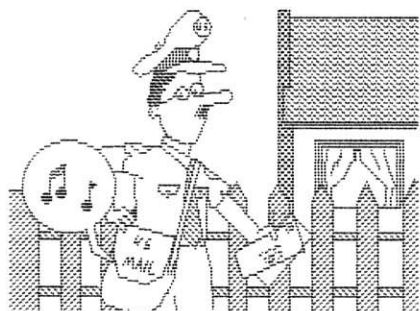
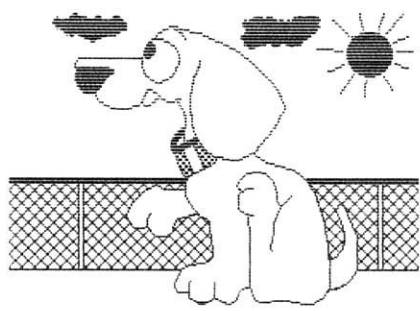
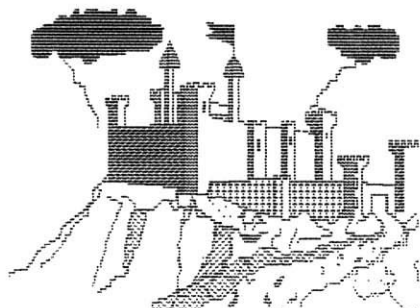


PROGRAM



BITEN

90-4



Medlemsenkät om utrustning	3
Funnelweb 4.30	3
Spell It! 1.01	3
McGoverns XB-skola 5	4-7
Sinus och Cosinus	7
The Missing Link för XB	8-11
Programbanken Fairware	12-15
CALL COINC med AVPC	15
TI-Base version 3.01	15
Myarc HFDC Problem	16-18
Baskonvertering	18
Asgard Reflections	19
Geneve 192k VDP RAM	19
Tigercub Tips #54	20-22
Swedlow XB 12-14	23-25
Multiplan GRAM	25
Hint, Tips, Answers	26-27
TI-74 PC-Interface	28-30
Medlemsmatrikel	31-32

REDAKTÖREN

Det saknas fortfarande insända artiklar från medlemmarna. Redaktören kan inte själv skriva allt i tidningen. Den beredskap som finns i form av utländskt material måste användas flitigt. Innehållet kommer att bli allt tråkigare vartefter det bästa från utlandet har publicerats.

Du som har gamla egna program, sänd in dessa till tidningen för publicering. Det som du själv tyckte var nytt för fem år sedan har kanske fortfarande stort intresse hos andra medlemmar. Skriv helst med TI-Writer, Funnelweb eller annan ordbehandlare som ger DIS/VAR 80-filer. Du bör helst använda den engelska versionen 1 av TI-Writer eftersom den svenska version 2 ej kan läsas av Funnelweb som redaktören använder. TI-Writer version 2 fungerar ej med två simultana diskkontrollkort (Myarc + TI) på grund av en bug i DSR. Du som har kassett kan även skriva in texten i ett Basic-program med en DATA-sats per rad. I undantagsfall kan kortare manus på papper tas emot. Insända flexskivor och kassetter returneras alltid. Om redaktören är på bra humör kanske du får tillbaka dessa med något skojigt program. Tag alltid en egen kopia innan du sänder in dina alster.

DIJIT har slutat tillverka AVPC 80-kolumnskort enligt uppgift från säker källa.

Funnelweb 4.30 DR80 kan eventuellt fungera felaktigt. Det finns per den 9 aug 1990 ingen garanterat felfri version. Lita inte på tidigare daterade uppgifter om att buggarna är fixade.

Myarc HFDC fungerar ej med hårddiskar med fler än 7 huvuden eftersom det endast finns tre trådar för val av huvud enligt Richard Twyning i EAR juli 90. Små hårddiskar använder pinne 2 för "write precompensation", medelstora använder den inte och stora hårddiskar har den som mest signifikant bit för val av huvud.

Redaktör: Jan Alexandersson
Utmanarredaktör: Anders Persson
TI-74 redaktör: Lars Herold Andersen
Programbankir: Börje Häll

Föreningens adress:
Föreningen Programbiten
c/o Schibler
Wahlbergsgatan 9 NB
S-121 46 JOHANNESHOV
Sverige

Postgiro 19 83 00-6
Medlemsavgiften för 1990 är 120:-

Datainspektionens licensnummer:
82100488

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

Övriga annonser kostar 200 kr för hel sida. För lösblad (kopieras av annonsören) som skickas med tidningen gäller 200 kr per blad. Föreningen förbehåller sig rätten att avböja annonser som ej hör ihop med föreningens verksamhet eller ej på ett seriöst sätt gäller försäljning av originalexemplar av program.

För kommersiellt bruk gäller detta: Mångfaldigande av innehållet i denna skrift, helt eller delvis är enligt lag om upphovsrätt av den 30 december 1960 förbjudet utan medgivande av Föreningen Programbiten. Förbudet gäller varje form av mångfaldigande genom tryckning, duplicering, stencilering, bandinspelning, diskettinspelning etc.

Föreningens tillbehörsförsäljning:
Följande tillbehör finns att köpa genom att motsvarande belopp insätts på postgiro 19 83 00-6 (porto ingår)
Användartips med Mini Memory 20:-
Nittinian T-tröja 40:-
99er mag. 12/82, 1-5, 7-9/83(st) 40:-
Nittinian årgång 1983 50:-
Programbiten årgång 84-89(styck) 50:-
TI-Forth manual 100:-
Hel diskett ur programbanken(st) 30:-
Enstaka program 5:- st + startkostnad 15 kr per skiva eller kassett (1 program=20kr, 3 program=30 kr).
Se listor i PB89-3 och PB90-4.

Telefon 08-771 05 69
Manustopp 1 okt(nr5), 15 nov(nr6).

Jan Alexandersson, Springarvägen 5,
3 tr, 142 61 TRÅNGSUND

MEDLEMSENKÄT OM DATAUTRUSTNING

Tillsammans med inbetalningskortet för årsavgiften fyllde medlemmarna i uppgifter om vilken utrustning som fanns. Svaren gällde endast dator-typ, diskkontrollkort och lagrings-media. Två medlemmar har svarat att de har TI med 360 kbytes. I sammanställningen nedan har de räknats som 90 kbytes. Om du har en specialtillverkad PROM som verkligen ger 360 kbytes = 1440 sektorer så skriv och berätta hur det hänger ihop. Av för-eningens 60 medlemmar har 50 besvar-at frågorna. Resultatet blev följ-ande i procent av antalet svarande:

DATORTYP	procent
TI-99/4A	96
Geneve	2
TI-74	8
CC-40	2

LAGRING (kort/skiva)	procent
kassett	20
TI-kontroll 90 kbytes	40
TI-kontroll 180 kbytes	20
Corcomp-kontr 360 kbytes	12
Myarc-kontr Quad EPROM	4
Myarc HFDC	6

Corcomp med 360 kbytes:

Egil Jensen
Bengt Jönsson
Douglas Lundqvist
Sten Gunnarsson
Lars Rogel
Anders Persson

Myarc FDC med Quad EPROM:

Sören Bernle 360 kbytes
Sven Lundgren 720 kbytes 5,25"

Myarc HFDC med hårddisk + flexskiva:

Bengt Jönsson 20MB + 360kB
Jan Alexandersson 20MB + 720kB 5,25"
Bertil Stenfeldt 40MB + 720kB 3,5"

Dator Myarc 9640 Geneve:

Sören Bernle

Dator TI-74:

Harry Nilsson 1,2 Mbytes PC
Anders Persson kassett
Lars Herold Andersen
Thomas Kolk

Dator CC-40:

Harry Nilsson 64 kB Quick Disk

FUNNELWEB 4.30

En ny version 4.30 av Funnelweb finns nu i programbanken. Nyheter:

Funnelweb kan konfigureras så att den laddas med huvudmenyn, User List eller Disk Review. Den senare kan undertryckas genom att hålla nere mellanslag under starten.

Disk review 80 kolumner har nu en fullständig disk manager, sektor-editor, programladdare, titta på alla filer och FW centrala meny. Multifilkopiering kan göras med fyra olika destinationsskivor. Den visar rätt även med 720 kbytes-skivor (liksom 40-kolumnsversionen).

Show Directory och Quick Directory visar även 720 kbytes skivor riktigt men visar samma fillängd som på en mindre skiva (DR visar filerna 1-2 sektorer längre).

Om du har en tidigare version av Funnelweb så bör du spara de gamla filerna SYSCON, UL och DS. DS måste döpas om till D1. DR/DS används nämligen för den nya Disk Review. Om du har äldre teckenfiler så kan du döpa om din svenska teckenfil till C1 (tidigare CHARA1) och din engelska teckenfil till C2 (tidigare CHARA2). Alla andra filer tas från den nya Funnelweb. Glöm inte att det finns hjälpskrmar till konfigureringen som fås fram genom "?".

McGovern har lånat en Myarc HFDC. Corcomp RS 232 fungerar ej tillsam-mans med detta. ■

SPELL IT!

av Jan Alexandersson

Det har kommit ett nytt rättstav-ningsprogram Spell It! från Asgard. Den version 1.01 som jag har funger-ar mycket dåligt. Jag rekommenderar inte köp innan dessa problem är av-klarade. Programmet fungerar inte med långa textfiler. Jag har miss-lyckats med filer som är 55, 75, 87, 89 och 219 sektorer långa. Text-filen blir helt förstörd efter pro-gramkörningen. Kortare filer verkar fungera utan problem. Horizon RAM-disk kan ej användas. Tills vidare köp inte programmet. ■

MCGOVERNS XB-SKOLA, DEL 5

av Tony McGovern, Australien

VII. ACCEPT AT OCH ANDRA UTFLYKTER

TI Extended Basic är ett mycket omfångsrikt språk. XB-modulen innehåller 12 kbytes ROM och drygt 3 (den 4:e är inte full) GROM-bankar på vardera 6 kbytes. Detta är utöver 8 kbytes med konsol-ROM och 6 kbytes konsol-GROM (endast GROM 0 används av XB, GROM 1-2 är endast BASIC som ej används av XB). Det sorgliga med TI-99 är att GROM och GPL överhuvudtaget blev till. Jag gissar att det var TI:s försök att hålla mjukvarumarknaden stängd. Slutresultatet som vi alla känner till är att de sköt sig i båda fötterna med skrämmande träffsäkerhet. Istället för att använda TMS 9900 CRU-adressering för att bank-switcha vanliga ROM eller att bara använda GROM som en källa för att ladda till RAM (som jag förstår görs i P-code kortet), skulle de kunna ha haft en maskin som gjorde rättvisa åt sin CPU, en verklig minidator för hemmabruk men detta är nu redan historia.

Jag har funderat på vad TI skulle ha gjort långt tillbaka när 99/4 konstruerades, vilket kunde ha gjorts utan besvär vid den tiden (eller också när den uppdaterades till 99/4A). Min slutsats är att denna maskin skulle ha getts 4 kbytes snabbt 16-bitars CPU RAM istället för futtiga 256 bytes. Det skulle ha funnits gott om utrymme för några förändringar och/eller bättre avkodning av memory mapped utrustningar (VDP, ljud, tal, GROM). Detta skulle ha betytt att Basic och XB system-adresser, sprite-tabeller, fullständiga skärmbuffertar, strängbuffertar, minnesstack, och så vidare skulle ha funnits i snabbt RAM, och t o m konsol-BASIC kunde ha haft fullt utrymme för tecken- och sprite-definitioner (som t ex TI-LOGO har). Dess moduler skulle då ha kunnat vara avsevärt bättre, låt oss erkänna det, många av de tidigare var helt hopplösa, och de senare har alla begränsats av bristen på hyfsat CPU-RAM. De enda moduler som har klarat sig genom alla år är de som

har använt 32 kbytes expansions-RAM. TI skulle då aldrig ha blivit indraget i marknadskriget till döds (det var TI:s) med den mycket underlägsna maskinen VIC-20. Jag miss-tänker att de 256 byten uppstod eftersom några TI-chefer ville skydda deras befintliga test-kort och lilla minidator-marknad.

Den viktigaste förbättringen som verkligen behövs i XB-underprogram är något sätt att undersöka variabelvärden i godtyckligt underprogram när programkörningen stoppas av BREAK eller felmeddelande. TI borde ha gjort det i XB genom att bibehålla EDIT-kommandot från konsol-Basic, och tillåta det att komma åt underprogram med namn. Någon som lyssnar på detta? Om möjligt lägg till enkla kommandon med matrisoperationer, full syntaxkontroll vid inmatning, 80-kolumnsmöjlighet med formatering, bitmappade skärmfunktioner, snabb programkörning och allt annat kommer att vara en extra bonus. Då skulle alla TI-99/4A ägare vara helt nöjda. Den tråkiga nyheten är att TI har börjat minska stödet för 9900-familjen trots dess utmärkta kvalitet, och således blivit mindre attraktivt för nya konstruktioner. När vi ser tillbaka så har vi fortfarande inte dessa saker i Basic, och eftersom Geneve Basic förblir ofullständig och full med buggar (mitten -89).

Nog med denna utflykt och tillbaka till XB-skolan! Vad är den mest kraftfulla egenskapen hos XB efter SUB och CALL? En bra kandidat är filsystemet, men eftersom detta redan är inbyggt i konsolen, kommer jag ha hålla mig till kommandon som är unika för XB. Huvudkandidaten är ACCEPT AT och dess villkor (även ett enkelt ACCEPT har vissa intressanta förbättringar jämfört med INPUT men detta har beskrivits tidigare). Detta underströks av en lång artikel i en datatidning (mitten -84) om maskinkod för att lägga till denna funktion till IBM PC Basic (som inte heller har underprogram). ACCEPT AT är mycket användbart och kraftfullt,

men har några odokumenterade egenskaper liksom några lömska buggar, vilka är värda att behandlas i denna artikelserie.

