i vanlig ordning välkommen. Tänk nu på att det finns en viss risk att det kan bli så här i framtiden om inte alltför våldsamma protester hörs. Ta därför chansen att komma med din åsikt redan nu!

Lars har valt en annan metod för genereringen av kontrollbokstäverna. Det blir alltså inte samma resultat som *BUGOUT* ger. Men meningen är ju att Lars program, kallat *CHECKLIST*, ska kunna användas av alla intresserade. Så länge alla har samma går det ju bra.

Ett program av Lars Thomasson

Detta är en kort beskrivning till CHECKLIST programmet.

Checklist är ett program vilket listar X-BASIC program direkt från minnet. Vid listningen genereras en kontrollbokstav för varje rad. Om denna bokstav ej stämmer överens med kontrollbokstaven i en annan lista, gjord av detta program, är det garanterat en skillnad mellan raderna. Tanken är att detta skall hjälpa till vid felsökning av program som operatören själv skrivit in efter en listad förebild.

I syfte att underlätta läsningen av listan skrivs endast en sats ut på varje rad. Då programraden innehåller mer än en sats delas denna rad före varje dubbelkolon. Den tidigare omnämnda kontrollbokstaven utskrives före radnumret till den rad bokstaven hänför sig.

Handhavande

Checklist laddas in i minnet antingen före eller efter det X-BASIC program det skall lista. Inladdningen sker på vanligt sätt, dvs:

CALL INIT
CALL LOAD("DSKn.CHECKLIST")

När både Checklist och X-B programmet finns i minnet är det bara att använda det förstnämnda till att lista det sistnämnda. De för denna operation användbara kommandona presenteras nedan.

CALL LINK("SCREEN")

Detta anger att listningen önskas på skärmen. Varje rad på skärmen upptar maximalt 30 kolumner. Detta för att undvika problem med vissa TV-mottagare, vilka kan vara problematiska att ställa in så att de yttersta kolumnerna blir fullt läsliga. Om en sats är för lång för att skrivas på en skärmrad kommer satsen att delas på erforderligt antal rader. För att möjliggöra läsning av texten avbrytes scrollningen då någon tangent nedtryckes för att återupptas vid nästa tryckning.

CALL LINK("DEVICE","enhetsnamn")

Här anges till vilken utenhet den genererade listningen skall skickas. I händelse av fel vid skrivningen, illegalt filnamn etc, kommer Checklist att returnera ett felmeddelande vilket talar om felets art. Därefter återlämnas kontrollen till X-BASIC systemet. Då den specificerade utenheten är en diskfil kommer denna att öppnas som DIS/VAR 80, vilket innebär att den kan läsas in med TI-WRITER eller EDITOR/ASSEMBLER editorn.

Om detta kommando inte används gäller den utenhet som användes vid förra listningen. Om det är första exekveringen för stunden är standard utenhet PIO.

Om både screen och device kommandona anges gäller det senast inmatade av de två.

CALL LINK("LINEL",radlängd)

Detta kommando användes för att upplysa Checklist om hur långa raderna i utfilen maximalt får vara. Tillåtna värden på radlängden är intervallet fr o m 10 t o m 80. Om en längd som ej uppfyller detta kriterium anges, lämnar Checklist felmeddelandet "BAD ARGUMENT". Då en sats befinnes vara för lång för utskrift på en rad, delas satsen och fortsätter på nästa rad. Anges ingen längd på detta sätt användes samma värde som vid föregående listning, eller 80 tecken om det är första körningen efter inladdningen.

OBS! Om utmatningen är dirigerad till skärmen har detta kommando ingen annan effekt än att ändra standardvärdet. På skärmen skrivs listan alltid med en radlängd om 30 tecken.

CALL LINK("LIST")

Med detta anrop till Checklist startas listningen. Skulle ett fel uppstå under exekveringen, t ex skrivare ej påsatt eller diskett full, kommer Checklist att avbryta sig och ge en felkod vilken beskriver felet.

CALL LINK("LISTP",start,stopp)

I de fall endast en del av XB-programmet önskas listat användes detta kommando. Start och stopp anger mellan vilka radnummer listning ska ske. Eventuella fel i inmatningen av radnumren kommer att upptäckas av Checklist och belönas med passande felmeddelande. Start- och stoppradnumren behöver inte ingå i programmet.

Checklist listar den del som börjar med radnumret vilket är lika med eller närmast större än start och slutar med radnumret som är lika med eller närmast mindre än stopp. Felhanteringen är identisk med den som gäller för kommandot list enligt ovan.

Sammanfattningsvis gäller följande:

"List" och "listp" listar ett X-BASIC program från minnet.

"Device" och "linel" anger vilken utfil respektive radlängd som ska användas.

Då utmatningen önskas på skärmen användes i stället "screen".

Vid kallstart (första körningen efter inladdningen) av Checklist är standard utfil PIO med max 80 tecken per rad. Vid varmstart gäller i stället de värden vilka användes vid förra exekveringen.

Programmet nedan hör egentligen till artikeln *"Frågor och svar"*. För att Du ska få en uppfattning om hur ett program listat med *Checklist* kan se ut visar vi det här.

Fråga: Hur får man sprites att röra sig i en cirkel?

Man kan använda en SIN-COS-funktion. Pröva följande lilla programsnutt för att få en uppfattning av vad som kan göras:

```
S 100 CALL CLEAR
O 110 CALL SPRITE(#1,42,2,96,128)
O 120 FOR I=0 TO 360 STEP 5
V 130 CALL LOCATE(#1,96+30*SIN(I*PI/180),
      128+30*COS(I*PI/180))
Q 140 NEXT I
E 150 GOTO 120
```

Körningen stoppas med CLEAR (FCTN-4)

GRAM KRACKER™

En recension av Jan Alexandersson

Gram kracker från Millers Graphics är en stor modul som innehåller GRAM och RAM i stället för GROM och ROM. Med hjälp av denna är det möjligt att lagra moduler på disk eller kassett och sedan ladda tillbaka till *Gram kracker*.

Det följer med en manual på 55 sidor och en "MILK"-disk med diverse användbara tillägg. Den kostar USD 189 (+ porto, moms och tull) och har funnits hos postorderfirmor i USA (jag köpte från *Tenex*) eller direkt från MG. Enligt en artikel i *Micropendium* december 1986 har MG lagt ned tillverkningen.

Disponering av minnet

Den innehåller 80 kbytes CMOS-RAM med batteri back-up uppdelat på 64 kb GRAM och 16 kb RAM. Det finns även 8 kb internt GROM. De olika bankarna som finns på samma minnesadresser aktiveras med switchar på fronten. RAM-bank 1 och 2 kan även kopplas om med intern bankswitchning. När bankswitchningen är aktiverad är GRAM 0, GRAM 3–7 och RAM-BANK 1–2 skrivskyddade.

CPU-adresser

>2A30 – >3EFF Memory Editor som laddas till EM
>6000 – >7FFF Bank 1(8k) och Bank 2(8k)

GROM/GRAM-adresser

>0000 GRAM 0(8k)	operativsystem(6k)	
>2000 GRAM 1(8k)	Basic-tolk(12k)	Loader(8k)
>4000 GRAM 2(8k)	– " –	
>6000 GRAM 3(8k)		
>8000 GRAM 4(8k)		
>A000 GRAM 5(8k)		
>C000 GRAM 6(8k)		
>E000 GRAM 7(8k)		

Dumpa och ladda moduler

Du placerar din programmodul i en extra modulport som sitter ovanpå *Gram krackern*. Modulen kommer att stå upp lodrätt. Du kan sedan dumpa modulen till kassett eller disk. Det krävs inget extra utöver grunddatorn. Även moduler som använder bankswitchning som Extended Basic och Atari-spel kan dumpas ut.

Den modul du dumpar ut kommer att lagras på flera filer i PROGRAM-format med en fil för varje 6k eller 8k-bank. Extended Basic som är en av de största modulerna kommer att lagras på 6 filer i följande ordning:

RAM-BANK 2
RAM-BANK 1
GRAM 6
GRAM 5
GRAM 4
GRAM 3

Du anger endast ett filnamn tex DSK1.XB eller CS1 och sedan dumpas alla bankar som finns i modulen. *Gram krackern* ger sedan automatiskt olika namn till samhörande filer. Detta gäller både disk och kassett. *Obs* ange inte CS1.X utan endast CS1 eftersom datorn själv stegar fram filnamnet.

Editor/Assembler eller TI-Writer ger bara en fil för GRAM 3. Du laddar in alla samhörande filer genom att ange ett enda filnamn tex DSK1.MODULEN eller CS1. Samma laddare kan även ladda in assemblerprogram till expansionsminnet från kassett eller disk.

Modulporten kan även användas när du vill köra en modul direkt med ROM och GROM. GRAM 3–7 och RAM-BANK 1–2 kopplas automatiskt bort när en modul stoppas in. GRAM 0–2 kan dock användas. Man behöver således aldrig ta bort *Gram krackern*.

Det är även möjligt att dumpa ut GROM 0–2 som sitter i grunddatorn och ladda in något på dessa GRAM-adresser om du har expansionsminne 32k.

GROM 0 operativsystemet

GROM 0 innehåller operativsystemet och genom att lägga över detta i GRAM (om du har 32 kb EM) kan man sedan gå in och ändra. Jag har i min egen modifiering lagt in äkta små bokstäver och svenska tecken så att alla moduler som körs använder dessa. Dessutom har jag lagt in svensk text i samtliga ledtexter för kassettrutinerna. Jag har även gått in och ändrat texten på titelskärmen med färgbalkarna och litet till. Du kan alltid gå tillbaka till det inbyggda GROM 0 med en switch på fronten.

GROM 1–2 TI BASIC

GROM 1–2 innehåller BASIC-tolken men används *ej* av Extended Basic. Det är därför möjligt att lägga in något annat på dessa GRAM-adresser (med 32 kb EM), tex TI-Writer och Editor/Assembler. Du får plats med båda eftersom de ej tar mer än en GRAM-bank. Om du sedan behöver BASIC kan du aktivera GROM 1–2 med switch på fronten.

Edit Memory (med 32 kb EM)

En mycket användbar rutin är en avancerad minneseditor som kräver 32k EM för att fungera. Den liknar mycket editorn i *Explorer*. Du kan editera GRAM, CPU-RAM och VDP-RAM med HEX, ASCII eller ASCII med basic offset.

Det finns även möjlighet att söka HEX- eller text-strängar. När sökningen stannar kan man gå in och editera.

En annan funktion är MOVE som kan flytta stora block mellan olika minnen. Jag har provat att flytta nästan 16 kbytes från HEX A000–DAF2 i expansionsminnet till GRAM 1–2. *Disk Manager 1000* kunde sedan köras direkt från menyn efter färgbalkarna. Det kräver ytterligare POKE i GRAM som fanns beskrivet i *Micropendium* sept 1986.

Med editorn kan du även dumpa minne till skrivare eller disk i HEX, ASCII eller ASCII med offset.

Explorer patch (MILK-disk)

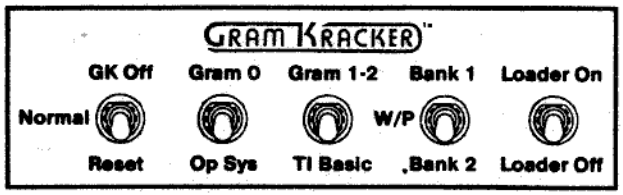
Det finns ett program på MILK-disken som modifierar *MG Explorer* så att äkta GRAM kan editeras och ej PSEUDO GRAM som originaldisken använder. Motsvarande problem finns i TI-Debugger som måste ändras om den skall skriva till GRAM.

GRAM-disk EA/TW (MILK-disk)

Det finns ett program som lagrar EDIT1 och ASSM1–2-filerna från EDITOR/ASSEMBLER i GRAM i stället för på disk. Alternativt kan EDITA1–2 och FORMA 1–2 från TI-Writer lagras i GRAM. Laddningen av dessa filer till expansionsminnet går sedan mycket fortare än via disk.

Flytta GRAM EA/TW (MILK-disk)

E/A-MOVER och TIW-MOVER gör det möjligt att flytta TI-Writer och Editor/Assembler från normalt GRAM 3 till valfri GRAM. Dessutom måste FORMA1-filen ändras med en sektor-editor så att byte 34–35 ändras till >1011 eftersom den annars kommer att läsa GRAM3. Jag har valt att lägga TI-Writer på GRAM 2 och Editor/Assembler på GRAM 7. Det finns dock programskrivare som anropar



GROM 3 direkt och då fungerar inte dessa flyttade moduler. Denna typ av programmering är dock osund men finns på några FAIRWARE (FREEWARE) som föreningen har. Program där detta förekommer är SMALL C version 1.3 och MENU LOADER på RAPID SCROLL-skivan.

För TI-Writer kan det även vara bra att ta bort valet av olika språk på menyn. Jag ändrade g6006 till >60CB vilket gör att endast valet av engelska finns kvar. Jag har på detta sätt möjlighet att ha fyra moduler tillgängliga på menyn som kommer efter färgbalkarna:

1 GRAM KRACKER
2 TI WRITER
3 EXTENDED BASIC
4 EDITOR/ASSEMBLER

Med en switch på fronten kan jag aktivera TI-BASIC i stället för *Gram kracker* och TI-WRITER.

Nya CALL för XB (MILK-disk)

XB-CALLS ger möjlighet att lägga in nya kommandon i Extended Basic. Följande finns:

CALL NEW	NEW även i program
CALL BYE	BYE även i program
CALL CLSALL	stänger alla filer
CALL CLOCK	startar interruptdriven klocka (går.fel p g a skillnaden mellan 50 och 60 Hz)
CALL CLKOFF	stänger av klockan
CALL CAT	visar diskkatalog på skärmen

Köra Basic i GRAM (MILK-disk)

Du kan välja att initiera GRAM 7 eller GRAM 3–7 för lagring och körning av BASIC-program (ej Extended Basic). Flera BASIC-program kan lagras samtidigt (även med bara GRAM 7) och kan sedan väljas på startmenyn. Programmet körs i GRAM så hela VDP-RAM blir ledigt för lagring av variabler. TI har gjort flera moduler på detta sätt tex Personal Record Keeping som är skriven i BASIC.

För den som är sugen på spel kan jag föreslå följande meny:

1 TI BASIC
2 CAR WARS
3 TI-WRITER
4 EDITOR/ASSEMBLER
5 BASIC-PROGRAM-A
6 BASIC-PROGRAM-B
7 BASIC-PROGRAM-C
8 MOON PATROL

Du kan då ha fyra moduler och flera egna BASIC-program tillgängliga. Med switch på fronten kan du även koppla in GRAM 1–2 och då ha tillgång till DISK MANAGER 1000 i stället för BASIC.

Ändra teckenset (MILK-disk)

NEWCHARS laddar in teckenset i GRAM 0 om det först har lagrats i en CHARA1-fil. Observera att tecknen endast får vara 7 pixel höga så alla CHARA1-filer kan ej användas. Detta betyder att äkta underslängar ej kan användas. CHARA1-filen ingår *inte* i disken till *Gram kracker* vilket den borde ha gjort. Det blev litet letande innan jag hittade någon fil som fungerade.

Maximem och GRAM-karte m m

Det finns även två andra moduler att köpa som kan användas på liknande sätt som *Gram krackern*.

Maximem från Kanada har i sin första version som jag provat 48 kb RAM utan batteri uppdelat på 8 kb BANK 1–2 och 6 kb GRAM 3–6 och 8 kb GRAM 7. Det är tillräckligt med 6 kb per GRAM-bank eftersom alla TI moduler har 6 kb. Mechatronic Extended Basic II Plus har dock 8 kb för GRAM 7 vilket således *Maximem* klarar. Det finns en förbättrad 56 kb-version av *Maximem* som har batteri och 8 kb GRAM 3–7.

Maximem levereras ej komplett så du kan inte dumpa till kassett men väl ladda in från kassett om du fått hjälp med dumpen av någon som har diskdrive. Du kan inte heller dumpa till disk utan extra LOAD/RESET-knappar eller Navarone *Widget*. Om du använder *Widget* måste du ha Editor/Assembler-modulen eftersom *Widget* ej kan blockera bort *Maximem* om den stoppas i *Widget* trots att *Maximem* har Editor/Assembler inbyggt i GROM.

Maximem har bara en switch som endast behöver flyttas en gång vid laddningen. *Gram krackern* är något krångligare när det gäller att manövrera switcharna vid laddningen.

Mechatronic *Gram-karte* (Gram-card) är ett kretskort som stoppas in i Expansionsboxen och körs helt utan modul i datorns modulport. *Gram karte* har 128 kb RAM utan batteri uppdelat på 64 kb GRAM + 64 kb CPU-RAM. Den senare används som flera RAM-bankar som kopplas in med bank-switchning.

Gram-karte kan även använda RAM-minnet som GRAM om man använder basadress >9820. Det finns då möjlighet att ha två moduler samtidigt på samma GRAM-adress men med olika basadresser för GROM, nämligen CPU-adress >9800 och >9820. Det finns dock den inskränkningen att endast en av modulerna får ha RAM >6000–>7FFF och då kan inte GRAM 7 användas på modul nr 2. Det finns även en minneseditor och möjlighet att köra Basic i GRAM 3–4.

Det finns ytterligare en sak från Tyskland som som kallas *GRAM-Loader-Modul*. Den fungerar ungefär som *GRAM-Karte* men sätts i modulporten i stället för expansionsboxen.

Module Emulator är en amerikansk modul som kan dumpa ut många moduler men klarar inte Extended Basic och Personal Record Keeping.

Gram Kracker med RAM-disk

Tillsammans med *Horizons* 192 kb RAM-disk som jag skaffat får man en mycket bra kombination. Datorn får 320 kb RAM varav 272 kb med batteri back-up. Jag har ändrat i GRAM så att alla modulerna söker på RAM-disken i stället för DSK1. Det är bara att gå in i SEARCH-funktionen i *Gram krackern* och leta upp var det står DSK1. Jag ändrade enligt följande:

Extended Basic	g6352 DSK3.LOAD
TI-Writer	g65A7 DSK3.
	g6760 DSK3.CHARA1EDITA1FORMA1
Editor/Assembler	g661E DSK3.EDIT1ASSM1UTIL1

Övriga ställen med DSK1 för TI-Writer ändrade jag inte eftersom dessa har med laddning av UTIL1-filer att göra. För Editor/Assembler finns endast ett val och det betyder att du ändrar disk för samtliga filer som modulen söker d v s EDIT1, ASSM1–2 och UTIL1.

