MPU-laren nr 9 · 1979

Tidskrift för Privatdataklubben PD68 · Årgång 2



Redaktionen

MPU-larenredaktionen består av följande medlemmar.

Alf Eriksson
Vallavägen 83
13641 Handen 08/7452667

Roger Holmstrand
Rapsodivägen 168
14200 Trångsund 08/7713852

Carl Axel Bruno
Timotejvägen 116
19177 Sollentuna 08/7543788

Lars Landqvist
Skagersvägen 20 1tr
12142 Johanneshov 08/912625

Bernt Svensson
Ploggatan 16 3tr
11634 Stockholm 08/403517

Annonser

ANNONSPRISER I MPU-laren.

Helsida	200:-
Halvsida	100:-
Kvartsida	50:-

Priserna gäller tryckfärdigt material.Radannonser för medlemmar är gratis om det inte gäller firma.Tidningens upplaga är för närvarande 225 ex.
För material till tidningen tala gärna med någon i redaktionen ovan eller sänd detta till,PD68 Box 98,12221 Enskede 1.

Styrelsen

PD68 nuvarande styrelse.

Nils Ohlsson Ordförande
Sandfjärdsgatan 68 12tr
12169 Johanneshov

Tommy Bladh V.Ordförande
Båstadsvägen 7
12151 Johanneshov

Åke Holm Kassör
Långsjövägen 15B
13554 Tyresö

Lars Olof Fagerström Sekreterare
Gärdebyplan 6 4tr
16374 Spånga

Bo Ljungblad V.Sekreterare
Brännkyrkagatan 117
11728 Stockholm

Gunnar Lilliesköld Suppleant
Mossvägen 1C
18499 Åkersberga

Anders Thunell Suppleant
co.Fittja R&TV Forvägen 15
14551 Norsborg

Gunnar Lovius Suppleant
Ragnebergsvägen 21
13667 Handen

MEDLEMSINTRÄDE

Vill DU bli medlem i PD68 ?
Sätt in 75:- på postgiro 960468-7
samt namn adress tel nr så kommer medlemskap på posten.

LEDAREN

Debatten om mikrodatorer fortsätter i dagspressen, och om man följt denna undgår man inte att märka att man svängt från en klart negativ hållning till en att beskriva som närmast frågande ställning. Nåväl, om detta har jag redan filosoferat i nr 3, men det har ytterliggare accentuerats genom olika artiklar. En av de intressantare var nyligen införd i Expressen (15 okt sid 20). Här hade man lösningen på hur man skall avskaffa behovet av reda pengar. Nåväl det är väl inget nytt utbrister man genast. Kontokort har funnits en längre tid, så det är väl inget speciellt. Det är i och för sig riktigt men man har här gått ett steg längre. Genom on-line förbindelser med en datacentral kan man omedelbart verifiera en presumtiv kunds ekonomiska status. Det innebär också att man kan kontrollera att kortet inte är stulet (omedelbar spärrning vid stöld), att man dessutom själv kan bli informerad om hur ens aktuella kapital ser ut utan att öppna plånboken kan ju vara intressant. Skämt åsido, utvecklingen går fort och i TV släppte man nyligen informationen om att ett företag lyckats intergrera en processor med själva kontokortet. Detta var ett stort steg mot decentraliserad datakraft. Man uppnår här ett avsevärt bättre skydd för den enskilde personens ekonomiska integritet Dina affärer blir åter en "affär" mellan dig och din plånbok. Givetvis kommer det att lysa rött vid en eventuell övertrassering, men konsekvenserna behöver inte bli så stora. Någon notering i ett centralt databas-system finns inte och detta är ett steg i rätt riktning. Nåväl, klubbens målsättning är inte att vara politisk eller ekonomisk-politisk. Vi vill bevaka utvecklingen och med våra begränsade resurser föra ut lite know-how till en bredare allmänhet.En viktig länk är Du då som medlem. Vi hoppas att Du vill ställa upp och berätta, inte bara om den skojiga blinkande cursorn, utan även om att datorn är på gott och ont, och att den blir vad alla påverkar den till. Arbeta aktivt för ett bättre datasamhälle.

N. Ol

REFERENSBIBLIOTEK OCH NYHETER

Klubben PD68 har inget organiserat utbyte av tips, idéér, hjälp med hårdvaror eller mjukvaror. En del tips har förmedlats av MPUlaren. Övrig hjälp och frågor har medlemmarna kunnat fått genom personliga kontakter. Denna kontakt tycker jag är bra och bör premieras. Det berikar såväl tidningen som individen mänskligt, mer levande och med ökad tekniskt kunnande.

All information kan nu inte förmedlas på detta sätt. En mer generell information är då bättre att förmedla centralt t.ex. genom en tid - ning eller att någon har hand om en speciell sak. Denna sida av en klubbs verksamhet är helt beroende av medlemmarnas aktivitet, behov och intresse. Så vi har den verksamhet och tidning som vi gör den till. Nu till saken.

Interelko har, genom att ge oss kostnadsfritt tre uppsättningar böcker, tagit initiativet till ett referensbibliotek. Det kommer att bestå av data-, system-, komponent- och manualböcker och nyheter som kommer att sändas till klubben kontunuerligt.

Tre personer inom klubben ska sköta denna verksamhet. Dessa kommer att ha likadana upp sättningar hos sig. Meningen är att medlemmarna ska kontakta dem för att kunna låna hem referensex och datablad..

Adresserna till oss är;

Sven Erik Ringström
Högbyvägen 18
175 46 JÄRFÄLLA

Bernt Svensson
Ploggatan 16 3 tr
116 34 STOCKHOLM

Jan Söderberg
Skjutbanevägen 2C nb.
723 39 VÄSTERÅS

Vi har följande för utlåning:

Optoelectronics Designers
Catalog 1979
Hewlett packard

Low power schottky data bok/M/

The european cmos selection/M/

The Switchmode Series /M/

Mcroprocessor cours /M/

M6800 Resident cobol;
Language reference manual /M/
Operations ——"—— /M/

Macro assemblers reference manual 6800,01,05,09 /M/

EXORterm 100, EXORciser Display Console Users Guide /M/

MPU Vocabulary /M/

Application note, engineering bullentin...Catalog /M/

Microcomputer components data bok 1
Memory products data bok 2
Microcomputer development systems and subsystems data bok 3
Motorola= /M/

Linear integrated circuits volym 6 /M/

EXORdisk 11/111 operating system Users Guide M68MDOS3 /M/

Co-resident /M6800/
Assembler reference manual /M/
Editor ——"—— /M/

Linking loader reference manual /M6800/ /M/

Resident MPL Language Reference Manual /M6800/ /M/

Handledning till M6800 systemet
Distributören = /D/

Även datablad och fler böcker.

BS

Programmeringskurs i BASIC.

Upprinnelsen till denna kurs var att ge tidningen en levande funktion. T.ex. att öka kommunikationen mellan medlemmarna i klubben och att vara ett medium för utbyte av enkla tips, idéer och erfarenheter i programmering.
Kursen kommer därför inte vara traditionellt upplagd från början till slut.
Jag har tänkt mig börja med förklaringar av olika satser (eng. statements). Därefter en beskrivning av hur de används och olika exempel. Detta är nu traditionellt. Om det inte ska förbli så beror på dig och alla andra. Genom att skicka enkla och mer komplexa program och idéer till tidningen kan vi få kursen och tidningen mer levande.

INTRODUKTION

Det ökande antalet datorer i samhället gör att allt fler människor kommer i beröring med datorer och dess verkningar och användning. Av detta har skapats ett behov av utbildning av många människor för programmering. Utifrån detta behov skapades BASIC (Begginers All-purpose Symbolic Instruktion).

En av orsakerna till att den rikhaltiga floran av programmerings-språk utökades med BASIC var att man ville skapa ett språk som var speciellt lämpligt för undervisning i programmering.
BASIC är enkelt och logiskt uppbyggt och både lätt att lära och använda. BASIC är speciellt lämpligt vid körning av program via terminal p.g.a. möjlighet till dialogprogrammering.

Sedan BASIC först presenterades har språket utvecklats till ett effektivare och mångsidigare programmeringsspråk.
Den enkla grundstommen har dock behållits och nybörjare kan använda sig av den utan kunskaper om de tillägg, som gjorts.

PROGRAM

Vad är ett BASIC-program?
Enkelt uttryckt är det en serie instruktioner som talar om för datorn hur något ska göras.
Dessa exakta satser eller instruktioner i BASIC-programmet ska utföras i ordningsföljd av datorn. Varje sats förses med ett radnummer. Man kan hänvisa till en viss sats genom att ange dess radnummer.
Detta om radnummer kommer att bli klarare längre fram. I övrigt hänvisar jag till MPU-laren 3 / 79 sid. 14,15 där Lars-Olof Fagerström i en del skrivit om programmering mer generellt.

Ni kan även ta fram MPU-laren 2/ 79 sid. 21. Där finns en lista över BASICsatser och lite exempel om dem.

DEL I

Basicsatsernas andvändning

Här är en lista på de vanligaste satserna. De är ordnade efter vad de gör, alltså vilken typ av funktion de har. Funktionerna motsvarar det som datorn kan göra: den kan läsa-, beräkna-, lagra-, skriva-, jämföra data och hoppa i programmets instruktionsföljd.

Forts. nästa sida

LET	-satsen används för att ge endast en variabel ett värde.
READ	-satserna används för att ge flera variabler ett värde.
DATA INPUT PRINT RESTORE	är kommandon för överföring av data; till och från datorn, till skrivaren, osv.
GOTO	Ovillkorligt hopp
IF- THEN	villkorligt hopp
ON	-satser; villkorligt hopp
FOR.... NEXT	Program-loopar (är ett programavsnitt, som genomlöps ett visst antal gånger.
GOSUB RETURN	Satser för hopp till och från subrutiner.
PEEK POOK	BASIC-programet kan kombineras binärt med dessa satser
STOP	Används för att avsluta ett program.

/ som sagts tidigare så är dessa satser vad datorn i huvudsak gör. Det är inte alla men de flesta och vanligaste. /

Nedan kommer nu några satser beskrivna en och en. Därefter några programexempel.

Först något om de data som datorn använder i sitt arbete. Data uppträder i ett programeringsspråk som variabler eller konstanter. Om det är numerisk databehandling så uppträder data som tal, och behandlas i huvudsak med de aritmetiska räknelagarna. Vid textbehandling används särskilda regler för bearbetning av bokstäver och textsträngar.

Datorn arbetar med binärt talsystem, vilket vi inte behöver bry oss om här. I basic anväder vi det decimala talsystemet, de vanliga talen 1,2,3...osv.

I numerisk behandling av data är en konstant talet självt. Vid programering blir programet kortare och överskådligare om man inte behöver skriva ett specifikt tal varje gång. Istället kan man oftast använda variabler.

En variabel i basic har samma funktion som i algebran. En variabel är alltså en symbol för ett tal, vilket som helst. Den representerars i basic av ett variabelnamn som vi själva väljer. Dessa namn är begränsade i basic till en bokstav (A-Z),ibland även å ä ö , till vilken en siffra (0-9) kan fogas.

Antag att vi vill summera två godtykliga tal, a och b, inget över 100 och inget som är negativt. Första satsen blir då en sats som läser in data(tal) till variablerna a och b:

```
1  READ  A;B
```

För att summera a och b använder vi det matemtiska formelspråket och skriver då A + B. Detta är ingen fullständig basic sats utan summan måste tilldelas en variabel t.ex. C. Genom basic verbet LET tilldelas summan den nya variabeln C.

```
2  LET   C = A+B
```

För att inte något av de lästa talparen (hur READsatsen läses av datorn beskrivs senare) är negativa eller över 100 tillkommer fyra jämförelsesatser. För detta användes de vanliga jämförelsetecknen:

>(större än)
<(mindre än)
=(lika med)

Satserna blir följande:

```
3 IF A>100 THEN 1    undersöker om A är större än 100 och
                     hoppar tillbaks till radnummer 1
4 IF B>100 THEN 1    som ovan fast variabeln B undersöks

5 IF A< 0 THEN 1     om variabel A är negativt, hopp till 1

6 IF B< 0 THEN 1     som ovan fast variabeln B undersöks.
```

Nu är summan beräknad och finns i C. Vi vill ha svaret tillgängligt och veta vilken operation vi gjort. Det kan vi göra med basicverbet PRINT.

```
7 PRINT "SUMMAN AV",A,"OCH",B,"=",C
```

Meddenna sats skriver datorn texten inom citationstecken samt de tal som variablerna A,B,och C representerar.

De ovanstående hoppsatserna (IF-satser) är villkorliga satser. Genom ovillkorlig sats GO TO radnummer n, i det här fallet 1, får vi programet att börjaom på nytt.

```
8 GO TO  1           hopp till början
```

När programet körs läses data in. Som programet nu är fattas data. Vilket tal som ska läsas in meddelar vi datorn iprogrammet med basic-ordet DATA, varefter vi räknar upp talen med kommatecken emellan.

```
9 DATA 1,-2,13,120,-15,3,17,5
```

Till sist ska programmet avslutas med satsen END. Det meddelar datorn att uppdraget och instruktionerna är slut.

```
10 END
```

Vad gör detta program för något? Trots den digra data listan så summerar det endast två tal och skriver ut PRINT-satsen .

```
SUMMAN AV 17 OCH 5 = 22
SUMMAN AV 17 OCH 5 = 22
SUMMAN AV 17 OCH 5 = 22
SUMMAN AV 17 OC.....
```

osv tills vi själva bryter programmet.

Programmet som **helhet** blir:

```
1 READ  A,B
2 LET   C = A+B
3 IF    A>100 THEN 1
4 IF    B>100 THEN 1
5 IF    A< 0 THEN 1
6 IF    B< 0 THEN 1
7 PRINT "SUMMAN AV",A,"OCH",B,"=",C
8 GO TO 1
9 DATA 1,-2,13,120,-15,3,17,5
10 END
```

Kommentarer till programmet

DATA-satsen innebär att datorn skapar en intern datafil av data, som utifrån datasatsernas radnummerföljd,läggs i ordning. Det innebär(data satsen läses alltid en variabel i taget från vänster till höger) att först 1till A och -2 till B läses in. Ett tydligare sätt att skriva detta på är: A=1 och B=-2.

Forts. nästa sida

I LET-satsen tilldelas variabeln C summan av talen 1 och -2 : C=-1. Detta skrivs inte ut för när datorn kommer till rad 6 hoppar den tillbaks till rad 1. Villkoret för satsen har uppfylts, B är negativt.

Ett nytt talpar inläses: A=13 och B=120. Även detta talpar är ogiltigt enligt rad 4. Märk väl att C har tilldelats den nya summan 133, vilket inte heller skrivs ut. Efter återhopp till rad1 läses nästa par. Samma förlopp som förut händer A=-15.

Sista talparet A=17 ochB=5 får programmet att utföras till PRINT-satsen så attresultatet skrivs ut. GO TO-satsen irad 8 gör att hela förloppet görs om ideligen utan stop. Hur man programmerar ett stopp kommer jag till senare.

Presstopp för MPUlaren har varit länge redan, så jag avslutar del 1 här. Om det finns oklarheter eller om ni har egna idéér så hör av er eller skriv till mig.

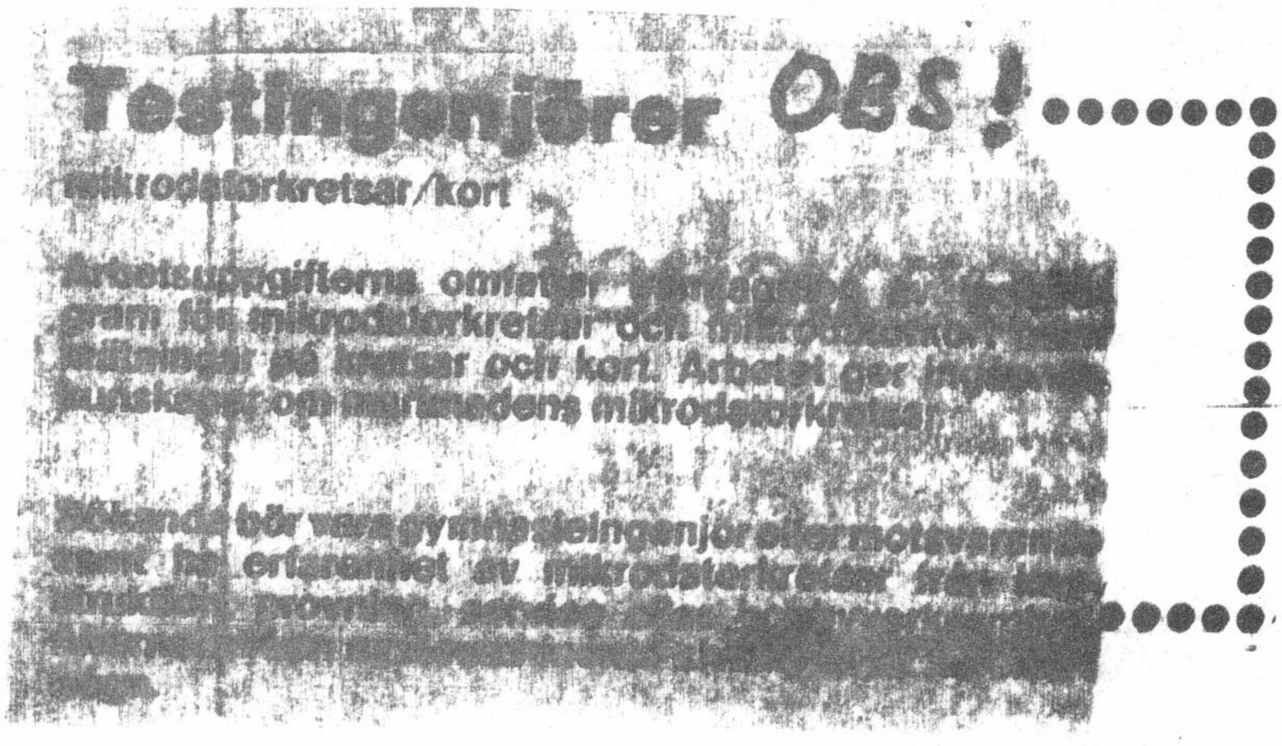
Bernt Svensson Ploggatan 16
tel; 403517 116 34 STOCKHOLM

Pröva om de här två programmen fungerar och hurﬁl

```
 10 READ T,F1,F9
 20 IF T=0 THEN 100
 30 PRINT
 40 PRINT
 50 PRINT "TABELL",T
 60 PRINT "FAHRENHEIT      CELSIUS"
 70 LET F=F1
 80 LET C=(5/9)*(F-32)
 90 PRINT F,C
100 LET F=F+1
110 IF F>F9 THEN 10
120 GO TO 80
130 DATA 1,30,36
140 DATA 2,200,212
150 DATA 0,0,0
160 END
```

```
 10 PRINT "SKRIV 2 TAL"
 20 INPUT T1,T2
 30 IF T1=T2 THEN 60
 40 IF T1 T2 THEN 80
 50 IF T1 T2 THEN 100
 60 PRINT "TALEN ÄR LIKA"
 70 GO TO 110
 80 PRINT "T1<T2"
 90 GO TO 110
100 PRINT "T1>T2"
110 END
```

INPUT är en ny sats här. Den hämtar talen T1 ochT2 från terminalen.



19:33:56 FÖR D2

```
00001                        NAM   KTIME
00002                    * KTIME IS AN EXAMPLE PROGRAM THAT USES THE
00003                    * 7-SEGMENT DISPLAYS TO EXHIBIT AN ELAPSED
00004                    * TIME, OR IF PRESET, THE TIME OF DAY.  THE
00005                    * PROGRAM USES SEVERAL JBUG ROUTINES AND THE
00006                    * DISPLAY ROUTINE (DSPLY) CLOSELY RESEMBLES
00007                    * THE SEQUENCE IN JBUG EXCEPT THAT IT WAS
00008                    * MODIFIED TO TAKE A CONSTANT AMOUNT OF TIME
00009                    * REGARDLESS OF THE NUMBERS BEING DISPLAYED.
00010                    *
00011                    * TO OPERATE IN ELAPSED TIME MODE, TYPE
00012                    *   0010G ON THE D2 KIT KEYBOARD.
00013                    * TO USE AS A CLOCK, INSERT THE HOUR (0-23)
00014                    *   IN LOCATION 0000, THE MINUTES IN 0001,
00015                    *   AND THE SECONDS IN 0002.  TYPE 0019G TO
00016                    *   START THE CLOCK.

00018                    * EQUATES TO JBUG
00019  E29A      MDIS2  EQU   $E29A
00020  E0E0      DLY1   EQU   $E0E0
00021  E3CA      DIGTBL EQU   $E3CA
00022  8020      DISREG EQU   $8020
00023  8022      SCNREG EQU   $8022

00025 0000               ORG   $0000
00026 0000 0001  HOUR    RMB   1          HOURS
00027 0001 0001  MIN     RMB   1          MINUTES
00028 0002 0001  SEC     RMB   1          SECONDS
00029 0003 0006  BUFF    RMB   6          DISPLAY BUFFER
00030 0009 0001  CTR     RMB   1          LOOP COUNTER
00031 000A 0002  XTEMP   RMB   2          TEMP
00032 000C 0001  SCNCNT  RMB   1          DIGIT SELECT

00034 0010               ORG   $0010      START OF PROGRAM
00035 0010 7F 0000 ENTRY1 CLR    HOUR      INIT COUNTERS
00036 0013 7F 0001        CLR    MIN
00037 0016 7F 0002        CLR    SEC
00038 0019 86 20   ENTRY2 LDA A  #$20      DISPLAY COUNTER
00039 001B 97 0C          STA A  SCNCNT
00040 001D 8D 1E   LOOPA  BSR    SHOW      SHOW THE CURRENT TIME
00041 001F CE 0002        LDX    #SEC      UPDATE CURRENT TIME
00042 0022 A6 00   TIMA   LDA A  0,X
00043 0024 8B 01          ADD A  #1        ADD 1 SECOND
00044 0026 19             DAA
00045 0027 A7 00          STA A  0,X
00046 0029 8C 0000        CPX    #HOUR
00047 002C 27 09          BEQ    TIMB
00048 002E 81 60          CMP A  #$60
00049 0030 26 EB          BNE    LOOPA
00050 0032 6F 00          CLR    0,X
00051 0034 09             DEX
00052 0035 20 EB          BRA    TIMA
```

Forts. nästa sida

```
00054 0037 81 24    TIMB    CMP A   #$24        RESET TO 0 AT MIDNITE
00055 0039 27 D5            BEQ     ENTRY1
00056 003B 20 E0            BRA     LOOPA
00058
00059                   * SUBR TO SHOW DATA FOR 1 SECOND
00059 003D 96 02    SHOW    LDA A   SEC         GET SECONDS
00060 003F BD E29A          JSR     MDIS2
00061 0042 D7 08            STA B   BUFF+5      STORE INTO BUFFER
00062 0044 97 07            STA A   BUFF+4
00063 0046 96 01            LDA A   MIN         DO MINUTES
00064 0048 BD E29A          JSR     MDIS2
00065 004B D7 06            STA B   BUFF+3
00066 004D 97 05            STA A   BUFF+2
00067 004F 96 00            LDA A   HOUR        DO HOURS
00068 0051 BD E29A          JSR     MDIS2
00069 0054 D7 04            STA B   BUFF+1
00070 0056 97 03            STA A   BUFF
00071 0058 86 72            LDA A   #114        DELAYS PER SECOND
00072 005A 97 09            STA A   CTR
00073 005C 8D 06    LOOPB   BSR     DSPLY
00074 005E 7A 0009          DEC     CTR         DONE?
00075 0061 26 F9            BNE     LOOPB
00076 0063 39               RTS

00078                   * SUBROUTINE TO DISPLAY THE BUFFER
00079 0064 CE 0003  DSPLY   LDX     #BUFF       GET POINTER TO BUFFER
00080 0067 A6 00    LOOPC   LDA A   0,X         GET DATA BYTE
00081 0069 4C               INC A
00082 006A 08               INX                 MAKE PTR TO NEXT CHAR
00083 006B DF 0A            STX     XTEMP       SAVE IT
00084                   * CONVERT BCD TO 7 SEGMENT
00085 006D CE E3C9          LDX     #DIGTBL-1   START OF CONV TABLE
00086 0070 08       LOOPD   INX
00087 0071 4A               DEC A
00088 0072 26 02            BNE     LOOPF
00089 0074 E6 00            LDA B   0,X
00090 0076 8C E3D4  LOOPF   CPX     #DIGTBL+10
00091 0079 26 F5            BNE     LOOPD
00092 007B 7F 8022          CLR     SCNREG      BLANK DISPLAYS-REDUCE FLICKER
00093 007E F7 8020          STA B   DISREG      OUTPUT CHAR
00094 0081 96 0C            LDA A   SCNCNT
00095 0083 B7 8022          STA A   SCNREG
00096 0086 CE 004D          LDX     #$4D        DELAY 1 MS.
00097 0089 BD E0E0          JSR     DLY1
00098 008C DE 0A            LDX     XTEMP       RESTORE BUFF PTR
00099 008E 8C 0009          CPX     #BUFF+6     DONE?
00100 0091 27 05            BEQ     OUT         YES
00101 0093 74 000C          LSR     SCNCNT      MOVE TO NEXT CHAR
00102 0096 20 CF            BRA     LOOPC
```