Den enklaste nivån av ACCEPT AT kombinerar INPUT-rutinen med dess åtkomst av editeringsfunktionerna, med placering av markören (cursorn) på skärmen med AT-villkoret. Så långt är detta endast inmatningsversionen av DISPLAY AT. Skillnaden mot INPUT är att det inte finns möjlighet till "prompt"-strängar, men ett DISPLAY AT ordnar detta. Det tar dessutom endast emot inmatning av en enstaka variabel och inte av en hel variabellista. Eftersom ACCEPT AT och DISPLAY AT inte skrollar skärmen, kan upprepad användning av dessa ge ett mycket bättre resultat än INPUT när snygg grafik är viktig. Gör dina egna exempel här eller arbeta med XB-manualens exempel. Kom ihåg att markören finns i XB-färggrupp 0 om du försöker att snygga till grafiken.

BEEP tillåter en hörbar prompt med endast en program-byte (vi kommer att behandla programlängd senare om denna artikelserie håller på tillräckligt länge). Naturligtvis kan upprepad användning av BEEP bli litet tröttsamt. ERASE ALL-villkoret ger ett alternativ till CALL CLEAR för att rensa skärmen. Jämfört med CALL CLEAR utförs ERASE ALL långsammare (det tycks ta en rad i taget) men tar mindre programutrymme. Dess resultat är även något annorlunda. Detta lilla program som använder ERASE ALL med DISPLAY kommer att göra både snabbheten och skärmresultatet enkelt att se.

```
100 CALL CLEAR :: CALL COLOR
(0,3,3)
110 FOR I=1 TO 100 :: CALL C
LEAR :: NEXT I
120 FOR I=1 TO 100 :: DISPLA
Y ERASE ALL :: NEXT I
130 CALL SCREEN(11):: FOR I=
1 TO 1000 :: NEXT I
```

Detta är de enkla delarna av ACCEPT AT -- nu börjar det bli intressant. VALIDATE tillåter att programmeraren bestämmer vilka tecken som får användas som svar. Datorn säger "HONK" (det är ordet som används i TI-FORTH) vid otillåtna inmatningar.

Tre förutbestämda typer finns tillgängliga. UALPHA tillåter endast stora bokstäver -- mycket användbart för filnamn och liknande. Detta är inte riktigt samma sak som att trycka ned ALPHA LOCK tangenten eftersom det endast tillåter bokstäver, och således inte kan användas vid inmatning till en numerisk variabel. DIGIT-typen gör precis vad dess namn antyder, och NUMERIC tillåter inmatning av godtyckligt flyttal men även vanliga positiva heltal. Liksom med INPUT, kan alla tal även matas in till en strängvariabel, men numeriska variabler är kinkigare.

Vad gör man om dessa fördefinierade typer inte är riktigt det du önskar? Antag att endast siffrorna 1 till 4 är tillåtna, som i en meny med val mellan fyra saker numrerade 1 till 4. I konsol-Basic skulle extra programrader vara nödvändiga för att undersöka INPUT, men ACCEPT AT hanterar detta med villkoret VALIDATE ("1234") eller VALIDATE(I_LIKE_IT\$) där strängvariabeln tidigare har satts till "1234". För att ange detta litet mera formellt, endast tecknen i sträng-argumentet till VALIDATE kan matas in via tangentbordet med ACCEPT.

SIZE-villkoret tillåter att ACCEPT AT används nästan utan påverkan på skärmutseendet. Det suddar det angivna antalet tecken, tillhanda-håller ett fönster med bestämd längd, och om längden är negativ, kommer befintliga tecken inom fönstret ej att suddas, och väntar på en omedelbar inmatning för godkännande. Detta är mycket praktisk när det gäller att skapa default-val som användaren lätt förstår. Låt oss knappa in ett litet program för att se det viktigaste.

```
100 CALL CLEAR :: DISPLAY AT
(12,1):RPT$("_",28)
200 ACCEPT AT(12,2)SIZE(3):A
$
300 DISPLAY AT(15,2):A$;LEN(
A$)
400 CALL KEY(0,K,S):: IF S>0
THEN 100 ELSE 400
```

Du har sannolikt ALPHA LOCK nedtryckt. Om så är fallet släpp upp den, och kör vårt lilla program.

Tryck bara på ENTER den första gången, och tryck först SPACE nästa gång, och slutligen SPACE innan någon annan tangent tryckes. Detta visar att mellanslag efter det sista godkända tecknet icke tas med. Prova några VALIDATE här också, om du önskar. Som nu programmet är skrivet, ändra SIZE(3) till SIZE(-3). Det accepterar nu vad som fanns inom det 3 tecken breda inmatningsfönstret.

Allt detta var nu mycket enkelt, men det för oss till gränsen av en odokumenterad snårskog. Ändra CALL KEY (0,K,S) till CALL KEY(3,K,S) och kör programmet igen, och mata nu in bokstäver. Lägg märke till vad som händer andra gången. Detta besvarar frågan vilket tangentbord som ACCEPT AT använder -- nämligen som CALL KEY (0,K,S) så att det blir samma tangentbord som senast använts. Prova med delat tangentbord (1-2) på den sista raden. På maskinkodsnivå måste en bestämd byte i CPU scratchpad-RAM sättas för tangentbord innan anropet av SCAN-rutinen göres. Jag tolkar uppförandet så att det visar att de XB-moduler som jag har erfarenhet av inte ändrar denna byte. XB-manualen beskriver emellertid inte detta uppförande överhuvudtaget. Om inte XB var ett dött språk skulle detta vara en signal till försiktighet. I ditt program måste du hålla reda på om det senaste CALL KEY valde det tangentbord du önskar för ACCEPT AT. En positiv sida av detta är att du kan kontrollera ACCEPT AT genom att låta det föregås av ett CALL KEY för att underlätta inmatningen för användaren. Ett exempel är när programmet frågar efter filnamn, så kommer tangentbord 3 alltid att ge stora bokstäver tillsammans med övriga tecken. Brian Rutherford, HV99, var den förste som tipsade mig om detta besynnerliga uppförande tillsammans med några ytterligare odokumenterade variationer av användningen av ACCEPT AT.

Nu är inte detta alltför illa, men det finns värre. Stoppa in VALIDATE ("123")-villkoret tillsammans med ACCEPT AT och kör programmet. Inget problem med SIZE(3), men SIZE(-3) är kinkigare. Du kan inte mata in o giltiga tecken genom tangentbordet, men icke ändrade "_" slinker igenom. VALIDATE tycks undersöka tecken som

matas in från tangentbordet, och inte när edit-buffertens innehåll överförs till variabeln. Efterföljande blanktecken kommer dock att ignoreras. Sannolikt laddar ett negativt SIZE skärmfönstret till edit-bufferten utan att kontrollera med VALIDATE. Slutligen är detta inte ett stort problem eftersom programmeraren kan kontrollera vad som finns på skärmen innan ACCEPT AT anropas. Det bör påpekas, ytterligare en gång, att XB-manualen inte binder ACCEPT AT till att fungera på detta sätt.

Detta uppförande lämnar en svag punkt i ACCEPT AT som måste betraktas som en bug, som dock kan spåras. Antag att du har en meny med val av nummer 1-4, med default 1 inskrivet i SIZE(-1)-fönstret och ett VALIDATE ("1234")-villkor för att säkerställa riktig inmatning av en numerisk variabel. Kan något överhuvudtaget gå fel? En ondskefull programtestare skulle omedelbart sätta default-värdet med FCTN-1. Ett försök att mata in den tomma strängen med ENTER kommer att skrolla skärmen med ett varningsmeddelande. Detta är inte ett ödesdigert fel, men kan nästan vara det om din bakgrund är en noggrant komponerad grafisk skärm. Att komma runt detta problem är inte svårt, men det bästa sättet kommer även att klara av en ännu värre bug, så jag lämnar detta ett tag. Jag betraktar undertryckning av fel- eller varningsmeddelande med det allmän-giltiga ON ERROR eller ON WARNING som dålig programmeringsteknik. Det bästa säkerhetsnätet är det som aldrig används utan bara testas.

Gå nu tillbaka till det ursprungliga testprogrammet och byt varje A\$ till ett fältelement A\$(2). Default-dimensioneringen (10 element) räcker. Inget ändras. Ändra sedan ditt A\$(2) i ACCEPT AT till A\$(1+1). Nu fungerar det bara om det dessutom finns ett VALIDATE-villkor, men SIZE-fönstret fungerar ej och inmatningen kan även spilla över till nästa rad. Detta kan dock inte användas som ACCEPT från flera rader! Lösningen på detta och det tidigare problemet är samma -- använd alltid ACCEPT till en temporär enkel strängvariabel, och bearbeta sedan svaret, och använd inte ACCEPT di-

SINUS OCH COSINUS

av Kent Edgardh

I grundbasic och X-basic finns ett antal inbyggda numeriska standardfunktioner. De är kortfattat och samlat beskrivna i bruksanvisningen på sidorna 92 t o m 97.

Vill man vid handräkning veta sinusvärdet för något tal är en metod att slå upp det i en tabell. Hittar man inte talet gör man en interpolation t ex mellan de två närmast liggande värdena. Hur gör då vår TI 99:a? Utan att förklara det verkliga tillvägagångssättet borde man kunna förstå att en stor tabell med drygt tiotal decimala siffror inte finns lagrade. Det får bara inte plats. Här sker någon form av approximativ beräkning. Tillräckligt approximativ att det blir användbart med god noggrannhet. Man har en bra källa

rekt med numerisk eller ACCEPT med fältvariabel med index som matematiska uttryck. Båda dessa problem upptäcktes av mina programtestare under skrivandet av TEX-BOUNCE, vilket visar att programtestning inte enbart skall göras av författaren. Samma sak gäller författare av programmeringsspråk!

Underprogrammet CALL ERR misslyckas med att klara fel när DSR-rutinen inte hittar en yttre enhet, som ett försök att komma åt en tom flexskiveenhet. Sättet att komma runt detta är att ha ett andra CALL ERR efter att ytterligare en gång ha försökt anropa en fil som man inte har lyckats öppna. Detta OPEN kan inte stängas med CLOSE utan att programmet kraschar eller att man har aktiverat detta extra steg för att rensa Peripheral Access Block.

Instruktionen ON BREAK NEXT är användbar, i synnerhet i spel, för att stänga av FCTN-4 (BREAK)-tangenten. Ett CALL SOUND som har längre varaktighet än 33, kommer emellertid att kringgå detta. Vad detta beror på är inte klart för en utomstående betraktare. ■

för att få ett trigonometriskt värde för ett godtyckligt tal precis som vid en större kalkylator och betydligt mer lätthanterligt än en tabell i bokform.

Att SIN 0 blir 0 och COS 0 blir 1 lär väl vara bekant. Gör man kommandot PRINT SIN(0) får man utskriften noll och PRINT COS(0) utskriften ett. Studera skillnaden mellan de två utskrifterna. Då upptäcker man att den senare har en vidhängande punkt efter sig. Vad har den där att göra? Skulle det inte räcka med bara ettan? Förklaringen får man när man inser att det är en decimalpunkt. När COS(0) blir beräknat har TI 99-an inte fått precis 1 utan en liten bit därifrån. Hur stor var den missen? Ge därför kommandot PRINT COS(0)-1 så får du differensen mellan beräknat och rätt värde. Det blev fel i 14:e decimalen men hyggligt nära ändå.

Det kan finnas tillämpningar då man inte vill ha något fel alls och för speciellt COS 0. Tänk efter vad du borde vilja ha för värde på SIN (COS(0)-1) och vad du i realiteten får med ett PRINT-kommando. Nu har det kanske blivit en väsentlig skillnad i detta sätt att använda COS-funktionen.

Så här skulle man kunna resonera för att bli av med denna egenhet. Skapa en ny numerisk funktion av DEF-begreppet. Denna ska ha egenskapen att den ger rätt värde även för 0. Dvs COS(X) för $X \neq 0$ men 1 för $X=0$. Jag kallar den nya förbättrade versionen för COSIN(X). DEF-satsen placeras med så lågt satsnummer som möjligt eller åtminstone före första användandet av det i programmet.

DEF COSIN(X) = -(X=0) - (X<>0)*COS(X)

Här skall observeras att vår maskin har den egenheten att lämna -1 för sannt och 0 för falskt när för mig alla andra kända basic-dialekter ger 1 för sannt. Den här tekniken blev antydd i en artikel av Mikael Nordlins i Programbiten.

Hittar du andra värden på numeriska standardfunktioner som inte faller till belåtenhet ändra det bara på motsvarande sätt. ■

THE MISSING LINK FÖR XB

av Jan Alexandersson

UTÖKAD EXTENDED BASIC

Det finns två sätt att utöka Extended Basic med färdiga programsnuttar som du kan anropa i ditt eget XB-program. Du kan använda underprogram som anropas med CALL NAMN(...) eller assemblerprogram som anropas med CALL LINK("NAMN",...).

Det finns en skiva från Tigercub som innehåller ett stort antal sådana underprogram (SUB ... SUBEND). Tigercub säljer den under namnet NUTS & BOLTS. Det bästa sättet är att ladda in dessa underprogram med MERGE i slutet av programmet och anropa dessa med CALL NAMN där de behövs.

UTÖKNING MED CALL LINK

Det finns även utökningar till Extended Basic som gör att du kan anropa assemblerrutiner med CALL LINK. Exempel på sådana program är:

- J&KH SXB
- Oak Tree Display Enhancement Pack.
- Enhanced Display Package (PB 88-1)
- STAR (se PB 88-1)

Dessa program kan växla mellan Graphic1 (32 kolumner) och Text1 (40 kolumner) under programmets körning. Med 40 kolumner finns det dock inskränkningar av valet av kommandon eftersom vissa rutiner i ROM också vill använda den nedre delen av 40-kolumnsskärmen. Om man enbart vill använda text på skärmen är dessa mycket användbara och det går alltid att växla till Graphic1 när man vill blanda text, grafik och färger.