Lagringsformat

Den information som måste finnas tillgänlig för laddaren förutom den data som skall hamna i GRAM och RAM är:
- minnesadress där man börjar skriva
- hur många bytes skall skrivas
- vilka ytterligare filer skall laddas till andra bankar.

Gram Kracker lagrar alltid varje 6 kb eller 8 kb-bank som en 34 sektorer lång fil i programformat. De första 6 byten är information till laddaren om:
- flera filer följer eller ej
- antalet bytes
- minnesadress

Den första filen har ett namn med högst 10 bokstäver för disk. Alla efterföljande filer får namnet med tillägg av en siffra. Detta skiljer sig något från vanliga assemblerfiler som t ex lagras som UTIL1, UTIL2 och UTIL3. Motsvaran-

laddas finns i en separat fil på disken. Denna fil har även en hel meny om alla moduler som finns på disken. Vid kassettladdning saknas detta helt och du måste själv komma ihåg till vilken bank som laddningen skall göras och hur många filer som finns.

Gram-karte ger en 34 sektorer lång fil för en 8 kb-bank och har de 6 första byten med information till laddaren om hur många bytes och till vilken minnesadress dock *ej* samma värden som för *Gram kracker*. Vilka filer som är samhörande lagras i en separat fil på disken.

Jag har provat att göra om *Maximem*-filer till *Gram kracker* med ett kort assemblerprogram utan problem.

Referenser

1. Micropendium maj -86: Gram kracker review
2. Micropendium sept -86: Adding DM1000 to Gram kracker
3. Programbiten 85-4.03: Maximem
4. Programbiten 86-3.16: Maximem
5. Micropendium JUN 86: Maximem review
6. Micropendium sept -86: Gram Card review
7. Micropendium dec -86: GK Utility I and GRAM Packer
8. Micropendium dec -86: MG to halt making GRAM Kracker

Rita cirklar med matematiska funktioner

av Jens Hovmöller

Denna artikel handlar om hur man gör mönster och cirklar med vanlig och Extended BASIC. Programidéerna är kanske inte så originella men visar hur korta programmen kan vara.

Cirkel i TI BASIC

```
100 CALL CLEAR
110 REM CIRKEL I VANLIG BASIC
120 J=J+1
130 CALL HCHAR(SIN(J)*11+12,COS(J)*11+16,42)
140 GOTO 120
```

SIN i Extended BASIC

```
100 RANDOMIZE
110 CALL CHAR(42,"3C7EFFFFFFFF7E3C")
120 CALL MAGNIFY(2)
130 CALL SCREEN(2)
140 CALL CLEAR
150 V=INT(RND*10)+10
160 H=INT(RND*10)+10
170 I=I+1
180 IF I=29 THEN I=1
190 CALL SPRITE(#I,42,INT(RND*14)+3,92+SIN
    (J*V)*50,128+SIN (J*H)*100)
200 J=J+0.1
210 GOTO 130
```

Med SIN-funktionen kan man lätt göra fina och mjuka svängar genom att lägga tex en sprite på en position som ökar eller minskar med hjälp av sinusfunktionen. Här ovan finns ett sådant litet program.

Obs! Radnummer 100 tom 130 kan slopas utan större förändring. Om Du ej har tillgång till Extended BASIC kan Du skriva av det program som står här nedan och få ett liknande resultat.

SIN i TI BASIC

```
100 CALL CLEAR
110 V=V+2
120 H=H+1
130 CALL HCHAR(12+SIN(V)*10,16+SIN(H)*12,46)
140 GOTO 110
```

CALL DISTANCE i Extended BASIC

Kommandot CALL DISTANCE lägger in avståndet mellan två angivna sprites eller en punkt på skärmen och en sprite. På detta sätt kan man göra cirklar som inte ser ut som sneda fyrkanter. Du kan ändra på rad nummer 220 och få andra cirklar. Prova tex D=D/2 eller D=D/90. Raderna 100 och 110 kan tas bort, även rad nr 120 kan slopas förutsatt att Du ändrar rad nr 170 till:

170 CALL COLOR(I,I+1,I+1)

Experimentera själv med nedanstående program, det är faktiskt mycket kul!

Detta program liknar inte det ovanstående på så vis att här blir det en figur och inte bara en "orm" som slingrar sig med mjuka svängar. Nej, här ritas en figur upp och stannar kvar på skärmen. Du kan också ändra figurens utseende genom att ändra raderna 110 och 120 på tex följande sätt:

V=V+1, H=H+2, V=V+2, H=H+3, V=V+3, H=H+2, V=3, H=1, V=1, H=3 osv.

```
100 REM GRAFIK MED CALL DISTANCE
110 REM AV JENS HOVMÖLLER 860406
120 DATA 16,15,12,11,10,9,7,14,8,6,4,13,3
130 CALL CLEAR
140 CALL SCREEN(2)
150 CALL SPRITE(#1,32,1,92,124)
160 FOR I=2 TO 14 :: READ F
170 CALL COLOR(I,F,F)
180 NEXT I
190 FOR V=1 TO 24
200 FOR H=1 TO 32
210 CALL SPRITE(#2,32,1,V*8-7,H*8-7)
220 CALL DISTANCE(#1,#2,D)
230 D=D/50
240 CALL HCHAR(V,H,D)
250 NEXT H :: NEXT V
260 GOTO 260
```

Frågor och svar

av Börje Häll

Fråga: Hur gör man för att kunna räkna med grader i stället för radianer när man använder trigonometriska funktioner i 99'an?

Man får räkna om radianerna till grader med hjälp av följande formel:

Grader=radianer*180/PI
och för att göra om grader till radianer:
Radianer=grader*PI/180

Detta gäller Extended BASIC där PI finns. I TI BASIC finns inte PI men följande kan användas som en god aproximation:

4*ATN(1)

vilket kan användas i en användardefinition:
DEF PI=4*ATN(1)

Fråga: Varför bör man inte ha programfiler och datafiler på samma kassett?

På kassett sparas inte filer med namn. Om man har en datafil framför en programfil kan datafilen skriva över programfilen om man läser in mer data. Om datafilen kommer efter programfilen kan samma sak hända om man modifierar programmet så att det blir större. Dessutom kan det vara svårt att hitta datafilen om kassettspelaren inte har ett bra räkneverk. Detta gäller inte bara för programfiler visavi datafiler. När man har flera filer på samma kasett bör man alltså alltid lämna tillräckligt mycket plats mellan olika filer. Detta för att man ska kunna göra filerna större utan att riskera att skriva över filen, som kommer efter på kassetten. Det här gäller inte för flexskivor där filerna lagras med namn.

Geneve (i Skåne inte i Schweiz)

av Anders Persson

Den 14:e februari var det världspremiär i hela Staffanstorp för Myarcs nya dator 9640, eller Geneve som den också kallas. Sören Bernle hade investerat massor av både tid och pengar för att få hit en demonstrationsmodell till mötet. Han har visserligen köpt en Geneve, men någon serieproducerad modell fanns inte tillgänglig då. Det hängde på håret att den kom fram och igång i tid, men vi lyckades till slut.

På mötet körde vi några demoprogram som följde med datorn. Främst var det grafiken som visades. Jämfört med 99:an är det onekligen skillnad. Ett grafikprogram med mus fanns också. Dessutom hade Sören provat att köra några program som han tidigare gjort till 99:an. Tanken är ju att Geneve ska klara av i stort sett alla program som går på 4A. Om det stämmer återstår att se. Den dator vi hade var ju ett förserieexemplar, vilket både hård- och mjukvara bar synliga spår av. Allt fungerade inte som det var tänkt, med andra ord.

All vår början bliver svår . . .

Det här visade sig redan när vi skulle försöka få igång den överhuvudtaget. Bland annat blev det nödvändigt att skaffa fram en professionell RGB-monitor som även klarade 60 Hz. Lou Phillips på Myarc påstod visserligen att det skulle gå med 50 Hz också, men datorn ville inte rätta sig efter det. Dessutom vägrade den fungera ihop med något annat än Myarcs eget diskkontrollkort, trots att det står något annat i annonserna. Eventuellt går det med Texas eget, men det hade vi inte tillgängligt. CorComps ville den i alla fall inte acceptera.

Förhoppningsvis kommer de här problemen att vara kurerade när "riktiga" exemplar börjar komma hit. Speciellt om Myarc tänker fortsätta med den snorkiga attityd mot kunderna som vi (läs Sören) stötte på nu. Deras inställning tycks vara: "Vi vet bäst, men alla andra är att betrakta som minst halvidioter tills vi blir överbevisade om motsatsen."

Ett exempel: efter våra inledande försök att ansluta Geneven till RGB-ingången på Lennart Thelanders monitor, vilket inte gav någon vettig bild, ringde Sören till Lou Phillips för att förhöra sig om det verkligen var riktigt att det skulle gå med 50 Hz. Det ville sig nämligen inte, sa Sören, trots att han tagit två kvalificerade tekniker till hjälp. Det skulle vara jag och Lennart, det. Hmm . . . Phillips menade att "teknikerna" måtte vara några riktiga klåpare, eftersom det hade fungerat hur bra som helst när han var i Tyskland och demonstrerade. Där använde han en tysk monitor. I och med att tyskarna använder 50 Hz var det därmed bevisat att det gick. Sören ställde då den i sammanhanget intelligenta frågan: "Är du säker på att den inte ställde om sig automatiskt då?" Tystnad i andra sidan av tråden.

Någon dag senare, när det visade sig att det gick bra med en 60 Hz-monitor, ringde Sören upp igen och meddelade detta. Phillips kommentar: "Uhum."

Tekniken då?

En Geneve ser inte mycket ut för världen. Det är ett kort som sätts i expansionsboxen. Med en kabel anslutes tangentbordet, vilket ser mer imponerande ut, åtminstone jämfört med det som sitter i en 99:a. Det är ett vanligt PC-bord, med separat numerisk del.

För att datorn ska fungera i boxen måste det anpassningskort som normalt sitter där tas ut. Även eventuell minnesexpansion och p-kodskort ska plockas bort innan Geneven tas i drift. Däremot kan RS232- och diskkontrollkort vara kvar. Eller skulle kunnat. Varken dator eller diskkontrollkort från Myarc visade sig fungera så som reklamens utlovade hundraprocentiga kompatibilitet antyder. Som jag redan nämnt fungerade Geneven bara tillsammans med Myarcs kontrollkort, vilket i sin tur inte fungerar som det borde tillsammans med 99:an.

Det visade sig nämligen att inget av de två kontrollkort som kom fungerar med Pascal. Ett av korten fungerar för övrigt inte som det ska annars heller, även om det nog kan skrivas på kontot för oturligheter. Men just för Pascal är det mycket värdefullt att man kan lagra dubbelt så mycket på disketterna, eftersom det finns så mycket program på diskett till Pascal. De kontrollkort som vi har tillgång till är båda av QUAD-typ, vilket innebär dubbel densitet och även 80 spår, förutsatt att drivverket klarar av det. Så långt är det alltså rätt. Men av någon anledning vägrar det att starta på avsett sätt. Varför vet jag inte.

Teoretiskt sett innebär varken dubbel densitet eller 80 spår några problem för Pascal. Själv har jag använt CorComps kort, med 360K byte per diskett, i ett och ett halvt år utan problem.

Varning för Myarc om du vill köra Pascal på 99:an alltså!

Enligt de senaste uppgifterna från Myarc är de medvetna om att Pascal inte är kompatibelt med deras diskkontrollkort. De säger sig ha löst problemet genom en ändring i operativsystemet för Pascal så att det ska gå. Ännu återstår dock att se om detta stämmer.

Geneve bit för bit

Låt oss nu titta på vad som egentligen finns inuti datorn. Siffrorna nedan motsvaras av kretsarnas placering enligt bilderna.

1. Mikroprocessorn som används är TMS 9995. Det är en efterträdare till TMS 9900, som sitter i 4A. Trots att 9995 bara har en åttabitars databuss (9900 har sexton) är den snabbare än 9900. Det beror dock inte på att klockfrekvensen skulle vara högre i 9995, något som felaktigt påståtts i mer än en artikel som jag har läst. Båda processorerna arbetar med 3 Mhz. Det är förvisso fullt riktigt att 9995 skall ha en kristall på 12 Mhz, men den frekvensen delas ner till 3 Mhz av processorn. Om man räknar på det viset utnyttjar 9900 en grundfrekvens på 48 Mhz, vilken sedan, med hjälp av klockkretsen TIM 9904, delas ner till en fyrfasig klocka på 3 Mhz. Så var det med det.

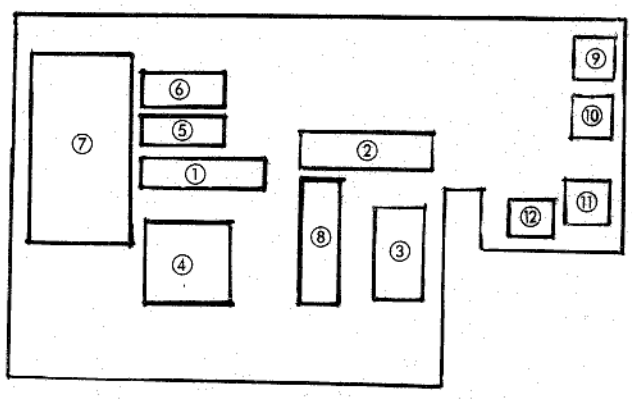
Skillnaden i snabbhet beror i stället på att 9995 inte behöver lika många klockcykler på sig för att göra samma sak. Detta beror i sin tur på flera olika saker.

För det första har 9995 256 bytes RAM inuti chipet. Maximal snabbhet kan man uppnå endast genom att både program och data finns i detta minne. Internt utnyttjas 16 bitars bredd på dataledningen.

För det andra utnyttjar 9995 pipelining i ett steg. Vitsen med det är att databussen blir jämnare utnyttjad. Normalt varierar behovet från outnyttjad till överlastad inom varje instruktion. Men genom att läsa in instruktioner vid de tillfällen då bussen inte behövs till något annat, sprids behovet ut på ett bättre sätt.

För det tredje är 9995 något effektivare konstruerad, med exempelvis interna register som inte finns i 9900.

Resultatet blir att en addition på 9900 kräver 14 klockcykler, medan 9995 nöjer sig med fyra. Motsvarande för BLWP är 26 mot 11, för DIV 92–124 (beroende på data) mot 28. Samtliga uppgifter är hämtade ur "The 9900 Family



Kretsarnas placering på Geneve-kortet

Data Book" och förutsätter att minnena är snabbare än processorn.

TMS 9995 är opkodskompatibel med 9900. Det innebär att den förstår samma instruktioner och dessutom kodar dem på samma sätt. Dessutom finns det fyra instruktioner till, bland annat multiplikation och division med tecken.

2. Videoprocessorn i Geneve är en vidareutveckling av den som sitter i 99/4A. Lustigt nog är den inte tillverkad av Texas Instruments, utan i Japan. TI började utveckla kretsen, men bestämde sig för att lägga ner den innan tillverkningen startade. Rättigheterna såldes till Yamaha, som nu tillverkar V9938.

Den har samma möjligheter som TMS 9918/9929, och naturligtvis mer därtill. 80 teckens bredd, med status längst ner, finns. Bit-map med högre upplösning, 512*212 finns också. Varje pixel kan ha valfri färg. Högst tio sprites kan visas på samma rad.

3. Här sitter VDP RAM. Högre upplösning kräver mer minne för att lagra skärmar och teckendefnitoner. Geneve ska ha 128K byte VDP RAM.

4. För att det ska vara möjligt att klämma in allt på ett kort som får plats i boxen, har Myarc varit tvungna att låta specialtillverka en "gate-array" till Geneve. Det rör sig alltså om en sammanslagning av en massa små logikkretsar för diverse avkodningsuppgifter. Kretsen, som är tillverkad av Mitsubishi, påstås vara det sista hindret för serieproduktion. Den första omgången lär nämligen ha varit felaktig.

5. EPROM. Här finns kod för flyttalsaritmetik och annat som behövs för att datorn ska starta ordentligt när strömmen slås till. Det ska bli intressant att se om Geneve någonsin kommer att bli så talrik att EPROM-kretsarna kommer att ersättas av maskprogrammerade ROM. Troligt lär det inte vara, med tanke på att den antagligen kommer lite för tidigt för att vara ordentligt avlusad.

6. Statiskt RAM utan wait-states. Det här motsvarar de 256 bytes RAM som sitter på 16-bitars buss i 99:an. Geneve har åtta kilobytes snabbt minne.

7. Dynamiskt RAM med ett wait-state. Dessa 512K bytes utgör alltså maskinens huvudminne.

8. I/O-krets. Kretsen är en TMS 9901, samma som sitter i 99/4A. Den har en inbyggd timer, avkodning av avbrott och ett antal in- och utgångar.

9. Anslutning för monitor. Genom att flytta en bygel kan man få antingen video eller analog RGB. Det sistnämnda rekommenderas för att 80 teckens bredd ska vara lätt att läsa.