Efter vår Extra MPU-laren har Interelko skaffat hem

MEK6800AB

MEK68R2

Forts. nästa sida

```
00104 0098 86 20    OUT     LDA A  #$20
00105 009A 97 0C            STA A  SCNCNT
00106                * THIS LOOP CAN BE USED TO TWEEK THE
00107                * DELAY FOR A PARTICULAR CRYSTAL.
00108 009C 86 0A            LDA A  #10       10 USEC/LOOP
00109 009E 4A       LOOPE   DEC A
00110 009F 26 FD            BNE    LOOPE
00111 00A1 39               RTS
00112       0000            END
TOTAL ERRORS 00000
```

```
PAGE 001  UG1175  .LX:1

S00B00004B5449D45202020 1A
S11E00107F00007F00017F00029620970C8D1ECE0002A6008B0119A7008C000F
S11E002B0027098160266B6F00920EB81242705520E09602BDE29AD7089707 22
S11E00469601BDE29AD70697059600BDE29AD70497038672970980D067A000960
S11E00626F939CE0003A6004C08DF0ACEE3C9084A2602E6008CE3D426F57FBD
S11E007C8022F7802096 0CB7802 2CE004DBDE0E0DE0A8C0009270574000C2050
S10E0097CF862097 0C860A4A25FD390C
S9030000FC
```

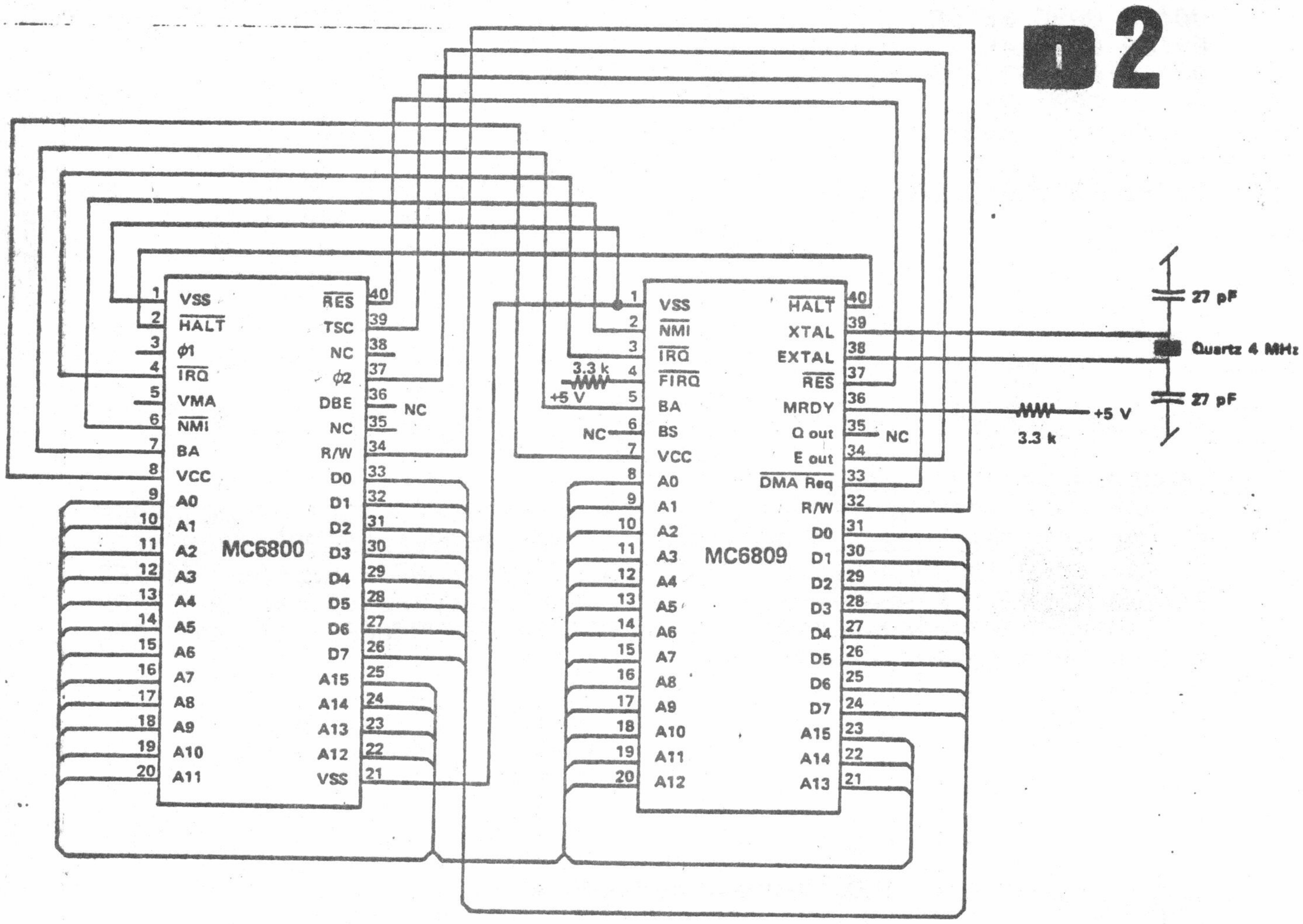
HEXADECIMAL ARITHMETIC

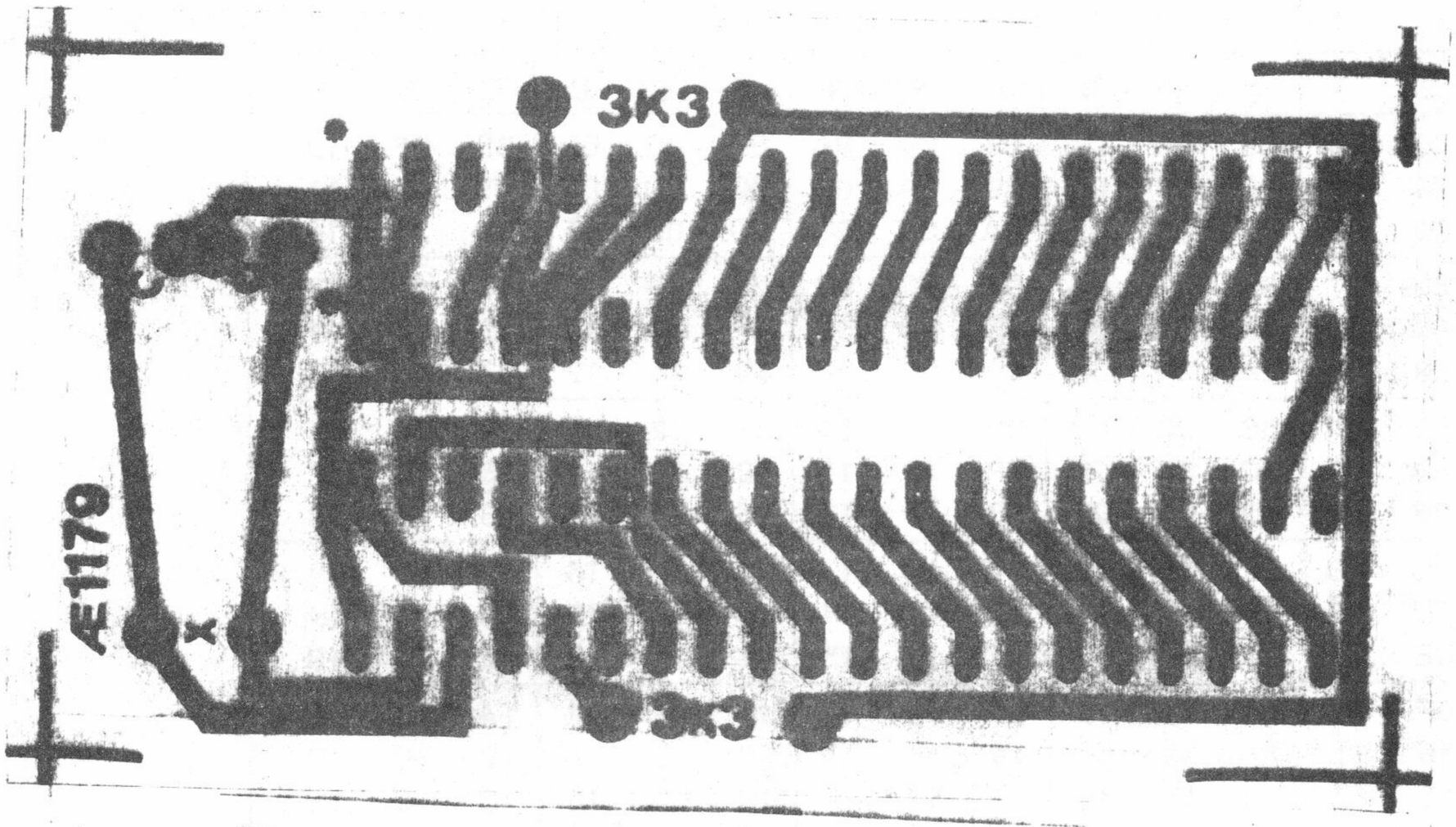
ADDITION TABLE															
0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
1	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10
2	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11
3	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12
4	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13
5	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14
6	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15
7	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16
8	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17
9	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18
A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19
B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A
C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B
D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C
E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D
F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E

6800 till 6809

INTERCEPT SOCKET FOR MC6809

D 2





Kretskortslayout i skala 2:1

Forts. nästa sida

GENERAL DESCRIPTION AND CAPABILITY.

When the following modifications are carried out to a standard D2 Kit Microcomputer Module Board (MMB.), the system will be able to run on an MC6809 M.P.U. It is intended that in conjunction with the Motorola EK9BUG4 Monitor Program, the modified MMB via its ACIA can be used with an RS-232 terminal. (The keyboard/display module is not required.)

Separate clocks are used, the on-chip oscillator and one for the generation of Transmit and Receive clocks for the ACIA. This gives flexibility in the choice of MPU clock frequency while utilising the original D2 Kit MC6871B - specially chosen for baud rate generation. The first choice baud rate is 300, but others may be generated (see "ACIA Clock Generation", below).

The greater number of registers on the 6809 compared with the 6800 demands changes to NMI timing, to allow stacking of all register information. The MC8316 binary counter setting is altered from 10 to 15 clock cycles.

MODIFICATIONS:

1. From the MEK6800 D2 Kit Microcomputer Module Board remove the following devices:-

MC6800	MPU
MCM6830	ROM with JBUG Monitor (SCM44520P)
MC6871B	Clock Generator

2. Insert the following DIP sockets in the user wire-wrap space on the board:-

   1 off 40-pin socket for MC6809 MPU

   1 off 24-pin socket for MC6871B clock generator (optional see "ACIA Clock Generation" below).

   2 off 14-pin socket for MC1488, MC1489 RS-232 type Driver and Receiver.

   Also; 1 off plug in crystal socket - to be mounted close to pins 38, 39 of MPU socket.

3. Carry out the wire-wrapping, soldering and track cutting as per Tables 1, 2, 4.

4. Either (i) carry out the strapping to either of the ROM sockets U10, U12 for the TMS2716 EPROM, as in Table 5.

   Or (ii) strap sockets U10, U12 to support any 2048 x 8 EPROM or 2 off 1024 x 8 EPROMS (with EK9BUG3 split between them).

5. If it is intended to have the MC6871B on a socket in the user wire-wrap area, carry out connections as per Table 3, or see "ACIA Clock Generation" below.

6. Insert devices as per 2. above, plus desired EPROM(S).

Forts. nästa sida

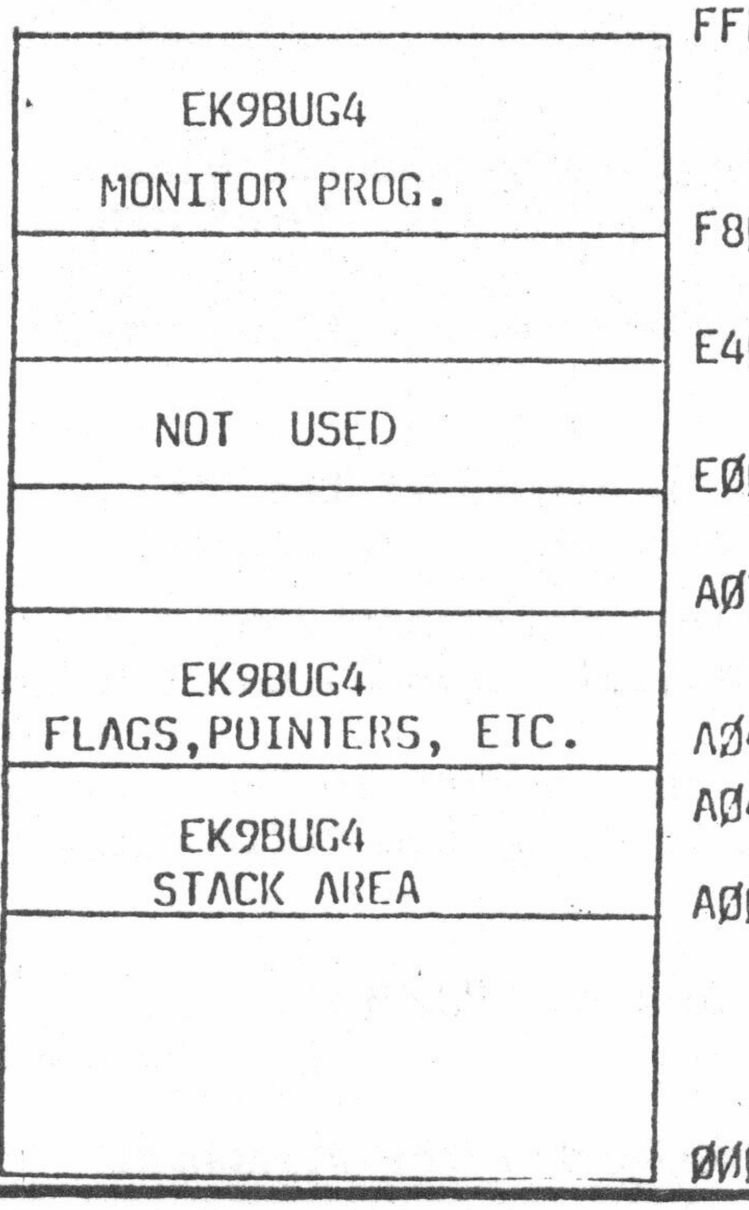
Also require an RS-232 25-Way D-Type connector for 3. above.

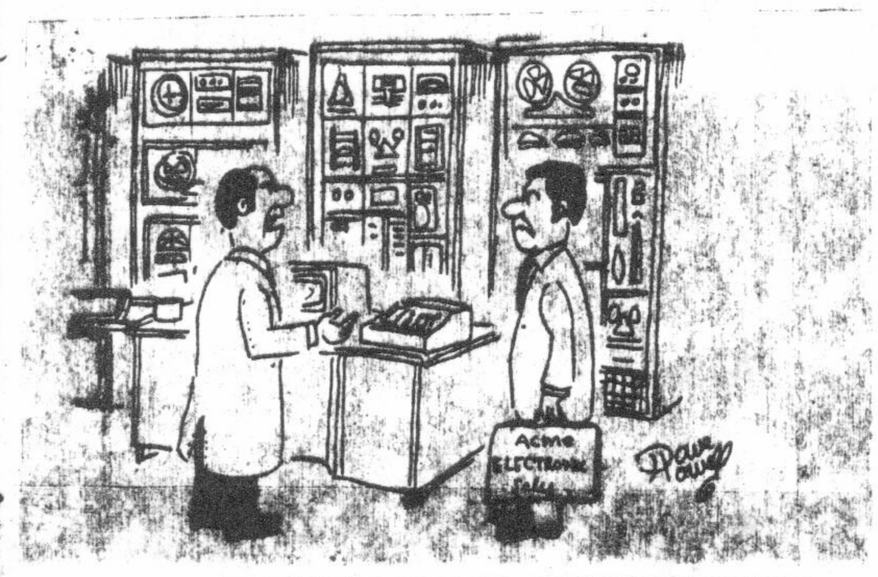
MEMORY ORGANISATION.

The present version of EK9BUG4, being developed from the MC6801 Lilbug monitor, is not position-independent.

The monitor program resides at $F800 - $FFE4
The interrupt vectors are at $FFF0 - $FFFF.
Locations $A000 - $A049 are used as stacking area.
Locations $A04A - $A07D are used for pointers, flags etc.

On a 6809-modified D2 Kit, with the JBUG ROM removed, the $\overline{ROM}$ signal is used for chip select ($\overline{S}$) on the TMS2716. Address line A12 is still a "don't care" for $\overline{ROM}$.



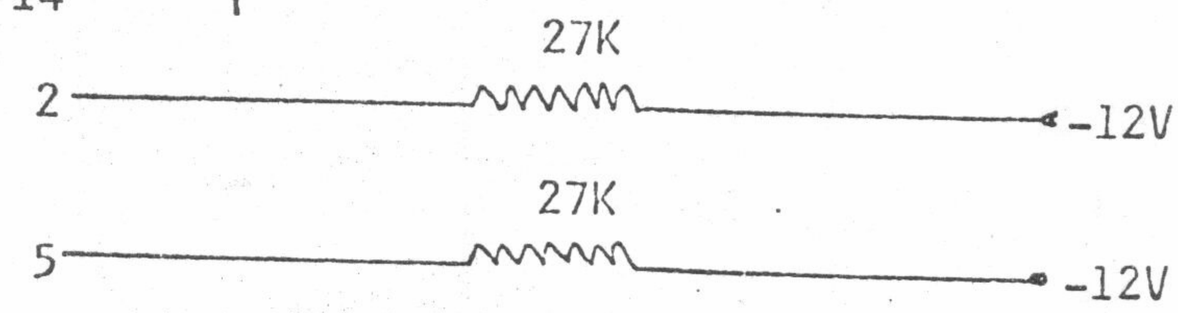


Du sa att DATØRN skulle betala sig själv – när händer det !

TABLE 2

ACIA AND RS - 232 CONNECTIONS.

FUNCTION	ACIA (U23) PIN	MC1488 PIN	MC1489 PIN	RS - 232 CONN. PIN
TXD	6	2		
RXD	2		6	
TX CLK	4 ← LINK			
RX CLK	3 ← LINK			
TX DATA		3		3
RX DATA			4	2
TA TERMINAL READY			1	20
G. OF ABOVE		4 5	3	
TA SET READY		6		6
EAR TO SEND		6		5
RRIER DETECT		6		8
OTECTIVE GROUND				1
GNAL GROUND		7	7	7
+12V		14		
-12V		1		
+ 5V				
SPONSE A ITROL				
SPONSE B ITROL				



Forts. nästa sida

CONNECTIONS TO MC6809 40 - PIN DIP SOCKET.

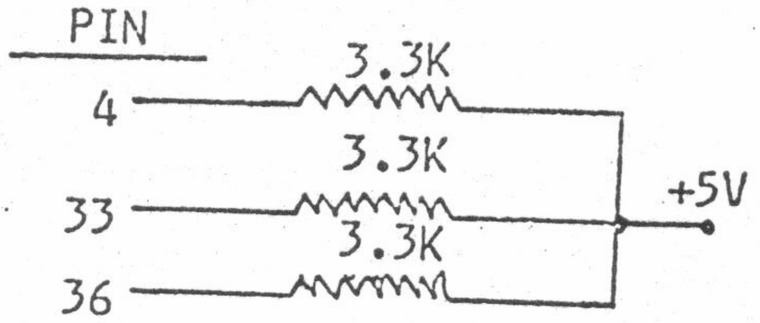
FUNCTION	FROM U6 PIN	TO 6809 PIN
ADDRESS BUS		
AØ	9	8
A1	10	9
A2	11	10
A3	12	11
A4	13	12
A5	14	13
A6	15	14
A7	16	15
A8	17	16
A9	18	17
A10	19	18
A11	20	19
A12	22	20
A13	23	21
A14	24	22
A15	25	23
DATA BUS		
DØ	33	31
D1	32	30
D2	31	29
D3	30	28
D4	29	27
D5	28	26
D6	27	25
D7	26	24

FUNCTION	FROM U6 PIN	TO 6809 PIN
$\overline{HALT}$	2	40
$\overline{IRQ}$	4	3
$\overline{NMI}$	6	2
BA	7	6
R/$\overline{W}$	34	32
E(Ø2)	37	34
$\overline{RESET}$	40	37
V_{DD}	1	1
V_{SS}	8	7
VMA	5	3

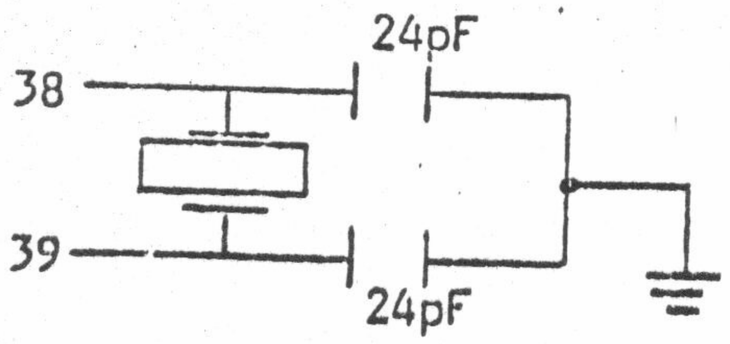


ALSO:

FUNCTION	TO 6809 PIN
FIRQ	4
DMA/$\overline{BREQ}$	33
MRDY	36



EXTAL	38
XTAL	39



ACIA CLOCK GENERATION.

The above modifications allow the MPU to have a 4MHZ oscillator and the ACIA 4800HZ TX,RX Clocks. The following are some alternative ways of generating TX,RX clocks.

1. Use the MC6809(E) signal and have no MC6871B module. Choose a 4.9152MHZ crystal for the MPU and feed E(1.2288MHZ) into the MC14040 (U17 pin 10). This will give a 300 baud data rate. This runs the MC6809 slightly beyond spec., and it might be neccessary to use a 2.4576MHZ crystal. In this instance remove connector U23 pin 4 to U17 pin 3 (÷ 256) and connect U23 pin 4 to U17 pin 4 (÷ 128).

2. Use the MC6871B module in socket U15. Cut tracks to U15 pin 12 (Ø2) and ensure that the new E(formerly Ø2) line runs from MC6809 pin 34 to U9 pin 12.

3. Baud rates other than 300 can be generated by taking the appropriate signal from the MC14040 12-bit binary counter (U17).

   e.g. 4800 baud for use with CRT-based PDS system:-

   cut track from U17 pin 13 to U23 pin 4

   connect U17 pin 5 to U23 pin 4

   The result is that with 1.2288MHZ into U17 pin 10, this will give 76.8KHZ into the ACIA.

Forts. nästa sida

På grund av att Electrolux dataanläggning havererat kom dagens kylskåpsleveranser att hamna på annan adress.

Ny leverans kommer att ske tisdagen den 25 september 1979.

Beträffande tillträde till lägenheten med huvudnyckel, gäller vad som tidigare meddelats.

Vi ber om överseende för det inträffade.

Vänligen

AB FAMILJEBOSTÄDER
Distrikt II

TABLE 3. MC6871B CLOCK GENERATOR SOCKET CONNECTIONS.

FUNCTION.	MC6871B SOCKET PIN.	TO.
+5V	7	-
GND	1,18	-
2 x FC	24	U17 PIN 10

TABLE 4. MC8316 BINARY COUNTER MODIFICATIONS.

	FUNCTION.	U25 PIN.
REMOVE (CUT TRACK)	+5V	3,(5)
CONNECT	GND	3,(5)

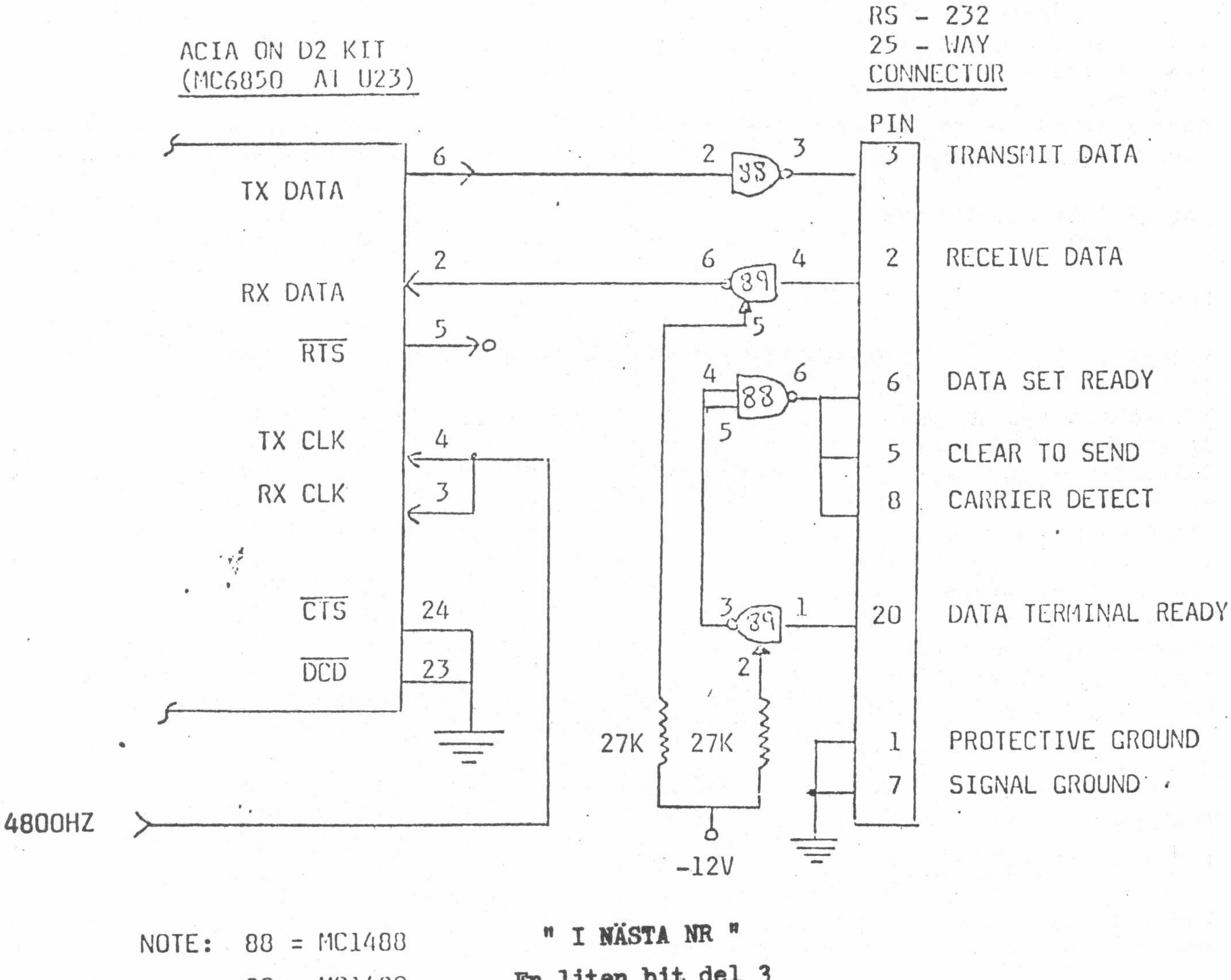
TABLE 5. STRAPPING OF U10 OR U12 FOR TMS2716 (2048 x 8 EPROM).