BITMAP-GRAFIK GRAPHIC2

Om du vill komma åt varje punkt (pixel) på skärmen oberoende av skärmen i övrigt måste man använda Graphic2 som inte gärna kan samsas med Basic. Det första program som försökte sig på detta fanns publicerat i 99'er magazine dec 1982. Detta ritade inte på skärmen utan i

den låga 8 kbytes-delen av expansionsminnet. När bilden var färdigritad överfördes den sedan till VDP-minnet för att visas på skärmen. Det var sedan inte möjligt att fortsätta programkörningen utan QUIT var enda möjligheten. Börje Håll har gjort en modifiering av detta program så att en del av VDP-minnet lagras undan innan bilden visas. Sedan stoppas det undanlagrade tillbaka och man är tillbaka i Basic.

Modulen Triton Super-XB version 120 har vidareutvecklat dessa tankar men alla kommandon skriver till höga delen av expansionsminnet (bilden syns ej) utom CALL LINK("SHOW") och CALL LINK("EDIT"). Du måste således låta GCLEAR, MOVE, DRAW, CIRCLE, SQUARE, LABEL, GDUMP, GSAVE och GLOAD skriva mot en skärm som inte syns. När bilden visas med SHOW eller EDIT kan man endast komma tillbaka till Extended Basic genom att trycka på E för EXIT.

THE MISSING LINK (TML)

Det finns nu ett program som kan använda Graphic2 med Extended Basic på ett radikalt nytt sätt. The Missing Link (TML) av Harry Wilhelm säljs av Texaments för USD 24.95 + porto till Europa USD 8. Du måste ha TI-99/4A + Extended Basic + 32 kbytes EM + flexskiveenhet. TML består av en flexskiva och en manual på 30 sidor. En DEMO-skiva kan fås från Jan Alexandersson mot en skiva plus returporto (5 kr eller rabattmärke).

TML arbetar med interrupt och fungerar i Graphic2 under hela programkörningen. När du editerar eller bryter programmet hamnar du alltid i Graphic1 som du känner till från de vanliga XB-programmen. Det finns inget särskilt kommando för att aktivera G2-mode utan så fort du skriver RUN så hamnar du automatiskt i Graphic2. När programmet är slut eller du bryter med CLEAR (FCTN 4) kommer du tillbaka till Graphic1.

Hela skärmen är ordnad så att du an-

vänder den för grafik. Det finns endast individuella punkter 192x240 stycken som man kan tända och släcka. Det finns dock rutiner som överför bokstäver till denna grafik-skärm. När du placerat dessa på skärmen kommer de endast att bestå av enstaka punkter vilket gör att du kan blanda många storlekar och färger på bokstäverna. När de väl har fastnat på skärmen så är de endast grafik-punkter. Det finns således inga vanliga rader och kolumner för bokstäver utan de kan placeras efter pixel i X- och Y-led. Det är fullt möjligt att få 32 rader och 60 kolumner med text på skärmen. Du kan alltid blanda text och grafik hur mycket som helst.

Alla trigonometriska funktioner som SIN och COS fungerar utan problem vilket inte gäller andra program som använder dessa rutiner i datorns ROM som brukar skräpa ner skärmen. Du som har försökt att rita en sinus-kurva i Graphic2 med FORTH känner till problemet.

Du kan använda 16 olika färger med den inskränkningen att inom en linje som är 1 pixel hög och 8 pixel bred kan endast en förgrundsfärg och en bakgrundsfärg användas. Det är själva videoprocessorn 9918A/9929A som sätter denna begränsning. Hela VDP-minnet är radikalt omstuvvat så de vanliga kommandona i XB för att skriva och läsa på skärmen kan inte användas men ersättes av nya kommandon som anropas med CALL LINK.

Du kan välja mellan att köra TML med 16 eller 2 färger. TI eller Corcomp diskkontrollkort kan ha max 3 öppna filer medan Myarc kan ha max 2 filer. Eftersom Graphic2 kräver mycket bildminne har stackutrymmet för variabelnamn, underprogramnamn, strängvariabler och återhopp-adresser minskats (424-5212 bytes):

16 färger + 3 filer ger	424 bytes
16 färger + 1 fil ger	1460 bytes
2 färger + 3 filer ger	4176 bytes
2 färger + 1 fil ger	5212 bytes

Detta kräver att du hushållar med namn (numeriska, sträng och underprogram). Hela expansionsminnet med 24488 bytes är ledigt för programrader och numeriska variabler.

Observera att numeriska variabler har namnet i stackutrymmet och data i programutrymmet. Undvik strängvariabler och om de används bör de nollställas med t.ex. A\$="" när de ej längre behövs.

Ytterligare några bagatellartade inskränkningar finns. Underprogram får ha högst 8 bokstäver i namnet och LINPUT #NR måste användas för DISPLAY-filer (INPUT går inte).

MEDFÖLJANDE EXTRA PROGRAM

Förutom själva TML-programmet som är ett XB-program med gömda maskinkods-rutiner finns följande program:

- DEMO-program som även finns på demonstrationsskivan till TML. Det är ett 97 sektorer stort XB-program.
- PAPER SAVER som visar hur en TI-Writer fil kommer att bli utan att du behöver skriva ut den. Det är ett 36 sektorer stort XB-program.
- CONVERT omvandlar XB från PROGRAM-format till INT/VAR 254-format.
- CHARDEF kan editera FONT-filer så att bokstavsutseendet kan ändras.

CALL LINK MED TML

Det finns 32 olika rutiner som anropas med t.ex. CALL LINK("PRINT", rad,kol,variabel). Alla rader och kolumner är pixel så radnummer är 1-192 och kolumnnummer är 1-240. Observera att 256 ej används eftersom 8 pixel till höger och vänster ej är åtkomliga.

a. skärmen

Det finns CLEAR och COLOR som påverkar hela skärmen. Även CALL SCREEN kan användas.

b. pennan

Det finns PENHUE, PD, PR, PE och PU för att ställa in hur pennan som skriver på skärmen ska fungera.

c. fönster

WINDOW bestämmer ett fönster som

sedan används för text- och grafikkommandona. REWIN gör att grafik kan skrivas utanför fönstret. FILL färgar en rektangel av valfri storlek.

d. text

Du skriver på skärmen med PRINT och gör inmatning med INPUT. Defaultvärdet för bokstäver är 6x8 punkter vilket ger 40 tecken per rad. Du kan ändra detta med CHSIZE till max 8x12 punkter men själva bokstaven är max 8x8 punkter så övriga punkter är blanka. Det finns 5 olika FONT-filer (min 4x6 och max 8x8) på skivan som kan läsas in och användas för PRINT/INPUT. Alla fonter är så att de är placerade högt så att den blanka pixelraden alltid kommer under tecknet. Detta skiljer sig från vanlig Extended Basic som alltid har den tomma pixelraden upptill. Detta kanske försvårar möjligheterna att definiera svenska ÅÄÖ. Den minsta FONT-filen ger möjlighet till 32 rader med 60 kolumner. Du kan blanda alla bokstavsstorlekar på skärmen samtidigt. För text används tecken 32-127 och för sprites tecken 1-32. Sprites har inget default-utseende så de måste alltid definieras med CHAR som även används för att ändra bokstävernans utseende. Det finns även ett FORMAT-kommando som liknar PRINT USING i XB.

PRINT suddar bokstavsytan innan tecknet skrives men du kan ändra detta med CALL LOAD(11110,16,0) så att bokstaven adderas pixel för pixel till tidigare grafik eller bokstäver (OR-funktion). De övriga grafikkommandona fungerar med OR.

Det finns ett fristående CHARDEF-program som kan editera FONT-filerna så att du kan skapa svenska bokstäver.

e. grafik

Det går att använda PIXEL, LINE, CIRCLE och BOX.

f. turtle

Det finns även kommandon för turtle-grafik som används i LOGO. TML kan dock aldrig råka ut för "out of ink". TI LOGO använder Graphic1 med

ett begränsat antal tecken medan TML använder Graphic2 utan begränsningar. Kommandon för TURN, FWD, PUTPEN och GETPEN finns.

g. sprites

Det går att använda 32 sprites samtidigt, även med rörelse vilket TML sköter eftersom detta inte finns inbyggt i videoprocessorn för Graphic2. Det finns kommandon för SPRITE, DELSPR, FREEZE, THAW, SPRPOS och DSTNCE. Även CALL COINC(ALL,C) och CALL MAGNIFY kan användas. Det går att starta många samhörande sprites som ett helt block. Detta gör att du får mycket stora sprites med möjlighet att ha flera färger med t.ex. CALL LINK("SPRITE",901,0,0,0,0,50,50) som startar 9 sammanhängande sprites. Nollorna betyder att dessa värden ej ändras.

h. bilder

Du kan ladda in TI-Artist bilder med LOADP och spara skärmen med SAVEP. Det är valfritt att undertrycka färgfilen _C så att endast pixelfilen _P laddas och sparas. Skärmen kan skrivas till en Epson-kompatibel printer med enkel density med DUMP. För att få en snygg utskrift är det bättre att spara till en disk-fil med SAVEP och sedan använda SMARTCOPY från Alexander Hulpke som skriver med quad density.

PROBLEM

Jag är mycket nöjd med programmet som på ett bra sätt använder de grafikmöjligheter som finns i videoprocessorn. Jag saknar dock ett kommando för att läsa skärmen med t.ex. GETPIX och möjlighet att flytta eller kopiera fönster med t.ex. MOVWIN och COPWIN. Utskrift med quad density vore mycket önskvärt. Sprite nr 2 kan endast användas om sprite nr 1 tidigare har definierats.

Alla grafikkommandon som t.ex. PIXEL ritar både förgrunds- och bakgrundsfärg samtidigt. Detta är ej tekniskt nödvändigt. Jag tycker att man alltid ska läsa bakgrundsfärgen och endast lägga till förgrundsfärgen. Datorn lagrar färger i COLOR TABLE med en gemensam byte för både

förgrund och bakgrund. Första halvan (4 bit = nybble) har förgrunds-färgen och den andra halvan (4 bit) har bakgrundsfärgen. Datorn bör läsa hela byten och byta ut första halvan men lämna bakgrundsfärgen oförändrad. Det kunde även finnas ett kommando som endast ritade bakgrund och behöll befintlig förgrund. Även TI-Artist och alla andra program jag sett ritar för- och bakgrund samtidigt vilket är mycket fånigt. Jag testade ovanstående principer i FORTH för många år sedan så jag vet med säkerhet att det kan fungera

mycket bättre.

TML är gjord för den normala video-processorn 9918A/9929A som finns i TI-99/4A men den fungerar naturligtvis även med 9938-processor i Graphic2 mode. Om du har DIJIT AVPC bör du byta till EPROM PWRUP2A. Version 1 av denna EPROM kommer att sudda de 16 första tecknen av XB-programmet. Du kan kompensera för detta genom att lägga en lång REM-rad först. Version 2 fungerar utan detta problem.

```

100 ! RITA SINUSKURVA TML
130 XSTART=1
140 XMAX=240
150 YSTART=96
160 CALL LINK("CHAR",93,"382
838447C444400")
170 CALL LINK("PRINT",4*8,1,
"* SINUS-KURVA *")
180 CALL LINK("PRINT",6*8,2,
"AMPLITUD (MAX 1)")
190 CALL LINK("INPUT",6*8,15
*8,AMPL)
200 IF ABS(AMPL)>1 THEN 180
210 CALL LINK("PRINT",8*8,2,
"PERIODER (ANTAL)")
220 CALL LINK("INPUT",8*8,15
*8,PERIOD)
230 CALL LINK("PRINT",10*8,2,
"PUNKTAVSTAND")
240 CALL LINK("INPUT",10*8,1
5*8,PKTAVST)
250 IF PKTAVST<=0 THEN 230
260 CALL LINK("CLEAR")
270 CALL LINK("LINE",YSTART,
XSTART,YSTART,XMAX)
280 FOR I=XSTART TO XMAX STE
P PKTAVST
290 CALL LINK("PIXEL",YSTART
-95*AMPL*SIN(PERIOD*2*PI*(I-
XSTART)/(XMAX-XSTART)),I,2,8
)
300 NEXT I
310 CALL SOUND(50,440,0)
320 CALL KEY(5,K,S)
330 IF S<1 THEN 320

```

```

100 ! AUTO SPRITE TML
130 CALL LINK("CLEAR")
140 RANDOMIZE
150 DEF RN=90*RND-45
160 CALL LINK("PRINT",1,1,"A
NTAL SPRITE")
170 CALL LINK("INPUT",1,90,A
NTAL)

```

```

180 IF (ANTAL>32)+(ANTAL<0)T
HEN 160
190 CALL LINK("PRINT",16,1,"
MAGNIFY")
200 CALL LINK("INPUT",16,90,
MAGN)
210 IF (MAGN>4)+(MAGN<1)THEN
190
220 CALL MAGNIFY(MAGN)
230 CALL SCREEN(16)
240 FOR I=1 TO ANTAL
250 CALL LINK("CHAR",I,"FFFF
FFFFFFFFFFFF")
260 CALL LINK("SPRITE",I,I,1
3*RND+2,120,160,RN,RN)
270 NEXT I
280 CALL KEY(5,K,S)
290 IF S<1 THEN 280

```

```

100 ! TML SPRITE TEST
130 CALL LINK("CHAR",9,RPT$(
"F",64))
140 CALL MAGNIFY(2)
150 CALL LINK("SPRITE",1,9,5
,100,100)
160 CALL LINK("SPRITE",2,9,1
3,100,116)
170 CALL LINK("SPRITE",3,9,7
,116,100)
180 CALL LINK("SPRITE",4,9,1
1,116,116)
190 CALL LINK("SPRITE",401,0
,0,0,0,50,50)
200 CALL KEY(5,K,S):: IF S<1
THEN 200 ELSE CALL LINK("DE
LSPR",4)
210 CALL KEY(5,K,S):: IF S<1
THEN 210 ELSE CALL LINK("FR
EEZE")
220 CALL KEY(5,K,S):: IF S<1
THEN 220 ELSE CALL LINK("TH
AW")
230 CALL KEY(5,K,S):: IF S<1
THEN 230

```


PROGRAMBANKEN FAIRWARE

av Börje Häll

Första raden anger diskettnamn, antal disketter och vilken modul som krävs för att köra programmet/programmen. En asterisk (*) anger att det kan krävas olika moduler för olika program på disketten eller att ett annat program krävs.