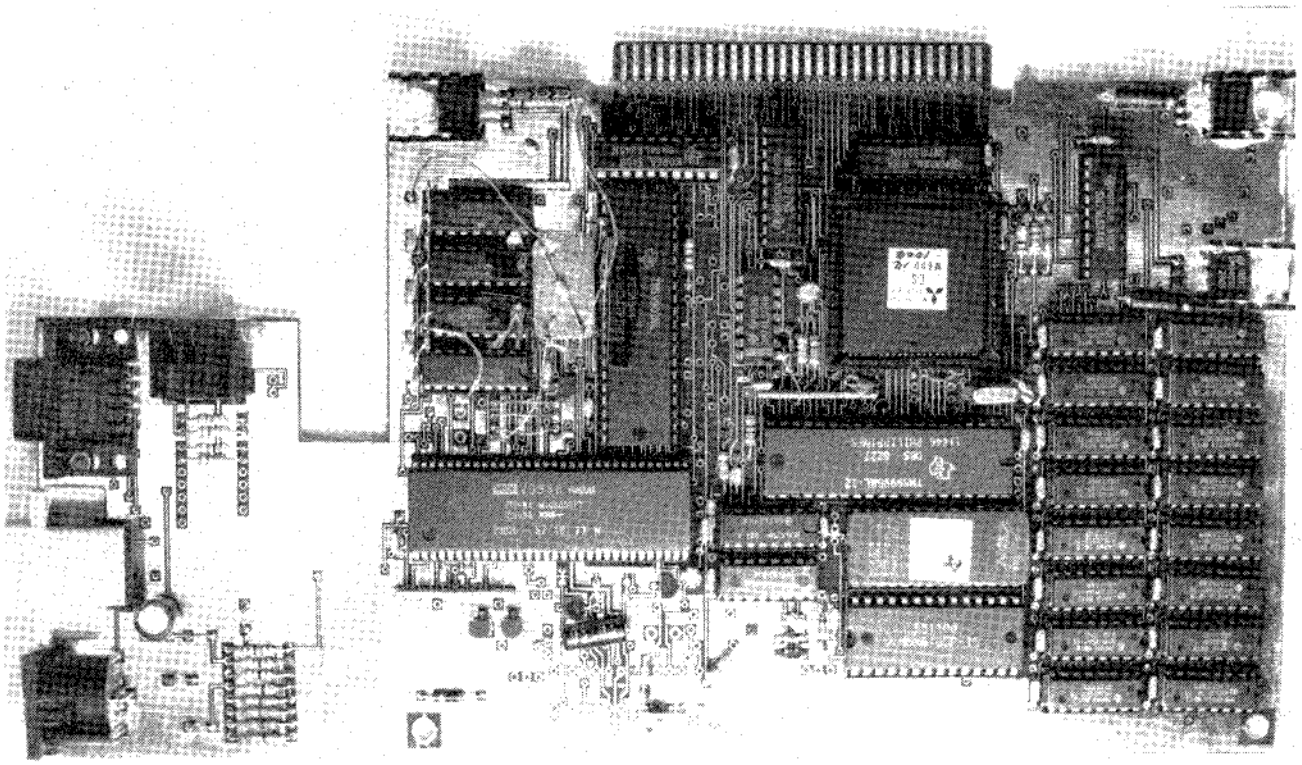
10. Kontakt för joysticks av vanlig TI-typ.

11. Här kopplas musen in. På mötet provade vi med en mus till en Commodore Amiga. Det fungerade, även om vi nog inte riktigt begrep hur programmet egentligen skulle hanteras.

12. Tangentbordets kabel anslutes här. Den är, som brukligt, en kabel ungefär lika grov som till en joystick. Glöm den brandslang till kabel som TI försåg oss med!

Sammanfattning

Ja, det blir inte lätt, det. Den maskin vid hade tillgång till gav ett så splittrat intryck att det är svårt att åstadkomma någon riktig sammanfattning. Vissa delar var mycket imponerande, exempelvis grafiken. Men att alls få det att fungera var så komplicerat att det knappast vore något för "normala" amatörer. Vi får nog avvakta med ett slutligt omdöme tills serieproducerade exemplar dyker upp. Om man ska tro Myarc *(om!)* skulle det kunna bli i början av maj. Naturligtvis återkommer vi med mer ingående rapporter när det nu blir.



Tillbehör

av Jan Alexandersson

Här följer en sammanställning av olika program och tillbehör som finns att köpa till TI-99/4A. För varje program anges en säljare och ett pris. Många av programmen går att köpa på flera ställen. Vid köp från USA tillkommer moms, tull och porto. Du kan räkna med att 1 dollar = 10 kr inkl allt. Vid mindre beställning är portot ofta 10 dollar oberoende av köpets storlek. Vissa program under Expansionsbox kan även användas till kassett t ex GramKracker.

Det är möjligt att fristående enheter som köps från USA kräver 110 V. Tag reda på om detta passar dig innan du köper. Om du köper utrustning med TV-utgång från USA bör du först ta reda på om det passar det europeiska PAL-systemet.

Tillbehör till TI-99/4A grunddator

Joystick-adapter (för WICO och ATARI)	
WICO	TEN USD 5
(bygg själv)	(se PB 84-1.06)
RGB-modulator	
TI PHA 2037	MAG SEK 350
Speech syntesizer	
TI PHP 1500	TEX USD 30
Klocka	
Corcomp Clock Stand Alone	TEX USD 70
Modul-expander	
Navarone WIDGET (för 3 moduler)	TEX USD 25
Expansionsminne, fristående	
Corcomp 32K Stand Alone	TEX USD 90
Mechatronic 32K RAM+Centronics	ALB DEM 290
Mechatronic 128K RAM+Centronics	ALB DEM 400
RS232/PIO, fristående	
Corcomp Stand Alone	TEX USD 100
(bygg själv)	(se PB 85-1.5)
Programmeringsspråk	
TI Extended BASIC	TEX USD 50
TI Mini Memory med Assembler+Ordbeh.	TEX USD 39
TI Teach Yourself BASIC	TEX USD 7
TI Teach Yourself Extended BASIC	TEX USD 7
TI Beginning Basic Tutor	TEX USD 7
Ordbehandling	
Navarone Console Writer, modul	TEN USD 18
DataBioTic Miniwriter III, modul	TEN USD 40
Tex-Scriber (för XB)	TEX USD 20
Extended Typewriter (för XB)	TEN USD 28
Databas	
TI Personal Record Keeping, modul	TEX USD 16
TI Personal Report Generator, modul	TEX USD 11
Extended Name-It (för XB)	TEN USD 28
Kommunikation	
TI Terminal Emulator II, modul	TEX USD 10
Datavision Modul	PB SEK 565
Expansionsbox	
TI PHP 1200 (se annonser i PB)	
Corcomp	TEX USD 380
CPS 99 DS/DD	REI DEM 1398
(bygg själv)	(se PB 85-3.22)
Spel (modul om ej annat anges)	
Tombstone City	TEX USD 5
TI Invaders	TEX USD 5
Munch Man	TEX USD 5
Tunnels of Doom	TEX USD 7
Alpiner	TEX USD 5
Parsec	TEX USD 5
Adventure	TEX USD 7
Microsurgeon	TEX USD 10
Super Demon Attack	TEX USD 10
Moonsweeper	TEX USD 10
Sneggit	TEX USD 6
Hopper	TEX USD 5
Burgertime	TEX USD 10
Jawbreaker II	TEX USD 6
Congo Bongo	TEX USD 16
Treasure Island	TEX USD 12
Return to Pirates Island	TEX USD 12
Buck Rogers	TEX USD 16
Star Trek	TEX USD 16
Fathom	TEX USD 10
Munchmobile	TEX USD 12
Slymoids	TEX USD 16
M*A*S*H	TEX USD 12
Computer War	TEN USD 25
Night Mission (kassett eller diskett)	TEN USD 18

Tillbehör till TI-99/4A med 32k RAM

Interrupt switch	
Corcomp Load Interrupt Switch	TEN USD 12
Programmeringsspråk	
Mechatronic Extended Basic II Plus	RYT USD 75
LOGO II	TEX USD 20
PB-FORTH (kassett eller diskett)	PB SEK 100
Turbo Assembler (för XB)	CK DEM 99
Ordbehandling	
LT-Writer (för XB)	LT SEK 10

Tillbehör till TI-99/4A med expansionsbox

Moduldump	
16K RAM CMOS modul	PB SEK 875
Guy Gournay Maximem 48K	RYT USD 145
Millers Graphics Gram Kracker 80K	TEN USD 195
Mechatronic Gram Karte 128K	RYT USD 250
Klocka+printbuffert 64K+plats för speech	
Corcomp Triple Tech Card	TEX USD 110
RS232/PIO	
Corcomp Card	TEX USD 80
Myarc Card	TEN USD 80
RAM 32K EM	
Corcomp 32K Card	TEX USD 100
RAM-disk	
Corcomp 256K inkl EM	TEX USD 170
Corcomp 512K inkl EM	TEX USD 230
Myarc 128K inkl EM	Texa USD 170
Myarc 256K inkl EM	Texa USD 190
Myarc 512K inkl EM	Texa USD 240
Horizon Kretskort utan komponenter	HOR USD 53
Horizon 104K utan EM	HOR USD 165
Horizon 192K utan EM	HOR USD 210
Disk-kontrollkort	
Corcomp DS/DD för 4 drivar	TEX USD 150
Myarc DS/DD för 4 drivar	Texa USD 150
Myarc Quad density för 4 drivar	Texa USD 190
Disk Manager	
DM 99 (ej kopiering)	PB SEK 50
DM 1000 version 3.5	PB SEK 50
QS Disk Manager IV	TEN USD 20
QS Quick Copier II (endast kop.)	TEN USD 18
QS The Duplicator (endast kopiering)	TEN USD 18
Navarone Super Duper (endast kop.)	TEX USD 25
Texcomp Nibbler (endast kop.)	TEX USD 20
80 kolumnskort	
Mechatronic 80 Column Card	TEN USD 220
Programmeringsspråk	
TI Editor/Assembler (m. filer+manual)	TEX USD 20
QS Assembler (utan EA-filer)	TEN USD 15
TI-FORTH (diskett + manual)	PB SEK 200
Forth&99 (3 disketter)	PB SEK 120
Wycove FORTH	TEN USD 43
Tiny Pascal (listning för FORTH)	SUN SEK 100

c99 version 2.1 (3 disketter)	PB SEK 150
Myarc XB Level IV för 128K RAM	Texa USD 65
CALL LINK-program	
OAK TREE Display Enhanc. 40 column	TEN USD 26
Super Extended Basic	TEN USD 75
Hjälpmedel, BASIC	
TI Programmimg Aids III (analys)	TEX USD 10
NEATLIST (analys)	PB SEK 50
XB Detective (analys)	TEN USD 18
QS XREF (analys)	TEN USD 18
QS Converter (DIS/VAR till BASIC)	TEN USD 20
XB Detective (analys)	Texa USD 18
OAK TREE SMASH (komprimering)	TEN USD 22
Hjälpmedel, disk	
MG Advanced Diagnostics	TEN USD 18
Texcomp The Explorer	TEX USD 20
Navarone Disk Fixer	TEX USD 30
Corcomp Peripherial Diagnostics	TEX USD 25
Hjälpmedel, assembler	
Millers Graphics Explorer	TEN USD 25
Navarone Super Debugger (diskett)	PB SEK 50
PB Dissasembler (Tony Rogvall)	PB SEK 50
Millers Graphics DISkASSEMBLER	MG USD 20
Ordbehandling	
TI-Writer	TEX USD 39
QS-Writer (utan TW-filer)	TEN USD 14
99 Writer II (med TW-filer)	TEX USD 20
Uppgradering TI-Writer/Multiplan	PB SEK 50
Auto Spell Check	TEN USD 30
Companion	INT USD 80
Databas	
Navarone Data Base Management	TEX USD 40
Oak Tree Acorn 99	TEN USD 50
QS Database 99	TEN USD 35
Microsoft Multiplan (kalkylering)	TEX USD 39
PR-BASE (2 disketter)	PB SEK 100
Kommunikation	
4A-Talk	TEN USD 20
Grafik	
GRAPH-X	TEX USD 50
TI Artist	Texa USD 20
Vaughn BITMAC	TEX USD 20
QS Draw 'N Plot	TEN USD 20
Navarone Paint 'N Print	TEN USD 28
Screendump	PB SEK 50
QS SDUMP II	TEN USD 20
FORIT av Bo Carleö (se PB 85-3)	
Joy Paint	GLS USD 40
Tal	
TI Text to Speech (english)	TEX USD 10
Mus	
Mechatronic Mouse	RYT USD 98
Spel	
Tennis	TEX USD 10
Computer War/Submarine Commander	TEN USD 30
River Rescue	

Adresser till försäljare

PB = PROGRAMBITEN (endast PB-markerade)
c/o Schibler
Wahlbergsgatan 6, 1 tr ned
121 46 JOHANNESHOV
postgiro 19 83 00-6

LT = Lennart Thelander (endast LT-Writer)
Dahlhemsvägen 103A
252 65 HELSINGBORG
Tel. 042-15 18 73

MAG = DIGITALSERVICE AB (fd Magnussons)
Box 20024
(Bjurholmsg. 13)
104 60 STOCKHOLM
Tel. 08-41 60 52

BLIJENBURGH ELECTRONICS
Box 8164
(Drottninggatan 81)
104 20 STOCKHOLM
Tel. 08-20 50 54

Douglas Lundqvist (stor sortering)
Kung Sverres gata 43
417 28 GÖTEBORG
Tel. 031-54 17 01 (kvällar och helger)

SUN = Computer Press (endast böcker)
Box 893
851 24 SUNDSVALL
Tel. 060-15 04 75

ALB = ALBS-ALLTRONIC
B. Schmidt
Postfach 1130
D-7136 ÖTISHEIM
Västtyskland

REI = Programm-Service REIS
Bergstrasse 80
D-5584 BULLAY
Västtyskland

CK = Verlag Rätz-Eberle/CK-Software (utvalda program)
Postfach 1640
D-7518 BRETEN
Västtyskland

TEX = TEXCOMP (stor sortering)
Users Supply
P.O. BOX 33084
GRANADA HILLS, CA 91344
USA

TEN = TENEX COMPUTER EXPRESS (stor sortering)
P.O. BOX 6578
SOUTH BEND, IN 46660
USA

MG = MILLERS GRAPHICS (endast MG-produkter)
1475 W. Cypress Ave.
San Dimas, CA 91773
USA

HOR = Horizon Computer (endast egna produkter)
P.O. BOX 554
WALBRIDGE, OH 43465
USA

RYT = RYTE DATA (mest Mechatronic)
Millennium Computers
210 Mountain Street
HALIBURTON,
ONTARIO K0M 1S0
Kanada

INT = INTELPRO (endast egna produkter)
13 Saratoga Dr.
Kirkland
QUEBEC H9H 3J9
Kanada

Texa = Texaments (TI-Artist och Myarc-produkter)
53 Center Street
New York 11772
USA

GLS = Great Lakes Software (endast Joy Paint 99)
804 E. Grand River Ave.
Howell, MI 48843
USA

FÖRENINGEN PROGRAMBITEN
BALANS- OCH RESULTATRÄKNING 1986

Ingående balans 1/1 1986

Tillgångar		Skulder och eget kapital	
Postgiro	44 860.28	Obetalda fakturor	6 110.00
Varulager	1 500.00	Medlemsavgifter -86	1 350.00
Inventarier	1 728.00	Prel. B-skatt -85	374.00
Fordran skatteåterb.	1 770.00	Utgående moms	3 869.35
Fordran Göran Nygren	257.00	Eget kapital	39 917.87
Ingående moms	1 505.94		
	51 621.22		51 621.22

Utgående balans 31/12 1986

Tillgångar		Skulder och eget kapital	
Postgiro	33 956.28	Obetalda fakturor	4 986.00
Varulager	8 265.30	Medlemsavgifter -87	720.00
Inventarier	1 728.00	Övriga skulder	1 801.10
Fordran skatteåterb.	760.00	Utgående moms	2 229.02
Fordran Michael Öhman	310.60	Eget kapital	39 484.34
Ingående moms	4 200.28		
	49 220.46		49 220.46

Resultaträkning för 1986

Intäkter		Kostnader	
Medlemsavgifter	30 910.00	Tryck tidskrift (5 nr)	22 353.00
Varuförsäljning	9 500.89	Varuinköp	15 099.94
Räntor	2 360.55	Porto	7 258.30
Årets underskott	7 244.58	Div. omkostnader	5 304.78
	50 016.02		50 016.02

Register för Programbiten 1986

av Jan Alexandersson

Detta register har skrivits med modulen Personal Record Keeping med mitt BASIC-program *Snabbfil* (se PB 85-2). Eftersom det ej går att använda printfunktionen mot disk har jag gjort ett BASIC-program som sköter konverteringen.

Förkortningar:

BA	=	BASIC
XB	=	Extended BASIC
XE	=	Extended BASIC + 32 kb expansionsminne
MM	=	Mini Memory (ofta även Editor/Assembler)
RK	=	Personal Record Keeping (eller Statistics)
AL	=	Assembler
FH	=	Forth
PA	=	Pascal
LO=	=	Logo
TW	=	TI-Writer
EM	=	Expansionsminne 32k
GRAM	=	Maximem/Gram Kracker/Gram Karte

Innan registret togs in i tidningen har jag gjort ytterligare konverteringar och bland annat ändrat till blandat små och stora bokstäver.

– Red.