MC2716 FUNCTION.	U10, U12 PIN.	SIGNAL SOURCE.
$\overline{S}$	18 (E 5)	U8 PIN 11 ($\overline{ROM}$)
V_{DD},+12V	19 (E 4)	-
* A10	20	U10 OR U12 (EØ) CONN.
V_{BB},-5V	21 (E 3)	-
A9	22 (E 2)	U10 OR U12 (E1) CONN.

REMOVE THE FOLLOWING CONNECTION(S) (CUT TRACK(S)):

$\overline{PROM\ Ø}$	TO	U10 PIN 20
	AND/OR	
$\overline{PROM\ 1}$	TO	U12 PIN 20

ACIA AND RS - 232 CONNECTIONS.



NOTE: 88 = MC1488
89 = MC1489

" I NÄSTA NR "

En liten bit del 3

Nybörjar Basik del 2

Spelhålan

TSC 6800 / 6809 Basic

TSC Sort / Merge

TSC Textprocessor

Microware A / Basic kompilator

Space --Komprimerande FLEX - utility

m.m

TILL SALU

MIKRODATOR i SCHROFF låda.
16 K RAM (2114 low power)
16 K ROM (Basic/ed-ass).
Videokort C-Å elektronik.
KCS interface, RS-232
5" TV monitor, kylfläkt
Nätagg 5V 10A,+-12V 3A.
Div. Software & program.

SÄLJES ALLT TILLSAMMANS
7000:-

ELLER

Videokort C-Å	**1100:-**
16 K RAM	**2000:-**
MOTOROLA EPROMKORT med 14 st 2708 (INTEL,NS) BASIC/Editor-Assembler m. manual	**1350:-**
Tangentbord LICON m. 19" panel	**650:-**
CPU (D-2) modifierad	**1200:-**
TV-monitor 5" Sv/v	**500:-**
Låda, nätagg, moderkort kontakter, bellkort, KCS kort, högtalare	**550:-**

Ulf T. Johansson
Vinddraget 12 B
802 30 GÄVLE

026-14 24 38 hem
026-10 03 40 arb

PROGRAMMVARA 6809

Det vimlar än så länge inte av program för 6809, men vi har gjort en liten sammanfattning av vad som finns i dag.

Monitorprogram:

Från SWTPC finns S-BUG 1.5, en monitor med ungefär samma funktioner som Mikbug. Den är på 2 k och har bl.a. inbyggd disk-boot. Den har adressområdet $F800-FFFF.
Från CÅ finns CBUG 1 och CBUG 2. Dessa båda är på 1 resp 2 k och har samma funktioner som TBUG1. CBUG 2 har dessutom disk-boot samt flera rutiner för sökning av bitmönster. Det går även att välja port (15 st) vid dumpning, punchning och laddning. CBUG är adresskompatibel med S-BUG.

Diskoperativsystem

TSC har en hel del till 6809. Det finns en DOS som heter FLEX 9.0 och som har samma egenskaper som FLEX 2.0, vilken vi har berättat om tidigare i MPU-laren. Den assembler som medföljer FLEX 9.0 klarar även macros. FLEX 9.0 kostar $90,-
Debug för 6809 är en mycket användbar "avlusare" för egna program. En välskriven manual ingår och DEBUG 6809 kostar $ 75,-
Basic finns givetvis också. Det är den välkända 6-siffrors basicen som i 6809-utförande är ännu snabbare. Priset är $ 65,-
TSC annonserar att de snart kommer med sort/merge och en 17-siffrors affärsbasic för 6809. Det är nog inte så långt tills en Pascal-kompilator finns tillgänglig också.
Motorola har debug för sin exorciser med 6809 samt diskoperativsystem med tillhörande editor, assembler och basic-interpretator.

MOTOROLA EXORset 30

Det nya utvecklingssystemet för M6809 — marknadens kraftfullaste 8-bitars MPU.



EXORset 30 är ett komplett utvecklingssystem i en kompakt enhet. Bland de många fördelarna med EXORset kan nämnas:

- Komplett 64 teckens ASCII tangentbord med 16 funktionstangenter
- 9" bildskärm för 22 rader/80 tecken eller 16 rader/40 tecken samt grafisk bild
- Dubbla mini-flexskiveenheter med 160K byte massminne och XDOS operativsystem som är kompatibelt med MDOS
- Utvidgad BASIC kompilator med interpretatormöjlighet. Kompilatorn genererar positionsoberoende re-entrant kod
- 48K byte RAM och hållare för 24K byte ROM/EPROM
- Plats för expansion för t ex extra RAM, I/O eller PROM-programmerare
- EXORset är helt kompatibel med EXORciser, EXORterm och Micromodules
- Lågt pris ca 25.000:-
- Leverans från lagret vid årsskiftet

MOTOROLA Semiconductor

DISTRIBUTÖREN

08/13 21 60

INTERELKO AB
Box 32
122 21 ENSKEDE

MPU STRATEGIC MARKETING EUROPE *GENEVA MAY 14th, 1979*

ALL REGIONS

Please find enclosed a proposal from ATX how to modify a 6800 D2-Kit for using it with a MC 6809. However the standard D2 key board and display is not used and an external terminal must be connected. The MBUG9 Monitor provides for the

L Load

P Punch

M Memory change

R Register display

G Go

commands, but does not have breakpoint - and trace capabilities.

The proposed solution has not been checked in GVA.

Best regards

K. Schneider

KS/ws
Attachment

Kopierigsservice av program införda i MPU-laren.Kopiering kann ske på kassett gratis eller på DISK system Flex och Smoke till självkostnadspris.Ring någon i redaktionen eller skriv till klubben.

```
AUSTIN,TEXAS--MICROCOMPUTER CAPITAL OF THE WORLD!
M6800-M6809 CROSS-ASSEMBLER  2.2
PAGE  001      MBUG9

00001                                   NAM    MBUG9
00002                8008       ACIAS   EQU    $8008  E708  8008
00003                8009       ACIAD   EQU    $8009  E709  8009
00004                A07F       STACK   EQU    $A07F  E7FF
00005                003F       SWI     EQU    $3F
00006 E000                              ORG    $E000  E800
00007                           *  I/O INTERRUPT SEQUENCES
00008                           *
00009                           *
00010                E000       FCTABL  EQU    *
00011 E000           4C                 FCC    /L/
00012 E001           E06A               FDB    LOAD
00013 E003           50                 FCC    /P/
00014 E004           E15D               FDB    PUNCH
00015 E006           4D                 FCC    /M/
00016 E007           E10C               FDB    CHANGE
00017 E009           52                 FCC    /R/
00018 E00A           E1EA               FDB    PSTAK
00019 E00C           47                 FCC    /G/
00020 E00D           E14C               FDB    GOTO
00021                E00F       FCTBEN  EQU    *
00022                           *  INITIALIZATION/RESET CODES
00023                           *
00024                           *
00025                           *
00026                           *
00027                E00F       START   EQU    *
00028 E00F 8E   A07F      3             LDX    #STACK
00029 E012 BF   A00C      6             STX    SP
00030 E015 8E   E238      3             LDX    #SWIS
00031 E018 BF   A027      6             STX    SWIJ
00032                           *
00033                           *  INITIALIZE ACIA
00034 E01B 86   03        2             LDAA   #3          MASTER RESET
00035 E01D B7   8008      5             STAA   ACIAS
00036 E020 86   51        2             LDAA   #$51
00037 E022 B7   8008      5             STAA   ACIAS
00038                           *
00039                           *  MAIN LOOP
00040                           *
00041 E025 10FE A00C      7  CONTRL LDS    SP
00042 E029 7F   A01A      7             CLR    OUTSW
00043 E02C 8E   E24E      3             LDX    #MCLOFF   TERMINAL INIT STRING
00044 E02F BD   E105      8             JSR    PDATA1    PRINT ITI
00045 E032 8D   1E        7             BSR    INCH1     READ COMMAND CHARACTER
00046 E034 1F   89 4D     8             TAB              SAVE IN B
00047 E037 BD   E1E5      8             JSR    OUTS      PRINT SPACE AFTER COMMA
00048                           *  B-REG CONTAINS CHAR - USE JUMP TABLE TO GO
00049                           *
00050 E03A 8E   E000      3             LDX    #FCTABL   X= ADDR OF J.T.
00051 E03D E1   84        4  NXTCHR CMPB   0,X        MATCH?
00052 E03F 27   0D        3             BEQ    GOODCH    YES!  GO TO RIGHT PLACE
00053 E041 30   01        5             INX
00054 E043 30   01        5             INX
```

Forts. nästa sida

```
USTIN,TEXAS--MICROCOMPUTER CAPITAL OF THE WORLD!
S800-M6809 CROSS-ASSEMBLER  2.2
AGE  002     MBUG9

0055 E045 30   01        5          INX
0056 E047 8C   E00F      4          CPX   #FCTBEN  END OF TABLE YET?
0057 E04A 26   F1        3          BNE   NXTCHR   NOPE - TRY AGAIN
0058 E04C 20   D7        3          BRA   CONTRL
0059 E04E AE   01        6 GOODCH   LDX   1,X      GET ADDR FM J.T.
0060 E050 6E   84        3          JMP   0,X      GO DO IT!
0061                       *
0062                       *
0063                       *
0064 E052 7E   E0EC      4 INCH1    JMP   INCH
0065                       *
0066                       *  INPUT HEX CHAR
0067                       *
0068 E055 8D   FB        7 INHEX    BSR   INCH1
0069 E057 80   30        2 INHEX2   SUBA  #$30
0070 E059 2B   CA        3          BMI   CONTRL   NOT HEX
0071 E05B 81   09        2          CMPA  #$09
0072 E05D 2F   0A        3          BLE   IN1HG
0073 E05F 81   11        2          CMPA  #$11
0074 E061 2B   C2        3          BMI   CONTRL   NOT HEX
0075 E063 81   1B        2          CMPA  #$1B
0076 E065 2E   BE        3          BGT   CONTRL   NOT HEX
0077 E067 80   07        2          SUBA  #7
0078 E069 39             5 IN1HG    RTS
0079                       *
0080             E06A      LOAD     EQU   *
0081 E06A 86   11        2          LDAA  #@21
0082 E06C 8D   6F        7          BSR   OUTCH    OUTPUT A DCW1
0083                       *
0084 E06E 7C   A01A      7          INC   OUTSW    DO NOT ECHO TAPE
0085                       *
0086 E071 8D   79        7 LOAD3    BSR   INCH
0087 E073 81   53        2          CMPA  #'S
0088 E075 26   FA        3          BNE   LOAD3    1ST CHAR NOT AN 'S
0089 E077 8D   73        7          BSR   INCH
0090 E079 81   39        2          CMPA  #'9
0091 E07B 27   A8        3          BEQ   CONTRL
0092 E07D 81   31        2          CMPA  #'1
0093 E07F 26   F0        3          BNE   LOAD3    2ND CHAR NOT A '1
0094 E081 7F   A029      7          CLR   CKSM     ZERO CHECKSUM
0095 E084 8D   2F        7          BSR   BYTE     READ BYTE
0096 E086 80   02        2          SUBA  #2
0097 E088 B7   A02A      5          STAA  BYTECT   BYTE COUNT
0098                       *
0099                       *  BUILD ADDRESS
0100                       *
0101 E08B 8D   1A        7          BSR   BADDR
0102                       *
0103                       *  STORE DATA
0104                       *
0105 E08D 8D   26        7 LOAD11   BSR   BYTE
0106 E08F 7A   A02A      7          DEC   BYTECT
0107 E092 27   0A        3          BEQ   LOAD15   ZERO BYTE COUNT
0108 E094 A7   84        4          STAA  0,X      STORE DATA
```

```
00109 E096 A1   84        4          CMPA  0,X       CHECK DATA
00110 E098 26   09        3          BNE   LOAD19    DATA NOT STORED
00111 E09A 30   01        5          INX
00112 E09C 20   EF        3          BRA   LOAD11
00113                               *
00114                               *  DOES CHECKSUM MATCH?
00115                               *
00116 E09E 7C   A029      7 LOAD15  INC   CKSM
00117 E0A1 27   CE        3         BEQ   LOAD3
00118 E0A3 86   3F        2 LOAD19  LDAA  #'?
00119 E0A5 8D   36        7         BSR   OUTCH     PRINT QUESTION MARK
00120                               *
00121                               *
00122 E0A7 8D   0C        7 BADDR   BSR   BYTE      READ 2 FRAMES
00123 E0A9 B7   A02B      5         STAA  XHI
00124 E0AC 8D   07        7         BSR   BYTE
00125 E0AE B7   A02C      5         STAA  XLOW
00126 E0B1 BE   A02B      6         LDX   XHI       (X) ADDRESS WE BUILT
00127 E0B4 39             5         RTS
00128                               *
00129                               *  INPUT BYTE (TWO FRAMES
00130                               *
00131 E0B5 17   FF9D      9 BYTE    LBSR  INHEX     GET HEX CHAR
00132 E0B8 48             2 BYTE2   ASLA
00133 E0B9 48             2         ASLA
00134 E0BA 48             2         ASLA
00135 E0BB 48             2         ASLA
00136 E0BC 1F   89 4D     8         TAB
00137 E0BF 8D   94        7         BSR   INHEX
00138 E0C1 34   04 ABE0  11         ABA
00139 E0C5 1F   89 4D     8         TAB
00140 E0C8 FB   A029      5         ADDB  CKSM
00141 E0CB F7   A029      5         STAB  CKSM
00142 E0CE 39             5         RTS
00143                               *
00144 E0CF 44             2 OUTHL   LSRA            OUT HEX LEFT BCD DIGIT
00145 E0D0 44             2         LSRA
00146 E0D1 44             2         LSRA
00147 E0D2 44             2         LSRA
00148                               *
00149 E0D3 84   0F        2 OUTHR   ANDA  #$F       OUT HEX RIGHT BCD DIGIT
00150 E0D5 8B   30        2         ADDA  #$30
00151 E0D7 81   39        2         CMPA  #$39
00152 E0D9 23   02        3         BLS   OUTCH
00153 E0DB 8B   07        2         ADDA  #$7
00154                               *
00155                               *  OUTPUT ONE CHARACTER
00156                               *
00157 E0DD 34   04        5 OUTCH   PSHB            SAVE B
00158 E0DF F6   8008      5 OUTC1   LDAB  ACIAS
00159 E0E2 57             2         ASRB
00160 E0E3 57             2         ASRB
00161 E0E4 24   F9        3         BCC   OUTC1     XMIT NOT READY
00162 E0E6 B7   8009      5         STAA  ACIAD
```

Forts. nästa sida

```
00163 E0E9 35   04        5          PULB
00164 E0EB 39             5          RTS
00165                       *
00166                       *  INPUT ONE CHAR TO A-REG
00167                       *
00168 E0EC B6   8008      5 INCH     LDAA  ACIAS
00169 E0EF 47             2          ASRA
00170 E0F0 24   FA        3          BCC   INCH       RCVER NOT READY
00171 E0F2 B6   8009      5          LDAA  ACIAD
00172 E0F5 84   7F        2          ANDA  #$7F       STRIP PARITY
00173 E0F7 81   7F        2          CMPA  #$7F
00174 E0F9 27   F1        3          BEQ   INCH
00175 E0FB 7D   A01A      7          TST   OUTSW      ECHO?
00176 E0FE 27   DD        3          BEQ   OUTCH      YES
00177 E100 39             5          RTS              NO - GO HOME
00178                       *
00179                       *  PRINT DATA POINTED AT BY X - REGISTER
00180                       *
00181 E101 8D   DA        7 PDATA2   BSR   OUTCH
00182 E103 30   01        5          INX
00183 E105 A6   84        4 PDATA1   LDAA  0,X
00184 E107 81   04        2          CMPA  #$04
00185 E109 26   F6        3          BNE   PDATA2
00186 E10B 39             5          RTS              STOP ON EOT
00187                       *
00188                       *  CHANGE MEMORY (M AAAA DD NN)
00189                       *
00190 E10C 8D   99        7 CHANGE   BSR   BADDR      BUILD ADDRESS
00191 E10E 17   00D4      9          LBSR  OUTS
00192 E111 BE   A02B      6 CHANG    LDX   XHI
00193 E114 17   00CC      9          LBSR  OUT2HS     PRINT OLD DATA
00194 E117 30   1F        5          DEX
00195 E119 8D   D1        7 CHA1     BSR   INCH       INPUT CHAR
00196 E11B 81   0A        2          CMPA  #$0A
00197 E11D 27   13        3          BEQ   LF         CHECK FOR LF
00198 E11F 81   5E        2          CMPA  #$5E
00199 E121 27   13        3          BEQ   UA         CHECK FOR ^
00200 E123 BD   E057      8          JSR   INHEX2
00201 E126 8D   90        7          BSR   BYTE2      GET NEW BYTE
00202 E128 A7   84        4          STAA  0,X        CHANGE MEMORY
00203 E12A A1   84        4          CMPA  0,X
00204 E12C 1026 FF73      6          LBNE  LOAD19     NO CHANGE
00205 E130 20   E7        3          BRA   CHA1
00206 E132 30   01        5 LF       INX
00207 E134 20   0E        3          BRA   UA1
00208 E136 86   0A        2 UA       LDAA  #$0A
00209 E138 8D   A3        7          BSR   OUTCH      PRING LF
00210 E13A 30   1F        5          DEX
00211 E13C BF   A02B      6 UA1      STX   XHI        SAVE DATA ADDR
00212 E13F 8E   E250      3          LDX   #MCL+1
00213 E142 8D   C1        7          BSR   PDATA1     PRINT CR
00214 E144 8E   A02B      3          LDX   #XHI
00215 E147 17   0087      8          LBSR  OUT4HS     OUTPUT DATA ADDR
00216 E14A 20   C5        3          BRA   CHANG
```

```
00217                                *
00218                                *
00219                                *  GOTO REQUESTED
00220                                *
00221 E14C BD   E1D2      8 GOTO     JSR    BADDRJ    GO GET ADDRESS FROM USE
00222                                *              XHI,XLOW HOLD ADDRESS
00223 E14F BE   A02B      6          LDX    XHI       GET ADDRESS TO GOTO
00224 E152 6E   84        3          JMP    0,X       TRANSFER CONTROL TO USE
00225                                *
00226                                *  PUNCH DUMP
00227                                *  PUNCH FROM BEGINNING ADDRESS (BEGA) THRU ENDING
00228                                *  ADDRESS (ENDA)
00229                                *
00230 E154      0D                   MTAPE1 FCB    $D,$A,0,0,0,0,'S,'1,4  PUNCH FOR
00231           E15D                 PUNCH  EQU    *
00232 E15D 86   12        2          LDAA   #$12      TURN PUNCH ON
00233 E15F BD   E0DD      8          JSR    OUTCH
00234                                *  PUNCH LEADER - 25 NULLS
00235 E162 C6   19        2          LDAB   #25       B HOLDS # NULLS
00236 E164 4F             2 PNULL    CLRA
00237 E165 BD   E0DD      8          JSR    OUTCH
00238 E168 5A             2          DECB
00239 E169 26   F9        3          BNE    PNULL
00240 E16B BE   A000      6          LDX    BEGA
00241 E16E BF   A02D      6          STX    TW
00242 E171 B6   A003      5 PUN11    LDAA   ENDA+1
00243 E174 B0   A02E      5          SUBA   TW+1
00244 E177 F6   A002      5          LDAB   ENDA
00245 E17A F2   A02D      5          SBCB   TW
00246 E17D 26   04        3          BNE    PUN22
00247 E17F 81   10        2          CMFA   #16
00248 E181 25   02        3          BCS    PUN23
00249 E183 86   0F        2 PUN22    LDAA   #15
00250 E185 8B   04        2 PUN23    ADDA   #4
00251 E187 B7   A02A      5          STAA   MCONT     FRAME COUNT THIS RECORD
00252 E18A 80   03        2          SUBA   #3
00253 E18C B7   A029      5          STAA   TEMP      BYTE COUNT THIS RECORD
00254                                *
00255                                *  PUNCH CR/LF/NULLS/S/1
00256                                *
00257 E18F 8E   E154      3          LDX    #MTAPE1
00258 E192 BD   E105      8          JSR    FDATA1
00259 E195 5F             2          CLRB             ZERO CHECKSUM
00260                                *
00261                                *  PUNCH FRAME COUNT
00262                                *
00263 E196 8E   A02A      3          LDX    #MCONT
00264 E199 8D   32        7          BSR    PUNT2     PUNCH 2 HEX CHAR
00265                                *
00266                                *  PUNCH ADDRESS
00267                                *
00268 E19B 8E   A02D      3          LDX    #TW
00269 E19E 8D   2D        7          BSR    PUNT2
00270 E1A0 8D   23        7          BSR    PUNT2
```

Forts. nästa sida

```
AUSTIN,TEXAS--MICROCOMPUTER CAPITAL OF THE WORLD!
M6800-M6809 CROSS-ASSEMBLER  2.2
PAGE  006     MBUG9

00271                          *
00272                          *  PUNCH DATA
00273                          *
00274 E1A2 BE   A02D      6          LDX   TW
00275 E1A5 8D   26        7 PUN32    BSR   PUNT2
00276 E1A7 7A   A029      7          DEC   TEMP
00277 E1AA 26   F9        3          BNE   PUN32
00278 E1AC BF   A02D      6          STX   TW
00279 E1AF 53             2          COMB
00280 E1B0 34   04        5          PSHB
00281 E1B2 1F   41        6          TSX
00282 E1B4 8D   17        7          BSR   PUNT2     PUNCH CKSM
00283 E1B6 35   04        5          PULB            RESTORE STACK
00284 E1B8 BE   A02D      6          LDX   TW
00285 E1BB 30   1F        5          DEX
00286 E1BD BC   A002      7          CPX   ENDA
00287 E1C0 26   AF        3          BNE   PUN11
00288 E1C2 8E   E24C      3          LDX   #MEOF
00289 E1C5 BD   E105      8          JSR   PDATA1    OUTPUT EOF
00290 E1C8 20   00        3          BRA   CTRL      BRANCH TO CONTROL
00291 E1CA 7E   E025      4 CTRL     JMP   CONTRL
00292                          *
00293                          *  PUNCH 2 HEX CHAR, UPDATE CKSUM
00294 E1CD EB   84        4 PUNT2    ADDB  0,X
00295 E1CF 7E   E1D5      4          JMP   OUT2H     OUTPUT TWO HEX CHAR AND
00296                          *
00297 E1D2 7E   E0A7      4 BADDRJ   JMP   BADDR     GO BUILD ADDRESS
00298                          *
00299 E1D5 A6   84        4 OUT2H    LDAA  0,X       OUTPUT 2 HEX CHAR
00300 E1D7 BD   E0CF      8 OUT2HA   JSR   OUTHL     OUT LEFT HEX CHAR
00301 E1DA A6   84        4          LDAA  0,X
00302 E1DC 30   01        5          INX
00303 E1DE 16   FEF2      5          LBRA  OUTHR     OUTPUT RIGHT HEX CHAR
00304                          *
00305 E1E1 8D   F2        7 OUT4HS   BSR   OUT2H
00306 E1E3 8D   F0        7 OUT2HS   BSR   OUT2H
00307 E1E5 86   20        2 OUTS     LDAA  #$20
00308 E1E7 16   FEF3      5          LBRA  OUTCH     (BSR+RTS)
00309                          *
00310                          *  PRINT STACK CONTENTS
00311 E1EA 8D   09        7 PSTAK    BSR   PRINT     PRINT STACK
00312 E1EC 7E   E025      4 CNTRL2   JMP   CONTRL    RETURN TO COMMAND LEVEL
00313 E1EF 7E   E1E3      4 OT2HS    JMP   OUT2HS
00314 E1F2 7E   E1E1      4 OT4HS    JMP   OUT4HS
00315                          *
00316 E1F5 8E   E24F      3 PRINT    LDX   #MCL      CR-LF-*
00317 E1F8 BD   E105      8          JSR   PDATA1    MAKE NEW LINE
00318 E1FB 8E   E258      3          LDX   #MCL1
00319 E1FE BD   E105      8          JSR   PDATA1
00320 E201 BE   A00C      6          LDX   SP        PRINT OUT STACK
00321 E204 8D   E9        7          BSR   OT2HS     CCR
00322 E206 8D   E7        7          BSR   OT2HS     ACCA
00323 E208 8D   E5        7          BSR   OT2HS     ACCB
00324 E20A 8D   E3        7          BSR   OT2HS     DPR
```

```
00325 E20C 8D   E4       7        BSR   OT4HS    X-REG
00326 E20E 8D   E2       7        BSR   OT4HS    Y-REG
00327 E210 8D   E0       7        BSR   OT4HS    U-SP
00328 E212 8D   DE       7        BSR   OT4HS    PC
00329 E214 8E   A00C     3        LDX   #SP
00330 E217 8D   D9       7        BSR   OT4HS    S-SP
00331 E219 39            5        RTS
00332                             *
00333                             *
00334                             *
00335                             *  JUMP TABLE FOR VECTORING
00336                             *
00337                             *
00338 E21A BE   A01D     6 I1     LDX   IOV      NMI
00339 E21D 6E   84       3  .     JMP   X
00340                             *
00341 E21F BE   A027     6 I2     LDX   SWIJ     SWI  PRINT OUT REGISTER
00342 E222 6E   84       3        JMP   X
00343                             *
00344 E224 BE   A01F     6 I3     LDX   IOV2     IRQ
00345 E227 6E   84       3        JMP   X
00346                             *
00347 E229 BE   A021     6 I4     LDX   IOV3     FIRQ
00348 E22C 6E   84       3        JMP   X
00349                             *
00350 E22E BE   A023     6 I5     LDX   IOV4     SWI2
00351 E231 6E   84       3        JMP   X
00352                             *
00353 E233 BE   A025     6 I6     LDX   IOV5     SWI3
00354 E236 6E   84       3        JMP   X
00355                             *
00356                             *  SOFTWARE INTERRUPT PROCESSING (SWI)
00357                             *
00358            E238             SWIS  EQU   *
00359 E238 10FF A00C     7        STS   SP       SAVE USERS SP
00360 E23C 1F   41       6        TSX
00361 E23E 6D   0C       7        TST   12,X
00362 E240 26   02       3        BNE   SWIS1
00363 E242 6A   0B       7        DEC   11,X
00364 E244 6A   0C       7 SWIS1  DEC   12,X
00365 E246 BD   E1F5     8        JSR   PRINT
00366 E249 7E   E025     4        JMP   CONTRL
00367                             *
00368                             *
00369                             *
00370                             *
00371                             *
00372                             *
00373                             *
00374 E24C       53      MEOF     FCC   /S9/
00375 E24E       13      MCLOFF   FCB   $13      READER OFF
00376 E24F       0A      MCL      FCB   $A,$D,$14,0,0,0,0,'M,4   LF,CR,PU
00377 E258       0A      MCL1     FCB   $A,$D,0,0,0,0
00378 E25E       43               FCC   /CC A  B  DP  X    Y    U    PC
```