ALEX-PRK01 1 PRK
Se PB 85-2, PB 85-4 och PB 88-3.

ALEX-STA01 1 ST

ALTUTORIAL 2 EA
Artiklar angående assembler på engelska. Även programexempel.

AMNION1 till AMNION8 8 XB
Diverse program för elektronik, matematik, statistik, fysik mm. Disketterna säljs var för sig. Även enstaka program säljs. Se PB 87-4.

ARCHIVER 1 XB EA TW
Packar/packar upp diskettfiler. Besparing 30-70%, av diskettutrymmet, beroende på filtyp.

AUTO-DIALER 1 EA TW
Automatisk uppringning tillsammans med FAST-TERM.

BASIC SORT 1 XB EA MM
Sorteringsrutiner, skrivna i assembler, för BASIC eller X-BASIC.

BASIC SPEL 1 1
Spelprogram i BASIC.

BASIC SPEL 2 1
Spelprogram i BASIC.

c99 2 EA
Subset av programmeringsspråket C. PB 86-3

c99 EXTRA 1 *
1 spel. Källkod till program för att skriva diskettkatalog. Program för att ladda TI-Writer från Mini Memory, även källkoden.

c99 STUFF 1 *
Diverse program skrivna i C99.

COLIST 1 XB
Listar BASIC-program. Se PB 90-2.

CREATIVE FILING 3 XB
Databashantering. 1000 records på en DS/DD diskett. Kompatibel med MYARC/HORIZON RAM-disk. Konverterar filer till DIS/VAR 80 för TI-Writer. Rapportgenerator, formaterare mm. Grafik för numeriska data. Grafik till printer eller diskett.

DIJIT 1 *
Textfiler från DIJIT. Bl a BBS, frågor och svar, 80-kolumnskort.

DISASSEMBLER 1 EA MM
Disassemblerar program i minnet. Utskrift till skärm, printer eller fil.

DISK AID 1 XB EA
Kan analysera hur sektorerna, på en diskett, används och även editera dessa.

DISK CAT 1 XB EA MM TW
Läser, sorterar och listar upp till 2000 filer.

DISK HACKER 1 XB EA MM
Läser vad som finns mellan sektorerna. CASSTRANS: överför asmprog från disk till kassett. CASSLOAD: Laddar asmprog från kassett. DIS/ASS: Disassembler i BASIC.

DISK MANAGER 1000 1 XB EA TW
Ersätter och är bättre än TI-modulen Disk Manager. PB 86-5

DISK MGR 1000 K 2 EA
Källkod till DISK MANAGER 1000 version 3.8

DISK MANAGER 99 1 XB EA MM
Enkel Disk Manager för BASIC och Extended BASIC. Kommandon kan ges från tangentbordet. PB 86-3, 86-5

DISK UTILITIES 1 XB EA TW
Disk manager, file editor, leta efter strängar, jämför disketter helt eller delvis mm. Möjlighet att skriva kommentarer till varje fil.

E/A MUSIC	1 EA	LINKER LIBRARIAN	1 XB EA
Musik. Amorada, In the mood, Flohwalzer och Charleston.		Underhåll av biblioteksrutiner, som används tillsammans med LINKER.	
EDITOR-80	1	LT-WRITER	1 XB
80 kolumns editor och Disk Review. Kräver 80-kolumnskort och Funnelweb.		Ordbehandling. PB 85-1	
EMUSIC PREPROC	2 XB	MACROASSEMBLER	3 EA
Assemblerhjälpmedel för att skriva musik i X-BASIC. Mycket avancerat men lätt att använda.		Ger möjlighet att använda macron vid skrivning av assemblerprogram.	
ENHANCED DISPLAY	1 XB	MACROASSEMBL NEWS	1 *
Subrutiner i assembler. Används för att förbättra in-utmatning, display mm i X-BASIC. Kan ge 40-teckens skärm. PB 88-1		Diskussion om macron till MACROASSEMBLER. Innehåller även program, bl a ett program för att skriva ut diskettkataloger.	
FAST TERM	1 XB EA MM TW	MASS TRANSFER	1 EA
Terminalemulator.		Överföring av filer via modem.	
FORTH&&99	3 XB	MAXRLE1 till MAXRLE13	
		Bilder till MAXRLEEA och MAXRLEXB. Disketterna säljs var för sig.	
FORTH-DEMO	1 XB EA MM	MAX-RLEEA	1 EA
		Ritar RLE-filer på skärmen. Möjlig- het till utskrift på skrivare. Kan även läsa filer från TI-ARTIST V/2 och GRAPHIX. PB 87-3	
FORIT	1 EA	MAX-RLEXB	1 XB
Ritprogram med möjlighet till ut- skrift på skrivare. Kräver TI- FORTH.		Ritar RLE-filer på skärmen. Möjlig- het till utskrift på skrivare. Kan även läsa filer från TI-ARTIST V/2 och GRAPHIX. PB 87-3	
FRACTAL EXPLORER	2 EA MM	MCOPIER	1 EA
Ritar kartliknande bilder i multicolor mode (Mandelbrot set). Möjlighet till utskrift på printer.		Kopierar disketter. Läger alla filers huvuden först på disketten. Det medför snabbare åtkomst.	
FUNNELWEB	3 XB EA MM TW	MIAMI-BOOT	1 XB
Menyprogram som kan ladda BASIC, X- BASIC och assemblerprogram. Laddar även TI-FORTH. Innehåller TI-Writ- er, assembler, Disk Manager 1000 mm.		Menyprogram. Laddar både X-BASIC- och assemblerprogram.	
GAME FORTH	1 FORTH	MULTIPLAN GRAM	1
		Multiplan för GRAM device som har support för 8K GRAM. Snabbare än originalversionen. GRAM Kracker filformat.	
HORIZON MENU	1 EA TW	MULTIPLAN STANDARD	1 MP
Menyprogram för Horizon RAM-disk. PB 88-1		Multiplan för modul. Snabbare än originalversionen.	
HORIZON MENU K	1 EA	MUSIC MAN	1 XB
Källkod till HORIZON MENU vers. 7.3		Musik.	
KNECHT-1	1 XB	MYARC SETUP	1 EA
Musikprogram. Kräver inte expansionsminne.		Setup-program för MYARC RAM-disk	
KNECHT-2	1 XB		
Musikprogram. Kräver inte exp-minne.			
LINKER	1 XB EA		
Gör om assemblerprogram i DIS/FIX 80-format till programformat. Biblioteksrutiner kan användas.			

NEATLIST	1 XB	S/FOSTER	1 XB
PB 86-3		Musikprogram.	
NITTINIAN 83	1 *	SCREEN DUMP	1 XB EA MM
Samlingsdiskett över program publicerade i Programbiten-Nittinian årgång 1983.		Dumpar skärmen till epsonkompatibel printer. PB 86-3.	
PATSY CLINE	1 XB	SOUTH PACIFIC	1 XB
Musik.		Musik.	
PB-FORTH	1 XB	SPIDER	1
Programmeringsspråket FORTH		Enbart text rörande anpassning mellan egna program och FUNNELWEB.	
PB-FORTH Källkod	3	STAR	1 XB
Källkod till PB-FORTH		53 assemblerrutiner, som anropas från XBasic. PB 88-1	
PERFECT PUSH	1 EA	STARTREK	1 XB
Spel		Musik.	
PERFECT PUSH K	1 EA	SUPERBUG II V1.0	1 XB EA MM
Källkod till PERFECT PUSH.		Debugger för assemblerprogram. Manual finns med på disketten.	
PILOT	2 EA	SUPERBUG II V2.1	1 XB EA MM
Programmeringsspråket PILOT.		Debugger för assemblerprogram. Manualen beställs från programförfattaren.	
PRBASE	2 XB EA MM TW	SUPER DEBUGGER	1 EA
Databashantering. PB 87-4		Texas Instruments debugger för assemblerprogram. PB 85-1	
PROGRAMBITEN 84	1 *	TE2 PROTOCOL MAN	1
Samlingsdiskett över program publicerade i Programbiten årgång 1984.		Protokollmanual till TI Terminal Emulator 2.	
PROGRAMBITEN 85	1 *	TELCO	2 XB EA
Samlingsdiskett över program publicerade i Programbiten årgång 1985.		Terminalemulator.	
PROGRAMBITEN 86	1 *	TEST	2 XB EA MM
Samlingsdiskett över program publicerade i Programbiten årgång 1986.		Texas Instruments program för att testa olika delar av TI99/4A-systemet inklusive olika interface.	
PROGRAMBITEN 87	1 *	TETRIS	1 EA XB
Samlingsdiskett över program publicerade i Programbiten årgång 1987.		Spel. Manual på tyska men den behövs inte.	
PROGRAMBITEN 88	1 *	TEXTLOAD/EA5LOAD	1 XB
Samlingsdiskett över program publicerade i Programbiten årgång 1988.		TI-FORTH	1 EA
PROGRAMBITEN 89	1 *	Programmeringsspråket FORTH	
Samlingsdiskett över program publicerade i Programbiten årgång 1989.		TI-FORTH Källkod	2 EA
PULSAR	1 EA	TI SINGS	1
Många subprogram skrivna i assembler. Används från X-BASIC.		Musik- och sångprogram i Basic.	
RAPID SCROLL	1 XB EA MM TW	TI WRITER	1 TIW EA XB
Läser textfiler i DIS-VAR 80-format. Möjlighet till scrollning både horisontal och vertikalt. PB 88-3		Förbättrad version av R.A.Green.	

TIW TUTORIAL 1
Kort manual till TI-Writer.

TK-WRITER 1 XB EA MM
Version av TI-Writer. Kräver inte
TI-Writermodulen. Funktionen Show
Directory finns inte.

TW/MP UPGRADE 1
Texas Instruments uppgraderingar
till TI-Writer och Multiplan.

UNIV DISASSEMBLER 1 EA MM
Disassembler. Utskrift på skrivare
eller skärm. Disassemblerar program
i minnet eller på diskett. Kan leta
efter strängar. Skrivet i FORTH.

WAGNER1 1 XB
Musik av Richard Wagner.

WAGNER2 1 XB
Musik av Richard Wagner.

WAGNER3 1 XB
Musik av Richard Wagner.

WAGNER4 1 XB
Musik av Richard Wagner.

WIZARD OF OZ 1 XB
Musik ur Trollkarlen från Oz.

XBASIC MUSIK 1 XB
Musikprogram skrivna i XBasic.

XBASIC SPEL 1 1 XB
Spelprogram i XBasic.

XBASIC SPEL 2 1 XB
Spelprogram i XBasic.

XB TOOLS 1 XB
Assemblerrutiner, som anropas från
XBasic. ■

CALL COINC MED AVPC

Ett nytt problem som jag upptäckt är
att CALL COINC(ALL,C) inte fungerar
med DIJIT AVPC. Prova följande som
fungerar perfekt med 9918A/9929A men
ej med AVPC 80-kolumnskort:

```
100 CALL MAGNIFY(4)
110 CALL SPRITE(#1,64,11,100,100)
120 CALL SPRITE
    (#2,64,13,100,100,0,10)
130 CALL COINC(ALL,C)
140 PRINT C
150 GOTO 130
```

Korta program med få sprites upp-
täckar aldrig COINC medan långa pro-
gram som XB-TUTORIAL nr 6 från TI
och ett tyskt spel SPACETAXI nästan
fungerar så att du inte ser problem-
et vid en första titt.

Eftersom DIJIT AVPC är interrupt-
driven så finns det risk att den
inte hinner med att undersöka COINC
tillräckligt ofta. Det märkliga är
att det inte hjälper att lägga
sprites ovanpå varandra och testa
COINC så även om de ligger stilla
och helt överlappar varandra så får
jag ingen COINC.

Eventuellt kanske DIJIT:s interrupt-
rutin läser statusregister #0 kontin-
uerligt. Varje läsning av detta re-
gister nollställer värdet så det
kanske inte finns något värde att
läsa för Extended Basics CALL COINC.

En annan möjlighet är att det finns
en skillnad i 9938-processorn så att
den endast upptäcker det dynamiska
förloppet när en sprite passerar
gränsen till andra sprite och ej det
statiska tillståndet av överlapp-
ning. Jan Alexandersson ■

TI-BASE VERS. 3.01

TI-BASE, ett databasprogram, finns
i version 3.01 från JUNE 13, 1990.
Du som har en tidigare version av
TI-BASE kan uppgradera till 3.01
genom att sända både system- och
tutor-skivan till Texaments till-
sammans med USD 14.95 + porto USD
8.00. Du får även 9 sidor uppdater-
ing av manualen. Ordinarie pris är
USD 24.95 + porto USD 8.00.
Nyheterna består bl.a. av:

- klarar filnamn med 29 tecken
- CLOSE klarar skrivskyddad skiva
- FOR kan även söka inne i sträng
- MODIFY STRUCTURE kan behålla data
- APPEND en databas till en annan
- ny REPORT-generator
- delete av sista pos input-fält OK ■

MYARC HFDC PROBLEM

av Jan Alexandersson

Jag har nu upptäckt flera allvarliga problem med Myarc HFDC hårddiskkontrollkort med EPROM H11 och DM V 1.30. Alla problem gäller användning av flexskivor. Hårddisken fungerar mycket bra och några problem finns inte med denna.