Artiklar i Programbiten 1986 sorterade efter modul/språk

Artikel	Källa	År-Nr. Sida	Ant Sid	Modul Språk
Lagring på band	Persson A.	86-2.06	1	AL
Input för AL	Häll Börje	86-2.22	1.4	AL
C för 99:an	Gustavsson B.	86-3.14	2.5	AL
Pixel till punkt	Bernle Sören	86-4.09	4	AL
Ladda AL,DSK&CS	Alexandersson	86-5.09	1.6	AL
WORD-S 1 och 2	Hovmöller N&J	86-5.15	2.6	BA
Helikopter-spel	Florin&Nilson	86-2.15	2.7	BA
Stjärnkikeri	Wennberg Arne	86-2.07	3	BA
P:Kopiera	Wind S-E	86-2.11	0.5	BA
CK-Program	Computer Kont	86-1.06	1	BA/XB
Programbanken	Thelander L.	86-3.08	1	BA/XB
Forth-spalten	Svahn L-E	86-3.10	1.6	FH
Forth Swefig	Lindberg L.	86-1.05	1	FH
Produktkostnad	Lindberg L.	86-4.08	0.8	FH
Forth-spalten	Svahn L-E	86-4.06	1.5	FH
Forth	Svahn L-E	86-2.10	1.4	FH
Maximem	Thelander	86-3.16	0.5	GRAM
Logo	Skyttner Lars	86-2.20	0.7	LO
Masken för MM	Landsberg B.	86-2.18	1.3	MM
Mini-ASMBAS	Landsberg B.	86-2.19	0.7	MM
Styr pratorn	Shaw Stephen	86-4.14	0.7	MM/XE
Pascal grupp	Persson A.	86-4.05	0.7	PA
Register 83-85	Alexandersson	86-1.09	3.3	RK
Bankkonto	Robertsson J.	86-5.18	4	RK
Register för PB	Alexandersson	86-4.05	0.3	RK
Ext. Precision	Prins Robert	86-5.14	0.5	TI-59
Vertikalkurvor	Nilsson Jan	86-3.12	2	TI-59
Polar plotter	Prins Robert	86-5.11	3	TI-59
Root Polynomials	Prins Robert	86-2.12	1	TI-59
Histogram	Prins Robert	86-2.13	1	TI-59
TI-74 BASIC	Schibler C.	86-5.14	0.3	TI-74
Autospell Check	Persson A.	86-3.05	1	TW
Mechatronic XB+	Persson A.	86-5.04	1	XB
Bug-out för XB	Persson A.	86-3.04	1	XB
Musik till XB	Svahn L-E	86-2.14	1	XB
Sprite i cirkel	Bourelius Ulf	86-2.17	0.3	XB
Frågor och svar	Häll Börje	86-4.16	0.4	XB
Garbage Collect	Alexandersson	86-1.03	0.2	XB
Parameterpassn.	Häll Börje	86-4.15	1.6	XB
Rev av Sverige		86-5.17	0.4	XB
Kalkylator 99:a	Sams Lars	86-1.12	1	XB
Korsord med 99	Wennberg Arne	86-1.08	0.7	XB
Sverige runt	Hovmöller N&J	86-3.17	2.7	XB
Sorteringsdemo	Östberg B-A	86-3.20	3.3	XB
Sprite maker	Eriksson T.	86-1.07	1	XB
Accept >28 teck	Alexandersson	86-1.04	0.2	XB
Neatlist	Häll Börje	86-3.09	0.3	XB+EM
Screendump	Häll Börje	86-3.09	0.3	XB+EM
LT-Writer	Thelander L.	86-2.20	1.3	XB+EM
Disk Manager 99	Häll Börje	86-3.09	0.2	XB+EM
RGB-modulator	Thelander L.	86-4.07	0.5	
Smart Programm.	Odelryd Peter	86-4.08	0.2	
Medlemsmatrikel	Schibler C.	86-4.08	8	
Advanced Diags	Persson A.	86-4.04	1	
Samlingsdisk 85	Alexandersson	86-4.13	0.6	
Bygg RAM-disk	Odelryd Peter	86-4.13	0.4	
Myarc Geneve	Odelryd Peter	86-3.24	0.1	
DM1000 V2.2 bug	Alexandersson	86-4.14	0.1	
Thorn-EMI spel	Odelryd Peter	86-3.24	0.1	
MICROpendium	Odelryd Peter	86-3.11	0.4	
Kontaktproblem		86-5.03	0.2	
Nat Assistance	Odelryd Peter	86-3.09	0.3	
Videosignaler	Fahlgren B.	86-5.05	1	
Disk Manager	Alexandersson	86-5.06	2.5	
Freeware	Häll Börje	86-5.08	0.5	
MG Explorer	Persson A.	86-3.07	1	
Staffanstorp	Persson A.	86-5.10	0.4	
Enkäten	Häll Börje	86-3.06	1	
Årsmötesprotok	Wickström H.	86-3.03	0.7	
Hitta på skiva	Persson A.	86-2.04	2	
Silverreed EB50	Bleeker Johan	86-1.05	0.2	
Annan 99-fören	Lindberg L.	86-1.04	0.2	
Staffanstorp	Persson A.	86-1.04	0.5	
Styr och ställ	Persson A.	86-5.22	3	
Årsmöte	Lindberg L.	86-1.03	0.5	

Artiklar i Programbiten 1986 sorterade efter år – nr – sida

Artikel	Källa	År-Nr. Sida	Ant Sid	Modul Språk
Årsmöte	Lindberg L.	86-1.03	0.5	
Garbage Collect	Alexandersson	86-1.03	0.2	XB
Staffanstorp	Persson A.	86-1.04	0.5	
Accept >28 teck	Alexandersson	86-1.04	0.2	XB
Annan 99-fören	Lindberg L.	86-1.04	0.2	
Forth Swefig	Lindberg L.	86-1.05	1	FH
Silverreed EB50	Bleeker Johan	86-1.05	0.2	
CK-program	Computer kont	86-1.06	1	BA/XB
Sprite maker	Eriksson T.	86-1.07	1	XB
Korsord med 99	Wennberg Arne	86-1.08	0.7	XB
Register 83-85	Alexandersson	86-1.09	3.3	RK
Kalkylator 99:A	Sams Lars	86-1.12	1	XB
Hitta på skiva	Persson A.	86-2.04	2	
Lagring på band	Persson A.	86-2.06	1	AL
Stjärnkikeri	Wennberg Arne	86-2.07	3	BA
Forth	Svahn L-E	86-2.10	1.4	FH
P:Kopiera	Wind S-E	86-2.11	0.5	BA
Root Polynomials	Prins Robert	86-2.12	1	TI-59
Histogram	Prins Robert	86-2.13	1	TI-59
Musik till XB	Svahn L-E	86-2.14	1	XB
Helikopter-spel	Florin/Nilsson	86-2.15	2.7	BA
Sprite i cirkel	Bourelius Ulf	86-2.17	0.3	XB
Masken för MM	Landsberg B.	86-2.18	1.3	MM
Mini-ASMBAS	Landsberg B.	86-2.19	0.7	MM
Logo	Skyttner Lars	86-2.20	0.7	LO
LT-Writer	Thelander L.	86-2.20	1.3	XB+EM
Input för AL	Häll Börje	86-2.22	1.4	AL
Årsmötesprotok	Wickström H.	86-3.03	0.7	
Bug-out för XB	Persson A.	86-3.04	1	XB
Autospell check	Persson A.	86-3.05	1	TW
Enkäten	Häll Börje	86-3.06	1	
MG Explorer	Persson A.	86-3.07	1	
Programbanken	Thelander L.	86-3.08	1	BA/XB
Neatlist	Häll Börje	86-3.09	0.3	XB+EM
Screendump	Häll Börje	86-3.09	0.3	XB+EM
Disk Manager 99	Häll Börje	86-3.09	0.2	XB+EM
Nat Assistance	Odelryd Peter	86-3.09	0.3	
Forth-spalten	Svahn L-E	86-3.10	1.6	FH
MICROpendium	Odelryd Peter	86-3.11	0.4	
Vertikalkurvor	Nilsson Jan	86-3.12	2	TI-59
C för 99:an	Gustavsson B.	86-3.14	2.5	AL
Maximem	Thelander	86-3.16	0.5	Gram
Sverige runt	Hovmöller N	86-3.17	2.7	XB
Sorteringsdemo	Östberg B-A	86-3.20	3.3	XB
Thorn-EMI spel	Odelryd Peter	86-3.24	0.1	
Myarc Geneve	Odelryd Peter	86-3.24	0.1	
Advanced Diags	Persson A.	86-4.04	1	
Pascal grupp	Persson A.	86-4.05	0.7	PA
Register för PB	Alexandersson	86-4.05	0.3	RK
Forth-spalten	Svahn L-E	86-4.06	1.5	FH
RGB-modulator	Thelander L.	86-4.07	0.5	
Produktkostnad	Lindberg L.	86-4.08	0.8	FH
Smart Programm.	Odelryd Peter	86-4.08	0.2	
Medlemsmatrikel	Schibler C.	86-4.08	8	
Pixel till punkt	Bernle Sören	86-4.09	4	AL
Samlingsdisk 85	Alexandersson	86-4.13	0.6	
Bygg RAM-disk	Odelryd Peter	86-4.13	0.4	
Styr pratorn	Shaw Stephen	86-4.14	0.7	MM/XE
DM1000 V2.2 bug	Alexandersson	86-4.14	0.1	
Parameterpassn.	Häll Börje	86-4.15	1.6	XB
Frågor och svar	Häll Börje	86-4.16	0.4	XB
Kontaktproblem		86-5.03	0.2	
Mechatronic XB+	Persson A.	86-5.04	1	XB
Videosignaler	Fahlgren B.	86-5.05	1	
Disk Manager	Alexandersson	86-5.06	2.5	
Freeware	Häll Börje	86-5.08	0.5	
Ladda AL,DSK,CS	Alexandersson	86-5.09	1.6	AL
Staffanstorp	Persson A.	86-5.10	0.4	
Polar plotter	Prins Robert	86-5.11	3	TI-59
TI-74 BASIC	Schibler C.	86-5.14	0.3	TI-74
Ext. Precision	Prins Robert	86-5.14	0.5	TI-59
WORD-S 1 och 2	Hovmöller N	86-5.15	2.6	BA
Rev av Sverige		86-5.17	0.4	XB
Bankkonto	Robertsson J.	86-5.18	4	RK
Styr och ställ	Persson A.	86-5.22	3	

Toady

ett program av Anders Månsson

Anders presenterar sitt program så här:

”Det på kassett bifogade spelprogrammet heter alltså *Toady*. Det är skrivet i Extended BASIC och upptar ca 8,5 kbytes av minnet. Programmet har en avsiktlig likhet med *Frogger* men är mycket ”fräckare”. Att få det att fungera med tangentbordet är alls icke svårt och därför överlåter jag åt var och en att göra detta på sitt eget vis.

VARNING! Bör ej spelas av en känslig person utan närvaro av tonåring eller annan person med perverterad humor.”

Så långt Anders egna ord. Som Anders skriver så är det en variant på Frogger. För den som funderat över varför grodorna absolut ger sig på att hoppa över vägar och floder har han en förklaring. Det var något år sen vi fick programmet från honom. Att jag väljer att publicera det nu beror bl a på att det är dåligt med inflödet av spelprogram till tidningen.

– Redaktören

```
  1 REM                                  **************
    **            *   T O A D Y   *              **
    **************
  2 !
  3 !
  4 !
  5 !
 10 ! 100-290 DEF TECKEN TILLSPRITES OCH FAST GRA
    FIK
 12 ! 300 EXEKVERAR PRESENTATIONEN OM PROGRAMMMET
     KORS FOR FORSTA GANGEN
 14 ! 310-590 SATTER UT FAST GRAFIK OCH SPRITES P
    A SKARMEN
 16 ! 600-650 HUVUDLOOP PA LAND
 18 ! 710-750 HUVUDLOOP OVER FLODEN
 20 ! OVRIGT: DELPROGRAM FOR OLIKA HANDELSER, POA
    NGBERAKNING, KONTROLL M.M
100 CALL CLEAR :: RESTORE 1330 :: REP=REP+1
110 NJ=0 :: TK=0.7 :: SK=2.92 :: MOT(1)=8 :: MOT(
    2)=-10 :: MOT(3)=8 :: MOT(4)=0
120 SS,K=1 :: TOAD=3 :: SC,FULL,TD=0
130 COLOUR=3
140 CALL CHAR(96,"00000C1221C00000")
150 CALL CHAR(104,"5AA55AA55AA55AA5")
160 CALL CHAR(112,"00")
170 CALL CHAR(114,"00FFFFFFFFFFFF00"):: CALL CHAR
    (115,"0000007E7E000000")
180 REM                                    FROG
190 CALL CHAR(128,"0000090B05030B0F0F0D18"):: CAL
    L CHAR(130,"000090D0A0C0D0F0F0B018")
200 CALL CHAR(136,"00000000001B0D0F0F0B03050B09")
    :: CALL CHAR(138,"00000000001BB0F0F0D0C0B0D09
    0")
210 CALL CHAR(140,"0000000407030103030103030704"):
    : CALL CHAR(142,"00000000CC90E8FCFCE890CC")
220 CALL CHAR(132,"00000000330917 3F3F170933"):: C
    ALL CHAR(134,"00000020E0C0B0C0C0B0C0E020")
230 REM                                ..........
    ........
240 CALL CHAR(108,"0022510905050503"):: CALL CHAR
    (109,"00"):: CALL CHAR(111,"00"):: CALL CHAR(
    110,"0010244A505060C0")
250 CALL CHAR(100,"0C7F3A3A3A7F0C"):: CALL CHAR(1
    01,"00"):: CALL CHAR(102,"6CFB3BB83BF86C"):: 
    CALL CHAR(103,"00")
260 CALL CHAR(124,"00361F1C1D1C1F36"):: CALL CHAR
    (125,"00"):: CALL CHAR(126,"0030FE5C5C5CFE30"
    ):: CALL CHAR(127,"00")
270 CALL CHAR(40,"0102CCFF9FF33F"):: CALL CHAR(41
    ,"00"):: CALL CHAR(42,"E080441FF39FFEE0"):: C
    ALL CHAR(43,"00")
280 CALL CHAR(36,"2E16837F3F03162E"):: CALL CHAR(
    37,"00"):: CALL CHAR(38,"1820E4FFFFE42018"):: 
     CALL CHAR(39,"00")
290 CALL CHAR(60,"0B3F7FFF7F7F7F3F7F7F7FFF7F3F07A
    4F0F8F8FCF8FCFEFFFFFEFEFEFCF86208")
300 IF REP=1 THEN GOSUB 1140
310 CALL CLEAR :: CALL SCREEN(3):: CALL COLOR(1,3
    ,14):: CALL CHAR(33,"FFFFFFFFFFFFFFFF"):: CAL
    L VCHAR(1,1,33,48)
320 CALL MAGNIFY(3):: FOR Q=2 TO 8 :: CALL COLOR(
    Q,16,14):: NEXT Q
330 CALL COLOR(9,6,8):: FOR Q=4 TO 12 :: CALL HCH
    AR(Q,3,96,30):: NEXT Q
340 CALL COLOR(10,12,10):: CALL HCHAR(12,3,104,30
    ):: CALL HCHAR(13,3,104,30):: CALL HCHAR(23,3
    ,104,30):: CALL HCHAR(24,3,104,30)
350 CALL COLOR(11,16,15):: FOR Q=14 TO 22 :: CALL
     HCHAR(Q,3,112,30):: NEXT Q
360 FOR Q=15 TO 21 STEP 2 :: CALL HCHAR(Q,17,114,
    2):: CALL HCHAR(18,3,115,30):: CALL HCHAR(18,
    16,112,4)
370 NEXT Q
380 GOSUB 480
390 GOSUB 410
400 GOTO 520
410 REM
420 CALL COLOR(13,12,14)
430 FOR T=8 TO 26 STEP 6 :: CALL VCHAR(3,T,128):: 
     CALL VCHAR(4,T,129):: CALL VCHAR(3,T+1,130):
    : CALL VCHAR(4,T+1,131):: NEXT T
440 RESTORE 1330 :: FOR Q=4 TO 14 :: READ CCR,COL
    ,VP,HP,V :: CALL SPRITE(#Q,CCR,COL,VP,HP,0,IN
    T(K$V)):: NEXT Q
450 FOR Q=16 TO 22 :: READ CCR,COL,VP,HP,V :: CAL
    L SPRITE(#Q,CCR,COL,VP,HP,0,V):: NEXT Q
460 RETURN
470 REM :: GOTO 499
480 DISPLAY AT(1,1)SIZE(11):"SCORE";SC
490 DISPLAY AT(1,13)SIZE(7):"TOADS";TOAD
500 DISPLAY AT(1,22):"SPEED";SS
510 RETURN
520 CALL SPRITE(#1,128,13,181,130):: PX=130 :: PY
    =181
530 N=0
540 GOSUB 480
550 IF FULL>2 THEN 580
560 RANDOMIZE :: T=INT(20*RND$8+1)+24 :: CALL SPR
    ITE(#15,108,13,94,T)
570 GOTO 600
580 RANDOMIZE :: T=INT(20*RND$8+1)+24 :: CALL SPR
    ITE(#15,36,3,94,T,0,6)
590 REM FOR Q=128 TO 140 STEP 4 :: CALL PATTERN(#
    1,Q):: FOR W=1 TO 50 :: NEXT W :: NEXT Q :: G
    OTO 900
600 REM
610 CALL JOYST(1,X,Y):: IF Y>0 THEN P=128 ELSE IF
     Y<0 THEN P=136 ELSE IF X<0 THEN P=132 ELSE I
    F X>0 THEN P=140
620 CALL COINC(ALL,C):: IF C THEN 1010 ELSE IF X=
    0 AND Y=0 THEN NJ=NJ+1 :: GOTO 640
630 CALL PATTERN(#1,P):: PY=PY-Y$4 :: PX=PX+X$4 :
    : CALL LOCATE(#1,PY,PX):: CALL SOUND(10,110,0
    ,111,0,112,0):: IF PY<84 THEN 700
640 CALL COINC(ALL,C):: IF C THEN 1010
650 IF PY>84 THEN 610
660 CALL COLOR(#16,8,#17,8):: CALL SOUND(800,-8,0
    )
670 CALL MOTION(#1,0,0):: CALL POSITION(#1,PY,PX)
680 CALL DELSPRITE(#1):: CALL SPRITE(#24,60,8,PY,
    PX):: FOR Q=1 TO 10 :: CALL SOUND(100,-8,0):: 
     FOR W=1 TO 3*Q :: NEXT W :: NEXT Q
690 TOAD=TOAD-1 :: CALL DELSPRITE(#24):: CALL COL
    OR(#16,2,#17,2):: IF TOAD=0 THEN 1090 ELSE 52
    0
700 N=1 :: CALL MOTION(#1,0,8):: CALL COINC(ALL,C
    ):: IF C THEN 710 ELSE 680
710 CALL POSITION(#1,D,I):: IF I<10 OR I>246 THEN
     680 ELSE CALL JOYST(1,X,Y):: IF Y THEN 730 E
    LSE NJ=NJ+1
720 IF N$RND>SK THEN 660 ELSE CALL JOYST(1,X,Y):: 
     IF Y=0 THEN 710
730 SIG=SGN(Y):: CALL PATTERN(#1,132-SIG*4):: N=N
    +SIG :: PY=PY-Y$4 :: IF N=0 THEN 1080
740 CALL POSITION(#1,D,PX):: CALL LOCATE(#1,PY,PX
    ):: CALL MOTION(#1,0,MOT(N)):: C=0 :: CALL CO
    INC(ALL,C):: CALL COINC(ALL,C)
750 IF N=4 THEN 760 ELSE IF C=0 THEN 680 ELSE CAL
    L SOUND(1,110,0,111,0,112,0):: GOTO 710
760 REM   made it over
770 CALL COINC(ALL,C):: IF C THEN 790
780 CALL POSITION(#1,V,H):: IF H>53 AND H<61 OR H
    >101 AND H<109 OR H>149 AND H<157 OR H>197 AN
    D H<205 THEN 810
790 CALL SOUND(200,-5,10)
800 CALL MOTION(#1,8,0):: FOR Q=1 TO 40 :: NEXT Q
     :: CALL POSITION(#1,PY,PX):: GOTO 660
810 CALL POSITION(#1,A,B)
820 IF H<61 THEN PX=57 ELSE IF H<109 THEN PX=105
    ELSE IF H<157 THEN PX=153 ELSE PX=201
830 CALL LOCATE(#1,A,PX)
840 FOR W=1 TO 10 :: CALL SOUND(5,110,0,-7,0):: I
    F X<>A THEN X=A ELSE X=A+3
850 CALL LOCATE(#1,X,PX):: NEXT W :: TD=TD+1 :: S
    C=SC+100-NJ :: NJ=0 :: GOSUB 480
860 IF TD<4 THEN CALL SPRITE(#(28-TD),128,13,X-5,
    PX)
870 IF TD<4 THEN 520 ELSE TD=0
880 REM    MELODI
890 IF RET THEN 960 ELSE CALL DELSPRITE(ALL)
900 CALL DELSPRITE(#1,#27,#26,#25,#24):: TK=TK-0.
    05 :: IF TK<0.05 THEN TK=0.55 :: SC=SC*2
```

```
910 FULL=FULL+1 :: SC=INT(1.2*SC):: IF FULL/3=INT
    (FULL/3)THEN TOAD=TOAD+1 :: IF TOAD>5 THEN TO
    AD=5 :: SC=SC*(1+FULL/10)
920 K=K+0.05 :: SS=SS+1 :: IF K>1.25 THEN K=1.25
    :: SS=6 :: SC=SC+50
930 GOSUB 440
940 IF SK<3.3 THEN SK=3.3
950 GOSUB 480
960 RESTORE 1510 :: FOR Q=1 TO 31 :: READ L,F ::
    CALL SOUND(INT(L*TK),F,0):: NEXT Q
970 IF RET THEN RET=0 :: RETURN
980 IF COLOUR=3 THEN COLOUR=5 ELSE IF COLOUR=5 TH
    EN COLOUR=2 ELSE COLOUR=3
990 CALL SCREEN(COLOUR):: CALL COLOR(1,COLOUR,14)
1000 GOTO 520
1010 REM TRAFIC-ACCIDENT
1020 CALL SPRITE(#24,60,7,PY,PX)
1030 CALL SOUND(100,-7,0):: CALL SOUND(300,-8,0)
1040 FOR Q=1 TO 150 :: NEXT Q
1050 TOAD=TOAD-1 :: GOSUB 490
1060 IF TOAD=0 THEN 1090 ELSE CALL DELSPRITE(#24)
1070 GOTO 520
1080 CALL POSITION(#1,O,PX):: CALL LOCATE(#1,PY,PX
    ):: CALL MOTION(#1,0,0):: GOTO 610
1090 REM   GAME OVER
1100 CALL DELSPRITE(ALL):: SC=SC+INT(185-PY):: TIM
    E=0 :: TOAD=0 :: GOSUB 480
1110 DISPLAY AT(24,3):"PRESS ENTER FOR A NEW GAME"
1120 CALL KEY(0,T,S):: IF S=0 THEN 1120 ELSE IF T=
    13 THEN 100 ELSE CALL CLEAR
1130 END
1140 REM :: PRESENTATION
1150 CALL SCREEN(2):: CALL MAGNIFY(2):: FOR Q=2 TO
     8 :: CALL COLOR(Q,3,2):: NEXT Q
1160 FOR Q=1 TO 28 :: DISPLAY AT(5,1):SEG$("
    I PRESENT TO YOU       ",1,Q):: NEXT Q
1170 RESTORE 1320 :: V=80 :: H=60 :: FOR Q=1 TO 5
    :: V=V+4 :: H=H+20 :: READ CCR :: CALL SPRITE
    (#Q,CCR,13,V,H):: NEXT Q :: RET=1 :: GOSUB 88
    0
1180 PRINT "   A GAME BY ANDREW MANSON"
1190 FOR Q=1 TO 400 :: NEXT Q :: CALL CLEAR :: CAL
    L DELSPRITE(ALL)
1200 PRINT "INSTRUCTIONS Y/N "
1210 CALL KEY(0,T,S):: IF S=0 THEN 1210 ELSE IF T<
    >89 AND T<>121 THEN CALL CLEAR :: RETURN
1220 CALL CLEAR :: CALL SCREEN(15):: FOR Q=3 TO 8
    :: CALL COLOR(Q,5,15):: NEXT Q :: CALL MAGNIF
    Y(3)
1230 CALL SPRITE(#1,128,13,1,17):: DISPLAY AT(2,4)
    :"YOUR MALE TOAD"
1240 CALL SPRITE(#2,128,11,25,17):: DISPLAY AT(5,4
    ):"FEMALE TOAD"
1250 CALL SPRITE(#3,100,02,57,17):: DISPLAY AT(8,4
    ):"CAR  WATCH OUT"
1260 CALL SPRITE(#4,108,13,81,17):: DISPLAY AT(11,
    4):"KILLERPLANT  WATCH IT"
1270 CALL SPRITE(#5,40,7,105,17):: DISPLAY AT(14,4
    ):"LOG  SAFE TO RIDE ON"
1280 CALL SPRITE(#6,36,2,128,17):: DISPLAY AT(17,4
    ):"CROCODILE  MIGHT GO DOWN"
1290 CALL SPRITE(#7,36,3,153,17):: DISPLAY AT(20,4
    ):"ALIGATOR  AVOID"
1300 DISPLAY AT(24,1):"WHEN READY  PRESS ANY KEY"
1310 CALL KEY(0,T,S):: IF S=0 THEN 1310 ELSE CALL
    CLEAR :: CALL DELSPRITE(ALL):: RETURN
1320 DATA 84,79,65,68,89
1330 DATA 124,5,105,50,20
1340 DATA 124,7,105,180,20
1350 DATA 100,5,121,140,-18
1360 DATA 100,7,121,100,-18
1370 DATA 124,2,137,190,23
1380 DATA 124,2,137,170,23
1390 DATA 100,14,153,190,-15
1400 DATA 100,14,153,140,-15
1410 DATA 100,3,153,40,-15
1420 DATA 124,3,169,76,19
1430 DATA 124,6,169,135,19
1440 DATA 36,2,41,135,8
1450 DATA 36,2,41,35,8
1460 DATA 40,7,57,50,-10
1470 DATA 40,7,57,150,-10
1480 DATA 40,7,57,250,-10
1490 DATA 40,7,73,100,8
1500 DATA 40,7,73,250,8
1510 DATA 250,392,125,330,250,262,125,330,250,392,
     125,523,250,392,125,330,250,349,125,294,250,2
     47,125,247
1520 DATA 250,392,125,349,250,330,125,40000,250,39
     2,125,494,250,587,125,494,250,523,125,659,250
     ,523,125,494
1530 DATA 250,440,125,392,250,492,125,440,250,349,
     125,294,250,262
```