Forts. nästa sida

```
JSTIN,TEXAS--MICROCOMPUTER CAPITAL OF THE WORLD!
3800-M6809 CROSS-ASSEMBLER  2.2
AGE  008      MBUG9

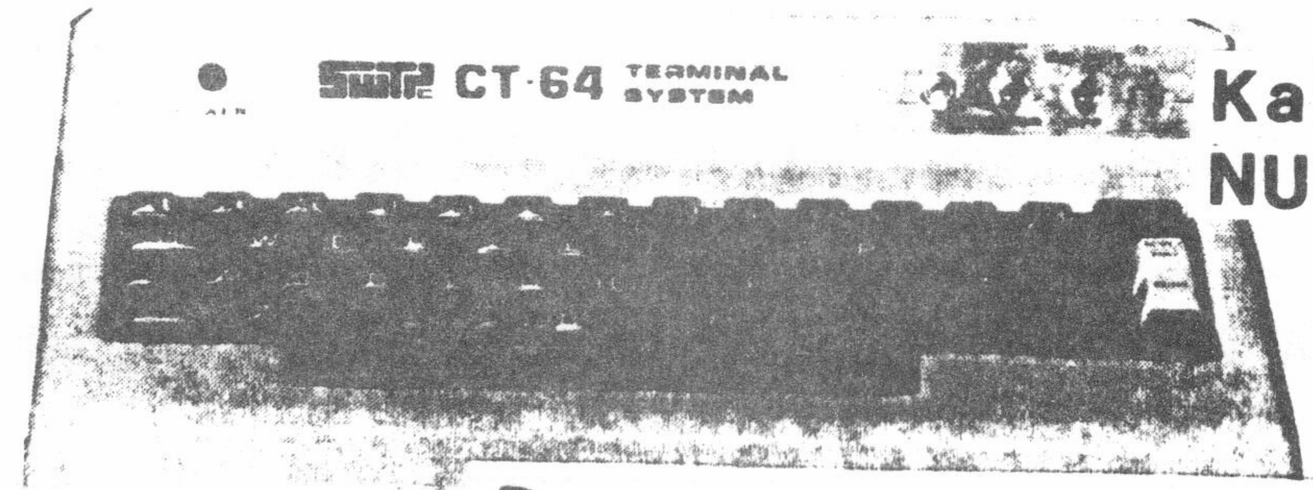
0379 E281       0A                      FCB    $A,$D,0,0,0,0
0380 E287       04                      FCB    04
0381                            *
0382                            *
0383                            *
0384                            *
0385                            *
0386                            *
0387                            *
0388                            *
0389                            *
0390 A000                               ORG    $A000  E600 (A00)
0391 A000       0002            BEGA    RMB    2
0392 A002       0002            ENDA    RMB    2
0393 A004       0002            NIO     RMB    2
0394 A006       0002            NIO1    RMB    2
0395 A008       0002            NIO2    RMB    2
0396 A00A       0002            NIO3    RMB    2
0397 A00C       0002            SP      RMB    2
0398 A00E       0002            SWI1    RMB    2
0399 A010       0002            SWI2    RMB    2
0400 A012       0008            BRINS   RMB    8
0401            A01A            BRANEN  EQU    *
0402 A01A       0001            OUTSW   RMB    1
0403 A01B       0002            TRCADR  RMB    2
0404 A01D       0002            IOV     RMB    2       NMI VECTOR
0405 A01F       0002            IOV2    RMB    2       IRQ VECTOR
0406 A021       0002            IOV3    RMB    2       FIRQ VECTOR
0407 A023       0002            IOV4    RMB    2       SWI2 VECTOR
0408 A025       0002            IOV5    RMB    2       SWI3 VECTOR
0409 A027       0002            SWIJ    RMB    2
0410            A029            CKSM    EQU    *
0411            A029            ASAVE   EQU    *
0412            A029.           TEMP    EQU    *
0413 A029       0001                    RMB    1
0414            A02A            BYTECT  EQU    *
0415            A02A            MCONT   EQU    *
0416 A02A       0001                    RMB    1
0417 A02B       0001            XHI     RMB    1
0418 A02C       0001            XLOW    RMB    1
0419            A02D            SSAVE   EQU    *
0420            A02D            TW      EQU    *
0421 A02D       0002                    RMB    2
0422            A02E            ENDIN0  EQU    *-1
0423 A02F       003A                    RMB    58   E7F2
0424 E3F2                               ORG    $E3F2      (E3F2)
0425 E3F2       E233                    FDB    I6         SWI3
0426 E3F4       E22E                    FDB    I5         SWI2
0427 E3F6       E229                    FDB    I4         FIRQ
0428 E3F8       E224                    FDB    I3         IRQ
0429 E3FA       E21F                    FDB    I2         SWI
0430 E3FC       E21A                    FDB    I1         NMI
0431 E3FE       E00F                    FDB    START      RESET
0432            0000                    END
```

```
AUSTIN,TEXAS--MICROCOMPUTER CAPITAL OF THE WORLD!
M6800-M6809 CROSS-ASSEMBLER  2.2
PAGE  010      MBUG9

**** SYMBOL TABLE
ACIAD  8009  ACIAS  8008  ASAVE  A029  BADDR  E0A7  BADDRJ E1D2
BEGA   A000  BRANEN A01A  BRINS  A012  BYTE   E0B5  BYTE2  E0B8
BYTECT A02A  CHA1   E119  CHANG  E111  CHANGE E10C  CKSM   A029
CNTRL2 E1EC  CONTRL E025  CTRL   E1CA  ENDA   A002  ENDIN0 A02E
FCTABL E000  FCTBEN E00F  GOODCH E04E  GOTO   E14C  I1     E21A
I2     E21F  I3     E224  I4     E229  I5     E22E  I6     E233
IN1HG  E069  INCH   E0EC  INCH1  E052  INHEX  E055  INHEX2 E057
IOV    A01D  IOV2   A01F  IOV3   A021  IOV4   A023  IOV5   A025
LF     E132  LOAD   E06A  LOAD11 E08D  LOAD15 E09E  LOAD19 E0A3
LOAD3  E071  MCL    E24F  MCL1   E258  MCLOFF E24E  MCONT  A02A
MEOF   E24C  MTAPE1 E154  NIO    A004  NIO1   A006  NIO2   A008
NIO3   A00A  NXTCHR E03D  OT2HS  E1EF  OT4HS  E1F2  OUT2H  E1D5
OUT2HA E1D7  OUT2HS E1E3  OUT4HS E1E1  OUTC1  E0DF  OUTCH  E0DD
OUTHL  E0CF  OUTHR  E0D3  OUTS   E1E5  OUTSW  A01A  PDATA1 E105
PDATA2 E101  PNULL  E164  PRINT  E1F5  PSTAK  E1EA  PUN11  E171
PUN22  E183  PUN23  E185  PUN32  E1A5  PUNCH  E15D  PUNT2  E1CD
SP     A00C  SSAVE  A02D  STACK  A07F  START  E00F  SWI    003F
SWI1   A00E  SWI2   A010  SWIJ   A027  SWIS   E238  SWIS1  E244
TEMP   A029  TRCADR A01B  TW     A02D  UA     E136  UA1    E13C
XHI    A02B  XLOW   A02C
```


S W T P C

CT-64



Kat pris 2.415:-
NU 1.750:-

Pascal för 6800 finnes och kommer för 6809

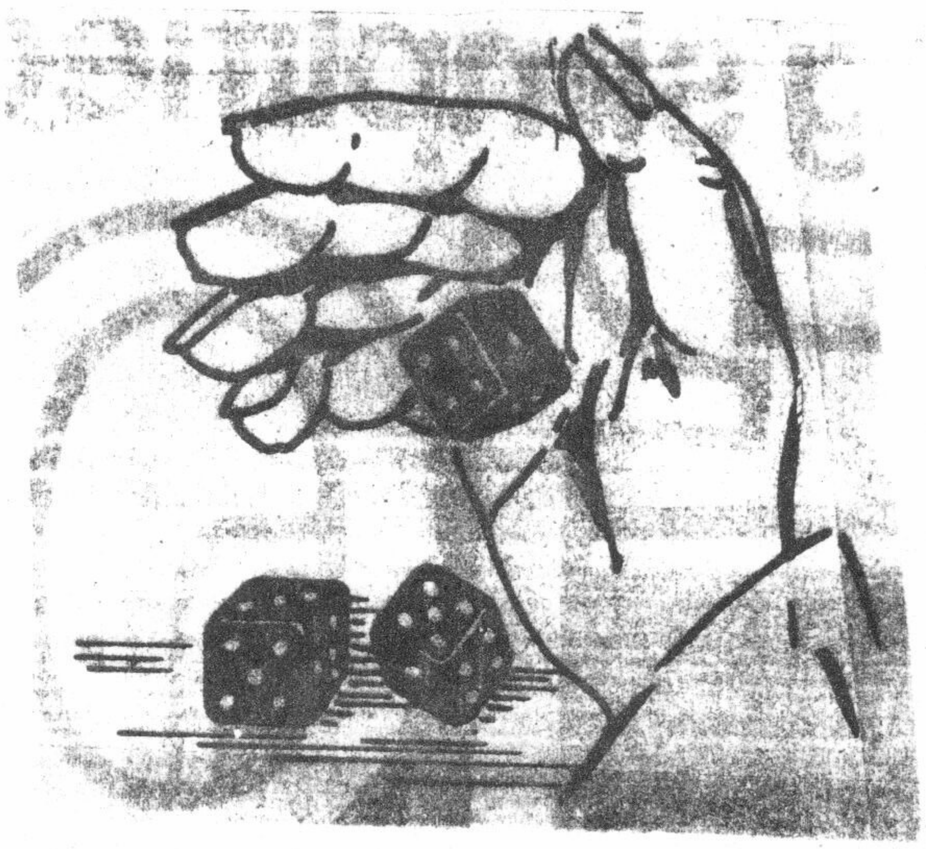
- 64 tecken per rad, 16 rader.
- 56 riktiga tangenter placerade som på en skrivmaskin.
- Page eller Scrolling. • Stora och små bokstäver, alla tecken och siffror för Basic, 128 ASCII tecken totalt. Ej sv tecken.
- 32 avkodade kontrolltecken, kan även skriva dessa på monitorn. • Svart eller vit text, fast eller blinkande cursor.
- Highlighting = vit bakgrund till enstaka tecken eller hela ord.
- 110—150—300 baud. RS 232 utgång.
- Två ospecifierade tangenter för eget bruk. • Automatisk repetition om en tangent nedtrycks mer än ca 0.5 sekunder.
- För 625 linjer 50 Hz. Helt komplett, inget extra behöves för att kunna göra det som nämnts ovan.

Swedish Electronics hb

BOX 2065, 750 02 UPPSALA. TEL. 018-100190

TÄVLING!

MOTOROLA SVERIGE har skänkt MPU-laren ett antal priser bland annat freesbee och tröjor



TÄVLINGEN är enkel gissa bara hur många många medlemmar det fanns i klubben den 1nov 1979!

Alla utom styrelsen och redaktionen får deltaga.

Svar senast 30-11-79 per post.

ENVÄLDIG DOMMARE REDAKTIONEN

String Search . . . added MIKBUG capability

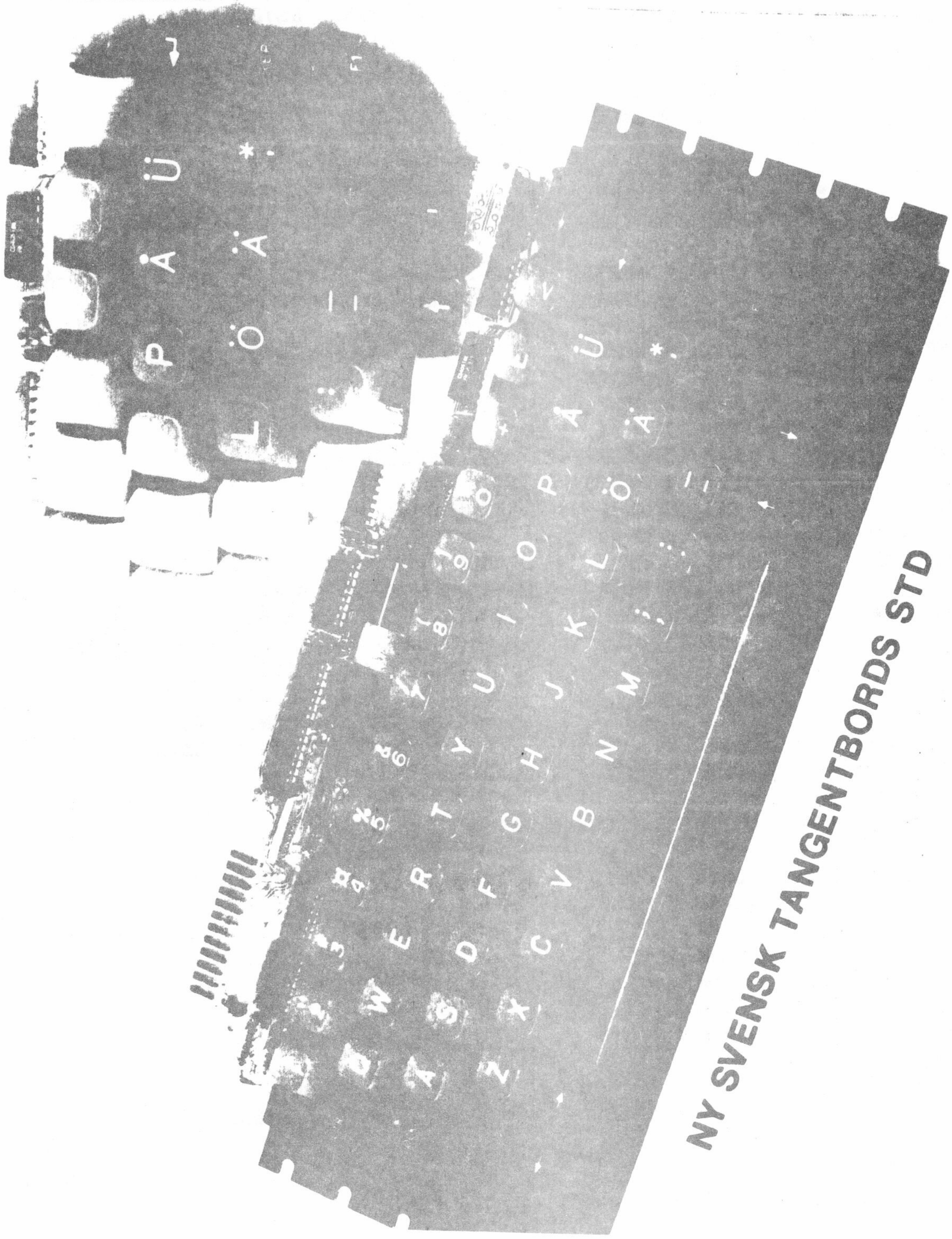
Dale Puckett
Quarters 674-D
Governors Island NY 10004

```
TARGET   RMB    2
TARTEM   RMB    2
FINSTR   JSR    BADDR     This is a MIKBUG routine that leaves a two byte address
                          in the X register. In this case it is used to input
                          the two byte string you are looking for.
         STX    TARGET
         LDX    #0        Point to the beginning of memory.
LOOP2    LDA A  0,X       Get a character or byte.
         CMP A  TARGET
         BEQ    FNDONE    Branch if found.
         INX              Bump pointer.
         CPX    #3FFF     See if done. $3FFF is the top of my memory.
         BEQ    MIKBUG    If so, go back to monitor.
         BRA    LOOP2
FNDONE   LDA A  1,X       Get second Byte.
         CMP A  TARGET+1
         BEQ    FNDTWO    Branch if second byte is correct also.
         INX              If not, bump pointer.
         BRA    LOOP2     Try again.
FNDTWO   STX    TARTEM    If you have found the two byte string, store the address
                          temporarily.
         LDX    #TARTEM   Point to it.
         JSR    OUT4HS    A MIKBUG routine which outputs 4 hex characters (2 bytes).
         LDX    TARTEM    Restore pointer.
         INX              Increment it.
         BRA    LOOP2     Continue till done.
```

FINSTR is a short routine that will help you locate a two byte string in memory and print out every address where it appears on the control terminal. I use it with a monitor program I use to supplement MIKBUG.

En svensk profil

- layout enligt SIS 66 22 41



```
00430  A000                OUTA   EQU  A ÅA000    PROM ADDR POINTER
00440  A002                BEGA   EQU  A ÅA002    MEM START ADDR  * LAST CHAR *
00450  A004                ENDA   EQU  A ÅA004    MEM END ADDR    *MUST BE SAME*
00460  A00C                DISF   EQU  A ÅA00C    JBUG WORK AREA 'DISBUF'
00470  A014                DIG4   EQU  A ÅA014    JBUG WORK AREA 'DIGIN4'
00480  A01A                XKEF   EQU  A ÅA01A    JBUG WORK AREA 'XKEYBF'
00490  E084                DISI   EQU  A ÅE084    JBUG SUBROUTINE 'DISNMI'
00500  E1F7                KEYC   EQU  A ÅE1F7    JBUG SUBROUTINE 'KEYDCC'
00510  E0E1                DLY1   EQU  A ÅE0E1    JBUG SUBROUTINE 'DLY'
00520  8004                PDRA   EQU  A Å8004    PA0-PA5=A4-A9 PA6=WE PA7=SET-A0-A3
00530  8005                PCRA   EQU  A Å8005    CA2=PP
00540  8006                PDRB   EQU  A Å8006    PB0-PB7=D0-D7
00550  8007                PCRB   EQU  A Å8007    CB2=AUTO-INCR A0-A3
00560  0100                       ORG  A Å0100
00570                      *
00580                      *** INITIALIZATION
00590                      *                                          --
00600  0100  CE FF34  INIT  LDX  I ÅFF34    PA0-PA5=A4-A9 PA6=CS/WE PA7 SETS A0-A3
00610  0103  FF 8004        STX  E PDRA     CA2 TOGGLE PROGRAM-PULSE
00620  0106  CE FF2C        LDX  I ÅFF2C    PB0-PB7=D0-D7
00630  0109  FF 8006        STX  E PDRB     CB2 PULSES TO INCREMENT A0-A3
00640  010C  CE 0A00        LDX  I Å0A00    LOAD
00650  010F  FF A00C        STX  E DISF       ADDRESS
00660  0112  CE 0000        LDX  I 0            TO
00670  0115  FF A00E        STX  E DISF  +2       ADDRESS-AREA
00680  0118  CE A010        LDX  I ÅA010    INDICATE
00690  011B  FF A01A        STX  E XKEF       FOUR CHAR
00700  011E  7C A014        INC  E DIG4         INPUT
00710  0121  BD E084        JSR  E DISI     DISABLE NON-MASK INTRRUPT
00720  0124  8D 02          BSR  R FXSP     SAVE PC HERE AND JUMP
00730  0126  20 08    RET   BRA  R GO         YOU WILL RETURN HERE
00740  0128  34      FXSP  DES                 BUT JBUG HAS A RTI ON GO-COMMAND
00750  0129  34            DES                   SO WE WILL HAVE TO
00760  012A  34            DES                     FIX A STACK WITH
00770  012B  34            DES                       CORRECT LENGTH BUT WE
00780  012C  34            DES                       HAVE NOTING TO SAVE
00790  012D  7E E1F7        JMP  E KEYC     GO DISPLAY  'A000''  '
00800                      *
00810                      ***  GO STARTS PROGRAM TRANSFER
00820                      *
00830  0130  B6 A000  GO    LDAA E OUTA     SHIFT
00840  0133  F6 A001        LDAB E OUTA  +1   OVER
00850  0136  44            LSRA                OUTADDRESS
00860  0137  56            RORB                  FOUR
00870  0138  44            LSRA                    STEPS
00880  0139  56            RORB                      TO
00890  013A  54            LSRB                        RIGHT
00900  013B  54            LSRB                        --
00910  013C  CA 40          ORAB Ä %01000000    SET UP CS/WE AND
00920  013E  F7 A031        STAB E SHIA     STORE IN SHIFTED-ADDRESS
```

```
00930                        *
00940                        *** LOOP IS MAIN-LOOP FOR MULTIPLE OF 16 BYTES
00950                        *
00960  0141  8D 19    LOOP    BSR  R UTHX      GO PROM 16 BYTES
00970  0143  FE A034          LDX  E MPOS      WAS IT THE
00980  0146  BC A004          CPX  E ENDA        LAST 16 BYTES
00990  0149  27 08            BEQ  R UT            YES JUMP
01000  014B  FF A002          STX  E BEGA      NO STORE NEW BEGIN ADDR
01010  014E  7C A031          INC  E SHIA      INCREMENT SHIFTED-ADDR
01020  0151  2A EE            BPL  R LOOP      GO PROM NEXT 16 BYTES
01030                        *
01040                        *** UT IS PROGRAM END
01050                        *
01060  0153  B6 8004  UT      LDAA E PDRA      SET
01070  0156  84 BF            ANDA Ä %10111111  DOWN  --
01080  0158  B7 8004          STAA E PDRA              CS/WE
01090  015B  3F              SWI                         AND HALT
01100                        *
01110                        *** UTHEX IS IMIDIATE-LOOP 100 TIMES FOR EATCH 16 BYTES
01120                        *
01130  015C  86 64    UTHX    LDAA Ä 100       INITIATE CYCLE
01140  015E  B7 A032          STAA E VARV        COUNTER
01150                        *
01160  0161  B6 A031  UTH1    LDAA E SHIA      LOAD SHIFTED-ADDR  --
01170  0164  B7 8004          STAA E PDRA        SET UP A4-A9 AND CS/WE
01180  0167  8A 80            ORAA Ä %10000000  SET A0-A3
01190  0169  B7 8004          STAA E PDRA          TO 'F'
01200  016C  86 10            LDAA Ä 16        LOAD
01210  016E  B7 A033          STAA E LCTR        BYTE-CNTR
01220  0171  FE A002          LDX  E BEGA      INITIATE
01230  0174  FF A034          STX  E MPOS        MEMORY-ADDR
01240                        *
01250                        *** BYTE IS MINOR-LOOP
01260                        *
01270  0177  FE A034  BYTE    LDX  E MPOS      LOAD
01280  017A  A6 00            LDAA X             NEXT BYTE
01290  017C  08              INX              INCREMENT
01300  017D  FF A034          STX  E MPOS        MEMORY ADDR
01310  0180  B7 8006          STAA E PDRB      SET UP D0-D7
01320  0183  86 3C            LDAA Ä %00111100  PULSE
01330  0185  B7 8005          STAA E PCRA          PROGRAM
01340  0188  CE 0050          LDX  I 80              PIN
01350  018B  BD E0E1          JSR  E DLY1              FOR
01360  018E  86 34            LDAA Ä %00110100           ONE
01370  0190  B7 8005          STAA E PCRA                  MILISECOND
01380  0193  7A A033          DEC  E LCTR      MORE BYTE TO PROGRAM
01390  0196  2E DF            BGT  R BYTE        YES JUMP
01400  0198  7A A032          DEC  E VARV      HAVE WE DONE IT
01410  019B  2E C4            BGT  R UTH1        ONE HUNDRED TIMES
01420  019D  39                RTS                YES GO BACK
01430  A031                    ORG  A ÅA031
01440  A031          SHIA      RMB  A 1         SIX HIGHEST ADDR BITS
01450  A032          VARV      RMB  A 1         PROGRAM PULSE CNTR
01460  A033          LCTR      RMB  A 1         CNTR FOR LOWEST ADDR BITS
01470  A034          MPOS      RMB  A 2         CURRENT MEMORY POS
01480  A036                    END  A
TOTAL ERRORS   0
```

Forts. nästa sida

```
OUTA    A000    830   840
BEGA    A002   1000  1220
ENDA    A004    980
DISF    A00C    650   670
DIG4    A014    700
XKEF    A01A    690
DISI    E084    710
KEYC    E1F7    790
DLY1    E0E1   1350
PDRA    8004    610  1060  1080  1170  1190
PCRA    8005   1330  1370
PDRB    8006    630  1310
PCRB    8007
INIT    0100
RET     0126
FXSP    0128    720
GO      0130    730
LOOP    0141   1020
UT      0153    990
UTHX    015C    960
UTH1    0161   1410
BYTE    0177   1390
SHIA    A031    920  1010  1160
VARV    A032   1140  1400
LCTR    A033   1210  1380
MPOS    A034    970  1230  1270  1300
PIA1DA  8004
PIA1CA  8005
PIA1DB  8006
PIA1CB  8007
ACIA1S  8008
ACIA1D  8009
PIA2DA  8020
PIA2CA  8021
PIA2DB  8022
PIA2CB  8023
IOV     A000
NIO     A006
SP      A008
CKSM    A00A
BYTECT  A00B
XHI     A00C
XLO     A00D
TEMP    A00E
TW      A00F
MCONT   A011
XTEMP   A012
STACK   A042
PUSHX   A2F4
PULLX   A2F4
READX   A2F6
SHIFT   A2F8
SWAP    A2F9
SHRDLS  A2FA
SHRDRS  A2FB
STX     0002
```

```
ETX     0003
EOT     0004
BSP     0008
FSP     0009
NL      000A
LF      000B
CR      000D
RN      0011
PN      0012
RF      0013
PF      0014
TCV260  C000
OUTCH   C003
INCH    C006
OUT4HS  C009
OUT2HS  C00C
PSPACE  C00F
PDATA1  C012
INADR   C018
INHEX   C01B
BREAK   C01E
PUNOUT  C021
KEYDC   E14E
```

```
                    PROG2708 V1.2    78-10-26

S00600004844521B
S1150100CEFF34FF8004CEFF2CFF8006CE0A00FFA00C64
S1150112CE0000FFA00ECEA010FFA01A7CA014BDE084D4
S11501248D022008343434347EE1F7B6A000F6A001C7
S113013644564456454CA40F7A0318D19FEA0348F
S1130146BCA0042708FFA0027CA0312AEEB68004D6
S114015684BFB780043F8664B7A032B6A031B78004A2
S11301678A80B78004861OB7A033FEA002FFA034AC
S1140177FEA034A60008FFA034B78006863CB78005E5
S1130188CE0050BDE0E18634B780057AA0332EDF77
S10901987AA0322EC439E6
READY
```

PRINTERS

Att nedteckna siffror och bokstäver på papper kan ske på många olika sätt, och många olika tekniska lösningar har sett dagens ljus på ritborden. För och nackdelar har varierat med de olika typerna och jag skall här försöka att grovt göra en sammanställning av de vanligast förekommande maskinerna.