SINGLE DENSITY 180 KBYTES

I en tidigare artikel om HFDC har jag beskrivit fysiska läs- och skrivproblem med single density. Du kan prova detta genom att ta en kopia på en fullpackad DS/SD-skiva och sedan använda Disk Utilities Compare Disk. Nästan varje gång kommer några sektorer att markeras som ej lika. Vid varje körning är det dock olika sektorer som markeras. Något liknande problem finns inte med double och quad density.

DOUBLE DENSITY 360 KBYTES

Jag har nu upptäckt att double density (och även single density) blandar ihop fragmenterade filer så att två olika filer får en sektor gemensam. Problemet fås så här med TI-Writer Load File och Save File eller BASIC OLD och SAVE:

1. Spara filen ADAM med 7 sektorer
2. Spara filen BERTIL med 5 sektorer
3. Spara filen ADAM med 8 sektorer
4. Spara filen CESAR med 8 sektorer
5. Spara filen ADAM med 10 sektorer

Det spelar ingen roll om du använder DIS/VAR 80-filer för TI-Writer eller PROGRAM-filer för BASIC. Det hemska resultatet kan ses med Hard Master som visar sektorerna hexadecimalt:

Diskname=360K	Free=588	Used=18
Filename	Sze	Typ Len FdrNo Clusters
ADAM	A	D/V 80 2 22-27 2C-2D 34-34
BERTIL	5	D/V 80 3 28-2B
CESAR	8	D/V 80 4 2D-33

Sektor >2D används av två olika filer ADAM och CESAR. Detta betyder att den sist sparade filen ADAM är riktig medan den tidigare CESAR har en sektor utbytt och således förstörd.

Du kan enkelt testa dina skivor med DM 1000. Tag katalogen och markera alla filer för kopiering. Du ser till höger hur många sektorer som ska kopieras. Detta skall vara 2 sektorer mindre än det totala antalet använda sektorer som visas upptill. Om du får ett högre värde så är det ett tecken på att skivan har förstörda filer. Ju större värde du får desto fler filer är förstörda.

Om du upprepar ovanstående experiment med Ottawa DM 1000 3.5 eller Myarc DM V 1.30 så kommer allt att fungera perfekt. Vad jag förstår beror detta på att dessa disk manager använder sektor-läs/skriv (underprogram >10) medan TI-Writer och Basic använder fil-läs/skriv (underprogram >14 och >15). Även den nya Funnelweb 4.30 Disk Review 80 använder fil-läs/skriv vilket gör att den får problem på grund av fel i Myarcs EPROM H11.

QUAD DENSITY 720 KBYTES

Även quad density har ett mindre problem som gör att inte alla frigjorda sektorer släpps i sektor 0. Om vi upprepar försöket med filerna ADAM, BERTIL och CESAR med quad density och TI-Writer, BASIC eller FW 4.30 DR80 blir resultatet följande:

Diskname=QD	Free=591	Used=5AF
Filename	Sze	Typ Len FdrNo Clusters
ADAM	B	D/V 80 2 22-27 2C-2D 36-37
BERTIL	5	D/V 80 4 28-2B
CESAR	9	D/V 80 6 2E-35

Samma försök med Myarc DM V 1.30 ger detta:

Diskname=QD Free=591 Used=5AF
 Filename Sze Typ Len FdrNo Clusters

ADAM	A	D/V	80	2	22-27
					2C-2D
					36-36
BERTIL	5	D/V	80	4	28-2B
CESAR	8	D/V	80	6	2E-34

Hard Master anger använda sektorer felaktigt för quad density skivor. Den övre tabellen visar att filerna alltid får ett jämt antal sektorer eftersom en allocation unit är 2 sektorer för QD-skivor. Den nedre tabellen visar att filerna endast får det antal sektorer som behövs. Detta betyder att det finns sektorer som är upptagna enligt sektor noll men som ej används enligt filhuvudet. Om man sedan använder DELETE "DSK..." från Basic så kommer inte dessa sektorer att frigöras. DELETE från Basic av den nedre skivan kommer att ge en tom skiva med Used=4 och Free=2876. Användning av DM V 1.30 ger med samma skiva efter delete av samtliga filer Used=2 och Free=2878.

Katalog för den första TIWR-kopierade QD-skivan ger följande (decimalt):

	FWEB	CALL	FWEB	MYARC
	ED80	DIR	DR80	DM V
ADAM	11	12	12	12
BERTIL	5	6	6	6
CESAR	9	10	10	10
Used	30	30	30	30

Katalog för den andra DM V-kopierade QD-skivan ger följande (decimalt):

	FWEB	CALL	FWEB	MYARC
	ED80	DIR	DR80	DM V
ADAM	10	11	12	12
BERTIL	5	6	6	6
CESAR	8	9	10	10
Used	30	30	30	30

HARDDISK

Jag har inte upplevt några problem med 20 Mbytes hårddisk. Om man gör samma kopieringstest till hårddisk blir resultatet enligt Hard Master:

TEST DIR 364
 Filename Sze Typ Len FdrNo Clusters

ADAM	B	D/V	80	368	7954-7959
					795E-795F
					7968-7969
BERTIL	6	D/V	80	36A	795A-795D
CESAR	9	D/V	80	36C	7960-7967

Jag får samma resultat oberoende vad jag använder för kopiering (TIWR, BASIC eller DM V). Katalog med CALL DIR eller DM V ger detta:

ADAM	11 sektorer
BERTIL	6 sektorer
CESAR	9 sektorer

SECTOR NOLL MED MYARC

Jag fick nyligen en FORTH-skiva med DS/DD som jag inte kunde läsa med min Myarc HFDC. Alla skärmar (även skärm 0) hade normala FORTH-skärmar vilket gjorde att skivan saknade skivhuvud på sektor 0 och 1. Det enda som jag kunde läsa var spår 0 med 18 sektorer som innehåller sektor >00 - >11. Alla övriga sektorer var helt oåtkomliga både med FORTH och en sektoreditor.

Lösningen på detta problem visade sig vara att kopiera in en sektor 0 från en normal DS/DD-skiva. Den tidigare sektor noll hade jag naturligtvis först kopierat till en annan skiva med en sektoreditor. Efter denna lilla ändring kunde jag läsa samtliga sektorer på skivan.

Det visade sig att det endast var tre bytes i sektor 0 som hade någon betydelse för möjligheten att läsa sektorerna:

- byte >A och >B max antal sektorer
 - byte >C sektorer per spår

Det räcker att >A->B har ett större nummer än den sektor du vill läsa. Om >C sätts fel kommer fel sektorer att läsas. Dessa tre bytes ska ha följande värden:

		A B C
SS/SD	90 kbytes	>016809
DS/SD	180 kbytes	>02D009
DS/DD	360 kbytes	>05A012
DS/QD	720 kbytes	>0B4012

Slutsatsen är att alla FORTH-användare bör använda ett normalt skivhuvud på sektor 0-1 och upptaget-markera alla sektorer genom att skapa en fil SCREENS som upptar samtliga sektorer så att man inte av misstag kan lagra någon ytterligare fil på skivan. En sådan FORTH-skiva bör alltid kopieras med en äkta sektorkopierare som t.ex. DM 1000. Använd inte filkopiering eftersom det finns en viktig skillnad mellan TI DM 2 och DM 1000. DM 2 kopierar filhuvudet före datasektorerna medan DM 1000 placerar filhuvud efter datasektorerna. Således kan filhuvudet hamna på olika sektorer beroende på kopieringsprogram så att FORTH-skärmar med låga nummer blir felaktiga.

Det finns flera sätt att åstadkomma en felaktig sektor 0 byte >C:

- Initiera en skiva med DM 1000 och TI-kontrollkort med double density utan att testa resultatet.

- Använda FORTH-skärmar även på sektor 0.

- Formatera med FW 4.30 DR80 som i sina första versioner skriver värdet >12 till byte >C oberoende av format. Detta kommer förhoppningsvis att rättas till i senare versioner.

Om en vanlig skiva med många filer får sektor 0 förstörd kanske du tror att hela skivan är förstörd eftersom du ej kan läsa de flesta sektorerna. Lösningen kan då vara att kopiera sektor 0 från en tom skiva och sedan använda Recover File i DM 1000.

Anders Persson och Sören Bernle hade ett liknande problem med Pascal P-Code kortet och Myarc FDC. Pascal läser aldrig sektor 0 vilket gör att diskkontrollkortet inte vet hur övriga sektorer ska läsas. Du kan läsa om lösningen på detta i:

PB 88-1 sid 8: Myarc DS/DQ-kort
PB 88-3 sid16: Myarc och P-systemet

Vad jag förstår uppför sig Myarc FDC (flexskiva) och Myarc HFDC (flexskiva + hårddisk) på samma sätt när det gäller behovet av en riktig sektor 0.

Ett problem kommer sällan ensamt så det visade sig att SECTOR ONE/40 version 1.1 har en allvarlig bug som gör att den försöker kopiera flera sektor än den klarar av med följd att de kopierade sektorerna blir felaktiga. Programmet klarar endast av >23 sektor och har än spärr för inmatning av >24 - >79 men accepterar underligt nog antalet >80 - >99. SECTOR ONE/80 version 1.2 för 80-kolumnskort fungerar utan detta problem, men varför har den fyra teckens bredd för inmatning när max >23 kan väljas? ■

BASKONVERTERING

(favorit i repris från PB 84-1)

```
100 REM BAS KONVERTERING. 15903005.
110 CALL SCREEN(2)
120 FOR A=1 TO 12
130 CALL COLOR(A,16,2)
140 NEXT A
150 CALL CLEAR
160 PRINT TAB(5);"BAS KONVERTERING!"
170 PRINT
180 INPUT "VILKEN BAS?":A
190 PRINT
200 INPUT " TILL BAS?":B
210 PRINT
220 INPUT " TAL ->":C$
230 FOR I=LEN(C$) TO 1 STEP -1
240 D$=SEG$(C$,I,1)
250 IF ASC(D$)>57 THEN 280
260 E=VAL(D$)
270 GOTO 290
280 E=ASC(D$)-55
290 F=F+(E*A^(ABS(I-LEN(C$))))
300 NEXT I
310 FOR J=INT(LOG(F+0.5)/LOG(B)) TO 0
STEP -1
320 G=INT(F/B^J)
330 F=F-G*B^J
340 IF G>9 THEN 370
350 H$=H$&STR$(G)
360 GOTO 380
370 H$=H$&CHR$(G+55)
380 NEXT J
390 PRINT ":" SVAR:";H$
400 H$=""
410 GOTO 170
```

PRINTER PAPPER

en låda, finns att hämta genom
Kent Edgardh tel. 08-777 2944.

Pappret är tunt ljusbrunt pyjamas med traktormatning. Det har hål på båda sidor för insättning i A4 ringpärm och är liggande A4. ■

ASGARD REFLECTIONS

Tidningen Asgard News har bytt namn till Reflections. Du kan prenumerera för ett år med fyra nummer för USD 20 med flyg till Europa. Adress: Reflections, P.O.Box 10697, Rockville, MD 20849, USA

Ordbehandlingsprogrammet PRESS kommer inte att ges ut eftersom alltför stora problem med att avlusa programmet uppstod. PRESS är helt enkelt för stort för TI-99/4A med dess begränsade minnesutrymme.

Asgard kommer att börja sälja Mechatronic 80 kolumnskort som nu nyttillverkas av en annan tysk firma. Pris USD 200 + flygpost USD 20. Ta reda på om spänningsomvandlaren klarar 220 V före köp.

Priset har sjunkit på Myarcs produkter. Man kan nu hitta en Geneve för USD 400 och en HFDC för USD 200. Skriv och fråga efter priset hos Texaments eller Asgard innan du beställer. Många HFDC kan ej använda den 3:e kontakten avsedd för WDS3.

Asgard Hardmaster har en liten bug som hindrar användning av små hårddiskar på mindre än 20 Mbytes (i undantagsfall även större). Kopiera över programmet till en nyformaterad skiva. Editera den 10:e sektorn av HM-filen. Ändra byte >BC från >1110 till >1A10. För 80-kolumn editera den 11:e sektorn av HM80-filen ändra byte >22 från >1110 till >1A10.

Page Pro har uppdaterats till version 1.6. Du kan få den nya versionen genom att sända programskivan och USD 4 till Asgard Software. ■

GAMMA RAY OBSERVATORY

by Tony McGovern, Australia

A new satellite is to be launched during 1990, the Gamma Ray Observatory. This at 13 or so metric tons will be the heaviest satellite ever launched, and is intended to study short bursts of energetic radiation (gamma rays are about a million times more energetic than UV or soft X-rays in the usual classification of electromagnetic radiation). The

EXPAND YOUR GENEVE TO 192k VDP RAM

by Alexander Hulpke, Germany

This article is meant to show, how to install additional 64k of VDP RAM (expansion RAM) on your Geneve. This RAM can NOT be used to display, but only as a quickly accessible interim storage.

Some programs, for example my YAPP painting program, make use of this RAM, that is standard on the Mechatronic 80 Zeichen Karte and easily upgradable on the DIJIT AVPC.

This Project requires soldering and handling of CMOS Memory chips. You should attempt it ONLY if you are experienced in working with computer hardware. This project is not for beginners. You should have done something similar yet, for example the 32k static RAM addition.

If you are not sure, whether you are able to do it, ask a friend with experience to do it. Mistakes may do irreversible damage to your computer!

* The complete project is not *
* printed here. You can obtain *
* the whole description from *
* Alexander Hulpke. *

Be aware, that you can access the new RAM ONLY by changing VDP R45. Also in G6/G7 mode only every SECOND Byte is used! See 9938 manual for details.