TI-74

av Peter Odelryd

Som vi nämnde i förra numret har TI presenterat en ny räknare/fickdator, TI-74/BASICALC. Sen dess har den börjat säljas och finns nu att köpa i varuhus och affärer som säljer räknare. Som vi skrev då så verkar TI ha hämtat en hel del inspiration från CC-40 datorn som kom (och försvann) 1983. Den presenterades ju när den kom som en medlem i samma "familj" som TI-99 och tillbehör som skulle knyta ihop de bägge datorerna fanns också framtagna, det s k *HEXBUS*-interfacet. Efter det att 99-an slutade tillverkas försvann också CC-40 i tysthet. CC-40 kom med en inbyggd BASIC, som dock inte var helt kompatibel med TI BASIC eller Extended BASIC. CC-40 skulle också kunna programmeras i assembler, en variant anpassad för den processor som satt i, en 8-bitars TMS 70C20.

TI-74 är byggd kring en processor i samma familj, men annars är det en hel del nyheter. För det första är den mindre till formatet, går ner i bakfickan om man vill. En BASIC-tolk finns inbyggd i 74-an också. Den här gången är den dock mera lik Extended BASIC. Ett program skrivet på 99-an kan flyttas till 74-an med små eller inga ändringar. Skillnaderna ligger förstås i att 74-an bara visar en rad i taget på sin LCD-display.

Till höger om displayen finns ett fack för moduler. Bland det som finns att köpa från början finns moduler för statistik och matematik, liksom en med 8 K RAM. Eftersom minneskretsarna är i CMOS behåller de informationen även om strömmen stängs av.

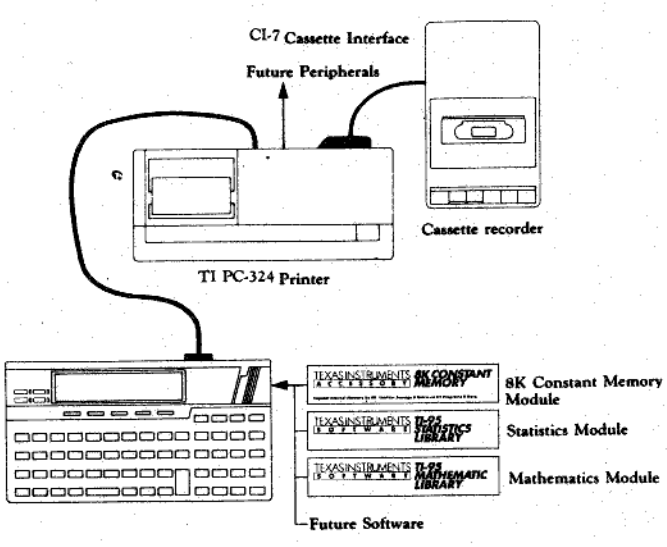
Förutom i BASIC finns det även möjlighet att lära sig Pascal på 74-an. En modul som innehåller en delmängd av UCSD Pascal finns framme. Det är dock inte kompilerad, utan tolkad, precis som BASIC och använder också radnummer. Med Pascal-modulen följer också en lärobok och en referensmanual (bägge på engelska).

Den kringutrustning som finns framme nu innefattar kassettbandsinterface och en skrivare. Skrivaren är av termisk typ och skriver 24 tecken per rad med en hastighet av 48 rader i minuten.

Något sätt att överföra program eller datafiler mellan 74-an och 99-an finns dock inte. Lagringen på kassett är inte lika mellan maskinerna. Det kommer dock en möjlighet att kommunicera via ett RS232-interface från 74-an, vilket öppnar möjligheter.

I fortsättningen kommer vi att lämna plats i Programbiten för material om 74-an. Vi är därför intresserade av att komma i kontakt med de som köpt en 74 och kan skriva om den. Hör av dig på tidningens adress!

Digitalservice erbjuder Föreningen Programbitens medlemmar att köpa 74-an till ett förmånligt pris. Se annons på sista sidan i detta nummer!



Kontrollerad input med exempel: Beräkning av dagavstånd

av Lennart Lindberg

Det finns behov av rutiner i FORTH för att på ett bra sätt mata in numeriska och på annat sätt avgränsade värden till olika program. Programmerare runt om i världen sysslar med att tillverka sådana eftersom FORTH:s kärna inte har definierade ord för detta. Här finns alltså utrymme för erfarenhetsutbyte.

I litteraturen har jag hittat några artiklar om saken som intresserat mig. Jag redovisar deras innehåll (delvis) här samt exemplifierar med ett litet program jag själv gjort för att beräkna avståndet i dagar mellan två datum. Det bygger på en algoritm för beräkningen som jag också hittat i en tidning. Algoritmen utgör emellertid endast en mycket liten – om också omistligt viktig – del av programmet, som strax framgår.

På skärmarna #55 och #56 finns några ord av Michael Ham, som är en professionell frilansande programmerare från Scotts Valley i Californien. Han har f ö numera fått en spalt i *Doctor Dobb's Journal of Software Tools,* som är en utmärkt tidning för professionella programmerare. Läs gärna hans artikel om FORTH i juli-numret -86.

Jag tror att Hams FORTH-ord är tämligen självförklarande. Jag har lagt till ett exempel med inmatning av JA/NEJ som kanske kan bidra.

Ett annat sätt att lösa samma problem visar Ken Takara från San José, också det en ort i soliga Californien, på skärmarna #57 och #58. Lösningen liknar den som används i ACCEPT i TI Extended BASIC. Man skapar alltså en sträng med de tecken som är godkända för inmatning. Med ordet **SELECT** åstadkommer man sedan att den lagrade strängen genomsöks med det tecken, som är under inmatning, som argument. Finns det i strängen accepteras det, annars inte. Acceptans markeras med ett positivt heltal på stacken, innebärande det accepterade tecknets position i strängen. Detta tal kan då användas för en flervalsrutin med t ex **CASE:** . Icke acceptans markeras med –1 på stacken.

Takara beskriver i den artikel där jag hämtat de här orden ytterligare ord för inmatning, som upptar tolv skärmar till.

I skärmarna #59–#61 återkommer Michael Ham med rent numeriska inmatningsord, sammanfattade i toppordet DIGITS. När det i programmet står **x DIGITS** förväntar sig datorn ett tal med högst x siffror. Ham tycker att man

```
SCR #54
  0 ( Laddskärm för input-ord + dagavstånd som tillämpning )
  1
  2 ö  1 FH  2 FH  THRU  ö char input check
  3 ö  3 FH  4 FH  THRU  ö data entry
  4
  5  5 FH  7 FH  THRU  ö numerisk input m DIGITS
  6  8 FH 17 FH  THRU  ö dagavståndsberäkning ( Julian date )
  7
  8
  9
 10
 11
 12
 13
 14
 15


SCR #55
  0 ( Char input check. FD V/5/19 M Ham  1 av 2 )
  1 ö Input-kontroll-ord med exempel CKxx och GETxx
  2
  3 : OK?     ( n ?--0 el n 1 )  IF 1 ELSE DROP 0 THEN  ;
  4 ö ger 0 el 1 till UNTIL i input-loop och droppar ej accept. data
  5 : ECHO    ( n--n )  DUP EMIT  ;
  6 ö visar det accepterade tecknet på skärmen
  7 : CAP     ( n--n )  DUP 96 > OVER 123 < AND IF 32 - THEN  ;
  8 ö förvandlar små bokstäver till stora (obs endast a-z här)
  9
 10 : CKA-E   ( n--n ? )  CAP DUP 64 > OVER 70 < AND  ;
 11 : CKO-9   ( n--n ? )  DUP 47 > OVER 58 < AND  ;
 12 : CKA,1,5  ( n--n ? )
 13    CAP DUP 65 = OVER 49 = OR OVER 5 = OR  ;
 14 ö CKxx accepterar A-E och a-e, 0-9 och A,1,5 respektive
 15 ö kan förstås lätt justeras för speciella kombinationer


SCR #56
  0 ( Char input check. FD V/5/19 M Ham  2 av 2 )
  1 ö Input-kontroll-ord med exempel CKxx och GETxx
  2
  3 : GETA-E ( --n ) BEGIN KEY CKA-E OK? UNTIL ECHO  ;
  4 : GETO-9 ( --n ) BEGIN KEY CKO-9 OK? UNTIL ECHO  ;
  5 : GETA,1,5 ( --n )
  6    BEGIN KEY CKA,1,5 OK? UNTIL ECHO  ;
  7 ö GETxx hämtar in till stack och skärm de tecken som avgränsats
  8 ö med CKxx-orden.
  9 ö Om operatören trycker fel tecken händer ingenting - datorn
 10 ö väntar på bara ett rätt. Tangentbordet verkar låst för andra
 11 ö tecken än de rätta. Ja/Nej svar t ex:
 12 : CKJN   CAP DUP 74 = OVER 78 = OR ;
 13 : GETJN  BEGIN KEY CKJN OK? UNTIL ECHO  ;
 14 : JN?    ( --? ) GETJN 74 =  ;
 15


SCR #57
  0 ( Data entry. FD VII/3/37 TAKARA. 1 av 2  )
  1
  2 : INSTRING     ( str ch--pos el -1 )
  3    0 4 ROLL 4 ROLL       ö duplicate string
  4    OVER + SWAP           ö start/stop adr in string
  5    DO OVER I C@ =        ö search in string, each position ...
  6     IF SWAP DROP 0 LEAVE  ö found: return offset and quit flag
  7     ELSE 1+               ö not found: increment offset pointer
  8     THEN LOOP
  9    IF DROP -1 THEN  ;    ö failure flag, return failure
 10
 11 ö med str förstås här kombinationen "startadr stränglängd" som
 12 ö fås med COUNT från "count-byte-adress"
 13 ö ordet kollar om ch finns i en tidigare lagrad sträng
 14 ö om inte läggs -1 på stack, eljest pos i strängen
 15


SCR #58
  0 ( Data entry. FD VII/3/37 TAKARA. 2 av 2  )
  1
  2 : SELECT       ( str--offset )
  3    BEGIN
  4     OVER OVER KEY  ö dup string, get key
  5     INSTRING       ö see if it is in string
  6     DUP -1 =       ö not found yet
  7    WHILE
  8     DROP           ö drop key and keep trying
  9    REPEAT
 10    >R DROP DROP    ö keep only position offset
 11    R>  ;
 12 ö SELECT kan användas för att acceptera endast input-tecken
 13 ö som finns lagrade i en speciell sträng. Pos i strängen
 14 ö returneras och kan användas för en CASE-rutin, t ex med CASE:
 15 ö Input-tecken som ej finns i strängen accepteras ej!
```

skall acceptera att folk inte alltid är datoruppfostrade, varför hans rutin tar emot bokstaven O/o som nolla och bokstaven L/l som etta. Han tar också emot mellanslag som nolla. I övrigt tar rutinen emot backspace och raderar felaktiga siffror samt tar emot ENTER som signal att talet är klart. Alla andra tangentnedslag betraktas som felaktiga och tecknet tas inte emot. Tangentbordet verkar låst för dessa tecken.