Om vi först tittar på sättet att skriva på pappret, kan en huvudgrupp genast urkristalliseras. De mekaniska printrarna.

Till de äldsta typerna av mekanisk skrivare hör den vanliga skrivmaskinen kompletterad med magneter som inderekt eller direkt påverkar typarmarna. Några konstruktioner med lösa ramar med magneter som placerades över själva tangentbordet kan väl betraktas som kuriosum i sammanhanget. Skrivhastigheten var dock låg och maskinerna förekommer väl numera endast i hobbydata.

Ett nytt steg i utvecklingen kom när man införde en vertikal trumma som kunde vridas runt sin axel och/eller höjas på centrumaxeln. Se fig 1. När den ställts till rätt position slog en hammare mot trumman. Mellan trumman och hammaren befinner sig papper och färgband. Texten blir tydlig och man kan genom att välja olika trummor få olika teckenuppsättningar. På den principen bygger den välbekanta TTY:n. Trots att antalet magneter är starkt nedbantat, innehåller Teletypen en sådan mängd skruvar och muttrar att man är böjd att utnämna den till värdens åttonde underverk. En konstruktion som står sig än idag, och väl funnit vägen till mången amatör. Nackdelarna fanns dock i form av mekaniskt slammer (100 dB) och låg skrivhastighet, 10 tecken/sek.

En vidare utveckling kom när trumman byttes ut mot en kula. Tekniken finns visserligen använd i vissa terminaler, men är nog vanligast i skrivmaskiner. Skillnaden mot trumman är att kulan måste styras runt två axlar.

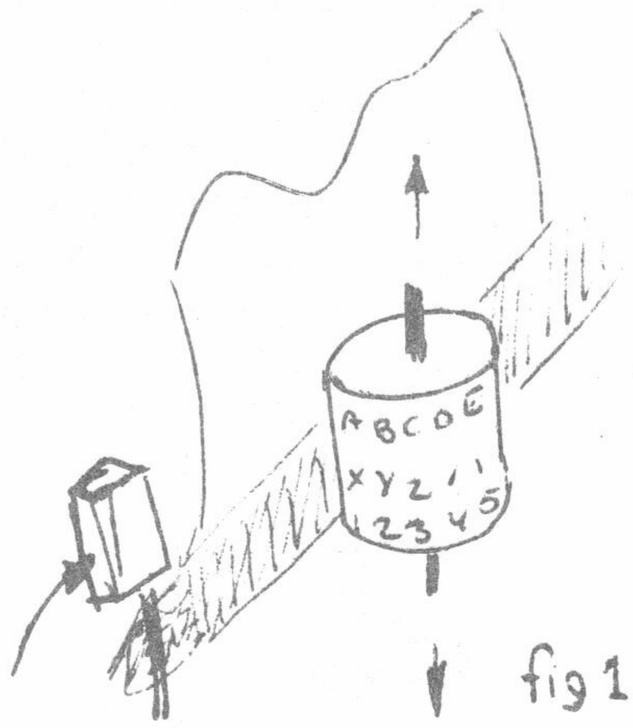


fig 1

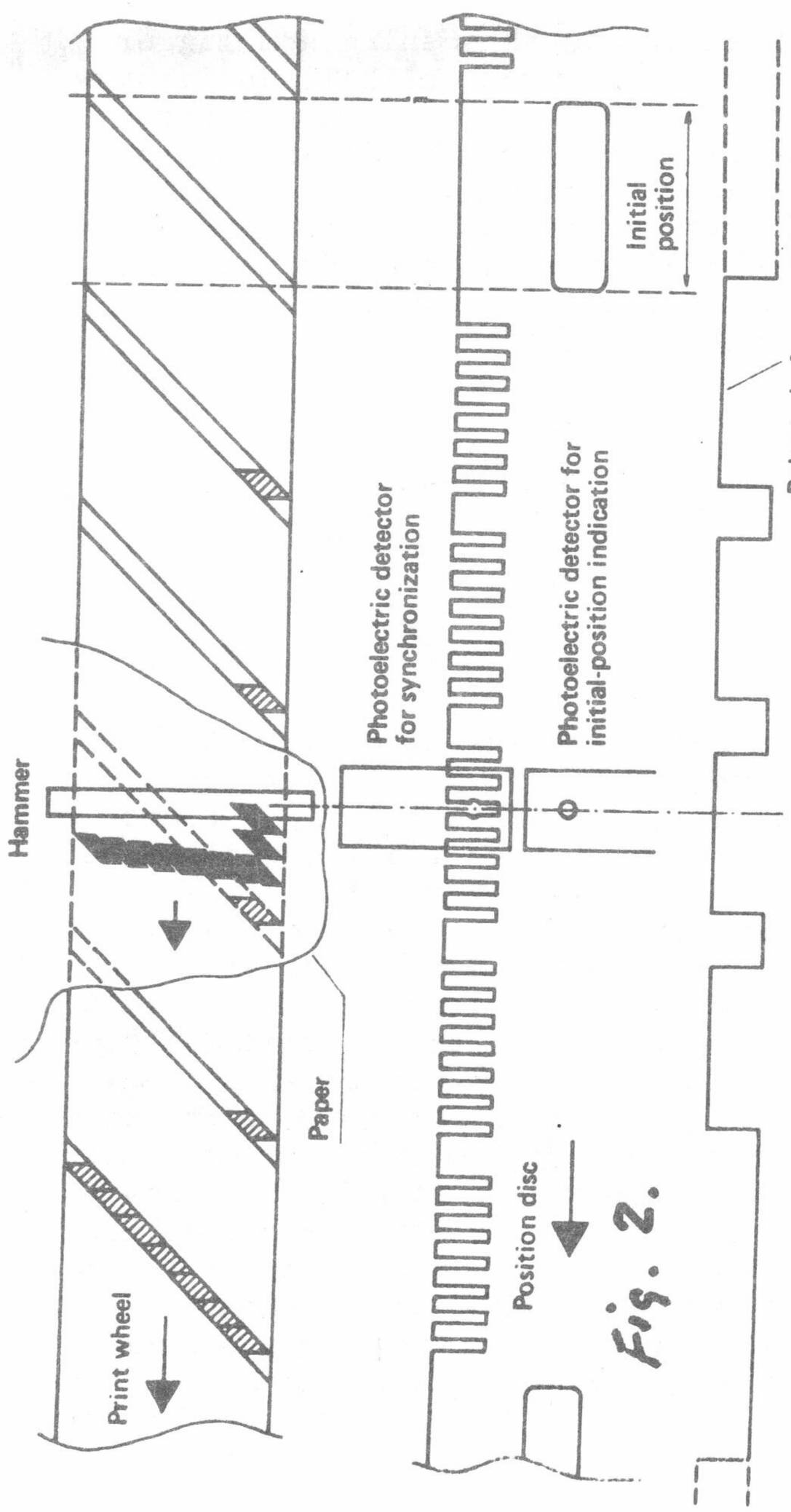


Fig. 2.

Ytterligare ett specialfall av trumma introducerades av Facit med printern 4555. Genom ett skrivhjul med olika långa diagonala linjer kan man konstruera de flesta tecken man önskar. Se fig 2. Man uppnår här en ganska hög skrivhastighet om cirka 30-60 tecken/sek. Nackdelen är att skriften blir något svårläst, närmast att beskriva som snedställd kantigt. Se fig 3.



fig 3

En annan modern lösning som ligger nära trumman är den så kallade Daisy - Chain-printern. Här har alla typer placerats på en roterande skiva med slitsar mot centrum. Se fig 4. Lätt att byta teckenrepotoar och snygga tecken med lite mekanik gör att denna typ blir allt vanligare. Jmf dymo-apparatens funktion.

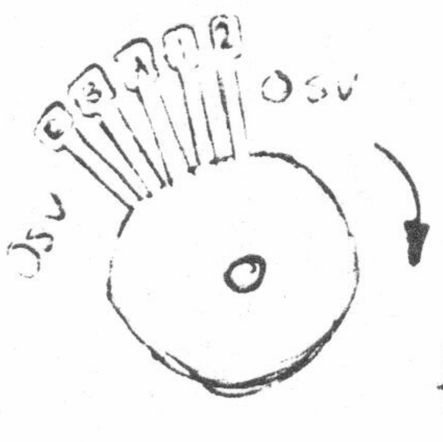


fig 4

En betydligt snabbare kusin till den vertikala trumman är den horisontella. Som man ser i fig 5 finns alla tecken i lika många uppsättningar som det finns kolumner att skriva i. Genom att ta in en hel teckenrad i ett minne och studera detta, kan man genom att rotera trumman och när rätt tecken passerar låta en hammare slå, på endast ett varv skriva alla positioner på en rad. Observera dock att tecknen inte skrivs från vänster till höger som normalt, utan helt slumpmässigt beroende på textens innehåll. En nackdel med denna printer är att om inte den justeras mycket noggrant, kommer tecknen att vagga upp och ner på raden. (ExemPelvis så här.)

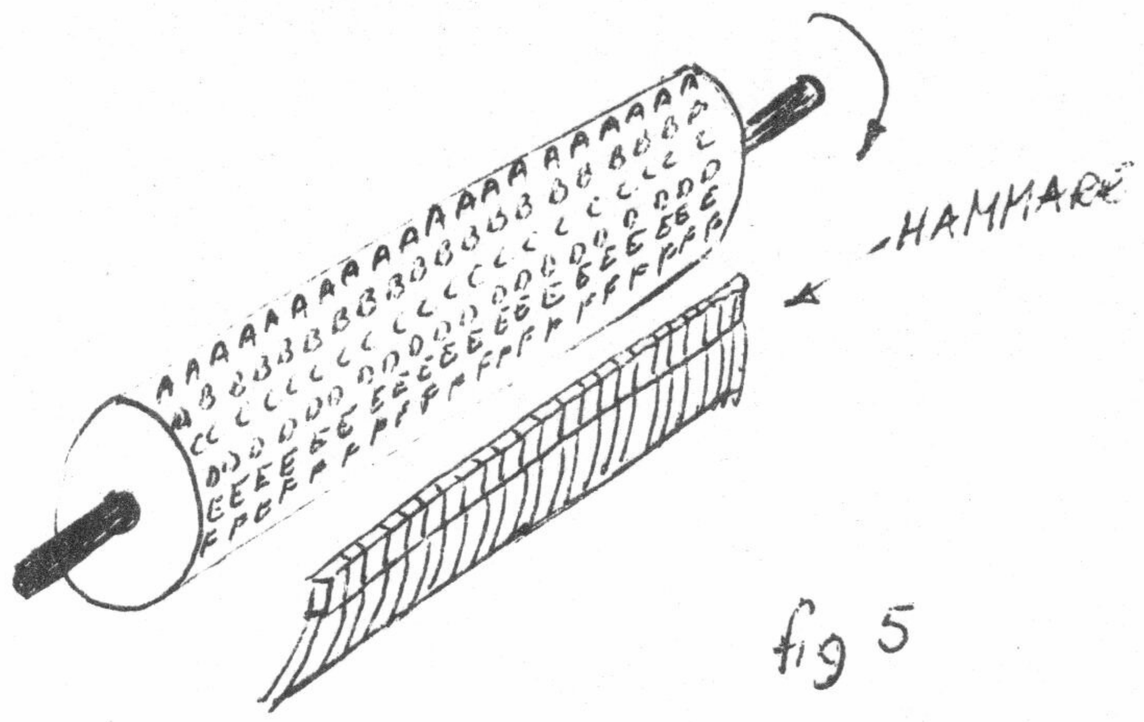


fig 5

Den horisontella trumskrivaren är mycket vanlig på proffesionella anläggningar där man önskar hög skrivhastighet.

En annan vanlig höghastighetsprinter är kedjeskrivaren. Här har man löst det hela så att tecknen finns på ett ändlöst roterande band. Det är alltså en kedja eller ett gummiband som går mellan två roterande valsar ställda på högkant. Se fig. 6. På grund av att kedjan blir relativt lång finns alla tecken som regel i två eller flera uppsättningar och man tar det första som passerar förbi skrivpositionen. Även här inser man att om hela raden finns känd kan man skriva tecknen slumpvis över raden och när en teckenuppsättning passerat vara klar.
I motsats till den horisontella trumskrivaren kommer det här att bli problem med rakheten i kolumnen om timingfel finns. (Exempel vis så här)

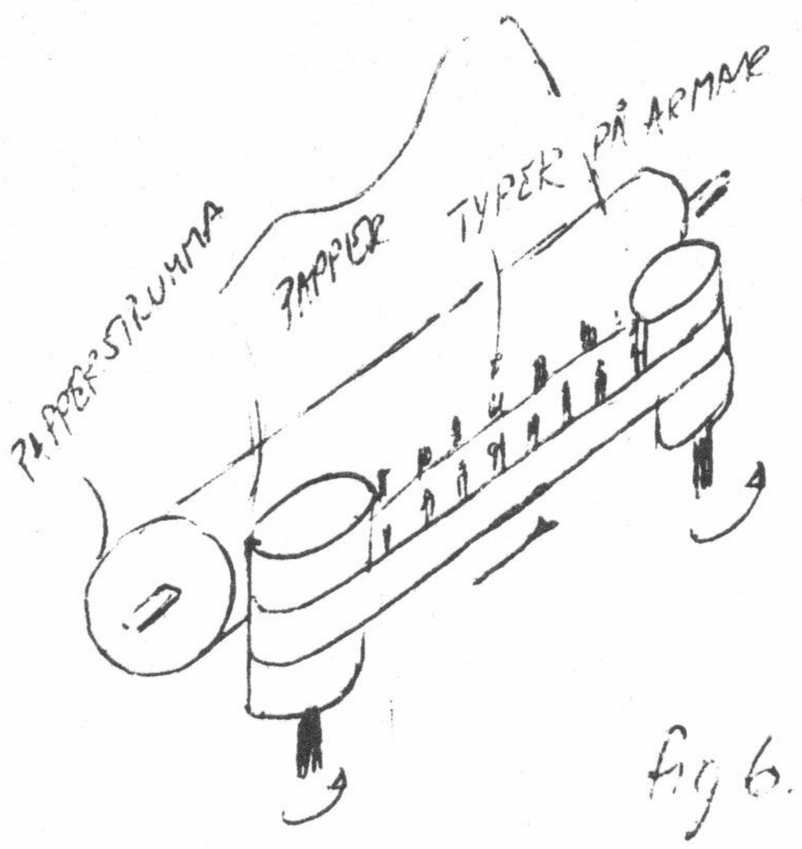


fig 6.

Rent tekniskt är dock likheterna stora mellan de hittils uppräknade skrivarna. Bortsett från specialfallet 4555 från Facit som skriver med diagonala streck, används hela typer gjutna i bly eller plast, alltså samma som i en vanlig skrivmaskin.

En ganska ny ideé är de så kallade matrisprintrarna. De avviker alltså helt från de tidigare omtalade genom att man bygger upp tecknet med ett antal punkter. Fördelarna med detta är stora, därför att man kan med programvara generera vilka tecken man önskar inom den matris som finns tillgänglig. Genom att öka antalet punkter i matrisen kan man skriva större bokstäver och få en bättre upplösning. Normalt använder man de vanliga standardmåtten om 5+7 eller 7+9, men i princip kan alla kombinationer finnas. Den stora fördelen finns dock i de grafiska möjligheterna. Konstigt nog utnyttjar man inte detta genomgående, utan pappersmatningen sker stötvis, i många skrivare. Man kan så att säga inte skriva mellan raderna.

De första matrisprintrarna arbetade med böjliga nålar som strålade samman mot en vertikal rad. Genom att dragmagneterna var ganska stora och den yta som nålarna skullę träffa var liten fick magneterna palaceras i en halvmåne. Detta gav dock upphov till problem med lagring av nålen och slitaget blev relativt stort med glapp som följd. En bättre lösning var då den av Facit patenterade matrisprintern med svängande armar. Se fig 7. Genom den styva armen med liten massa var det möjligt att nå skrivhastigheter på upp till och över 200 tecken/sek. Dessutom slapp man problem med lagringen.

Fortfarande är dock alla de uppräknade typerna av skrivare konstruerade med mekaniska delar. För att komma runt detta började man utnyttja ett tryckhuvud bestående av mycket små motstånd. Genom att leda en ström genom ett motstånd brändes en punkt in i pappret. Trots att dessa motstånd var mycket små lagrade de dock energi under en tid och skrivhastigheten måste alltså begränsas. En annan närliggande typ var de elektrostatiska. Här ledèr man en elektrisk ström in i pappret som alltså måste vara ett specialpapper. Den stora nackdelen med dessa båda metoder är dock att man inte kan få kopior direkt, utan får i såfall gå omvägen över en kopieringsmaskin. I vissa fall kan dessutom pappret åldras efter en tid. Fördelarna med att slippa mekanik kan dock mer än väl uppväga dessa nackdelar.

En helt annan typ av skrivhuvud, om det nu kan kallas så, använder bläckskrivaren. Den arbetar med en mycket tunn stråle av bläck som focuseras och avlänkas mot pappret. Likheten med ett vanligt katodstrålerör är slående. (Katodstrålerör = bildrör) Hög skrivhastighet är kännetecknade, men problem med torkat bläck finns. När man inte önskar någon färgning låter man nämligen strålen träffa en platta och sedan rinna ner från denna. Att sedan hålla bläcket i omlopp ger vissa problem. På en del veckotidningar som man premumererar på kan eventuellt adressen vara gjord med bläckskrivare.

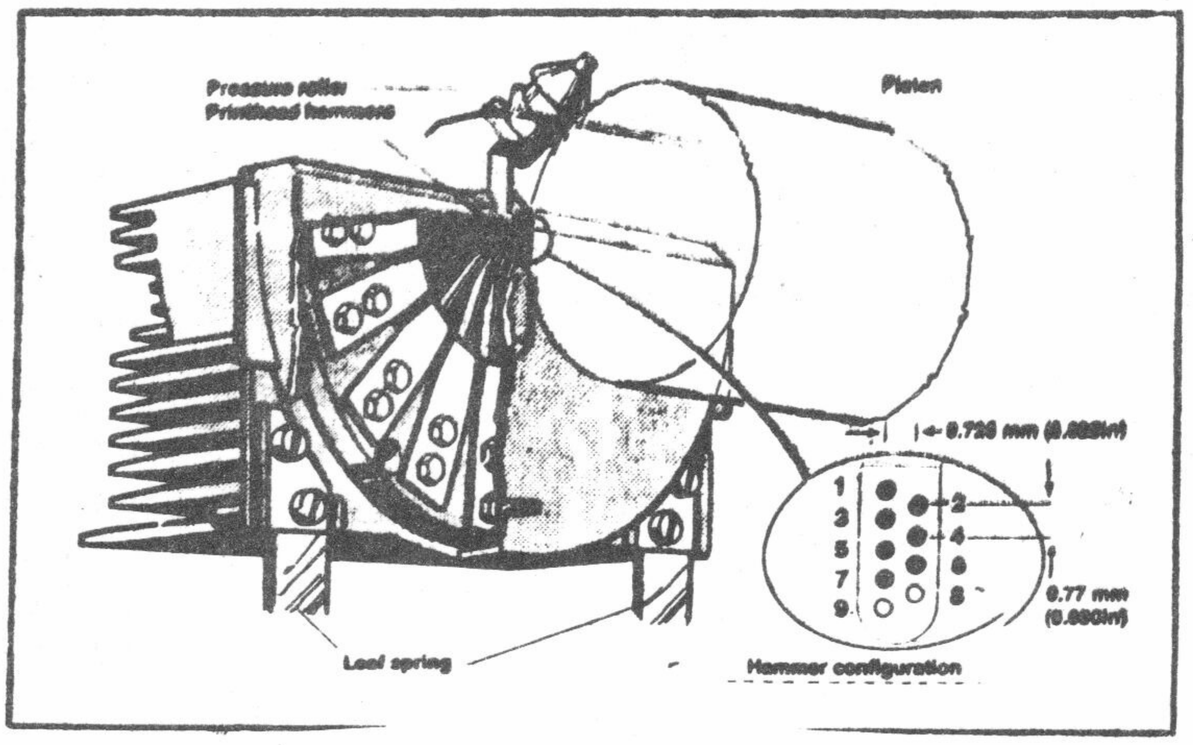


Fig 7

Till sist den verklige tuffingen. En laserstråle avlänkas mot ett papper. Man bränner in texten och kan här nå svindlande höga hastigheter, de har nämts upp till 1000000/ $_t$ sek. Frågan är väl i såfall bara hur man skall hinna mata fram pappret.

Detta leder in på en annan viktig del i samband med skrivare. Hur skall jag få pappret att matas fram. En väg är att man perforerar pappret i kanten med styrhål. Genom att låta piggar hugga tag i dessa hål kan en mycket säker frammatning ske. Nackdelen är att det inte ser särskilt trevligt ut, samt att man inte kan stoppa i vilka papper man önskar,för att få en utskrift på dessa. Delvis kan man komma runt detta genom att använda ett speciellt bärpapper som mina blanketter är limmade på. Det andra alternativet är då friktionsmatning med en gummivals. Här kan utskrift ske på alla papper, men så småningom styr det som regel snett och den som önskat långa listor har säkert sagt något fult ibland. Ett svårt val då de flesta skrivare inte medger båda möjligheterna.

Slutligen kan man konstatera att utvecklingen går mot mindre och mindre mekaniska delar och man försöker att ersätta dessa med elektronik på alla nivåer upp till mikrodatorer. Faktum är att den printer jag själv använder innehåller fler 40-pinnars kapslar än den kära computern brevid.

NO.

SPELHÅLAN

Bidraget till Spelhålan kommer denna gång ursprungligen från tidningen BYTE:s juli nummer. Det är en verklig pärla. De flesta spelprogram man ser i Basic är tämligen enkla, ja rent av heldumma. Inte värda mödan av inknappningen. Så inte denna gång.

När vi knappade in programmet passade vi på att översätta det till svenska. Sedan var det ett par dialektala Basic skillnader som fick fixas. Nu är den så generell att de flesta Basic tolkar bör vara nöjda.

Alla kommentarer är fortfarande kvar. Det gör att programmet är lite onödigt svulstigt och drar ganska mycket minne. Med SWTPc version 1. behövs 16k arbetsminne. För TSC disk Basic skulle det betyda att närmare 30k måste finnas. Eftersom kommentarer och blanka tecken inte behövs när man kör ett program har jag gjort en speciell FLEX utility (COMPRESS.CMD) som tar bort onödiga kommentarer och mellanslag. Den kan faktiskt ta bort 5k från QUEST. Det programmet kommer i nästa MPU-lare.

Åter till spelprogrammet. I QUEST skall man söka sig fram i en grotta och leta rätt på en skatt. Man kan ge sex kommandon för att ta sig fram med. De är:

E (EAST) för öster
W (WEST) för väster
N (NORTH) för norr
S (SOUTH) för söder
U (UP) för upp
D (DOWN) för nedåt

Vidare kan man ge P för att erhålla den poäng man då har.