(Geneve-ägare kan ta kontakt med Jan Alexandersson för en kopia av detta) ■

reason it is of interest here is that its extensive computer systems are based on TI SPB-9989 microprocessors, a software compatible member of the 9900 family that powers the TI-99/4a. It is a bit of a cross between a 9995 and a 99000 and if I have the right one in mind is built in bipolar I-squared-L technology. TI-99 assembly programmers would easily recognize what lay behind the general descriptions given of how the satellite systems operate. ■

TIPS FROM THE TIGERCUB

#54

Copyright 1988

TIGERCUB SOFTWARE
156 Collingwood Ave.
Columbus, OH 43213

Distributed by Tigercub Software to TI-99/4A Users Groups for promotional purposes and in exchange for their newsletters. May be reprinted by non-profit users groups, with credit to Tigercub Software.

This program uses the program that writes a program technique to create a program that can be used over and over to create quiz programs.

When you key in the these routines, DON'T change any line numbers!

First key in this routine and run it to create a D/V 163 file named ASCII on the disk in drive 1.

```
100 OPEN #1:"DSK1.ASCII",VARIABLE 163,OUTPUT
110 FOR J=1 TO 125 :: X$=X$&CHR$(J):: X2$=X2$&CHR$(J+125):: NEXT J
120 PRINT #1:CHR$(0)&CHR$(230)&"X$"&CHR$(190)&CHR$(199)&CHR$(125)&X$&CHR$(0)
130 PRINT #1:CHR$(0)&CHR$(240)&"X2$"&CHR$(190)&CHR$(199)&CHR$(125)&X2$&CHR$(130)&"J$"&CHR$(190)&"X$"&CHR$(184)&"X2$"&CHR$(0)
140 PRINT #1:CHR$(255)&CHR$(255)
```

Next, key in this part -

```
220 CALL CLEAR :: CALL SCREEN(5):: FOR SET=1 TO 12 :: CALL COLOR(SET,2,8):: NEXT SET :: DIM L$(250,4)
230 !skip to line 280!
280 READ M$ :: DISPLAY AT(2,14-LEN(M$)/2):M$ :: FOR J=1 TO C :: READ M$ :: DISPLAY AT(6+J,4):J:M$ :: NEXT J
290 DISPLAY AT(12,1):"Category to match? (1-"&STR$(C)&")" :: ACCEPT AT(12,26)SIZE(1)VALIDATE("1234"):M :: IF M>C THEN 290
```

```
300 IF C=2 AND M=1 THEN A=2 :: GOTO 320 ELSE IF C=2 AND M=2 THEN A=1 :: GOTO 320
310 DISPLAY AT(14,1):"Match against (1-"&STR$(C)&")" :: ACCEPT AT(14,21)SIZE(1)VALIDATE("1234"):A :: IF A>C OR A=M THEN 310
320 DISPLAY AT(16,1):"How many choices (2-5)" :: ACCEPT AT(16,25)SIZE(1)VALIDATE("2345"):CH :: IF CH>N-1 THEN 320
330 FOR J=1 TO N :: FOR L=1 TO C :: READ L$(J,L):: NEXT L :: NEXT J
340 X$=SEG$(J$,1,N):: FOR J=1 TO CH :: RANDOMIZE :: X=INT(LEN(X$)*RND+1):: Y(J)=ASC(SEG$(X$,X,1)):: X$=SEG$(X$,1,X-1)&SEG$(X$,X+1,255):: NEXT J
350 Z=INT(CH*RND+1):: IF L$(Y(Z),1)=Y$ THEN 350 ELSE Y$=L$(Y(Z),1)
360 DISPLAY AT(8,1)ERASE ALL:L$(Y(Z),M):: FOR J=1 TO CH :: DISPLAY AT(10+J,4):J:L$(Y(J),A)
370 NEXT J :: DISPLAY AT(23,1):""
380 DISPLAY AT(20,1):"(1-";STR$(CH);")?" :: ACCEPT AT(20,8)SIZE(1)VALIDATE(DIGIT):Q :: IF Q=0 OR Q>CH THEN 380
390 IF L$(Q,M)<>L$(Z,M) THEN 410 :: DISPLAY AT(23,1):"CORRECT!"
400 CALL SOUND(100,659,5):: CALL SOUND(100,784,5):: CALL SOUND(400,1047,5):: GOTO 340
410 DISPLAY AT(23,1):"WRONG!" :: CALL SOUND(300,110,0,-4,5):: GOTO 380
Enter MERGE DSK1.ASCII and then SAVE DSK1.QUIZ,MERGE
Then key in -
100 OPEN #1:"DSK1.QUIZ",VARIABLE 163,INPUT :: OPEN #2:"DSK1.QUIZ/2",VARIABLE 163,OUTPUT
110 FOR J=220 TO 410 STEP 10 :: LINPUT #1:M$ :: CALL LINE(J,LN$)
120 PRINT #2:LN$&CHR$(156)&CHR$(253)&CHR$(200)&CHR$(1)&"1"&CHR$(181)&CHR$(199)&CHR$(LEN(M$))&M$&CHR$(0):: NEXT J
130 PRINT #2:CHR$(255)&CHR$(255):: CLOSE #1 :: CLOSE #2
140 SUB LINE(LN,LN$):: LN$=C
```

```
HR$(INT(LN/256))&CHR$(LN-256
*INT(LN/256)):: SUBEND
```

Run that to convert the merge file QUIZ into another merge file QUIZ/2. Then key this in -

```
100 CALL CLEAR :: CALL SCREE
N(5):: FOR SET=1 TO 12 :: CA
LL COLOR(SET,2,8):: NEXT SET
:: DISPLAY AT(2,5):"TIGERCU
B QUIZWRITER"
110 CALL CHAR(64,"3C4299A1A1
99423C"):: DISPLAY AT(4,1):"
@ Tigercub Software for free
":"distribution - no copying
":"fee may be charged."
120 DISPLAY AT(8,1):"This pr
ogram will write":"multiple-
choice quizzes of":"the cate
gory match type."
130 DISPLAY AT(11,1):"It wil
l accept up to 250":"records
, if memory permits,":"and u
p to 4 categories per":"reco
rd."
140 DISPLAY AT(15,1):"For in
stance, a quiz on the":"tabl
e of elements could have":"t
he element name, its symbol"
:"and its atomic weight."
150 DISPLAY AT(19,1):"The pr
ogram will allow you":"to se
lect which two cate-":"gorie
s to match."
160 DISPLAY AT(23,8):"PRESS
ANY KEY" :: DISPLAY AT(23,8)
:"press any key" :: CALL KEY
(5,K,S):: IF S=0 THEN 160 EL
SE CALL CLEAR
170 DISPLAY AT(2,1):"The Qui
zwriter can be used":"over a
nd over to write any":"numbe
r of different quizzes,"
180 DISPLAY AT(5,1):"and eac
h quiz can be SAVED":"and ru
n again and again."
190 DISPLAY AT(12,1):"Place
a disk in drive 1 with":"eno
ugh space available for":"th
e quiz."
200 DISPLAY AT(15,1):"What f
ilename will you use":"for t
he quiz?":"DSK1." :: ACCEPT
AT(17,6):F$ :: CALL CLEAR
210 OPEN #1:"DSK1."&F$,VARIA
BLE 163,OUTPUT
220 !skip to line 420!
420 DISPLAY AT(8,1):"TITLE O
F QUIZ?" :: ACCEPT AT(10,1):
T$
430 T$=CHR$(147)&CHR$(200)&C
```

```
HR$(LEN(T$))&T$ :: DISPLAY A
T(12,1):"NUMBER OF CATEGORIE
S (2-4)?"
440 ACCEPT AT(12,28)SIZE(1)V
ALIDATE("234"):C :: PRINT #1
:CHR$(0)&CHR$(250)&"C"&CHR$(
190)&CHR$(200)&CHR$(1)&STR$(
C)&CHR$(0)
450 FOR J=1 TO C :: DISPLAY
AT(12+J*2,1):"CATEGORY #";ST
R$(J);" TITLE?" :: ACCEPT AT
(13+J*2,1):C$(J)
460 T$=T$&CHR$(179)&CHR$(200
)&CHR$(LEN(C$(J)))&C$(J):: N
EXT J
470 PRINT #1:CHR$(1)&CHR$(14
)&T$&CHR$(0)
480 DISPLAY AT(2,1)ERASE ALL
:"INPUT DATA":;:"(input END
when finished)"
490 N=N+1 :: Z$="" :: DISPLA
Y AT(6,1):"RECORD #"&STR$(N)
&RPT$(" ",200):: FOR J=1 TO
C :: DISPLAY AT(7+J,1):C$(J)
:: ACCEPT AT(8+J,1)SIZE(20):
Y$
500 IF Y$="END" THEN N=N-1 :
: GOTO 530
510 Z$=Z$&CHR$(200)&CHR$(LEN
(Y$))&Y$&CHR$(179):: NEXT J
520 LN=1000+N*10 :: CALL LIN
E(LN,LN$):: PRINT #1:LN$&CHR
$(147)&SEG$(Z$,1,LEN(Z$)-1)&
CHR$(0):: GOTO 490
530 PRINT #1:CHR$(1)&CHR$(4)
&"N"&CHR$(190)&CHR$(200)&CHR
$(LEN(STR$(N)))&STR$(N)&CHR$
(0)
540 PRINT #1:CHR$(255)&CHR$(
255):: CLOSE #1
550 DISPLAY AT(8,1)ERASE ALL
:"Enter NEW":;:"Enter MERGE
DSK1."&F$;:;:"Enter SAVE DSK1
."&F$;:;:"RUN" :: END
560 SUB LINE(LN,LN$):: LN$=C
HR$(INT(LN/256))&CHR$(LN-256
*INT(LN/256)):: SUBEND
Enter MERGE DSK1.QUIZ/2
and SAVE the result as your
completed QUIZWRITER.
```

This truly remarkable one-line disk cataloger tinygram by John Martin was published in the Jackson County news-letter -

```
1 IF F THEN INPUT #1:A$,A,J,
K :: IF J THEN PRINT A$;TAB(
12);J;TAB(18);SEG$(B$,ABS(A*
2)+1,2);K;TAB(27);A<0 :: GOT
O 1 ELSE RUN ELSE B$="AVDFDV
IFIVPG" :: INPUT "DSK":F ::
```



```

OPEN #1:"DSK"&STR$(F)&".",IN
TERNAL,RELATIVE,INPUT :: GOT
O 1 ! BY JOHN M

```

And an ingenious tinygram version of Wheel of Fortune, in the West Penn newsletter.

```

1 ! *** FORTUNE OF WHEELS **
*       A TINYGRAM       *
* by Mike & Ed Machonis*
*****
2 CALL CLEAR :: INPUT "ENTER
THE MYSTERY PHRASE      ":M$
:: CALL CLEAR :: L=LEN(M$)
3 D$=RPT$(CHR$(30),L):: FOR
J=1 TO L :: IF SEG$(M$,J,1)<
>" " THEN 4 ELSE D$=SEG$(D$,
1,J-1)&" "&SEG$(D$,J+1,L)
4 NEXT J :: PRINT D$
5 T=T+1 :: PRINT "TRY No.";
T:: :: INPUT "TYPE LETTER O
R ENTIRE PHRASE":A$ :: IF LE
N(A$)>1 AND LEN(A$)<L THEN 5
6 W=L+1-T :: IF A$=M$ THEN 9
7 FOR J=1 TO L :: IF SEG$(M$,
J,1)=A$ THEN D$=SEG$(D$,1,J
-1)&A$&SEG$(D$,J+1,L)ELSE 8
8 NEXT J :: PRINT :D$ :: GOT
O 5
9 FOR J=1 TO M :: CALL SOUND
(200+J*10,330+40*J,0):: NEXT
J :: PRINT "YOU WIN ";STR$(
W);",000 WHEELS!"::: :: INP
UT "PRESS ENTER TO PLAY AGAI
N":G$ :: T=0 :: GOTO 2

```

```

100 ON WARNING NEXT :: DISPL
AY AT(3,10)ERASE ALL:"KALKUL
ATOR"::::"Input 1st value an
d Enter."::"Input other value
s preceded"::"by +,-,* or / a
nd Enter." ! by Jim Peterson
101 DISPLAY AT(8,1):"Input =
and Enter to get"::"final re
sult."
110 R=14 :: C=1 :: ACCEPT AT
(12,1):N :: V=N :: F=1 :: N$
=STR$(N):: GOSUB 200
120 ACCEPT AT(12,1)VALIDATE(
"+-*/=",NUMERIC):N$ :: A=POS
("+-*/=",SEG$(N$,1,1),1):: G
OSUB 200 :: IF A=0 THEN 120
:: IF A=5 THEN 160
130 ON ERROR 140 :: N=VAL(SE
G$(N$,2,LEN(N$)-1)):: GOTO 1
50
140 CALL SOUND(200,110,5,-4,
5):: C=C-LEN(N$):: DISPLAY A
T(R,C):"" :: RETURN 120
150 IF A=1 THEN V=V+N :: GOT
O 120 ELSE IF A=2 THEN V=V-N
:: GOTO 120 ELSE IF A=3 THE

```

```

N V=V*N :: GOTO 120 ELSE IF
A=4 THEN V=V/N :: GOTO 120
160 DISPLAY AT(R,C):STR$(V):
: F,V=0 :: GOTO 110
200 DISPLAY AT(R,C):N$ :: C=
C+LEN(N$):: IF C>20 THEN C=1
:: R=R+1 :: RETURN ELSE RET
URN

```

Here is the world's short-est tic-tac-toe game, by R. Walters, converted to a tiny gram by Jim Peterson

```

2 DISPLAY AT(5,1)ERASE ALL:"
LET'S PLAY TIC-TAC-TOE": "T
HE BOARD IS NUMBERED": :TAB
(10);"1 2 3": :TAB(10);"8 9
4": :TAB(10);"7 6 5":
3 A=9 :: GOSUB 8 :: S=B
4 DEF F(X)=X-4+4*SGN(8.5-X)
5 C=F(S+1):: GOSUB 6 :: C=F(
S+3):: GOSUB 6 :: C=F(S+6)::
IF S/2=INT(S/2)THEN 7 :: DI
SPLAY AT(20,1):"I MOVE TO";F
(S+4):"":"THE GAME IS A DRAW
" :: STOP
6 A=C :: GOSUB 8 :: H=B :: I
F H<>F(C+4)THEN 7 ELSE RETUR
N
7 DISPLAY AT(20,1):"I MOVE T
O";F(C+4);"AND WIN!" :: END
8 DISPLAY AT(20,1):"I MOVE T
O";A: "":"WHERE DO YOU MOVE T
O?" :: ACCEPT AT(22,23)VALID
ATE("12345678"):B :: RETURN