DIGITS lämnar på stacken ett dubbeltal under ett värde som talar om hur många siffror som verkligen matades in. Om färre än fem siffror tas in blir alltid resultatet i enkelprecision. Den nolla som då blir den andra delen av dubbeltalet kan därför droppas.

På skärmarna #62–#71 har jag satt ihop en rutin för dagavståndsberäkning. Den körs med kommandot **DAGAV**, varvid programmet på skärmen frågar efter från-datum och till-datum. Denna inmatning kontrolleras med **DIGITS**, vilket var hela poängen med exemplet från början. Sedan fann jag att det blev roligare ändå. Bl a blev jag ganska förtjust i rutinen med en tabell för radnummer – **PRADER** – som styr formuläret på skärmen. Man kan gå in med radnumret och få tabelläget –stegnumret – eller gå in med stegnumret och få radnumret. Det här ger kontroll- och styrmöjligheter, tycker jag. Från början är den här lösningen avsedd för en editor där jag vid radbyte vill hamna enbart på vissa skärmrader och där vissa kontroller är kopplade till vissa steg i tabellen. Det gällde också att "gå runt", dvs efter sista raden skulle följa första igen. I skärmbildsorden använder jag **AT** som synonym för **GOTOXY**, som jag tycker är för tungt.

De datum-värden som matas in lagrar jag i en matris – **YRMADA. DIGITS** har jag byggt in i **SIF-IN** som gör gränsvärdeskontroller på data – 1/31 för dag, 1/12 för månad och 1/3000 för år. Jag lade till en rutin för editering efter inmatning om man skulle finna att man gjort något fel. När all inmatning är klar sätts sedan beräkningsalgoritmen i arbete. Data hämtas från matrisen, till-datums dagnummer beräknas först och sedan från-datums varefter de subtraheras. Resultatet skrivs ut.

Jag noterar att algoritmen för beräkning tar upp en halv skärm medan hela rutinen kräver 13 skärmar. De tolv är till för att klara av användarens umgänge med maskinen. Och där kan man förstås göra mer ändå:

* kontrollera att olika månader inklusive februari får rätt gränsvärde för dag (28,29,30,31)
* tillåta användaren att vid editering bara behöver mata in de siffror han vill ändra genom att utnyttja de redan inmatade
* osv

Här är det nästan bara fantasin som är gränssättande. Men det finns förmodligen också en gräns för vad som är rimligt!

```
SCR #59
  0 ( Numeric input. FD VI/3/23 M Ham. 1 av 3 )
  1
  2 : BELL 7 EMIT  ;
  3 : BSP  8 EMIT  ;   ö raderar vid backspace
  4
  5 : BS?    ( n--? )  8 = ;   ö ?=1 om backspace-tangent
  6 : CR?    ( n--? ) 13 = ;   ö ?=1 om <ENTER>-tangent
  7
  8 ö gör etta av L/l och nolla av O/o samt mellanslag
  9 : SP->0  ( n--n )  DUP 32 = IF DROP 48 THEN  ;
 10 : L->1   ( n--n )  DUP 76 = OVER 108 = OR IF DROP 49 THEN  ;
 11 : O->0   ( n--n )  DUP 79 = OVER 111 = OR IF DROP 48 THEN  ;
 12 : FIX    ( n--n )  SP->0  L->1  O->0  ;
 13
 14 : STOP?  ( n--n )  DUP  2 = IF ABORT THEN  ;
 15


SCR #60
  0 ( Numeric input. FD VI/3/23 M Ham. 2 av 3 )
  1
  2 : #?      ( n--n ? )      ö ?=1 om n är ASCII för en siffra 0-9
  3    DUP 47 > OVER 58 < AND  ;
  4 : #BSCR? ( n--n ? )      ö ?=1 som #? plus backspace och <ENTER>
  5    #? OVER BS? OR  OVER CR? OR  ;
  6
  7 ö  : GET#BSCR  ( --ascii ) BEGIN KEY FIX #BSCR? UNTIL  ;
  8 : GET#BSCR  ( --ascii )
  9 ö tar in ett tecken med FIX och #BSCR-kontroll
 10    0 BEGIN DROP KEY STOP? FIX #BSCR? UNTIL  ;
 11
 12 : OK?    ( n ?--0 el n 1 )   ö gör ingenting om ?=1
 13 ö tutar  och droppar n om ?=0
 14    IF 1 ELSE BELL DROP 0 THEN  ;
 15



SCR #61
  0 ( Numeric input. FD VI/3/23 M Ham. 3 av 3 )
  1
  2 ö n=max # of digits to collect; m=# of digits entered
  3 : DIGITS       ( n--d m )
  4    DUP 1 + 0       ö loop-gränserna
  5     DO BEGIN GET#BSCR  DUP BS?
  6 ( bs)   IF DROP I IF BSP R> 1- >R ELSE BELL THEN FALSE
  7 ( #+cr) ELSE DUP PAD I + C! DUP CR?
  8 ( cr)    IF DROP TRUE LEAVE
  9 ( siffr) ELSE OVER I = IF DROP BELL FALSE  ELSE EMIT  TRUE
 10         THEN THEN THEN
 11       UNTIL
 12    LOOP  DROP 0 0 PAD 1- (NUMBER) PAD -  ;
 13 ö <ENTER> ensamt registreras som noll med m=noll också
 14 ö obs! lagringen av 13 (=cr) i PAD för att avbryta (NUMBER)
 15 ö obs! manipuleringen av loop-index vid bs


SCR #62
  0 ( Dagavståndsberäkning, skärmrader )
  1
  2 CREATE2 PRADER
  3    16 , 2 , 6 , 8 , 9 , 10 , 12 , 13 , 14 , 16 , 2 ,
  4 9 CONSTANT ANT.PRADER ö ant olika radnr.
  5 ö Extra vid start/slut tillkommer
  6
  7 : éPRAD    ( n1--n2 ) ö n1=steg# i tabell  n2=innehåll
  8    2 * PRADER + @  ;
  9 : RAD.OK?  ( n1--n1 ? )  ö n1=rad# Om n1 finns i PRADER är
 10 ö ?=TRUE=steg# i tabellen. Eljest är ?=FALSE
 11 ö sökningen börjar alltid på steg 1 (alltså ej på 0)
 12 ö högsta rad# hittas därför alltid näst sist i tabell vid sök
 13    FALSE ANT.PRADER 1+ 1
 14    DO OVER I éPRAD = IF
 15       DROP I LEAVE THEN LOOP  ;


TI FORTH --- a fig-FORTH extension      7sep86LL

SCR #63
  0 ( Dagavståndsberäkning, skärmbild  1 av 2  )
  1 ö stack-diagram för alla orden ( kol rad-- )
  2 : DAB-RUB1
  3    AT ." Rutin för dagavståndsberäkning"  ;
  4 : DAB-RUB2
  5    AT ." Du vill beräkna dagavståndet: "  ;
  6 : DAB-FR
  7    AT ." Från" ;
  8 : DAB-TO
  9    AT ." Till" ;
 10 : DAB-AR
 11    AT ." Årtal (nnnn) : "  ;
 12 : DAB-MAN
 13    AT ." Månad  (nn)  : "  ;
 14 : DAB-DAG
 15    AT ." Dag    (nn)  : "  ;
```

Annonser

Säljes:
Fristående minnesexpansion och RS232-gränssnitt säljes. 1 000 kr/st eller högstbjudande.

Jonas Fjellstedt
Albert Målares väg 11 D
183 45 Täby

Säljes:
TI Disk-kontrollkort
2 (ev. 3) diskdrivar

Sören Bernle, tel 042/933 05

Säljes:
Editor/assembler-modul
Compute!'s TI Collection: Volume 1
Programmers Reference Guide to the 99/4A

Mikael Jonsson
046/11 97 00 (efter 18.00)

Säljes:
P g a byte till större dator säljes i paket:
Hemdator TI-99/4A med nätdel, PAL-modulator, bandspelarkabel
Extended BASIC
Demomodul
Litteratur:
TI-99/4A Users Reference Guide
TI-99/4A bruksanvisning
Lär Dig använda TI-99/4A
Div nr av Programbiten

Har för ett år sedan kostat mig 1 100:–. Giv ett bud!

Allan Johnson
Norrby, 156 00 Vagnhärad
tel arb 08/14 53 00, bostad 0156/200 16

Säljes:
Hårdvara
TI-99/4A (2 st) med nätaggregat och modulatorer. Expansionsbox med diskdrive, RS232, 32 K exp-minne och diskkontrollkort.
Extern kontrollbox med extra diskdrive.
Separat nätaggregat för 2 diskdrives.
Bandspelare.
Speech synthesizer.
Joystick Prostick II.
Matrisskrivare Star Gemini DP 510, parallell.
Diskettbox med 36 disketter.
5 kassettband.
13 programmoduler.
Kablar och anslutningar till allt.

Språk
X-Basic, Logo II, TI-Forth 4.5, Editor-Assembler

Program
Multiplan, TI-Writer, Music Maker, Programming Aids, Parsec, Munchman, TI Invaders, Moonsweeper, Tunnels of doom, Adventure (Pirate, Ghost town) + 50–60 andra program på diskett.

Litteratur
Fundamentals of TI-99/4A Assembly Language, Using and programming the TI-99/4A, 24 nummer av 99'er–HCM. Manualer till hårdvara och program.

Pris 6 000:– för hela paketet.

Säljes även i delar.

Bo Carleö,
Vivstavarvsvägen 245, 122 43 Enskede
tel 08/91 68 92

Säljes:
P-kodskort med programvara 650:–
Supersketch ritprogram med bräda 475:–
Michael Öhman, tel 761 51 62

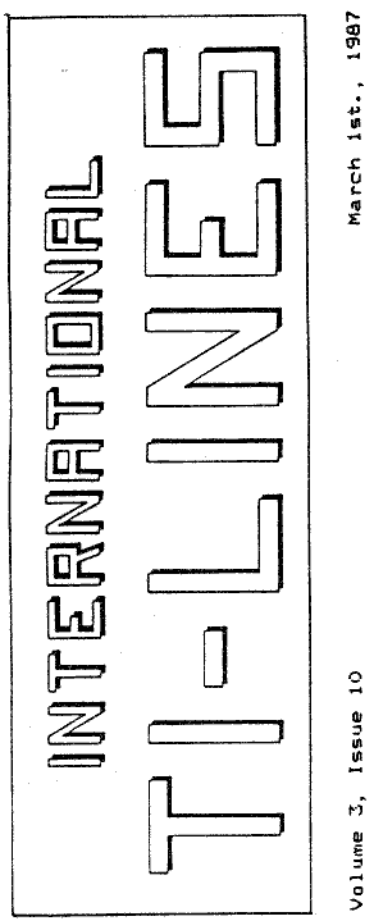
```
SCR #64
  0 ( Dagavståndsberäkning, skärmbild  2 av 2  )
  1 : DAFORM    CLS
  2    3 1 #PRAD DAT-RUB1
  3    3 2 #PRAD DAT-RUB2
  4    3 3 #PRAD DAT-FR     9 3 #PRAD DAT-AR
  5    9 4 #PRAD DAT-MAN    9 5 #PRAD DAT-DAG
  6    3 6 #PRAD DAT-TO     9 6 #PRAD DAT-AR
  7    9 7 #PRAD DAT-MAN    9 8 #PRAD DAT-DAG  ;
  8
  9 : DAT-OK
 10    AT ." Data OK? Tryck då <ENTER>! " CR 3 SPACES
 11       ." Annars valfri tg för editering"  ;
 12 : CLKOM  0 9 #PRAD AT 80 SPACES  ;  ö raderar 2 skärmrader
 13 : DAOKRAD  CLKOM
 14    3 9 #PRAD DAT-OK  ;
 15


SCR #65
  0 ( Dagavståndsberäkning, matris för datumlagring )
  1 6 ARRAY YRMADA
  2 ö från-dag lagras i 0,1,2 (aar,man,dag)
  3 ö till-dag lagras i 3,4,5 (aar,man,dag)
  4
  5 : RAD?>ARRAY  ( --n )
  6 ö n=steg# i YRMADA baserat på cursorns läge på skärmen
  7 ö obs! samband steg# i PRADER/YRMADA ( 3 - ) om skärmen ändras
  8    CURPOS # 40 / RAD.OK? DUP
  9    IF SWAP DROP 3 -    ö beräkna steg# om rad# OK
 10    ELSE ABORT THEN  ;  ö abortera om cursor på fel rad
 11
 12 : TO.YRMADA   ( n-- ) ö n=värde som skall lagras i YRMADA
 13    RAD?>ARRAY YRMADA !  ;
 14
 15


SCR #66
  0 ( Dagavståndsberäkning, inläsningsrutin 1 av 2 )
  1 : BSPS   ( n-- )  ö n=antal backspaces. OBS! Raderar ej!
  2    CURPOS # 40 /MOD ROT MINUS UNDER+ AT ;
  3
  4 : SIF-IN       ( lo hi #-- )
  5 ö lo=lägsta acceptabla värde   hi=högsta acc. värde
  6 ö #=högst antal siffror i talet in  n=det inmatade talet
  7    >R                        ö spara # på R-stacken
  8    BEGIN R SPACES R BSPS     ö radera för talet på skärmen
  9    R DIGITS BSPS DROP DUP    ö läs in talet t stack o skärm. BS
 10    2OVER BETWEEN IF          ö kolla gränsvärden
 11    TRUE ELSE DROP FALSE      ö rigga stacken för UNTIL
 12    THEN UNTIL
 13    R> DROP ROT ROT 2DROP     ö rensa R-stack och P-stack
 14    TO.YRMADA  ;              ö placera talet i YRMADA-tabellen
 15


SCR #67
  0 ( Dagavståndsberäkning, inläsningsrutin 2 av 2 )
  1
  2 : AAR-IN ( --n )     1 3000 4 SIF-IN  ;
  3 : MAN-IN ( --n )     1 12   2 SIF-IN  ;
  4 : DAG-IN ( --n )     1 31   2 SIF-IN  ;
  5
  6 : C>K23        ( n-- )  ö n=steg# i PRADER
  7    23 SWAP #PRAD AT  ;
  8
  9 : DATIDIN      ( n-- )  ö n=3 för FRAN, 6 för TILL
 10    DUP C>K23 AAR-IN  DUP
 11    1 + C>K23 MAN-IN
 12    2 + C>K23 DAG-IN  ;
 13
 14
 15


SCR #68
  0 ( Dagavståndsberäkning, editering )
  1 : DAT-ED
  2    AT ." F för editering av Från-tid, " CR 3 SPACES
  3       ." T för Till-tid, <ENTER> för OK. "  ;
  4
  5 : ED-RAD   CLKOM
  6   3 9 #PRAD DAT-ED  ;
  7
  8 : ED-LOOP
  9   BEGIN FALSE KEY
 10    CASE
 11    70 OF 3 DATIDIN ED-RAD ENDOF
 12    84 OF 6 DATIDIN ED-RAD ENDOF
 13    13 OF DROP TRUE ENDOF
 14    ENDCASE UNTIL CLKOM  ;
 15


TI FORTH --- a fig-FORTH extension      7sep86LL
```

Pressgrannar

Under denna rubrik kommer vi då och då att presentera tidningar i olika länder som skriver om 99'an. I nummer 86-3 presenterades *MICROpendium*, den ledande nyhetstidningen. Den här gången är det dags för TI-Lines, som utges av en kille i Oxford, England. Han heter Peter Brooks och har hållit på med 99'an sen den började säljas i Europa. Tidningen, som är i A5-format och körd på kopieringsmaskin, innehåller 32 sidor och kommer ut varje månad.

Innehållet är blandat med hårdvarutips och programtips. I senare nummer har exempelvis TI-Writer gåtts igenom i detalj. På hårdvarusidan brukar det finnas intressanta bitar. I mars-numret 1987 fanns en artikel om hur man modifierar ett fristående diskkontrollkort så att det kan hantera dubbelsidiga diskdrivar. En tidigare artikel beskrev hur man för en förhållandevis billig kostnad bygger in 32K expansionsminne inuti konsolen. Speciellt det senare tror vi kan vara av intresse för många medlemmar. Hör av dig till föreningen Programbitens adress om du vill veta mera om detta!