Orsaken till att den engelska nomeklaturen bibehållits för de sex kommandona är att man önskar kunna ge endast en bokstav i kommandot. Norr och Ned blir då inte unika.

Fortsättning sid 46

KCS INTERFACE

Standard/höghastighets kassettinterface

Här presenteras ett kassettinterface som är omkopplingsbart mellan 300 och 1200 baud. Dessutom kostar det inte alltför mycket att bygga. Kansas City Standard (KCS) har visat sig vara en bra och pålitlig överföringsstandard. Men den är för långsam för att fylla alla behov som en hobbydatorutövare har. De kretsar som presenterats tidigare har antingen varit långsamma 300 baud KCS eller höghastighetskretsar utan gemensam standard.

I båda dessa fall har kretsarna varit inte bara mer komplicerade än nödvändigt, utan också för svåra att använda. Kretslösningen i denna artikel är inte lösningen på alla problem men den fyller ett behov av en en-krets KCS och höghastighets-I/O som är enkel att bygga och använda för vilken amatör som helst.

Kretslösningen

Detta kassettinterface ha följande data: Det kostar lite och använder bara vanliga 7400-serie IC-kretsar som är enkla att få tag i. Det arbetar med både 300 och 1200 baud KCS genom att bara slå om en omkopplare. Inga inställningar är nödvändiga att göra när man ändrar datahastigheten.

Kretsen använder bara +5 volt för strömförsörjningen. Den har en enkel tre-trådars anslutning till datorn och är enkel att använda efter det att den installerats.

Figur 1 visar hur enkelt det är att ansluta kretsen till ett datorsystem. De inringade siffrorna i diagrammet motsvarar anslutningarna i tapeinterfacets kretsschema.

AMI Proto EVK 6800 mikroprocessorbaserade utvecklingskort, tillsammans med Davis Labs 16k RAM-kort och en helt hembyggd TV-terminal (TVT) är grunden i författarens system. TVT-lådan innehåller tangentbord, TVT logik-kort och detta kassett[illegible]-interface[illegible]-kort. Data in- och data ut-ledningarna är RS-232, alla andra är TTL. Data och X16 klocka för kassett-interfacet fås i TTL-nivåer vid TVT-ingångarna.

Data tillbaka till datorn matas in i en två-till-ett-väljare som tillåter automatisk datorkontroll via RTS-ledningen eller manuell kontroll för bara tape eller bara tangentbord. Därför är tape-interfacet alltid aktivt men beroende av [illegible] andra kontroller

Mjukvarukontrollen för denna krets är (åtminstone för 6800-system) en del av det vanliga monitorprogrammet. Alla 6800-system innehåller Punch- och Load-subrutiner för S-formatet, som utvecklades av Motorola som en minnesladdare/pappersrems-puncher och användes i MIKBUG.

För att spela in en tape använder man Punch-kommandot. Slå sedan på bandspelaren (inspelning) och slå en CR (Return). Data visas på skärmen samtidigt som inspelning sker. För att ladda ett program är det bara att spela tapen innan data börjar, slå ett L och en CR. Omkopplaren S2 i figur 1 måste också vara placerad i rätt läge.

Forts. nästa sida

Hittills har alla interfaceenheter som byggts fungerat utan att ge några som helst fel vid överföringen av data, varken vid 300 eller 1200 baud.

Interfacet kan delas upp i två delar, sändaren och mottagaren. Här nedan beskrivs hur vardera delen fungerar.

Sändaren

Sändaren består av IC-kretsarna U1, U2, U3a, U4 och U5 liksom filtret R2-3-4-5 och C1-2-3. Baud-Select-ingången kontrollerar två-till-ett-väljaren (U5), vilken väljer antingen X16 klockan för 300 baud (4800Hz) med ingången låg eller en uppskalad (till 4800 Hz) X16 klocka för 1200 baud (19,2 kHz) med ingången hög.

I båda fallen är utgången av väljaren (punkt A) en 4800 Hz fyrkantvåg. U2 delar denna med två (punkt B) och med 4 (punkt C) för att få de 2400 Hz och 1200 Hz som används för ljudtonerna ETT respektive NOLL.

Data (punkt D) klockas tillsammans med de erhållna 1200 Hz i U3a. Data-ettor leder de 2400 Hz genom output-multiplexern (punkt F) och data-nollor styr de 1200 Hz (punkt E). Dessa kombineras av U4b (punkt G) och bildar data-outputen.

Data-ettor vid 1200 baud är två hela vågor med 2400 Hz och nollor är en hel våg med 1200 Hz. Vid 300 baud är vågorna fyra gånger så många för etta respektive nolla.

Användandet av datorns X16 klocka ger minstta möjliga fel mellan data och de erhållna tonerna, vilket ger den bästa signalen ut. Filter-komponenterna är valda för att ge så lika amplitud som möjligt mellan 2400 Hz-och 1200 Hz-tonerna vid H.

Mottagaren

Mottagaren består av ingångsbegränsaren (C4, R6, CR1-2), U6, U7, U8 och U3b. Den starka signalen från bandspelaren är växelströmskopplad via C4/R6 och begränsas till 1,4 V p-p av CR1/CR2. U6 omvandlar den begränsade signalen till fyrkantvåg som rundas av C6. Schmitt-triggern U7a ger en skarpare fyrkantvåg för att trigga U8 (punkt J i figur 5).

Nivån för den blandade 1200/2400 Hz datasignalen är alltid riktig för en etta eller nolla. Detta händer två gånger för en etta och en gång för en nolla (1200 baud data). Vi tar bort den blandade 1200/2400 Hz signalen och får riktiga data vid U3b Q-utgång. Data tas från U3b Q och buffras och inverteras av U7c och går till baka till datorn (punkt H). U7d tänder en LED för data-ettor.

Mottagaren innehåller den enda punkten som måste justeras i hela kretsen - potentiometern R8 som styr känsligheten när U8 skall trigga. När kretsen är installerad i ditt system ansluter du interfaceutgången till sin egen ingång.

När ingen signal kommer från datorn skall data hela tiden vara ettor vilket ger 2400 Hz vid I/O-punkten. U8 triggas vid varje vågkant och pulsen ut vid U8 pinne 4 justeras till 312 mikrosekunder med R8. Därefter behövs inga inställningar göras.

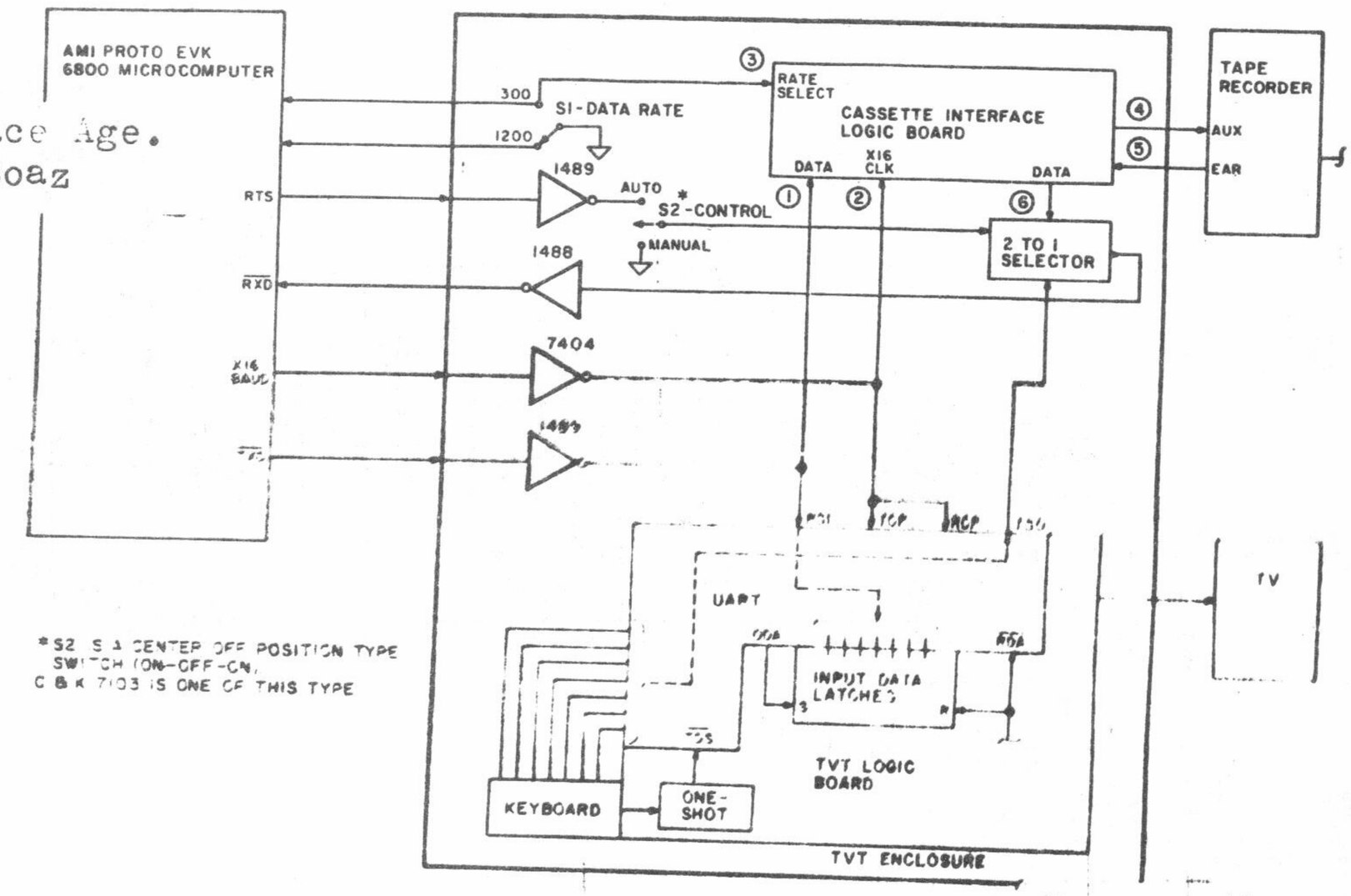
Använd en 0,1 mikrofarad kondensator för varje två IC, mellan +5 V och jord. Strömförsörjningen till U6 bör kopplas till jord via C5 så nära ICn som möjligt. En kondensator på minst 15 mikrofarad (tantal) bör användas där +5 V och jord går in på kortet.

Komponentlista

Motståndsvärdena är i ohm, kondensatorvärdena i mikrofarad.

U1, U2, U3	7474
U4, U5	7400
U6	LM311
U7	74132
U8	74123
CR1, CR2	1xN914
DS1	LED
R1, R3, R7	1k
R2	470
R4	100
R5	510
R6	10k
R8	50k (pot)
R9	270
C1, C2, C5, C9-13	0,1
C3	1,0
C4	0,0022
C6	0,01
C7 (Mylar)	0,033
C8 (tantal)	15,0

Ur Interface Age.
Wilson R Boaz



*S2 IS A CENTER OFF POSITION TYPE SWITCH (ON-OFF-ON). C B K 7103 IS ONE OF THIS TYPE

Forts. nästa sida

Fig. 1. Overall diagram of how system is interconnected.

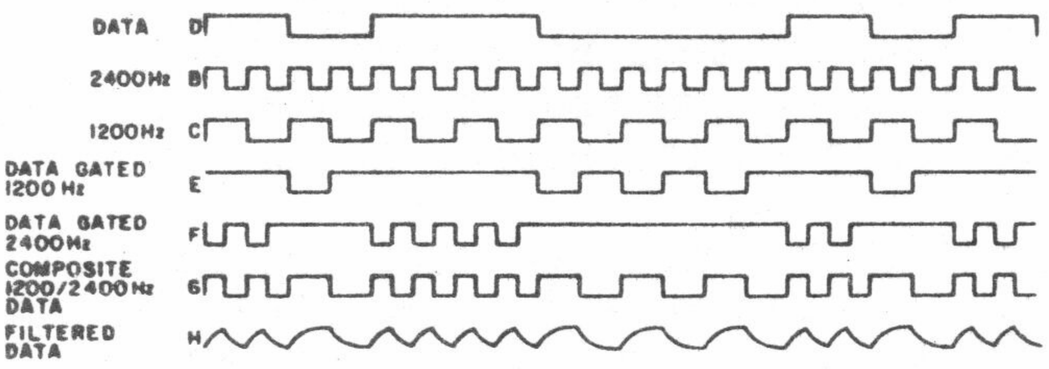


Fig. 2. Transmitter circuit timing diagram for key points and their relationships. All waveforms are for 1200 baud.

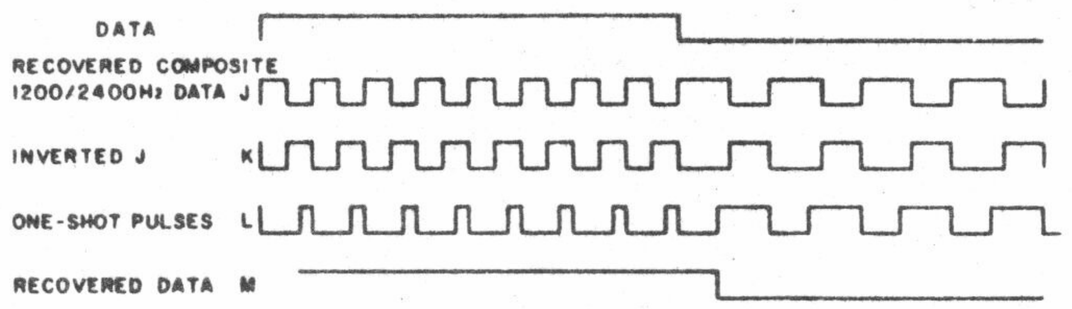


Fig. 4. Receiver waveforms for 300 baud data.

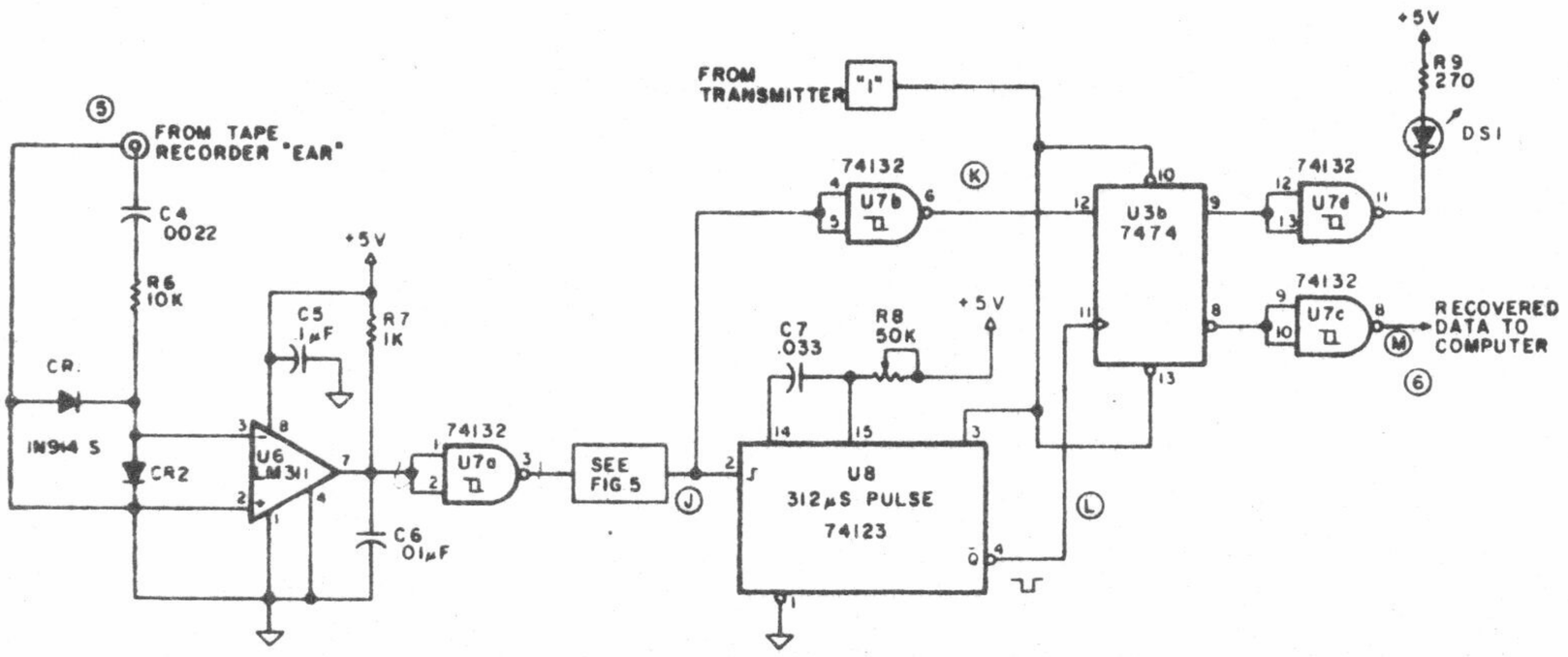


Fig. 5. Receiver recovers digital data from 1200/2400 Hz audio tones.

Fig. 7. Circuitry to be added for signal inversion.

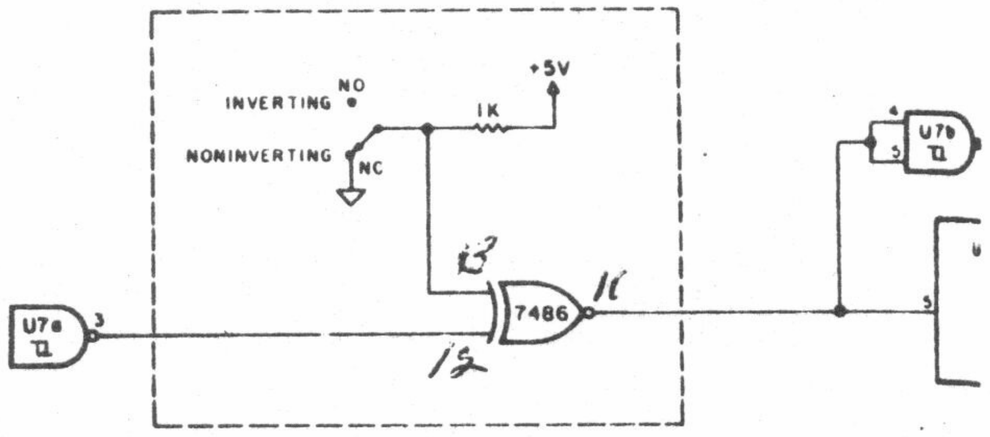


Fig. 6. Receiver waveforms for 1200 baud.

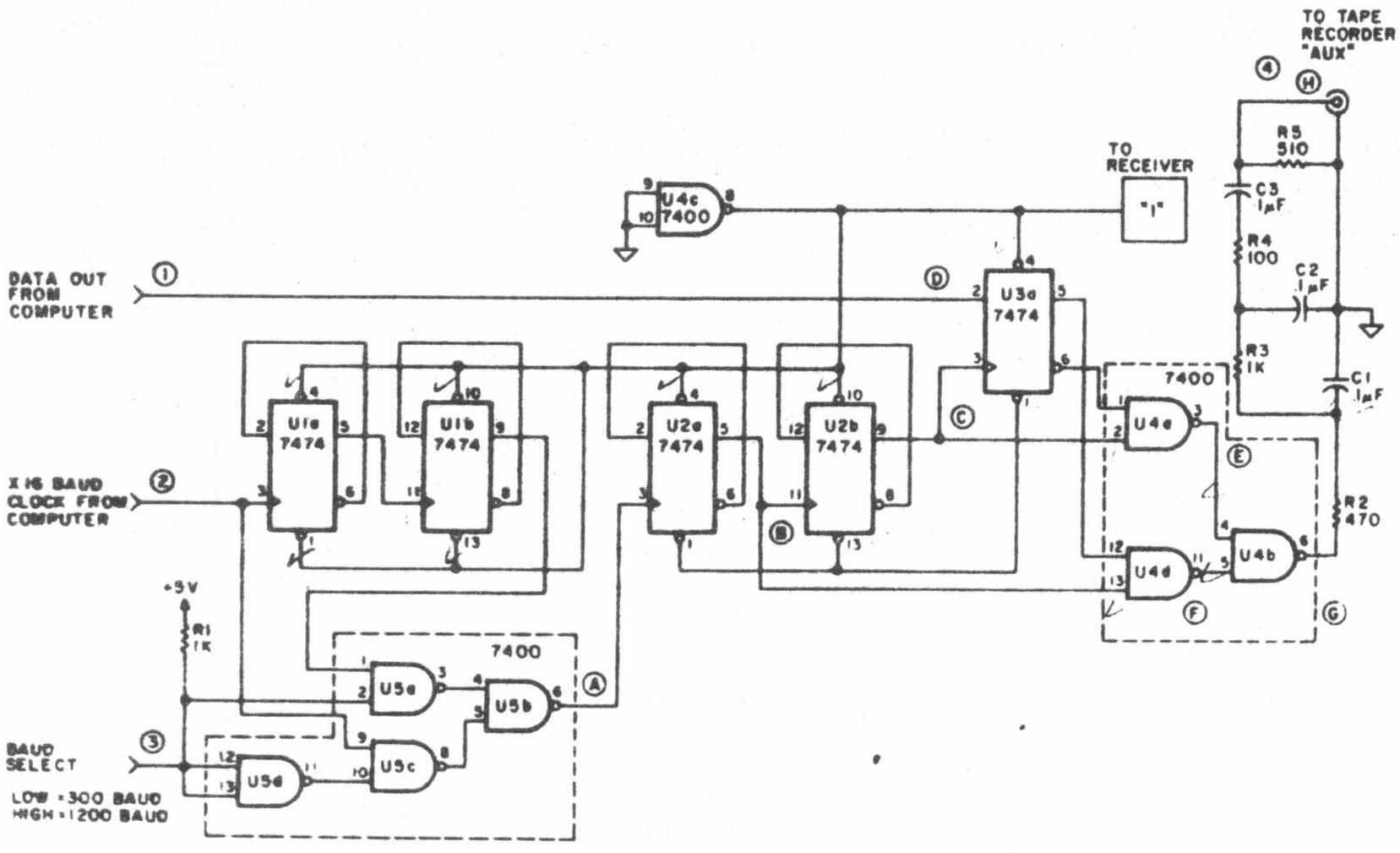
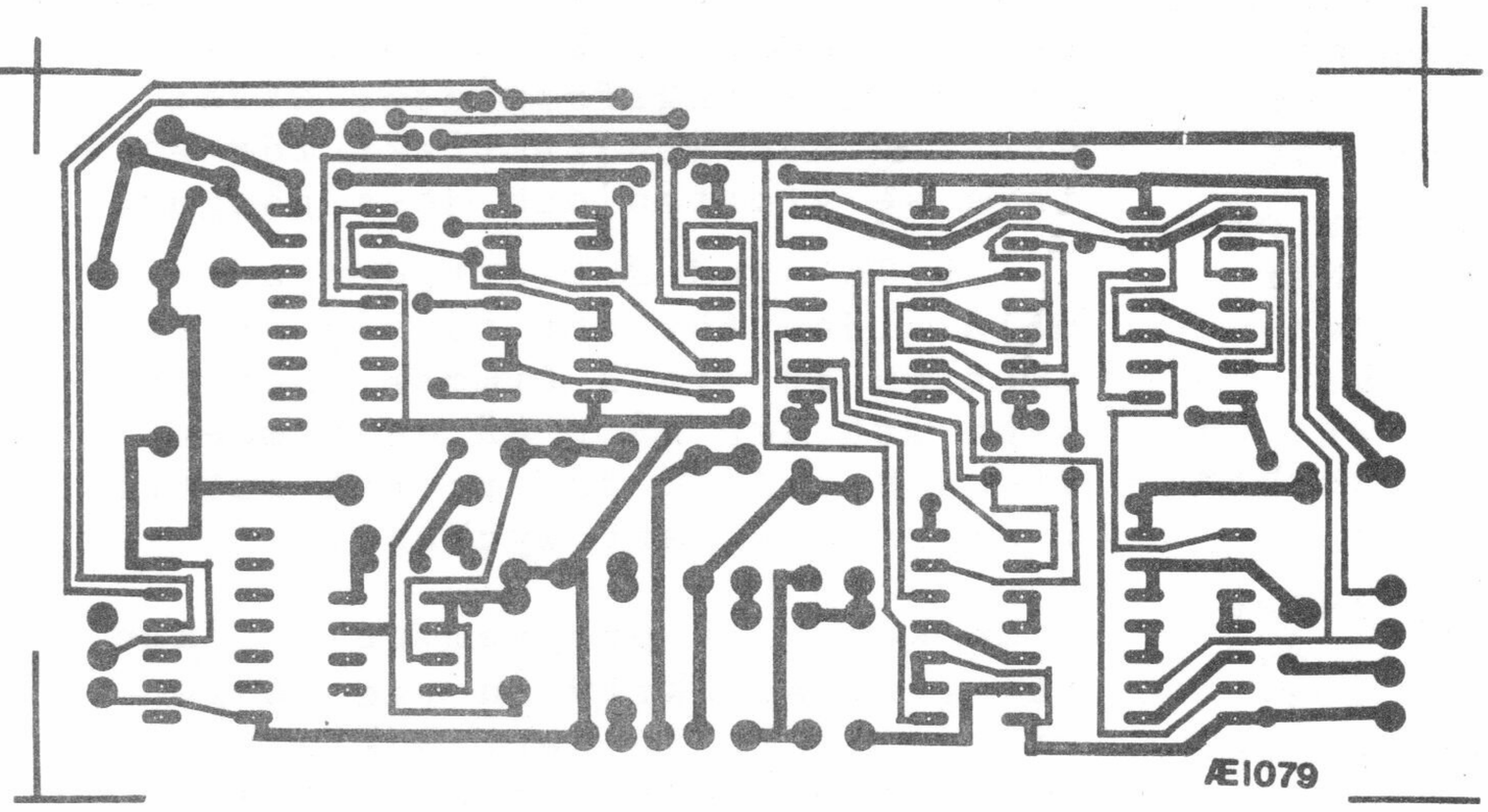
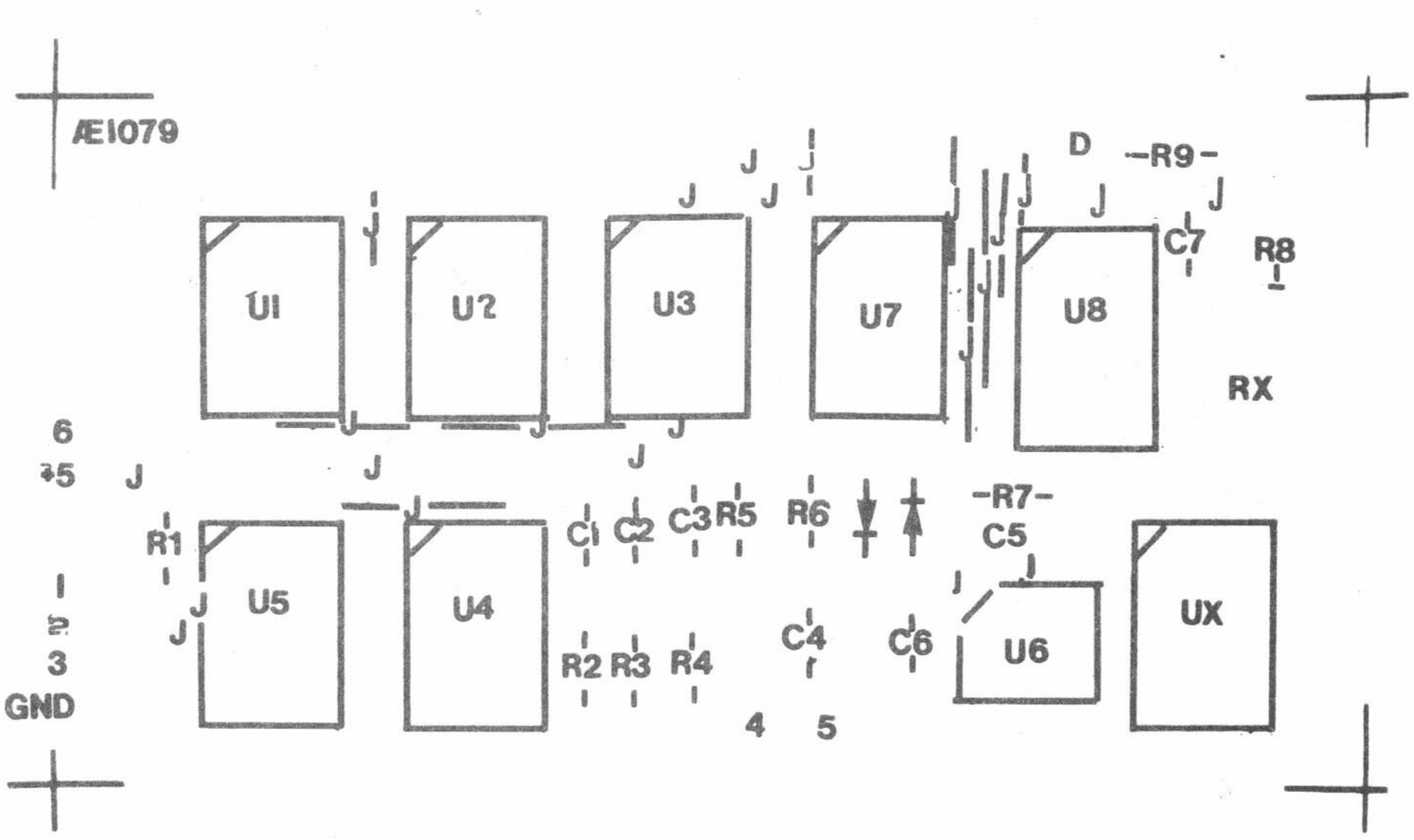