```

```

1 ! STRAIGHT-LINE CALCULATOR
TINYGRAM by Jim Peterson
Accepts input such as
6+6-11*2+3/4
2 T,F=0 :: C$="+-*/" :: ACCE
PT AT(12,1)ERASE ALL VALIDAT
E(NUMERIC,C$):F$ :: L=LEN(F$
):: FOR J=1 TO L :: X$=SEG$(
F$,J,1):: P=POS(C$,X$,1):: I
F P=0 THEN 5
3 IF F=0 THEN T=VAL(SEG$(F$,
1,J-1)):: F=1 :: A=J+1 :: P2
=P :: GOTO 5
4 V=VAL(SEG$(F$,A,J-A)):: A=
J+1 :: GOSUB 7 :: P2=P
5 NEXT J :: V=VAL(SEG$(F$,A,
255)):: GOSUB 7 :: DISPLAY A
T(12,L+1):"";STR$(T)
6 DISPLAY AT(24,1):"PRESS AN
Y KEY" :: CALL KEY(0,K,S)::
IF S=0 THEN 6 ELSE 2
7 IF P2=1 THEN T=T+V ELSE IF
P2=2 THEN T=T-V ELSE IF P2=
3 THEN T=T*V ELSE T=T/V
8 RETURN

```

That's all, folks!■

SWEDLOW EXTENDED BASIC

```

X  X  BBBB      # 12 to 14
X X   B  B
  X   BBBB      By
X X   B  B      Jim
X  X  BBBB      Swedlow

```

HEX	DECIMAL
0082 37B5	130 14261
0078 37C2	120 14274
006E 37C6	110 14278
0064 37D3	100 14291

(This article originally appeared in the User Group of Orange County, California ROM)

MORE ON PRINT SEPARATORS

If, in a PRINT USING, you follow the print list with a print separator, your 99/4A will honor it. Consider this:

```

10 FOR I=1 TO 30
20 PRINT USING "## ## " : I, I*I;
30 NEXT I

```

Instead of one column of numbers and their squares (which you would have without the semi-colon after the I*I) you will get four right justified columns on your screen. Also works with printers and with DISPLAY USING.

TERMINAL EMULATOR II ABILITY

With a program in memory and your Speech Synthesizer attached, enter LIST "SPEECH" and see what happens. Strange! (Source: Regina in COMPUTE)

LINE NUMBER TABLE

What, you ask, is a line number table? Glad you asked! When your TI stores a program in memory or on disk, it is broken into two parts: the line number table and the line contents. In this program:

```

100 ! T1
110 FOR I=1 TO 10
120 PRINT I
130 NEXT I :: CALL CLEAR

```

The line number table looks like this:

Note that the line numbers are in decending order. The second number is the memory location of the line contents. For example, the contents of line 100 start at >37D3. The > symbol means that the number is in hex or hexadecimal or base 16.

When you RUN a program your 4A, during pre-scan, reserves memory for variables and other things. That's why you get different responses to SIZE before and after RUNNING a program.

While your computer always keeps the line numbers in decending order, it shifts the memory locations of the line contents around without regard to order as you edit/modify your program.

If you save a program to disk in MERGE format, enter NEW and then MERGE your program back into memory, your line number table should be returned to order. This may or may not have any real advantage, however.

What use is this? It depends. If you want to go looking in memory and learn how TI stores and executes a program, this is the first step.

If not? Well, now you know that each line number takes four bytes of memory (two for the number and two for the memory address). A line separator (::) takes only one byte. Conclusion: multi-line statements save memory!

FILE*PRINT

Many files are stored in DISPLAY VARIABLE 80 format: TI writer files, Editor Assembler source files, text files, documentation files, etc.

This month's program reads your disk for DV80 files and displays a menu on your screen.

You can then print up to 99 copies of any file with a single keystroke. Further, if your printer is GEMINI EPSON compatible, you can activate double strike and/or emphasized print.

If you get strange error messages when you first run this program, change the ON ERROR statements in lines 170 and 270 (twice) to ON ERROR STOP and change the !@P- in line 150 to !. After you have finished debugging, restore these commands.

```
*****
* Programlistningen för FILE*PRINT *
* se PB 89-1 sid 19 under namnet *
* PRINTART *
*****
```

ON PROGRAM FILES

When you catalog a disk, there are five types of files: Program, Display Fixed, Display Variable, Internal Fixed and Internal Variable.

A program file varies from the others as they are a collection of individual records (data) while a Program file is a "memory image".

Memory image means essentially that, when writing to disk, your 4A takes what is in RAM and dumps it to disk. In other words, it is a copy or image of what is in memory.

There are at least three types of program files: Basic and XB programs, Assembly language files saved in program format and data files produced by some modules (Personal Record Keeping, Adventure, etc).

Your computer cannot distinguish between the various types of program files. They all look the same. It is only when you attempt to load and execute that your computer notes the difference (and it often locks up if you load a program file that does not match the application).

Program files (be they programs or

data) load significantly faster than other files. Why? Well, think of a bag of marbels. When in memory image, loading the file into RAM is a matter of pouring the bag from disk to CPU. In any other format, however, you take the marbels one by one from disk to memory. Dumping the bag is simply faster.

Finally, Assembly language programs in mirror image format can be loaded from the Editor/Assembler module or from TI WRITER's option 3. In both cases, the module will auto boot a program entitled DSK1.UTIL1.

IF THEN ELSE WITH GOSUB

I bumped into this while debugging a program. What output would you expect from this:

```
10 FOR I=1 TO 3
20 IF I=2 THEN GOSUB 40
    ELSE PRINT "TEST"; ::
    GOSUB 40
30 NEXT I :: STOP
40 PRINT I :: RETURN
```

You would expect this:

```
TEST 1
2
TEST 3
```

If you enter and run this program, you will actually get this:

```
TEST 1
2
2
TEST 3
```

Why two 2's? Best I can figure is that when I=2, the program flow is thus:

Line 20 tests I and finds that it is 2. We then GOSUB 40, PRINT I and RETURN. Upon return to line 20, we look for the next instruction.

Upon return from the subroutine, your 4A looks for the next instruction. It does not consider the ELSE. Hence, PRINT "TEST" is ignored while the second GOSUB 40 is executed as it follows a line separator.

Either of these line 20's work:

```
20 IF I=2 THEN GOSUB 40
   ELSE GOSUB 40

20 IF I=2 THEN GOSUB 40 ::
   PRINT "TEST" ELSE PRINT
   "TEST" :: GOSUB 40
```

The first one works because there is no line separator after the ELSE so control passes to the next line, as it should.

The second one works because there is a line separator after the GOSUB so the ELSE is functional.

The message here is that when using a GOSUB inside an IF THEN ELSE, do not execute a GOSUB immediately before an ELSE that is followed by a line separator.

AN ALTERNATE FOR ELSE

The following two lines execute in the same manner:

```
20 IF A=3 THEN 200 ELSE A=5

20 IF A=3 THEN 200 :: A=5
```

This only works when the sole code between THEN and ELSE is a line number. (Source: a TIGERCUB program).

TOKENS AND LINE CONTENTS

In the October, 1985 ROM, I covered the line number table. Briefly (if you missed it), it is a list, in RAM, of a program's line numbers and the memory locations of the line contents.

This month, we'll look at how the line contents are stored. Let's take a fairly simple line:

```
IF I=2 OR A=B THEN PRINT "HI"
:: GOTO 120
```

If you dumped this from RAM, it would look like this (the numbers are in HEX):

```
16 84 49 BE C8 01 32 BA 41 BE 42 B0
9C C7 02 48 49 82 86 C9 00 78 00
```

What does this mean? The translation:

HEX Dec Meaning

```
16 22 This line has 22 bytes
84 132 Token for IF
49 73 ASCII for I
```

(For the rest I will omit 'Token for' and 'ASCII for')

```
BE 190 =
C8 200 flag, unquoted value next
01 1 value is one byte long
32 50 2
BA 186 OR
41 65 A
BE 190 =
42 66 B
B0 176 THEN
9C 156 PRINT
C7 199 flag, quoted value next
02 2 value is two bytes long
48 72 H
49 73 I
82 130 :: (line separator)
86 134 GOTO
C9 201 flag, line number next
00 0 High byte of line #
78 120 Low byte of line #
00 0 End of Line marker
```

Tokens do not get you on a bus but do store TI Basic and XB commands as one byte each. This same system is used when a program is saved on disk.

Enjoy! ■

MULTIPLAN GRAM

Multiplan GRAM-versionen från R.A. Green kräver 8 kbytes GRAM-bankar för GRAM 3 och GRAM 4. Alla filer lagras i det format som används av GRAM Kracker. Filerna har 6 bytes med övergripande information innan själva data börjar:

```
RAM2-filen har FF0A 2000 6000
RAM1-filen har FF09 2000 6000
GRAM4-filen har FF05 2000 8000
GRAM3-filen har 0004 2000 6000
```

Detta betyder att den inte kan användas med Maximem. Du måste även ha originalmodulen eftersom alla GRAM-filer ej finns på skivan. ■

HINTS, TIPS & ANSWERS (HTA) - INTRODUCTION

Bill Sponchia
1051 Harkness Avenue
Ottawa, Ontario
Canada K1V 6N9
(member of Ottawa TI-99/4A UG)

This package was totally prepared by myself. The information used was gleaned from various User Group newsletters and magazines that I have read. Please note that you will find NO references as to where I got the information (eg - this tip comes from "XYZ User Group) however right here I will publicly announce that 95% of the contents was taken from newsletters; 4.9% was received by direct communication (eg - verbal) with someone; and the balance is my own findings. I have taken the liberty to edit the information as I saw fit. Many of the items have been reworded, some expanded upon and some reduced. This was all done for the sake of clarity (I hope!).

There are two reasons for not giving credit. The first is to save me file space and typing effort. Typing in who it came from for each item would have been time consuming and taken up more space. The second is that in many cases the origin was not known (eg - I saw the same tip in three newsletters with none giving any type of credit to another source) or the origin given was incorrect (eg - in a couple of cases I saw articles that I had written reproduced in other newsletters. Although my name was given the name of my User Group was improperly given - they stated I was from another Group).

I'm telling you this upfront so that when you later read a "tip" originating from you there should be no anger or hurt feelings. I am not trying to steal your ideas or take credit for myself. When you read that item, be proud that you thought of it and accept the thanks of not only myself but of all TIers.

If you wish a copy of the complete

(and most up-to-date) package please write to either myself or the Ottawa Users' Group. Please note that only requests containing a reasonable payment to cover the costs of diskette and mailing will be filled. All other will be returned or thrown out.

If you feel that you would like to make a contribution for the time and effort put into this package please send your donation to the Ottawa Users' Group.

BACKGROUND ON THE "HTA" PACKAGE:

Originally I started reviewing the various newsletters that we (meaning my Users' Group - The Ottawa TI-99/4A Users' Group) received from other groups and making photocopies of interesting articles that I felt would be of immediate benefit to me or something that I might later possibly want to look into. I then proceeded to file them by subject. In many cases these articles covered one or more pages (such as tutorials, etc.) however I was finding that much of what I had was only one or two line "tips".

Since copying a full page for only a couple of lines was not only wasteful it also proved to be space consuming in my filing cabinet. I therefore decided to put them on the word processor. Well the more I read, the more items I found and the more this grew.

I am now putting it out, not because I think it is the FINAL package, but because I feel that there is enough here to be of value to others. I plan to keep adding. If you have any Hints, Tips or Answers on any subject (please - stick to the TI computer subjects; I do not wish to include a section on Sex) pass in on so that I can include it in future updates. Remember - what may be common knowledge to you could very well be the most important discovery to the person next to you.

BASIC & XB TIPS (HTA) - 1

from Bill Sponchia, Canada

1. XB users with Ram Disks (or equivalent). The following program will load, run and then remove itself from memory and will put you in XB without going to disk drive #1 looking for the program "LOAD".

```
100 CALL INIT::
    CALL LOAD(-31952,255,0,255,0)
110 END
```

2. The following program should be saved in the MERGE format as a handy utility:

```
1 CALL CLEAR
2 OPEN #1:"DSKn.",INPUT,RELATIVE,
  INTERNAL !!n = disk drive #
3 INPUT #1:A$,J,J,K
4 DISPLAY "SIZE=";K;" USED=";J-K
5 FOR LOOP= 1 TO 127
6 INPUT #1:A$,A,J,K
7 IF LEN(A$)=0 THEN 10
8 DISPLAY A$;TAB(12);J
9 NEXT LOOP
10 CLOSE #1
11 INPUT "NAME OF PGM TO DELETE ELSE
  TYPE GO ":DEL$
12 IF DEL$="GO" THEN 15
13 DELETE "DSKn."&DEL$ !!n = disk
  drive #
14 GOTO 11
15 STOP
```

When programming this program can be used to catalog a diskette without getting out and loading in a disk manager. It is also written to allow for the deleting of files from the diskette.

To use just merge in with the program you are presently working on (of course it is assumed you started it at line 100 and did not use lines 1 to 15).

3. To turn off the QUIT key (FCTN =): CALL LOAD(-31806,16) Note: Now you must use "BYE" to quit BASIC and get back to the title screen.

4. In an IF-THEN-ELSE statement if you refer to a variable without any other type of relationship then this means "does not equal zero"

eg - IF X THEN 140 ELSE 100 - means If X does not equal 0 then go to 140 but if it equals 0 then go to 100.