Adress till TI-Lines:
Peter G. Q. Brooks
96 Banbury Road
Oxford OX2 6JT
England

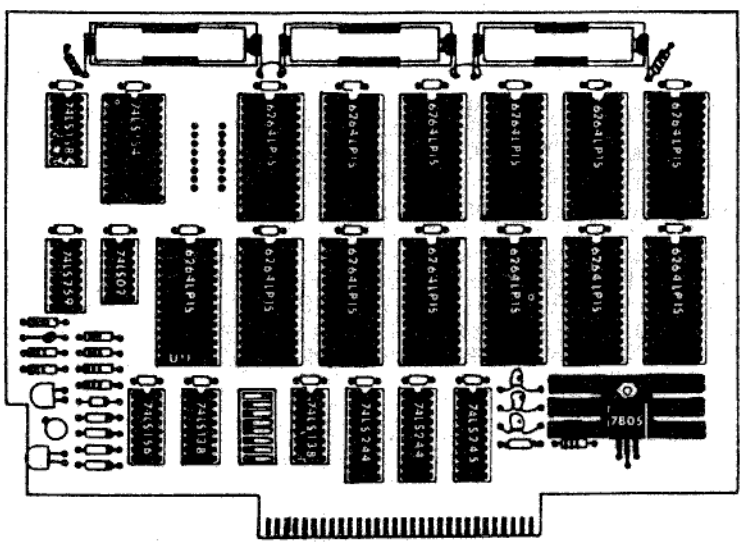



```
SCR #69
  0 ( Dagavståndsberäkning, huvudslinga för inmatning )
  1
  2
  3 : DAGENTRY
  4    DAFORM  FALSE    ö skärmbilden o värde för UNTIL
  5     BEGIN   DUP     ö DUP för UNTIL
  6      3 DATIDIN      ö från-datum
  7      6 DATIDIN      ö till-datum
  8      DAOKRAD        ö info-rad
  9      KEY CR?        ö vänta på en tangent
 10       IF DROP TRUE ö slut direkt om <ENTER>
 11       ELSE ED-RAD ED-LOOP DROP TRUE ö annars edit först
 12       THEN
 13     UNTIL DROP  ;  ö droppa första FALSE
 14
 15


SCR #70
  0 ( Exempel ur FD vol III/5/137 :  JULIAN DATE     )
  1
  2 ö av Peter B. Dunckel, San Francisko
  3 ö JULIAN DATE dvs dagnummer ur dag,månad
  4 ö dagnummer kommer i dubbeltalsform på stacken
  5
  6 : JD      ( mån dag år--dagnr ) ö ex: 3 20 1982 ger 2445049
  7    >R SWAP
  8    DUP 9 + 12 / R + 7 * 4 / MINUS
  9    OVER 9 - 7 / R + 100 / 1+ 3 * 4 / -
 10    SWAP 275 9 */ +
 11    + S->D 1.721029 D+
 12    367 R> M* D+ ';
 13
 14
 15


SCR #71
  0 ( Dagavståndsberäkning, beräkning o utskrift )
  1
  2 : DA.DIFF
  3    4 YRMADA é 5 YRMADA é 3 YRMADA é  JD   ö beräkna till-dag#
  4    1 YRMADA é 2 YRMADA é 0 YRMADA é  JD   ö beräkna från-dag#
  5    DMINUS D+       ö negera från-dag# och addera till o från
  6    D.  CR CR CR ;  ö skriv ut skillnaden
  7
  8 : DAß-KL
  9    AT ." Dagavståndet är " ;
 10 : DA-KLAR   CLKOM
 11    3 9 éPRAD  DAß-KL ;
 12
 13 : DAGAV
 14    DAGENTRY DA-KLAR DA.DIFF  ;
 15
```

RAM-diskbygget

I Programbiten 86-4 (sid 13) fanns ett upprop för att kolla intresset för att bygga RAM-diskar för expansionsboxen. Så småningom fick vi ihop 5 medlemmar och beställde korten från USA. De har nu anlänt och när detta skrives håller vi just på och bygger färdigt korten. Till nästa nummer av tidningen hoppas vi kunna återkomma med en "recension" av kortet och hur det gick med bygget.

Bilden nedan ur byggbeskrivningen.



Hur man styr och ställer

Av Anders Persson

(Artikeln är en fortsättning från Programbiten 86-5)

Tänd och släck

För att kontrollera just PIO-porten används en kombination av de två första metoderna. Vi ska nu ge oss på att tända och släcka vilken lysdiod som helst.

Beväpna dig med Mini Memory. Välj Easy Bug från menyn, och tryck på något så att du kommer förbi hjälpskärmen. Om du inte är bekant med Easy Bug kan det säkert vara bra att läsa anvisningarna i häftet som medföljer Mini Memory.

De kort som sitter i expansionsboxen delar minnesutrymmet >4000->5FFF. Endast ett kort i taget får vara inkopplat. För att det ska gå att skilja på korten har de olika CRU-adresser. På sidan 407 i E/A manualen kan du bland annat se att "Disk controller" har adresserna >1100->11FE och RS232-kortet >1300->13FE. Att adresserna slutar på >FE och inte >FF beror på att alla CRU-adresser är jämna. Den sista biten har ingen betydelse. >1300 och >1301 är alltså samma sak.

Detta innebär att det blir 128 bitar till varje kort. Men normalt används bara ett fåtal av dessa. RS232-kortet utnyttjar ovanligt många, hela 72 stycken.

Gemensamt för alla kort är att den första biten, med adressen >XX00, används till att koppla in kortet med. När en etta skickas ut här blir kortet aktivt, medan en nolla stänger av det igen. Prova själv med styrkortet till skivminnet:

Skriv C1100 ENTER.
Easy Bug svarar med
C1100 =00 ->
Skriv 1 ENTER. Nu kopplas kortet in, vilket syns genom att lysdioden på framkanten tänds.

Nästa adress som programmet kan modifiera är nu >1101. Men eftersom det är likvärdigt med >1100 kan du koppla bort kortet igen utan problem.

Skriv 0 ENTER. Nu ska lysdioden på kortet slockna igen. Genom att trycka på punkt (.) avslutas kommandot i Easy Bug.

Nästa steg blir nu att koppla in RS232-kortet. Eftersom det har adressen >1300, börjar du med C1300. Sedan en etta, ENTER och därefter punkt. Men vad nu? Det lyser ju inte!

På RS232-kortet är lysdioden inte hopkopplad med den bit som aktiverar kortet. Lysdioden har i stället en egen bit, med adressen >130E. Genom att sätta den biten till ett, enligt samma metod som vi använt tidigare, tänds även lysdioden. Du kan släcka den igen om du skickar ut en nolla.

Än så länge har inget hänt med lysdioderna som är anslutna till PIO-porten. Databitarna där har uppenbarligen inte ändrat sig. För att modifiera dem kan vi dock inte använda kommandot "C". Databitarna är nämligen ett exempel på memory-mapped I/O. Alla åtta bitarna påverkas alltså samtidigt, precis som när en byte skrivs i minnet. Adressen till porten är >5000, men avkodningen är inte mer avancerad än nödvändigt. Adresserna >4000->4FFF används av det ROM som finns på kortet. Den enda byte som PIO-porten utgör kan alltså husera i hela området >5000->5FFF. Det gör den också, på så sätt att alla jämna adresser från >5000 till >5FFE motsvarar >5000. Detta faktum underlättar lite när vi manövrerar porten med Easy Bug.

PIO-porten är bidirektionell. Det innebär att det går att både skicka ut och läsa in data. För att vår styrning ska fungera som avsett måste porten vara i läge för utmatning. Riktningen på porten ställs in med en CRU-bit på adressen >1302. Skriv C1302 och kontrollera att biten är en nolla. Noll betyder utmatning. Om så är fallet kan du trycka på punkt med en gång. Annars får du först ge den värdet noll.

Nu kan vi påverka datautgången enligt följande:

Skriv M5000 ENTER och därefter 0 ENTER. Nu ska alla lysdioderna ha slocknat. Dessutom har Easy Bug avancerat till byten med adressen >5001. Eftersom PIO bara reagerar på jämna adresser ska du trycka på ENTER en gång till. Nästa adress blir då >5002, vilken går till PIO-porten.

De värden som läses av från porten nu kan vara lika med det du skrivit ut, men det är inte säkert. Programmet läser visserligen på rätt ställe, men någon egentlig inläsning från omvärlden kan inte ske utan att porten vändes med CRU-biten >1302. Elektriskt sett innebär den läsning som sker nu att ingångar kopplas till ingångar, något som kan ge mystiska resultat.

Genom att skicka ut >01 på porten tänder du den gröna diod som är kopplad till D0. >02 tänder den gula vid D1, >20 den röda vid D5. Om du skriver ut det hexadecimala talet binärt ser du säkert hur det fungerar. Olika kombinationer från >00 till >3F ger olika kombinationer av tända lysdioder. De två bitarna längst till vänster i byten har ingen betydelse, eftersom vi bara kopplat in sex lysdioder, inte åtta.

Strömbrytarna

Nu har vi alltså kommit så långt att vi kan påverka vilka lysdioder som ska vara tända. Men inmatning då?

För att läsa av strömbrytarna får vi återvända till CRU igen. Du kan avbryta M-kommandot med en punkt. Handshake IN har adressen >1304, medan Spare IN finns på >1306. Skriv C1304. Easy Bug svarar då med C1304 =01 ->. Det betyder att det just nu finns en etta på ingången. Nu vill vi inte mata ut något, eftersom den utgången ändå inte är ansluten till någonting. Genom att bara trycka på ENTER kan du läsa av biten en gång till. Programmet svarar då med C1305 =01 ->. Detta är ju samma bit som >1304.

Tyvärr finns det inte någon "på stället marsch", utan adressen räknas upp vid varje avläsning. Men genom att nu trycka på minustecknet backar adressen ett steg, till >1304. Samtidigt läses biten av igen. Du kan alltså omväxlande trycka på ENTER och minustecknet, och på så sätt pendla mellan adresserna >1304 och >1305.

Så länge strömbrytaren inte trycks ner, kommer du att läsa in en etta. Men när brytaren kopplar ihop ingången med jord, returneras en nolla. Prova själv, men tänk på att du måste ha strömbrytaren nertryckt samtidigt som du trycker på ENTER eller minustecknet. Genom att använda adressen >1306 kan du läsa av den andra strömbrytaren i stället.

För att återställa RS232-kortet skall du nollställa CRU-bitarna >1300 och >130E. Om du lämnar kortet tillslaget kan du få problem när du exempelvis vill utnyttja skivminnet.

I/O-programmering

Nästa steg blir nu att automatisera styrningen med ett eget program. För att klara det är det naturligtvis nödvändigt att förstå hur CRU-instruktionerna fungerar. Som jag tidigare konstaterade råder det många olika uppfattningar om detta, de flesta felaktiga. Därför börjar jag med att försöka förklara hur det går till.

Programmeringen omfattas av vissa regler. Dessa kan verka omotiverade och mystiska om man inte vet vad de beror på. För att råda bot på detta ska vi först se på hur det fungerar elektriskt.

Som vi sett har varje bit en adress. Den skickar TMS 9900 ut via adressbussen, logiskt nog. Tre bitar längst till vänster ingår inte i adressen. De bitarna används till att skilja på CRU-operationer och de speciella instruktionerna IDLE, RSET, LREX, CKON och CKOF. De sistnämnda utnyttjas inte i 99/4A, men däremot i Texas mikrodatorserie TM 990. Respektive mnemonic kommer från deras betydelse i de maskinerna. Då de inte har någon funktion att fylla i 99:an, finns det inte heller någon avkodningselektronik för dem.

Därför ska du inte använda de instruktionerna. Datorns kringkretsar kan helt enkelt inte skilja på dem och en CRU-operation. Resultatet blir att någon bit, som händelsevis råkar adresseras, sätts till ett eller noll. Vilket beror på vad som råkar finnas på CPU:ns CRUOUT-ben. Möjligheten att skapa bekymmer är alltså formidabel.

Biten längst till höger används inte heller. Det beror på att TMS 9900 elektriskt sett adresserar 32768 16-bitars ord i minnet, inte 65536 8-bitars bytes. Därför har den bara 15 bitar i adressbussen. Biten längst till höger existerar helt enkelt inte i sinnevärlden. Du kan visserligen använda 16-bitars adresser till olika bytes när du programmerar, men processorn läser alltid två bytes i taget. Sedan använder den den byte du ville ha tag i.

Kvar blir tolv bitar till CRU-adressen, vilket ger just 4096 olika adresser. Den intresserade kan återigen slå upp databoken över TMS 9900. I den finns ett diagram över "CRU interface timing". Där kan du se hur signalerna läggs ut.

Först placeras adressen till biten på adressbussen. Om det är en utmatning läggs en etta eller nolla, vilket det nu gäller, på CRUOUT. Därefter kommer en puls på CRUCLK. Den talar om för elektroniken på adressen ifråga att den ska läsa av CRUOUT. Inläsning fungerar ungefär likadant. När adressen finns utlagd läser CPU:n av CRUIN. Någon puls på CRUCLK ges inte.

En-bits utinstruktioner

När en viss bit ska ändras, ska dess adress först lagras i R12. R12 är alltså ett av de fem register som har en speciell funktion. Sedan kan biten ifråga sättas hög eller låg. Båda varianterna har en egen instruktion. SBO (Set Bit to One) skickar ut en etta, SBZ (Set Bit to Zero) en nolla.

Men de är mer sofistikerade än så. Som operand till instruktionerna ingår ett offset från adressen i R12. Tack vare den möjligheten blir instruktionerna betydligt mer flexibla än de annars skulle varit.

Låt oss ta ett exempel. Vi konstaterade tidigare att RS232-kortets adress är >1300. Diverse andra bitar av intresse finns sedan på högre adresser. Genom att utnyttja ett lämpligt offset kan vi använda samma basadress i R12 hela tiden. Om R12 laddas med >1300, kan du koppla in kortet med SBO 0 och tända lysdioden med SBO 7.

Vadå sju? Adressen till lysdioden var ju >130E!

Jodå, det är riktigt. >130E − >1300 blir visserligen 14 (decimalt). Men det är ju bara varannan adress som verkligen motsvaras av en bit. Därför är lysdioden kopplad till den sjunde biten räknat från >1300. Intelligent nog har Texas valt att låta det offset som hör till instruktionerna multipliceras med två innan det används. Därigenom går det att komma åt 256 olika bitar med samma adress i R12, eftersom operanden får ligga inom området –128..127. Instruktionerna för att koppla in RS232-kortet, tända lysdioden, släcka den igen och koppla bort kortet kan se ut så här:

```
LI  R12,>1300    * Adress till RS232.
SBO 0            * Kort till.
SBO 7            * LED till.
SBZ 7            * LED från.
SBZ 0            * Kort från.
```

Enbits inläsning

Det finns också en instruktion som läser av en bit. Den heter TB (Test Bit). Liksom SBO och SBZ har den ett offset från adressen. När TB exekveras lagras en kopia av det som finns på ingången i "lika-med-biten" i processorns statusregister. Utfallet av testet kan därför kontrolleras med JEQ eller JNE. Om du vill läsa av den brytare som är kopplad till Handshake in kan du göra så här:

```
LI  R12,>1300    * Adress till RS232.
TB  2            * Kolla ingången.
JNE PRESSED      * Hopp om nertryckt.
```

Vår konstruktion jordar ju ingången när knappen trycks ner, varför vi får en nolla in när knappen är påverkad. Därför JNE ovan, i stället för JEQ.

Flerbits CRU-instruktioner

Instruktionerna ovan är praktiska när enskilda bitar ska påverkas. Men om man exempelvis vill läsa in ett helt tecken, åtta bitar, blir det aningen opraktiskt. Därför finns det två andra instruktioner som arbetar med flera bitar i taget. De kan hantera mellan en och sexton bitar i ett svep.

Utmatningsinstruktionen heter LDCR (LoaD Communication Register), medan den för inläsning heter STCR (STore Communication Register). Båda har två parametrar. Den första anger varifrån data kommer, eller var det ska lagras. Den andra anger antalet bitar.

För programmeraren ser det ut som om flera bitar åker in eller ut samtidigt. Men CPU:n skickar dem faktiskt en och en. För varje bit ökas CRU-adressen på adressbussen automatiskt, även om detta inte påverkar R12. Därför är den tid det tar för instruktionen att exekveras beroende på hur många bitar som ska överföras.

Något offset från adressen i R12 finns det inte plats till här. Det finns ju redan två parametrar ändå. Den första får utnyttja de fem olika adresseringssätt som exempelvis MOV kan använda. Den andra är bara ett tal i intervallet 0..15. Noll bitar verkar ju inte så meningsfullt, men nollan används för att ange att sexton bitar ska flyttas.

När det gäller åtta eller färre bitar betraktas adressen som en byte-adress. Nio eller fler bitar ger en ord-adress. Detta är av betydelse för exempelvis adresseringssättet indirekt med autoinkrement. Om adressen gäller en byte, räknas registret upp med ett, annars med två. Processorns statusbit för paritet påverkas endast när överföringen gäller åtta eller färre bitar.

När antalet bitar inte räcker till för att fylla ut en byte eller ett ord, lagras bitarna till höger i byten eller ordet. Vid inläsning (STCR) nollställes de resterande bitarna. Om fem bitar läses in till en byte, kommer alltså de tre bitarna längst till vänster i byten att sättas till nollor.

Om fem bitar från någon byte skickas ut (med LDCR) till CRU-adressen >1300, hamnar biten längst till höger i byten just på adressen >1300. Biten närmast till vänster kommer sedan på >1302. Till sist lagras den femte biten, från höger räknat, på CRU-adressen >1308.

Just den här möjligheten att ge portarna valfri bredd är närmast unik. De flesta andra processorfamiljer har I/O-kretsar som ger en viss bredd på portarna, kanske fyra eller åtta bitar. Tack vare den speciella lösningen här är det närmast bagatellartat att åstadkomma elektronik med portar om exempelvis tre, sju och elva bitar vid sidan om varandra.

Nu vet du tillräckligt för att skriva assemblerprogram som sköter om in- och utmatningarna. Men för att det ska bli mer intressant krävs också programvara som, på något intelligent sätt, bestämmer när och varför de olika lysdioderna ska vara tända. Det går naturligtvis att skriva alltihop i assembler, men du håller nog med om att det finns enklare sätt.

Samarbete med BASIC

Ett sådant är att skriva styrprogrammet i BASIC. Det går bra, under förutsättning att det går att manövrera CRU-bitarna. Nu är det inte möjligt direkt från BASIC, men det är lätt avhjälpt. Två assemblerrutiner kan vara lämpligt att ha, en som manövrerar lysdioderna och en som läser av brytarna.

Med hjälp av Easy Bug har vi redan konstaterat att lysdioderna avspeglar bitmönstret i ett tal som skickas ut på PIO-porten. Detsamma kan åstadkommas från BASIC. Ett underprogram, som vi kan kalla LEDS, tar emot ett tal och skickar ut det på porten. Ett annat program, kallat SWITCH, kan göra tvärtom med strömbrytarna. Deras lägen ger ju också ett bitmönster, om än bara två bitar långt. Låt oss se hur assemblerrutinerna kan se ut.