Fig. 3. Transmitter converts digital data to 1200/2400 Hz audio signals.



SKALA 1:1



ANNONS

2 st 8 K Ram -kort utan minnen (2102) tel 0758 -39951
Gustav Månstrand Box 75 17522 JÄRFÄLLA 1

Med endast dessa direktiv tar man sig fram genom tunnlar och schakt. Upp och ned i olika nivåer. Efter varje kommando får man en utskrift som berättar om den plats man kommit till. När man hittat skatten gäller det att ta sig ut med den. Då kan man plötsligt råka ut för att en tunnel som förut varit passerbar nu är för trång.

Sedan har man kanske trasslat sig fram nära öppningen. Då dyker en pirat upp och snyter åt sig skatten. Bara att gå in och leta igen.

Spelet pågår alltså i tre dimensioner. Det ställer stora krav på orienteringsförmågan. Efter ett tag kan man faktiskt för sitt inre se grottan och dess gångar.

I BYTE stod det att den normala tiden för att ta sig ut med skatten är c:a 2 timmar. Det har hänt att det gått på en timme. Vissa har dock frustrerat suttit i dagar utan att lyckas!!!

En kopia av programmet kan fås gratis. Skriv en rad. Ange vilken Basic du har. Enklast är det för oss att kopiera till en 5" FLEX diskett. Annars får det bli 300 b KCS kassett. Pappersstape är vi inte direkt överlyckliga att framställa.

TB1

```
1 REM QEST AV ROGER LHAFFEE
2 REM INSPIRERAD AV WILL CROWTHER"S 'ADVENTURE'
3 REM INKNAPPAD AV TBL & GRU (SVETT!!)
13 REM VARIABELBESKRIVNING
14 REM N  NODE ( GROTT) NUMMER
15 REM MO DRAG RÄKNARE
16 REM T  NUVARADE LÄGE FÖR SKATTEN ( -1 FÖR BÄRANDE )
17 REM T1,T2 FÖRSTA OCH ANDRA GÖMSTÄLLEN ( NOD NUMMER )
18 REM M6  SPARAR DRAGNUMRET NÄR HAN SADE NEJ, HAN VILLE
19 REM INTE TA SKATTEN MED SIG.
20 REM M   KARTA ÖVER MELLANFÖRBINDELSER
21 REM M(I,J) ÄR NÄSTA NOD FRÅN NOD J NÄR DU GÅR
22 REM N,E,U,D,W,S FÖR I = 1,2,3,4,5,6
23 REM
24 REM -  STUDS -
25 REM ATT GÅ TILL (NOD-2 ) BETYDER ATT STUDSA TILLBAKA TILL
26 REM DEN NOD DU KOM FRÅN.
27 REM
28 REM TVÅNGS- OCH ELLER SLUMPDRAG
29 REM M(1,N)=-2 BETYDER TVÅNGSDRAG SÅ FORT DU NÄR
30 REM NODE N. I SÅ FALL M(2,N) AV DE GÅNGER DU GÅR TILL
31 REM NODE M(3,N). OM DU INTE GÅR DIT, M(4,N)
32 REM AV GÅNGERNA DU GÅR TILL NOD (5,N), OCH RESTEN AV GÅNGERNA
33 REM GÅR DU TILL NOD 6,N)
34 REM
35 REM NODERNA > 100
36 REM NODE N + 100 BETYDER NOD N OM DU INTE HAR SKATTEN
37 REM OCH NODE N+1 OM DU HAR DEN.
38 REM NODE N + 200 BETYDER NODE N+1 OM DU HAR SKATTEN ANDRA
39 REM GÅNGEN, OCH NOD N ANNARS.
40 REM NODE N+ 500 BETYDER SNURRA I EN DELAY LOOP OCH SEDAN
41 REM GÅ TILL NOD  N. DETTA ANVÄNDS FÖR ATT FALLA NED I UTGÅNGS
42 REM SCAKTET, OCH DU KAN ÖNSKA ATT JUSTERA FÖRDRÖJNINGSTIDEN.
43 REM (RAD 6250).
```

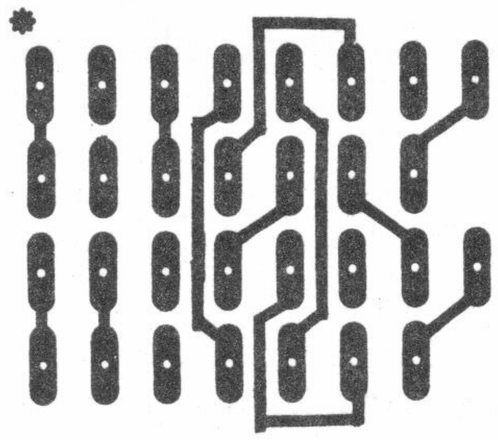
```
44 REM
45 REM Q$   INPUT STRÄNG
46 REM A$   TECKEN ATT JÄMFöRA I UTPUT RUTINEN.
47 REM A2   ANTAL TECKEN I A$
48 REM A1   OUTPUT FRÅN INPUT RUTINEN.U
49 REM P    PIRAT FLAGGA  1 OM PIRATEN HAR TAGIT DIG, 0 ANNARS.
50 REM P1   RÄKNARE FöR PIRATRUTINEN
51 REM N9   SPARAR GAMMAL NOD I DRAG RUTINEN, FöR STUDS
52 REM N8   SPARAR NOD VI STUDSADE FRÅN I DRAG RUTIN ,
53 REM FöR PRINT FLAGGA
54 REM NO   SPARAR GAMMAL NOD I DRAG RUTIN. FöR ÅTERVÄNDSGATA
55 REM AO   SPARAR GAMMAL RIKTNING I DRAGRUTINEN
56 REM D    DEBUG FLAGGA ( EJ NOLL FöR PRINTNING)
57 REM I,J DIVERSE RÄKNARE
58 REM W    FÄRD RÄKNARE. ANVÄNDS I POÄNGSÄTTNING. W(I)=1 OM HAN
59 REM VARIT TILL NOD I, 0 ANNARS.
60 REM S    POÄNG.
61 REM M9 MAXIMALA ANTAL NODER
80  D=0
100 REM ----------------------------------------------
105 PRINT CHR$(5)
110 REM GE DEM NÅGOT ATT FUNDERA -Å MEDANS JAG SÄTTER UPP DATABASEN.
120 PRINT "                QUEST/V1"
130 PRINT
140 PRINT "DU PROMENERADE GENOM"
150 PRINT "SKOGEN OCH DU STöTTE PÅ INGÅNGEN"
160 PRINT "TILL EN GROTTA, TÄCKT MED BUSKAR."
170 PRINT
180 PRINT "FOLK SÄGER ATT FöR MÅNGA ÅR SEDAN"
190 PRINT "GöMDE EN PIRAT SIN SKATT I DESSA SKOGAR"
200 PRINT "MEN INGEN HAR NÅGONSIN FUNNIT DEN."
210 PRINT "SÅVITT JAG VET, KAN DEN ÄNNU VARA DÄR"
400 READ M9,T1,T2
500 DIM W(42),M(6,42)
510 REM LÄS KARTAN TILL M ARRAYEN
520 FOR I = 1 TO M9
530 READ N
540 IF I=N THEN 570
550 PRINT "DATABAS PROBLEM ",I,N
560 STOP
570 FOR J = 1 TO 6
580 READ M(J,I)
590 NEXT J
600 NEXT I
900 PRINT
905 PRINT "NÄR DU SVARAR PÅ EN FRÅGA SÅ TITTAR JAG"
906 PRINT "BARA PÅ DET  FÖRSTA TECKNET. DU KAN DOCK"
907 PRINT "GE HELA ORDET OM DU VILL."
920 GOSUB 7500
1000 REM ---------------------------------------------
1010  N=5
1020  M0=0
1030  M6=0
1040  T=T1
1050  P=0
1060  P1=0
1070 FOR J=1 TO M9
1080  W(J)=0
```

Fortsättning sid 59

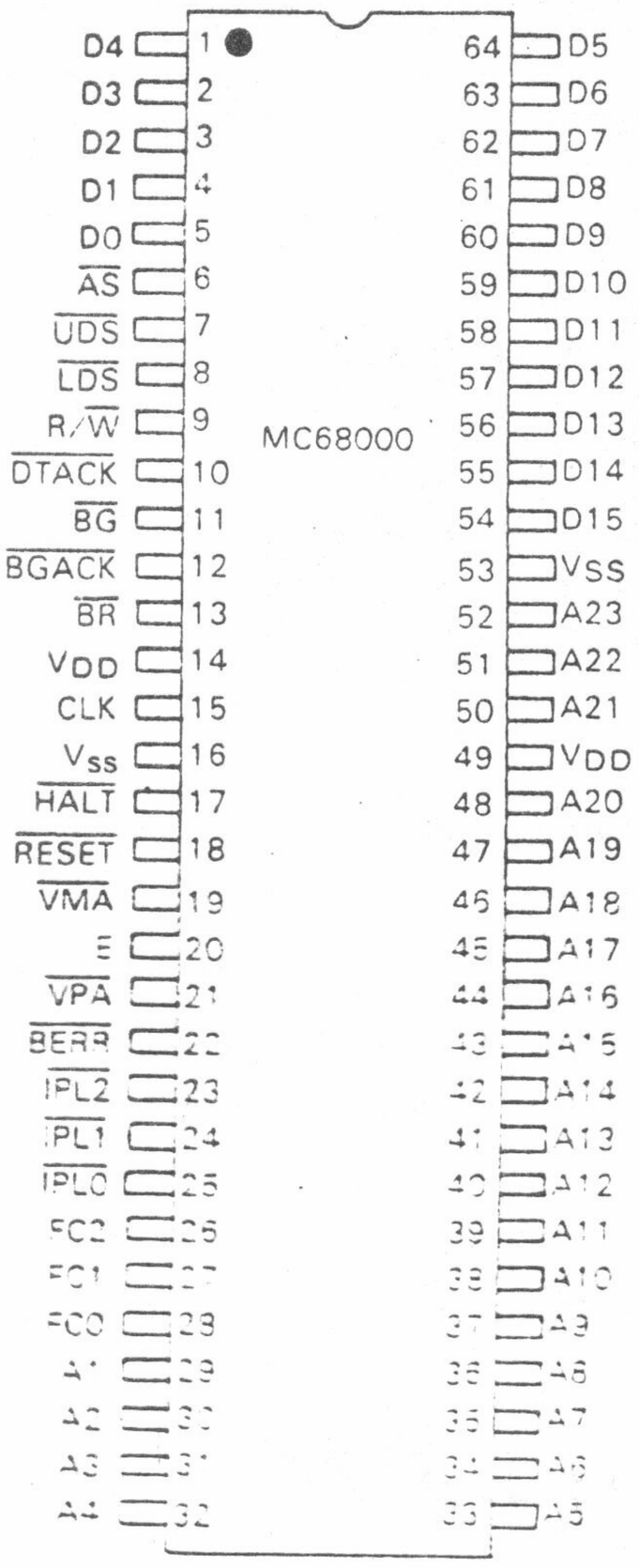
" MACKA "

Tillgången på databuffer 8T26 och 8T28 är svår.Här finnes en lösning.
Gör en "Macka"ersätt 74LS242 eller 74LS243,denna är brukbar i fall inte skilda in och utgångar användes.Kretslösning i skala 2:1 enl figur.

A.E.



8T28		LS243	8T28		LS243
1	-	1	9	-	8
nc 2		nc2	10	-	6
3	-	3	nc11		
4	-	11	12	-	9
5	-	10	13	-	5
6	-	4	nc14		
nc7			15	-	13
8	-	7	16	-	14



SEPTEMBER 1979 . . .

MOTOROLA HAS

INTRODUCED :

– MC 68000

– EXORset

And more to come . . .

MOTOROLA Semiconductors • Distributor product marketing • Geneva

68001-N SENASTE NYTT !

Ryckten säger att MOTOROLA arbetar på en ny NORSK version av 68000 som inte är så komplicerad.Norrmännen vill bara ha 1 pinne. Helst märkt NC.

H.Degerman

D.S.
Här har Ni senaste 'BUGGEN' jämför den med Er.Samma utrymme 1 K byte.
Den är gjort av PD68 medlemmen Herve Tireford i Schweits.Vilka Kombinationer PD68 +6809..

EXORbug accepts both upper and lower case characters. All values entered are assumed to be hexadecimal. If an invalid command or a non-hexadecimal value is entered, EXORbug responds by displaying WHAT ?, ringing the bell, and then issuing another prompt.

Control-X can be used at any time (at command level) to delete the current entry and cause another prompt to be displayed.

When entering hexadecimal values, only the last four digits are taken into account. If less than four digits are entered, leading zeroes are assumed.

Table 4-1 EXORbug Commands

COMMAND	FUNCTION	PAGE
PRNT	Display memory block in HEX & ASCII	04-04
PNCH	Dump memory block to audio cassette	04-05
LOAD	Load an oject tape from audio cassette	04-06
VERF	Verify an object tape against memory	04-07
DUMP	Dump memory block to EXORciser memory	04-07
DWLD	Download an EXORciser memory block	04-08
LINK	Download an EXORciser disk file	04-08
XDOS	Invoke the disk operating system	04-09
TMAP	Toggle address map	04-09
XCOM	Switch to terminal mode	04-09

Forts. nästa sida

RA	Display/change target A accumulator	04-10
RB	" " B accumulator	04-10
RC	" " CC register	04-10
RD	" " D register (A,B accumulator pair).	04-10
RP	Display/change target Direct Page register.	04-10
RL	Display/change target Location counter (program counter).	04-10
RS	Display/change target Hardware Stack pointer.	04-10
RU	" " User stack pointer	04-10
RX	" " X index register	04-10
RY	" " Y index register	04-11
RR	Display all registers	04-09
SM	Select memory location for display trace function and breakpoints.	04-11
SP	Select display speed	04-12
MV	Move memory block.	04-13
IS	Insert ASCII string to memory	04-13
EV	Extend breakpoints to WRITE condition	04-11
TC	Enable track change of memory location	04-11
DT	Disable track change of memory location	04-12
nnnn;G	Go to the target program at the specified address.	04-18
;G	Go to the target program through the user RESTART vector.	04-17
nn;I	Initialize memory block with the specified bit pattern.	04-21
;M or $M	Specify the memory search address range and mask.	04-19
nnnn;N	Trace nnnn instructions	04-19
;N	Trace one instruction	04-19
;P	Proceed with program execution	04-18
nnnn;P	Same as ;P but skip nnnn breakpoints	04-18
$T	Enable trace to ending address function	04-16
;T	Disable trace to ending address function	04-17
nnnn;V	Set breakpoint at address nnnn	04-14
;V or $V	Display breakpoints	04-16
nnnn;U	Remove breakpoint at address nnnn	04-16
;U	Remove all breakpoints	04-16
nnnn;W	Search memory for nnnn (1 or 2 bytes)	04-20
;Z	Line printer ON/OFF	04-21
nnnn/	Open location nnnn for memory change function and display contents.	04-22
followed by :		
[nn]LF	Open next memory location, display address & contents on next line.	04-22
[nn]SPACE	Open next memory location, display contents on same line.	04-22

[nn]COMMA	Open next memory location, no display	04-22
[nn]UA	Open previous memory location, display address & contents on next line.	04-22
[nn]SLASH	Reopen current memory location, display address & contents on next line.	04-23
'CHARACTER	Insert one ASCII character	04-22
nnnn;O	Calculate short relative offset to nnnn	04-23
nnnn;L	Calculate long relative offset to nnnn	04-23
[nn]CR	Terminate memory change function	04-23
CTRL-W	Suspend command execution	04-04
CTRL-X	Abort current command or entry	04-04
ESC-8	Display background change	04-25
ESC-C	Clear display from cursor	04-25
ESC-E	Erase screen	04-25
ESC-F	Display format toggle (40/80 characters)	04-24
ESC-G	Sound bell	04-25
ESC-H	Cursor to previous location	04-24
ESC-K	Kill line from cursor	04-25
ESC-L	Cursor home	04-25
ESC-N	Cursor to next location	04-24
ESC-O	Return to stand-alone mode of operation	04-25
ESC-S	Alphanumeric display ON	04-24
ESC-T	Alphanumeric display OFF	04-24
ESC-U	Cursor up one line	04-24
ESC-V	Cursor down one line	04-24
ESC-Y	Graphic display ON	04-24
ESC-Z	Graphic display OFF	04-24

Note : Brackets indicate optional entries

DATALITTERATUR

STUDENTLITTERATUR har givit en ny förlagskatalog innehållande,

Grundläggande böcker

Systemarbete

Databehandlingsteknik

Digital teknik och maskinvara

Numerisk analys

Programering

Tillämpningar

Katalogen kallas INFORMATINSBEHANDLING. Kann rekvireras från

STUDENTLITTERATUR Box 1719,22101 LUND. tel 046 - 307070

MICROPRO-CESSOR

INTERFACING TECHNIQUES (andra utökade uppl.)
av Austin Lesea och Rodnay Zaks

en oumbärlig bok för dig som vill veta mer om bl.a.

+ CPU: 6800,Z-80,8080,8085
+ INPUT/OUTPUT: Memory- och I/O-mapping Polling,interrupt,DMA. Parallell och serie-överföring, metoder och kretsar
+ PERIFERIENHETER: Interface mot tangentbord, LED-display,teletype,remsläsare,line printer,kassett,CRT,floppy disk. CRTC, FDC, CRC.
+ ANALOG/DIGITAL-OMVANDLING: Teori,komponenter,interface-metoder m.m.
+ BUS STANDARDS: S-100,6800,IEEE-488 (HP-IB),EIA RS232C,RS422,RS423
+ DIGITAL FELSÖKNING: Metoder och instrument,VOM,DVM,oscilloskop,logikprober,logikanalysatorer m.m.

416 sidor, 320 illustr. Engelsk text.

BESTÄLL IDAG SÅ SÄNDER VI BOKEN OMGÅENDE MOT POSTFÖRSKOTT. PRIS ENDAST 97:50 INKL MOMS OCH PORTO. (Normalt marknadspris c:a 160 kr)

DIGITAL CONNECTION HB
Box 288 175 25 JÄRFÄLLA

Hexadecimal	Decimal	Hexadecimal	Decimal
01 000	4 096	20 000	131 072
02 000	8 192	30 000	196 608
03 000	12 288	40 000	262 144
04 000	16 384	50 000	327 680
05 000	20 480	60 000	393 216
06 000	24 576	70 000	458 752
07 000	28 672	80 000	524 288
08 000	32 768	90 000	589 824
09 000	36 864	A0 000	655 360
0A 000	40 960	B0 000	720 896
0B 000	45 056	C0 000	786 432
0C 000	49 152	D0 000	851 968
0D 000	53 248	E0 000	917 504
0E 000	57 344	F0 000	983 040
0F 000	61 440	100 000	1 048 576
10 000	65 536	200 000	2 097 152
11 000	69 632	300 000	3 145 728
12 000	73 728	400 000	4 194 304
13 000	77 824	500 000	5 242 880
14 000	81 920	600 000	6 291 456
15 000	86 016	700 000	7 340 032
16 000	90 112	800 000	8 388 608
17 000	94 208	900 000	9 437 184
18 000	98 304	A00 000	10 485 760
19 000	102 400	B00 000	11 534 336
1A 000	106 496	C00 000	12 582 912
1B 000	110 592	D00 000	13 631 488
1C 000	114 688	E00 000	14 680 064
1D 000	118 784	F00 000	15 728 640
1E 000	122 880	1 000 000	16 777 216
1F 000	126 976	2 000 000	33 554 432

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
000	0000	0001	0002	0003	0004	0005	0006	0007	0008	0009	0010	0011	0012	0013	0014	0015
010	0016	0017	0018	0019	0020	0021	0022	0023	0024	0025	0026	0027	0028	0029	0030	0031
020	0032	0033	0034	0035	0036	0037	0038	0039	0040	0041	0042	0043	0044	0045	0046	0047
030	0048	0049	0050	0051	0052	0053	0054	0055	0056	0057	0058	0059	0060	0061	0062	0063
040	0064	0065	0066	0067	0068	0069	0070	0071	0072	0073	0074	0075	0076	0077	0078	0079
050	0080	0081	0082	0083	0084	0085	0086	0087	0088	0089	0090	0091	0092	0093	0094	0095
060	0096	0097	0098	0099	0100	0101	0102	0103	0104	0105	0106	0107	0108	0109	0110	0111
070	0112	0113	0114	0115	0116	0117	0118	0119	0120	0121	0122	0123	0124	0125	0126	0127
080	0128	0129	0130	0131	0132	0133	0134	0135	0136	0137	0138	0139	0140	0141	0142	0143
090	0144	0145	0146	0147	0148	0149	0150	0151	0152	0153	0154	0155	0156	0157	0158	0159
0A0	0160	0161	0162	0163	0164	0165	0166	0167	0168	0169	0170	0171	0172	0173	0174	0175
0B0	0176	0177	0178	0179	0180	0181	0182	0183	0184	0185	0186	0187	0188	0189	0190	0191
0C0	0192	0193	0194	0195	0196	0197	0198	0199	0200	0201	0202	0203	0204	0205	0206	0207
0D0	0208	0209	0210	0211	0212	0213	0214	0215	0216	0217	0218	0219	0220	0221	0222	0223
0E0	0224	0225	0226	0227	0228	0229	0230	0231	0232	0233	0234	0235	0236	0237	0238	0239
0F0	0240	0241	0242	0243	0244	0245	0246	0247	0248	0249	0250	0251	0252	0253	0254	0255

HEXADECIMAL-DECIMAL INTEGER CONVERSION

The table below provides for direct conversions between hexadecimal integers in the range 0-FFF and decimal integers in the range 0-4095. For conversion of larger integers, the table values may be added to the following figures:

SÄNDER ETT BIDRAG TILL MPU-LAREN

IDEN ÄR TILL HÄLFTEN KNYCKT FRÅN
MPU-LAREN NR2 78 /ARNE TILLBOM/S
PROGRAM.
MIN VERSION ÄR FÖR OSS SOM HAR
TBUG 1 I SNURRAN.
STARTADR. A0F2
PROGRAMMET FRÅGAR EFTER START OCH
STOPP ADRESSER PÅ DEN SNUTTEN SOM
MAN VILL SYNA NÄRMARE IBYTARNE.
FINNES DET TEXT STRÄNGAR INNE I
PROGRAMMET BLIR DET LITE KUFISKA
LOOPAR MEN DET KANSKE GER NYA IDEER
TILL PROGRAM HI HI .