5. When using XB prescan it will enable you to run programs that the computer will normally reject. Two things that I have been able to do are - (a) mixing FOR-TO statements and IF-THEN-ELSE statements under one line number; and (b) input two NEXT's but have only one FOR. Here is the program showing these:

```
100 J=0
110 !@P-
120 FOR J=1 TO 20 :: IF J/2=INT(J/2)
  THEN PRINT J ELSE NEXT J
130 IF J>19 THEN 150
140 NEXT J :: !@P+
150 END
```

6. To automatically return to the Master Title Screen (or Menu for Vermenu users) instead of "END" insert the following:

```
CALL INIT::CALL PEEK(2,A,B)::CALL
LOAD(-31804,A,B)
or
CALL INIT::CALL LOAD(-31804,0,36)
```

7. To restrict a CALL KEY statement to taking only one input at a time, no matter how long a particular key is held down -

```
100 CALL KEY(0,K,S)::IF S<1 THEN
  100
```

By restricting the status to +1 this will overcome the problem of the sometimes repeating key.

8. To get TRUE random numbers install this line into your program:

```
CALL PEEK(-31880,A,B)::CALL
INIT::CALL LOAD(-31808,A,B)
```

9. To erase the program from memory but not erase the screen (and not disturb any assembly routines in lower memory:

```
CALL INIT::CALL LOAD(-31952,
255,231,255,231)
```


GØR EN PC TIL SLAVE OM PC-INTERFACE TIL TI-74

av Lars Herold Andersen, Danmark

Aldrig så snart TI-74 var på gaden, beklagede CC-40 brugere forskellige mangler ved den nye maskine. CC-40 kunne beeepe og havde 7 redefinerbare karakterer. Underprogrammer som peek, poke, load og exec gjorde det muligt at udvikle programmer i dyrets "native code"; TMS7000 assembler. TI-74 har umiddelbart et mere tilknapet væsen. Det går ikke at peeke under klædningen, hverken i Basic eller Pascal-ingen adgang for uvedkommende! I nogen tid har man imidlertid kunnet købe sig fra de fleste af manglerne, og med i prisen får man flere nye features.

PC-Interface er en kombination af hardware og software, som muliggør kommunikation mellem TI-74 og en IBM kompatibel PC XT/AT. (Eller en TI Professional Computer.) Hardware er en lille sort box (2*4*10 cm), som plugges i PC'ens parallelle port (25 ben DIN stik). Boxen har 1 meter hale, og ender i en Dock-Bus connector, som 74 og 95'ere's 10-benede stik kaldes. I sig selv er hardwaren ganske uanvendelig. Software må der til: På 5 1/4" DOS floppy'n, som er i pakken, findes programmel i forskellige kategorier. Jeg beskriver kun den del, som har med Basic at gøre, men PC-Interface kan også anvendes fra Pascal, og med TI-95 PROCALC.

Hovedprogrammet PCIF er en EXE fil, som er central for hele PC-Interface's funktion. Når man lader PC'en eksekvere PCIF, sættes den i et "slave-mode", så den reagerer på kommandoer fra TI-74. Al kommunikation over interfacen foregår nemlig på initiativ af TI-74. For at kommunikere anvendes ganske almindelige Basic kommandoer og statements som t.ex. OPEN, PRINT og SAVE. Som TI-74 operatør har du følgende muligheder:

Du kan LISTe og PRINTe til PC'ens parallelle printer.

Du kan LISTe og PRINTe til PC'ens monitor. Output til printeren PC-324

kan automatisk omdirigeres til monitoren.

Basic programmer kan LISTes til ASCII filer på PC'ens diske. Programmer kan også SAVEs i et binært format på PC'ens diske, hvorfra de senere kan OLDes eller RUNnes. Programmer kan oprette, anvende og slette datafiler på PC'ens diske.

TIC74.EXE hører til kategorien "utility programmer". Programmet konverterer en ASCII fil af Basic statements til en kørbare programfil. Med andre ord: Når du skriver Basic programmer til din TI-74, er du ikke tvunget til at anvende det minimale display og den indbyggede line-editor. Du udvikler dine 74'er programmer på PC'en med din favorit "full-screen editor". Siden lader du TIC74 konvertere til en programfil, som kan OLDes eller RUNnes over PC-Interface. Se et eksempel på proceduren nedenfor.

Et andet vigtigt utility program er LOAD.PGM. Programmet er i 74'er format, og RUNnes over PC-Interface. Det giver mulighed for at load'e nye subprogrammer til 74'ere's System RAM, altså nye CALLs. På min floppy findes følgende subprogrammer, som kan overføres til 74'eren een for een ved hjælp af LOAD.PGM:

CALL PEEK() Læser bytes fra memory.

CALL POKE() Skriver bytes til memory.

CALL EXEC() Eksekverer et assembler-program på en given adresse. Programmet må være POKet på plads i forvejen.

CALL GETMEM() Reserverer memory til assemblerprogrammer.

CALL RELMEM() Frigør reserveret memory.

CALL IOX() Udvidet CALL IO(). Tillader avanceret kommunikation med

ydre enheder.

CALL CHAR() Redefinerer CHR\$(0) -
CHR\$(6).

CALL INDIC() Styrer indikatorer i
display.

CALL CLEANUP Raderer uanvendte Basic
variabler.

Alle CALLs og deres anvendelse find-
es beskrevet på disken. For at kunne
bruge alle subprogrammerne, kræves
imidlertid information, som ikke kan
hittes på disken eller i 74'erenes
manualer. Det har ingen nytte at
kunne POKE, hvis man ikke ved HVOR
man skal POKE! En del relevant
information findes sandsynligvis i
TI-74 Technical Data Manual, som jeg
har i bestilling hos Texas.

Subprogrammer som er LOADet med PC-
Interface, SAVes ikke med Basic pro-
grammer som anvender dem. CALLene
optræder naturligvis i Basic rader-
ne, men eftersom System RAM ikke
følger med ved SAVE, får man "E13
Not found" hvis ikke subprogrammerne
er loadet i forvejen. CALL PUT()
lægger derimod al System RAM på et
8K RAM modul, inklusiv nye loadede
subprogrammer. Man kan altså lave
et slags Extended BASIC modul, som
indeholder et image af memory (CALL
PUT()) med de nye subprogrammer.

En ren Basicalc svarer "7710" på
CALL FRE(0). Om man loader alle sub-
programs, som er tilgængelige på PC-
Interface disken, får man svaret
"6588". De nye faciliteter forbruger
altså tilsammen 1122 bytes.

Nedenfor finder du et lille "anima-
tions program", som demonstrerer et
par af de nye subprogrammer. Des-
værre fungerer det ikke før du får
tag i PC-Interface, eller et memory
image med de nødvendige subprogram-
mer. Hvis du sender mig et 8K RAM
modul + 20 kr(kontant - ikke check),
skal jeg returnere det med alle sub-
programmer og printet dokumentation.

```
100 ! DEMO AF NYE SUBPROGS.. lha  
      8.90  
110 ! HOLD -> OG <- FOR AT AENDRE  
      FART, "BREAK" FOR STOP  
120 DLAY=20:INDIC=0:ON BREAK NEXT
```

```
130 CALL INDIC(11,0):CALL INDIC(3,0)  
140 BLK$(0)="1F1911111913111F":  
      BLK$(1)="1F1315111311131F"  
150 BLK$(2)="1F1115111119111F":  
      BLK$(3)="1F1511151111191F"  
160 CALL CHAR(0,"0000000018070000")  
170 CALL CHAR(1,"00000000031C0000")  
180 CALL CHAR(2,"0000071800000000")  
190 CALL CHAR(3,"00001C0300000000")  
200 WAVES(0)=CHR$(0)&CHR$(1)&  
      CHR$(2)&CHR$(3)  
210 WAVES(1)=CHR$(1)&CHR$(2)&  
      CHR$(3)&CHR$(0)  
220 WAVES(2)=CHR$(2)&CHR$(3)&  
      CHR$(0)&CHR$(1)  
230 WAVES(3)=CHR$(3)&CHR$(0)&  
      CHR$(1)&CHR$(2)  
240 DISPLAY AT(15),CHR$(4);"      "  
      ;CHR$(4);: 4 SPACES  
250 FOR LOOP=0 TO 3  
260 DISPLAY AT(16),WAVES(LOOP);  
270 CALL CHAR(4,BLK$(LOOP))  
280 CALL KEY(KEY,STAT):IF STAT=0  
      THEN 320  
290 IF KEY=252 AND DLAY<40 THEN  
      DLAY=DLAY+1  
300 IF KEY=254 AND DLAY>0 THEN  
      DLAY=DLAY-1  
310 IF KEY=231 THEN CALL  
      INDIC(10,0):CALL INDIC(3,1):END  
320 FOR DLOOP=0 TO DLAY:NEXT DLOOP  
330 NEXT LOOP  
340 CALL INDIC(10,INDIC):INDIC=NOT  
      INDIC  
350 GOTO 250
```

Da jeg skrev programmet, anvendte
jeg et par af faciliteterne i PC-
Interface : Programraderne er skrev-
et i Microsoft Word 4.0, inklusive
radnumre. Jeg sparede filen som
"text & LF only" på PC'ens harddisk
under navnet C:DEMO.B74 Herefter lod
jeg utility programmet TIC74 konver-
tere tekstfilen til 74'er program
format. TIC74 leverede filen
C:DEMO.PGM Når TI-74 og PC'en er
koblet sammen og PCIF kører, kan jeg
gøre OLD "100.C:DEMO.PGM" fra 74'er
Basic, programmet overføres og kan
RUNnes. ("100" i OLD kommandoen er
device nummeret som PC- Interface
tildeler PC'ens diskdrev.)

Med investeringen på 440 kr i en PC-
Interface, har jeg forøget anvende-
ligheden af min 74'er betydeligt.
Ikke alene har jeg fået harddisk og
floppydrev til min BASICALC (
praktisk set), jeg har også øget
mulighederne for at anvende 74'eren

sammen med de computere jeg anvender dagligt. Nu er mit ønske om at bruge TI-74 til dataopsamling "i farten" ikke urealistisk. Det er blevet mere attraktivt at skrive programmer til TI-74, nu hvor jeg ikke besværes af det minimale display. Med de nye subprogrammer og det øvrige utility i pakken, er assemblerprogrammering af maskinen blevet muligt. (Der er endnu meget at udforske, men vi har de basale værktøjer.)

Der findes mange argumenter for at købe et PC-Interface, men argumenter mod findes også, og de kan let være afgørende: PC-Interface kræver at du har adgang til en PC XT/AT eller en kompatibel maskine. Interfacet er parallelt og helt dedikeret. Det er hurtigt, men kan f.eks. ikke anvendes med TI-99/4A, Amiga eller Macintosh. Hvis man kan dække sine behov med en RS232 Seriel Interface, får man et standardværktøj, som er langt mere fleksibelt. Mere om RS232 til TI-74 i næste Programbiten. ■

FORNYELSE AF PROGRAMBITEN

av Lars Herold Andersen, Danmark

I PB 90-2 efterlyser Anders Persson forslag til fornyelse af foreningen. Her er et: Lad os beskæftige os mere med maskiner som er taget frem i de senere år. Det er sørgeligt at foreningen kun har 100 medlemmer, og skal den fornyes, må der nye medlemmer til. Selvom TI-99 har levet længe og lever endnu, er det kun lidt sandsynligt, at der fødes flere 99'er entusiaster. Jeg håber og tror at nye folk kan hverves på nye maskiner; lad tage TI-74 op for alvor!

Lad det være sagt; jeg (ungefær) elsker min TI-99 og kan aldrig skille mig af med den, og det er klart at Programbiten skal holde på med den maskine. Jeg synes også at PB skal koncentrere sig om Texas produkter. Ikke fordi der ikke findes andre maskiner der dur, for det gør der. Jeg anvender selv alt muligt andet end Texas i det daglige. Mere fordi det er en tradition

i foreningen - ingen tvivl om det.

Hvorfor nu TI-74? Først og fremmest fordi det er en Texas - eller fordi det er en god lille computer.. vælg selv. TI-74 har mange fine egenskaber, og en af de fineste er at den stadig støttes af Texas Instruments. Angiveligt udvikles der stadig grej-er til dyret. Jeg har netop købt en PC-Interface, som blandt andet giver mulighed for udvikling af assembler-programmer. RS232 interfacen HX-3000 til CC-40 er kompatibel med TI-74 og er stadig tilgængelig. Der går rygter om 32K RAM moduler.

Der er udsigt til en helt ny verden i denne maskine - lad os komme i gang med at udforske den. Derfor - har du en 74'er så skriv ind og beret. Hvad gør din 74'er når den ikke er slukket? Hvad ville du gerne at den skulle gøre? Ok, hvis vi stadig kun er tre i foreningen med TI-74 / CC-40, jævnfør matriklen 89, så kan korrespondencen nok snart tage slut, men der kan jo være kommet flere til siden. Og du som kender 74'er ejere; har du berettet om PB for dem? Ellers gør det! Jeg skal forsøge at skrive regelbundet om TI-74 og dens familie i de kommende numre af PB. Hvis jeg er den eneste som læser spalten er den unødigt og kun spild af godt papir, så derfor - sig frem - du som har spørgsmål eller artikler til 74'er spalten.

Venlig hilsen

Lars Herold Andersen
Anholtsgade 9 1.th
DK-8000 Århus C
DANMARK
tlf +45 86 19 33 41

I den seneste uge er jeg blevet ganske optaget af at studere LPC, speech modellen som speech synten forstår og decoder. Jeg fandt flere bøger på et teknisk bibliotek som beskriver transformationen af det analoge talesignal til LPC. Jeg forstår ikke alt endnu, men det er min vilde ambition at skrive programmer til 99'an som klarer konverteringen fra analog signal til LPC! I den forbindelse vil jeg spørge, om du har set noget materiale om emnet. ■