Nedan följer först en lista som passar till Extended BASIC. För att kunna utnyttja den behöver du alltså minnesexpansion och skivminne. Nästa lista, vars program är avsett att anropas från TI BASIC, kräver dock enbart Mini Memory. Det kan alltså matas in med radassemblatorn.

```
* ASSEMBLERPROGRAM FÖR STYRNING AV PIO
* FRÅN EXTENDED BASIC.
*
* LEDS MEDGER MANÖVRERING AV UTDATABITARNA.
* SWITCH LÄSER AV CRU-LINJERNA FÖR
* HANDSKAKNING OCH RESERV.
*
* A-DATA 861111
*
       DEF LEDS,SWITCH

NUMREF EQU >200C      FÖR ÖVERFÖRING AV PARAMETRAR
NUMASG EQU >2008
XMLLNK EQU >2018
CIF    EQU >20
CFI    EQU >12B8

FAC    EQU >834A
GPLST  EQU >837C

RS232  EQU >1300   CRU BAS FÖR RS232-KORTET
CARD   EQU 0       OFFSET FÖR INKOPPLING AV KORTET
DIR    EQU 1       OFFSET FÖR RIKTNINGSBITEN
LED    EQU 7       OFFSET FÖR LYSDIODEN
LATCH  EQU >5000   MINNESADRESS FÖR DATABITARNA

LEDS
       CLR  R0       HÄMTA PARAMETER ETT (DEN ENDA)
       LI   R1,1
       BLWP @NUMREF
       BLWP @XMLLNK OMVANDLA TILL HELTAL
       DATA CFI

       LI R12,RS232  LADDA CRUBASEN FÖR
*                    RS232-KORTET
       SBO  LED      TÄND LYSDIODEN
       SBO  CARD     KOPPLA IN KORTET
       SBZ  DIR      SÄTT PIO-PORTEN I LÄGE FÖR
*                    UTMATNING
       MOVB @FAC+1,@LATCH  FLYTTA PARAMETERN TILL
*                          PORTEN (ENDAST BYTE)
       SBZ  CARD     STÄNG AV KORTET
       SBZ  LED      SLÄCK LYSDIODEN
       CLR  @GPLST   ÅTER TILL X-BASIC
       B    *R11

SWITCH
       LI R12,RS232+4  LADDA CRUBASEN FÖ HANDSKAK-
*                      NINGSBITEN
       SBO LED-2      TÄND LYSDIODEN
       CLR @FAC       STCR NOLLSTÄLLER ENDAST DEN
*                     HÖGRA BYTEN
       STCR @FAC+1,2  LÄS AV BÅDA BRYTARNA
       SBZ LED-2      SLÄCK LYSDIODEN

       BLWP @XMLLNK   OMVANDLA LÄST INFORMATION
       DATA CIF       TILL FLYTTAL
       CLR  R0        SKICKA TILL X-BASIC
       LI   R1,1
       BLWP @NUMASG
       CLR  @GPLST    ÅTER TILL X-BASIC
       B    *R11

       END
```

Motsvarande program för Mini Memory

```
7D00 04C0 CLR R0
7D02 0201 LI R1,1
7D04 0001
7D06 0420 BLWP @>6044
7D08 6044
7D0A 0420 BLWP @>601C
7D0C 601C
7D0E 1200 DATA >1200
7D10 020C LI R12,>1300
7D12 1300
7D14 1D07 SBO 7
7D16 1D00 SBO 0
7D18 1E01 SBZ 1
7D1A D820 MOVB @>834B,@>5000
7D1C 834B
7D1E 5000
7D20 1E00 SBZ 0
7D22 1E07 SBZ 7
7D24 04E0 CLR @>837C
7D26 837C
7D28 045B B *R11
7D2A 020C LI R12,>1304
7D2C 1304
7D2E 1D05 SBO 5
7D30 04E0 CLR @>834A
7D32 834A
7D34 34A0 STCR @>834B,2
7D36 834B
7D38 1E05 SBZ 5
7D3A 0420 BLWP @>601C
7D3C 601C
7D3E 7200 DATA >7200
7D40 04C0 CLR R0
7D42 0201 LI R1,1
7D44 0001
7D46 0420 BLWP @>6040
7D48 6040
7D4A 04E0 CLR @>837C
7D4C 837C
7D4E 045B B *R11
```

Med Mini Memory måste du också manuellt lägga in namnen på och adresserna till rutinerna i REF/DEF tabellen. Med adresserna ovan ska följande värden stoppas in.
7FE0: 53 57 49 54 43 48 7D 2A
7FE8: 4C 45 44 53 20 20 7D 00

Dessutom ska värdet på 701E ändras till 7FE0.

Observera det lilla knepet i SWITCH. Eftersom STCR måste ha den rätta adressen i R12, lagras >1304 där, i stället för >1300. Tack vare möjligheten till offset tillsammans med instruktionerna SBO, SBZ och TB, kan lysdioden manövreras lika enkelt ändå.

BASIC-program

Efter inmatning av programmet och ändring av REF/DEF-tabellen i Mini Memory är den versionen klar att användas. När det gäller Extended BASIC laddas programmet in i 8k-delen av minnesexpansionen med CALL INIT::CALL LOAD("DSK2.LIGHT").

Detta under förutsättning att programmet assembleras till objektfilen LIGHT i DSK2.

I X-BASIC går det naturligtvis att exekvera laddningen i början av programmet, men det innebär att den görs om varje gång programmet startas med RUN. Det är ju onödigt, eftersom SWITCH och LEDS ligger kvar, även efter NEW. Om du då laborerar med olika BASIC-program är det enklast att ladda in assemblerprogrammet från kommandoläge. Sedan låter du det vara kvar i minnet bara.

För Mini Memory gäller det att låta bli CALL INIT! Om den satsen utförs glömmer TI-BASIC att programmet finns i modulen. Det raderas visserligen inte, men du måste stoppa in det i REF/DEF-tabellen på nytt.

Med anledning av detta kommer de program som följer att förutsätta att SWITCH och LEDS finns åtkomliga där TI respektive Extended BASIC förväntar sig att de ska finnas.

Det går, i vanlig ordning, att utnyttja SWITCH och LEDS direkt från kommandoläge, om man så önskar. Prova med att skriva CALL LINK("LEDS",0). Om allt fungerar ska alla lysdioderna slockna. CALL LINK("LEDS",63) tänder dem igen. CALL LINK("LEDS",7) låter endast tre lysdioder vara tända. En röd, en gul och en grön.

CALL LINK("SWITCH",A)::PRINT A ger som resultat läget på brytarna. När ingen knapp är nedtryckt får A värdet tre. Det beror på att båda brytarna ger en etta tillbaks när de inte är påverkade. Prova att hålla ner en eller båda knapparna när du låter datorn utföra anropet till SWITCH. Båda knapparna nere ska ge värdet noll, medan en nertryckt ger ett eller två, beroende på vilken som du trycker på.

Nu kan vi skriva ett program som tänder och släcker lysdioderna beroende på vilka knappar som är nertryckta.

```
100 FOR I=0 TO 3
110 READ TABLE(I)
120 NEXT I
130 CALL LINK("SWITCH",PRESSED)
140 CALL LINK("LEDS",TABLE(PRESSED))
150 GOTO 130
160 DATA 63,56,7,0
```

Programmet tänder tre av lysdioderna när en knapp trycks ner. Det enklaste sättet att få en samhörighet mellan nertryckta tangenter och tända lysdioder är just att utnyttja en tabell. En tabell kan också innehålla en hel sekvens som ska genomlöpas. Studera nästa program.

```
100 FOR I=0 TO 7
110 READ TABLE(I)
120 NEXT I
130 FOR I=0 TO 7
140 CALL LINK("LEDS",TABLE(I))
150 FOR DELAY=1 TO 200
160 NEXT DELAY
170 NEXT I
180 GOTO 130
190 DATA 36,38,33,34,36,52,12,20
```

Du känner nog igen signalernas växlingar, eller hur?

Ett sådant här system brukar kallas tabelldrivet. En av fördelarna med det är att det är lätt att ändra. Om du råkar bo i Stockholm, eller i någon annan avkrok, tycker du kanske att den kontinentala ljusväxlingen verkar främmande. Genom att ändra dataföljden i rad 190 till 36,38,33,35,36,52,12,28 får du den svenska varianten. Vi skåningar har ju båda versionerna på nära håll.

För att det ska bli mer likt den verkliga förebilden måste naturligtvis fördröjningen vara olika mellan de olika tillstånden. Men det får du själv ordna. Prova exempelvis med att ha ett tvådimensionellt fält, där den första kolumnen anger vilken kombination av lysdioder som ska lysa. Den andra kolumnen kan då tala om hur länge tillståndet ska behållas.

Genom att låta knapparna motsvara de givare som normalt finns i vägbanan, kan du göra ett BASIC-program som styr ljussignalerna i en gatukorsning. Du har visserligen bara halva uppsättningen med lysdioder och knappar, men det är ju enkelt att parallellkoppla två lysdioder på varje utgång respektive två brytare på varje ingång.

Styr med Forth

Ett eventuellt signaleringsprogram i BASIC får du själv knåpa ihop. Däremot visar jag hur man kan gå tillväga i TI-Forth. Det är samma program som jag tidigare visat på någon träff i Staffanstorp.

För att nu inte lusa ner tidningen med en massa Forth kommer programmet att publiceras i ett utskick från SIG-Forth. När det nu finns en sådan ska den ju utnyttjas.

Utvecklingsmöjligheter

Komplexitetsnivån på den här kretsen är förvisso inte speciellt hög. Men det går naturligtvis att gå vidare. I Staffanstorp i november 1986 visade jag en datorstyrd hiss och en kontrollbox med bland annat en analog joystick. Den kräver dock en A/D-omvandlare, något som inte finns på RS232-kortet. Jag har byggt ett eget I/O-kort med digitala in- och utgångar såväl som analoga ingångar.

Hissen däremot var ansluten till PIO-porten. Ett problem när lite mer avancerade saker ska styras med den porten är att den är bidirektionell. Det finns två bitar som alltid är utgångar och två som alltid är ingångar. Men de åtta databitarna är antingen allihop utgångar samtidigt, eller också ingångar. Hissen krävde åtta bitar ut och åtta bitar in, samtidigt. Det går inte an att tappa bort utgångarna bara för att man vill läsa in något. För att lösa detta byggde jag en expansion utanför porten. Med två åttabitars latchar (LS373), två AND-grindar (LS08) och en inverterare uppnådde jag den önskade effekten. Val av latch samt klocksignal skickar jag ut med de två extra utbitarna.

Det går naturligtvis lika lätt att utöka porten till 16 bitar ut och inga alls in (förutom de två CRU-linjerna). Om du har behov av fler bitar än så, kan du tillgripa multiplexing av adresser och data på bussen. Med den metoden går det att bygga en tämligen enkel konstruktion som ändå ger 1024 bitar, både ut och in, samtidigt. Men då är du nog inte nybörjare längre...

Ny version av c99

En ny version av C-kompilatorn för 99-an har nyligen kommit. Den första versionen beskrevs av Björn Gustavsson i Programbiten 86-3 (sid 14–16). Sen dess har ett par versioner kommit. Den senaste, som kallas version 2.1 (och också, lite förvillande, release 3.0) innehåller en hel del förbättringar. Det finns nu formaterad in- och utmatning med scanf respektive printf. Ytterligare nyheter inkluderar extern, case, switch, do ... while och for.

Programmet finns i Programbanken. Kompilatorn, inklusive dokumentation och biblioteksfunktioner tar upp två disketter. För 100:– får Du alltså c99 med utprintad dokumentation. En tredje diskett med c99-program från olika håll finns också. För att kunna använda c99 måste Du ha Editor/Assembler-modulen, expansionsminne och disksystem. I nästa nummer av tidningen kommer vi att publicera ett c99-program.

Göteborgarna på bettet

WEST 99 heter en nystartad dataklubb i Göteborg. Klubben är öppen för alla som använder TI-99/4A datorn. Vi har även tillgång till en databas i regionen (YARN).

Vi erbjuder alla TI-99 ägare att bli medlemmar i klubben. Du kan söka medlemskap på tre sätt.

1: Skriv ett brev till klubben (adress nedan). Ange Ditt namn, adress och telefonnummer så hör vi av oss.

2: Ring till databasen YARN, tel 031/99 23 58. Gå in i area 19. Följ instruktionerna i första meddelandet och lämna namn, adress och telefonnummer så hör vi av oss med mer information.

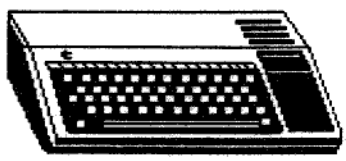
3: Du kan ringa till klubben och anmäla Dig (telefonnummer nedan).

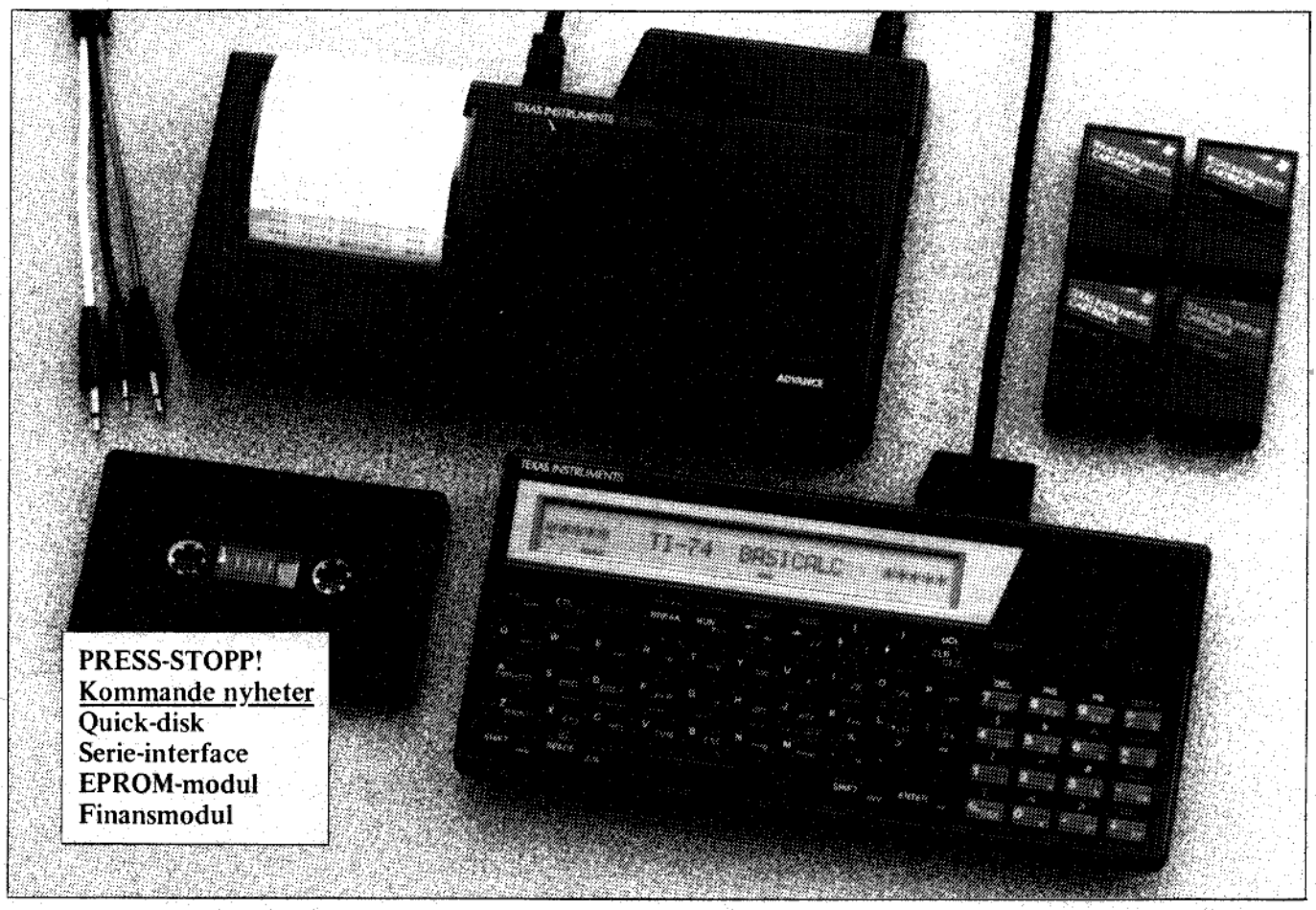
Inriktningen med WEST 99 är att kunna hålla kontakten med andra TI-99 ägare, hjälpas åt med problemlösningar och utbyta programmeringserfarenheter.

Välkomna med Era ansökningar

Dataklubben WEST 99 i Göteborg
Box 8897
402 72 Göteborg

tel 031/91 70 04 Sten
tel 031/28 54 26 Bert





TI-74 Basicalc.
Portföljdator och avancerad räknare.

TI-74 Basicalc är den enda bärbara datorn som kan programmeras både i Basic och Pascal. 8K användarminne som kan utökas till 16K med minnesmodul (extra tillbehör). Minnesmodulen kan även användas för att lagra program.

TI-74 Basicalc har stora, logiskt placerade tangenter och teckenfönster med kontrastkontroll. 113 Basic-kommandon.

TI-74 är också en mycket avancerad teknisk räknare med mer än 200 funktioner för matematik, statistik, teknik och fysik.

Som extra tillbehör finns moduler med färdiga program för matematik och statistik samt modul för inlärning av Pascal.

Kringutrustningen till TI-74 Basicalc omfattar bl a kassettinterface och termisk skrivare med 24 tecken per rad.

TI-74 Basicalc levereras med utförlig bruksanvisning och hårt skyddsfodral.

INTRODUKTIONSERBJUDANDE TILL FÖRENINGEN PROGRAMBITENS MEDLEMMAR!

Vid köp av TI-74 får du kassettinterface utan extra kostnad (värde ca 229:-). Vid samtidigt köp av TI-74 och skrivaren PC-324 får du en modul utan extra kostnad (värde ca 495:-). Ej 8K RAM-modul.

Vi erbjuder även 10% rabatt på vårt övriga Texas Instrument-sortiment.

Jag beställer att levereras mot postförskott

____st TI-74 Basicalc	å	1.595:-
____st skrivare PC-324	å	1.195:-
____st kassettinterface	å	229:-
____st statistikmodul	å	495:-
____st matematikmodul	å	495:-
____st Pascal-modul	å	495:-
____st 8K RAM-modul	å	795:-

Jag beställer TI-74 och skrivaren PC-324 samtidigt och vill ha _____-modulen utan extra kostnad. Digitalservice bjuder på frakten.

Namn_____

Adress_____

Postnr/Ort_____

Medlemsnummer_____

Beställningen skickar du till

DIGITALSERVICE AB

Box 20024, 104 60 Stockholm
Telefonbeställningar 08-41 60 52