HÄLSNINGAR FRÅN LIDKÖPING

THOMMY KARLSSON

```
*G A0F2
A0F2 A14C
A0F2 BD F131     JSR E
A0F5 BD F09B     JSR E
A0F8 F6 A004     LDA B E
A0FB F1 A002     CMP B E
A0FE 26 08       BNE R
A100 F6 A003     LDA B E
A103 F1 A005     CMP B E
A106 22 44       BHI R
A108 BD E131     JSR E
A10B CE A002     LDX I
A10E BD E0C8     JSR E
A111 FF A002     LDX E
A114 A6 00       LDA A X
A116 27 2B       BEQ R
A118 81 8C       CMP A I
A11A 27 18       BEQ R
A11C 81 8F       CMP A I
A11E 27 14       BEQ R
A120 81 CF       CMP A I
A122 27 10       BEQ R
A124 84 F0       AND A I
A126 81 20       CMP A I
A128 27 16       BEQ R
A12A 81 60       CMP A I
A12C 25 15       BCS R
A12E 84 30       AND A I
A130 81 30       CMP A I
A132 26 0C       BNE R
A134 BD E0CA     JSR E
A137 BD E0C8     JSR E
A13A FF A002     STX E
A13D 7E A0F8     JMP E
A140 BD E0CA     JSR E
A143 BD E0CA     JSR E
A146 FF A002     STX E
A149 7E A0F8     JMP E
A14C 7E F0F3     JMP E
*                FND.
```

```
#LIST

0010 REM PROGRAM FÖR ATT OMVANDLA
0015 REM HEX TILL DECIMALT.
0020 REM SKRIVET AV T.KARLSSON
0025 REM DEN 19 MAJ 1979.
0040 LET A$=" 0123456789ABCDEF"
0050 INPUT "HEX",B$
0060 FOR E=1 TO 17
0070 IF LEFT$(B$,1)=MID$(A$,E,1) THEN A=E-2
0080 IF MID$(B$,2,1)=MID$(A$,E,1) THEN B=E-2
0090 IF MID$(B$,3,1)=MID$(A$,E,1) THEN C=E-2
0100 IF RIGHT$(B$,1)=MID$(A$,E,1) THEN D=E-2
0110 NEXT E
0120 LET F=A*4096+B*256+C*16+D
0130 PRINT
0140 PRINT "DECIMALT  ":F
0150 END

READY
#RUN
HEX£ 6800

DECIMALT  26624

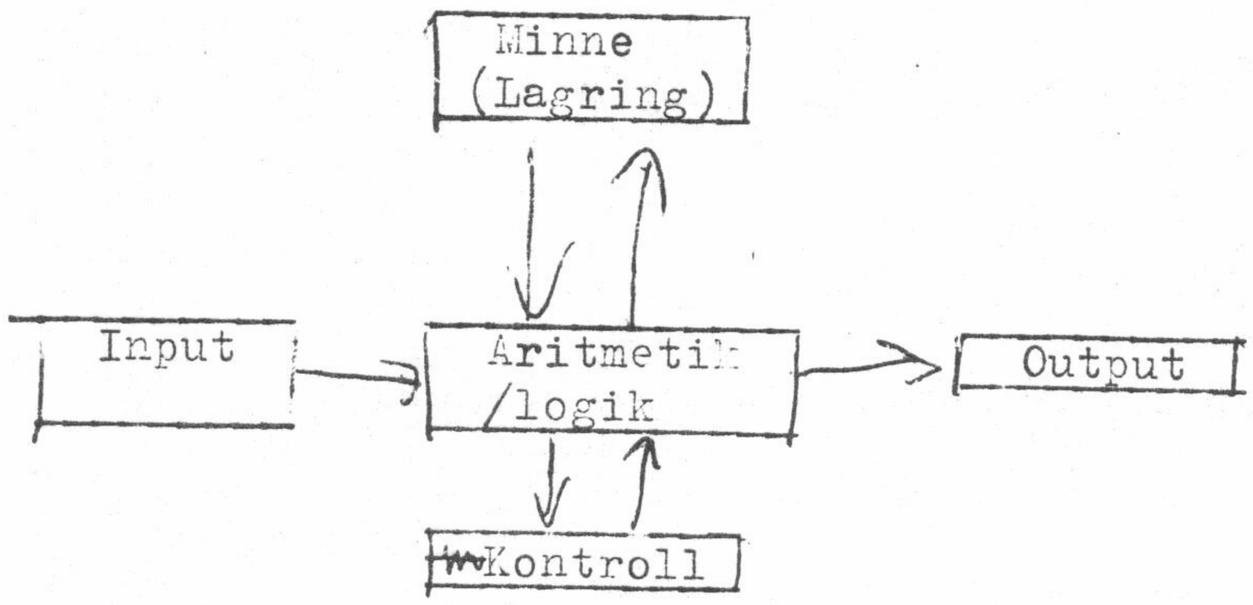
READY
#
```

EN LITEN BIT!! del 2

I förra numret pratade vi bara lite allmänt om vad en dator är. I detta nummer fördjupar vi oss lite i datorns olika delar. Dessutom börjar vi titta på programmering av 6800.

En dator anses vanligen bestå av fem grundläggande delar (se figur 1).

1. Input-enheter
2. Output-enheter
3. Minne
4. Aritmetisk/logisk enhet
5. En kontrollenhet - används för att synkronisera de andra delarna.



En del mikrodatorer har flera av dessa enheter i samma integrerade krets. Hur dessa enheter än är placerade måste alla finnas i ett system innan detta kan kallas "dator". Definitionen av en dator är centrerad kring minnesbegreppet och förmågan att lagra data och instruktioner i detta minne. Innan vi diskuterar instruktionerna bör vi dock bättre förstå hur datorn använder var och en av delarna.

Input

Som namnet anger är input-enhetens funktion att få information in i datorn. Normalt är denna funktion ett tangentbord kopplat till en bildskärm eller en skrivare, men input-enheter är inte begränsade till bara människoskötta tangentbord. Andra typer av input-enheter kan vara onkopplare, hålkorts- eller hålremseläsare. tjuvalarmsbrytare, termostater, fotoeellutrustning; nästan vilken utrustning som helst som kan visa ett på- eller av-läge kan användas. Även enheter som ger kontinuerligt ändrade tillstånd mellan av och helt på (analoga enheter) kan användas som input-engeter genom användande av en analog-till-digital-omvandlingskrets.

Output

Den andra av de fem grundläggande funktionerna är också en interface-enhet mellan datorn och verkligheten omkring. Men nu är det istället fråga om att visa information som kommer ut från datorn. Denna information är vanligen resultatet av några beräkningar gjorda efter flera inputs, men det kan också vara enbart en överföring av data som tidigare matats in. Bildskärmar och skrivare är de vanligaste formerna av output-enheter. Men en output-signal kan också användas för att slå av eller på en lampa, liksom att sköta någon elektrisk apparat. Tre bitar - de minsta dataenheterna, var och en av- eller på signal, nolla eller etta - kan t ex användas så här: en för att styra en motor att gå framåt eller bakåt, en andra bit för att ange hög eller låg hastighet och den tredje för att slå på eller av motorn.

Möjligheterna är oändliga.

Vissa enheter används för både in- och output. Kassetspelare och diskenheter är en mycket unik form av input/output- enheter. De kan användas både som input och output men de är också ett slags datorminne.

Apropå minne...

Den tredje huvudfunktionen i en dator är minnet. Alla instruktioner och data som datorn ska göra något med kommer att finnas i minnet vid någon tidpunkt.

En bra liknelse med ett datorminne är en vägg med postboxar. Varje minnesposition har ett eget "boxnummer" (adress) och varje box kan innehålla ett tecken (BYTE).

En BYTE består av 8 bitar (BITS på engelska). Därför finns det 256 olika symboler (2 upphöjt i 8 =256) som kan lagras i en minnesposition. Men bara en av dessa kan lagras i en minnesposition vid en given tidpunkt.

Vilken som helst av dessa symboler skulle kunna ha flera funktioner; de skulle kunna användas för att representera instruktioner, data eller delar av en instruktion eller ett datablock.

Det är mycket viktigt att komma ihåg att innehållet och adressen för en minnesposition är två olika saker. En instruktion som syftar mot en adress - angiven med 16 bitar - (vilket innebär att den kan vara från 1 till 65 536 - = 2^{16}) - får datorn att göra något speciellt med innehållet i den minnesposition som adressen anger. Innehållet kan förändras, däremot kan aldrig adressen förändras.

Aritmetisk/logisk enhet

Den fjärde funktionsenheten i datorn är den aritmetiska och logiska enheten ALU. Kontrollenheten och ALUn är de delar som ger varje dator dess egen "personlighet" och gör att en viss dator är mer effektiv för vissa typer av problem än en annan med annorlunda arkitektur (uppbyggnad).

8085, t ex, har en 8-bitars ackumulator som används för aritmetiska och logiska operationer. 8085 har sex 8-bitars register som kan paras ihop till tre olika 16-bitars register. Dessa 16-bitars register är användbara för att manipulera adresser. 6800, å andra sidan, har två 8-bitars ackumulatorer. 6800 har ett enda 16-bitars indexregister för adressmanipulation.

Trots att både 6800 och 8085 har vad som kallas "indexregister", så opererar de olika i de två systemen. Båda har en 16-bitars stackpekare som används för att kontrollera vissa maskinfunktioner, som t ex subrutinanrop. Användandet av stackpekare kommer att beskrivas senare i serien.

Kontroller

Den femte delen i ett datorsystem är kontrolldelen, den del av maskinen som programmerare blir mest välbekanta med. I denna del dekodas instruktionerna och här genereras olika kontrollsignaler för att kontrollera olika operationer i systemet. Instruktionsrepertoaren i både 6800 och 8085 innehåller fem grundläggande typer av instruktioner:

D a t a ö v e r f ö r i n g - flytta data mellan register eller mellan register och minne.

A r i t m e t i k - addera, subrahera, öka eller minska data i register eller i minne.

L o g i k - AND, OR, EXCLUSIVE OR, jämföra, rotera eller komplementera data i register eller i minne.

H o p p - villkorliga eller ovillkorliga hoppinstruktioner, subrutinanrop, och return-instruktionerr. ⊂⊃

S t a c k, I/O, m a s k i n k o n t r o l l - denna grupp innehåller I/O-instruktioner så väl som instruktioner för att upprätthålla stacken och interna kontrollflaggor (-bitar).

Programinstruktioner kan vara en, två eller tre byte långa. Vilken instruktion som helst som är längre än en byte måste ha en andra och/eller tredje byte i följande minnespositioner.

Det finns ytterligare ett mycket speciellt register som är en del av kontrollenheten i datorn. Det är programräknaren eller PC-registret. Alla datorminnen innehåller det program som ska följas, d v s den serie instruktioner som tillsammans utgör programmet. Följdaktligen måste datorn veta vilken instruktion som står i tur att utföras nästa gång, vilket PC håller reda på.

Adressmoder i 6800

6800 har sex moder för adressering. De adress-bytes som skrivs efter instruktionerna skrivs alltid så att den högsta delen av adressen anges efter instruktions-byten, medan den lägsta delen kommer sist - på samma sätt som vi skriver vanliga tal, de högsta före de lägre. Moderna är:

I n h e r e n t - instruktionen specificerar ett eller flera register som innehåller operander eller som resultatet sparas i. Inherent-instruktioner är alltid en byte långa.

I m m e d i a t e- den andra eller den andra och tredje byten i instruktionerna innehåller det aktuella data-värdet.

E x t e n d e d - dessa instruktioner är alltid tre bytes långa. Den andra och tredje innehåller adressen.

D i r e c t - dessa är alltid två bytes långa. Adra byten är ett värde från 0 till 255 som direkt kommer att adressera de första 256 byten i minnet. (6800 sätter själv in 00 som den högsta byten i adressen).

I n d e x e d - alla indexed-instruktioner är två bytes långa. Den andra byten har ett värde från 0 till 255 som läggs till det aktuella värdet i indexregistret. Detta skapar ett 16-bitars värde som används som dataadress. Innehållet i indexregistret ändras inte permanent genom denna addition.

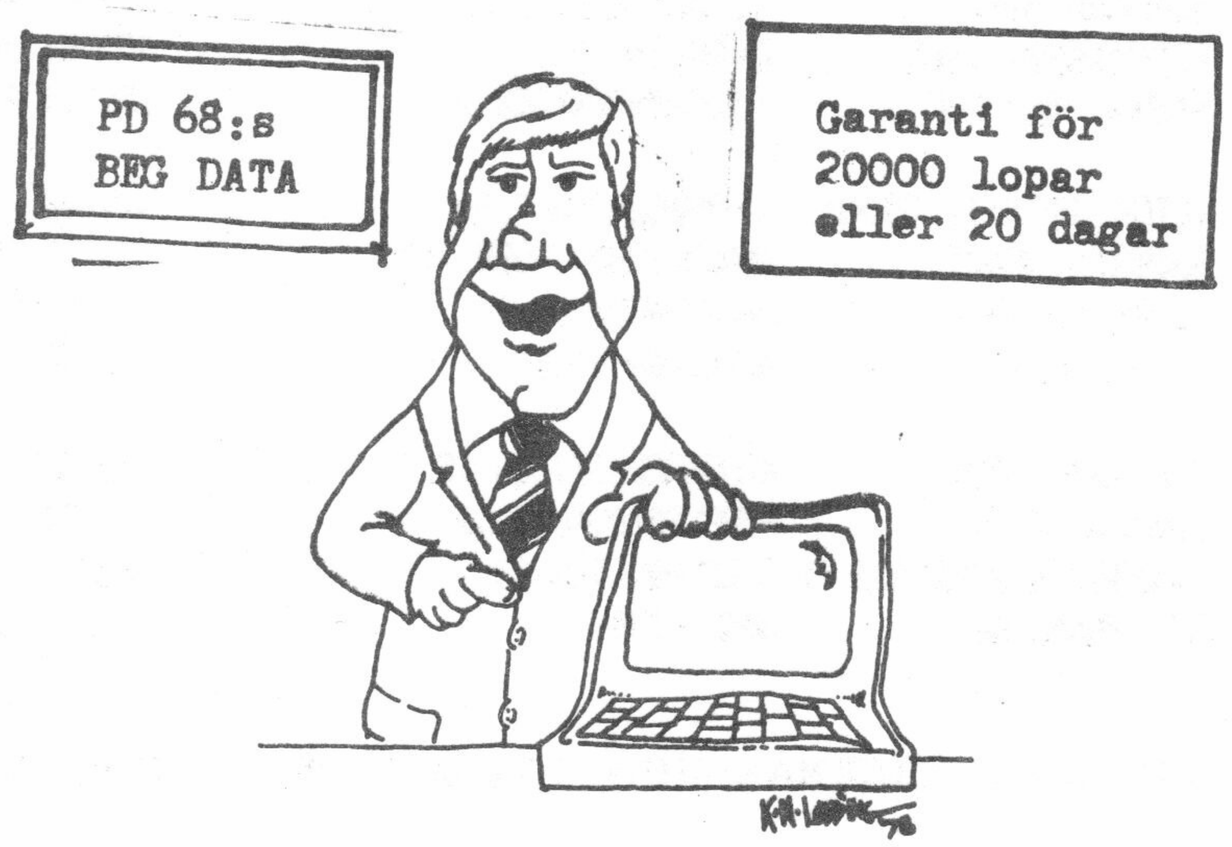
R e l a t i v e - alla hoppinstruktioner (utom JMP och JSR) använder denna mode. Alla dessa är två bytes långa. Användandet av relativ-mode är mycket lika indexed-moden men några skillnader finns.

Den andra byten i relativ-modens instruktion används som ett värde från -128 till +127, som adderas till värdet i PC. Det resulterande 16-bitars värdet lagras i PC, vilket gör att nästa instruktion utföres från en ny minnesposition. Eftersom PC pekar på början av instruktionen som följer efter hoppinstruktionen när additionen görs så innebär ett hopp till relativadressen noll att instruktionen som följer på hopp-instruktionen utförs i alla fall.

Ett hopp till relativadressen -2 orsakar en permanent loop då datorn hela tiden hoppar tillbaka till hoppinstruktionen.

Fortsättning följer i nästa nummer, då om I/O och med enkla program-skrivningar.

L-O.F.



"-och denna har ägts av en gammal dam, som endast har brukat densamma en gång per månad för kontroll av check-konto"

Denna sida innehöll medlemsregister.
Ej inskannat pga GDPR

```
1090 NEXT J
1100 PRINT
1110 REM BESKRIV
1120 GOSUB 8000
1400 REM ---------------------------------------------------------
1405 REM * * *          HUVUD LOOPEN BÖRJAR HÄR    * * *
1410 REM RÄKNA DRAGEN
1420  M0=M0+1
1430 REM DRAG
1440 GOSUB 6000
1450 REM KOLLA FÖR HITTANDE AV SKATTEN
1460 GOSUB 2000
1470 REM FÖRSÖK MED PIRATEN
1480 GOSUB 4000
1490 REM LOOPA TILLS DET ÄR KLART
1500 IF T>0 THEN 1400
1510 IF N<>5 THEN 1400
1700 REM BERÄKNA POÄNGEN
1710 GOSUB 3000
1720 PRINT
1730 PRINT "GRATULATIONER! DU FANN SKATTEN"
1740 PRINT "I ";M0;
1750 PRINT "DRAG OCH DU FÅR ";S+10;" POÄNG!!"
1760 PRINT "VILL DU JAGA IGEN ";
1770  A$="JN"
1780 GOSUB 5000
1790 ON A1 GOTO 1000,9999,1760
2000 REM ---------------------------------------------------------
2010 REM FUNNEN ?
2100 IF T<>N THEN RETURN
2110 IF T<0 THEN RETURN
2120 IF M6+3>M0 THEN RETURN
2200 PRINT "VILL DU TA DEN MED DIG (J/N) ";
2210  A$="JN"
2220  A2=2
2230 GOSUB 5000
2240 ON A1 GOTO 2300,2400,2250
2250 PRINT "..HUR SKA DU HA DET.."
2260 GOTO 2210
2300  T=-1
2310 PRINT
2320 PRINT "OK, LÅT OSS TA OSS UT HÄRIFRÅN!"
2330 RETURN
2400 PRINT
2410 PRINT "VI LÄMNAR DEN HÄR OCH DU KAN UTFORSKA"
2420 PRINT "LITE TILL."
2430  M6=M0
2440 RETURN
3000 REM ---------------------------------------------------------
3010 REM - POÄNG -
3020  S=0
3030 IF T=-1 THEN S=S+5
3040 IF P=1 THEN S=S+10
3050 FOR J=2 TO M9
3060  S=S+W(J)
3070 NEXT J
```

```
3080 RETURN
4000 REM ------------------------------------------------------------
4010 REM - PIRATEN -
4020 IF N=T2 THEN RETURN
4030 IF P=1 THEN RETURN
4040 IF T1=T2 THEN RETURN
4050 IF T<>-1 THEN RETURN
4060 REM HAN ÄR VID UTGÅNGEN MED SKATTEN. STRAFFA HONOM.
4065 REM (ARRGH, HUR KOM HAN HIT EGENTLIGEN ??)
4070 IF N=16 THEN P=160
4080 REM RÄKNA DRAG SEDAN TRÄFFAT TRÅNGA TUNNELN MED SKATTEN
4090 IF P1>0 THEN P1=P1+1
4100 IF N=3 THEN P1=P1+1
4110 REM GE HONOM NÅGRA DRAG TILL. SEDAN STRAFFA HONOM
4120 IF P1<15 THEN RETURN
4130 PRINT
4140 PRINT "PLÖTSLIGT STORTAR PIRATEN UT FRÅN"
4150 PRINT "SKUGGAN OCH SLITER SKATTEN IFRÅN DIG!"
4160 PRINT "- HAH! SKRIKER HAN - DU FANN MIN"
4170 PRINT "SKATT VA! NÄJA - JAG SKALL GÖMMA DEN"
4180 PRINT "BÄTTRE NÄSTA GÅNG!!"
4190 PRINT "OCH HAN FÖRSVINNER UT I MÖRKRET"
4200 PRINT "MED SKATTEN."
4210 P=1
4220  T=T2
4230 RETURN
5000 REM ------------------------------------------------------------
5010 REM - INPUT -
5020 REM FÖRSTA TECKNEN I A1 INPUT STRÄNGEN JMFÖRS MED
5030 REM BOKSTÄVERNA I I A$.VID MATCH RETURNERAS INDEXET I A$
5040 REM I A1. OM EJ TRÄFF, RETURNERAS STORLEKEN+1 AV A$.
5050 REM
5060 REM GET INPUT STRING
5070 INPUT Q$
5080 REM ANVÄND ENDAST FÖRSTA TECKNET
5085  Q$=MID$(Q$,1,1)
5090 FOR A1= 1 TO A2
5100 IF Q$=MID$(A$,A1,1) THEN RETURN
5105 NEXT A1
5110  A1=A2+1
5120 RETURN
6000 REM ------------------------------------------------------------
6010 REM - DRAG -
6020 REM KOM IHÅG VAR VI ÄR FÖR STUDS
6030  N9=N
6040 REM SET N8 TO VADSOMHELST MEN DU KÄN INTE GÅ DEN VÄGEN
6050  N8=0
6060 REM FRÅGA OM VÄGEN
6070 GOSUB 7000
6080 REM KOM IHÅG VAR VI ÄR FÖR SÄVITT EJ ÅTERVÄNDSGATA
6090 IF N=1 THEN 6120
6100  N0=N
6110  A0=A1
6120 PRINT
6130  I=M(A1,N)
6200 IF I=-2 THEN I=N9
6210 IF D<>0 THEN PRINT " * * DEBUG ";N;"TILL ";I
```

```
6220 IF I<500 THEN 6300
6230 REM FÖRDRÖJNINGSLOOP
6240  I=I-500
6250 FOR J=1 TO 100
6260 NEXT J
6270 GOTO 6200
6300 ON I/100+1 GOTO 6310,6340,6370
6310 REM NORMAL RUTT--MINDRE ÄN 100
6320  N=I
6330 GOTO 6400
6340 REM N+100 ADDERA 1 OM BÄRANDE PÅ SKATTEN
6350  N=I-100
6355 IF T=-1 THEN N=N+1
6360 GOTO 6400
6370 REM N+200 ADDERA 1 OM BÄRANDE PÅ SKATTEN ANDRA GÅNGEN
6380  N=I-200
6390 IF T=-1 THEN N=N+P
6400 IF N<>1 THEN 6500
6410 REM ÅTERVÄNDSGATA. VÄND DEN SÅ ATT DU KAN KOMMA UT ANDRA VÄGEN
6420 FOR J= 1 TO 6
6430  M(J,N)=2
6440 NEXT J
6450  M(7-A0,N)=N0
6500 REM PRINTA NOD BESKRIVNINGEN
6510 IF N8<>2 THEN GOSUB 8000
6520 REM KOM IHÅG ATT VI VARIT   HÄR
6530  W(N)=1
6540  N8=N
6600 IF M(1,N)<>-2 THEN 6800
6610 REM TVÅNGSDRAG, MED SLUMP MÅL.
6660  I=M(6,N)
6670  J=-1
6680 IF M(4,N)>100*RND(0) THEN I=M(5,N)
6690  J =-1
6700 IF M(2,N)>100*RND(0) THEN I=M(3,N)
6710 IF D<>0 THEN PRINT " * * DEBUG STUDS TILL ";I
6720 REM NU HAR VI EN NY DESTINATION. GÅ TILLBAKS OCH GÖR OM DET
6730 GOTO 6200
6800 RETURN
7000 REM ----------------------------------------
7010 REM - VILKEN VÄG -
7100 PRINT
7110 PRINT "                    VILKEN VÄG?";
7120  A$ = "NEUDWSP"
7130  A2 = 7
7140 GOSUB 5000
7150 IF A1<8 THEN 7300
7160 PRINT "VILKEN VÄG VÄLJER DU"
7170 REM GE INSTRUCTIONER
7180 GOSUB 7500
7190 REM BESKRIV POsITIONEN IGEN
7200 GOSUB 8000
7210 GOTO 7100
7300 IF A1<7 THEN 7400
7310 REM BERÄKNA OCH SKRIV UT POÄNGEN
7320 GOSUB 3000
7330 PRINT "DU HAR JUST NU ";S;" POÄNG"
```

```
7340 REM BÖRJA PÅNYTT
7350 GOTO 7100
7400 RETURN
7500 REM ==========================================
7510 REM SUBRUTINE FÖR ATT GE INSTRUCTIONER
7520 PRINT
7530 PRINT "SKRIV N,S,E,W,U,D FÖR NORTH,SOUTH"
7550 PRINT "EAST,WEST,UP,DOWN.SKRIV P FÖR POÄNG"
7560 PRINT
7570 RETURN
8000 REM ==========================================
8010 REM BESKRIV DEN NUVARANDE POSITIONEN
8050  I = INT(N/5)
8060  J = N - 5*I + 1
8070 REM HÄR FINNS TILLRÄCKLIGT MED SATSNUMMER FÖR ATT TA HAND OM
8080 REM NODERNA 0,49.DU MÅSTE LÄGGA TILL FLER OM DU ADDERAR
8090 REM NODERNA 50 OCH UPPÅT.

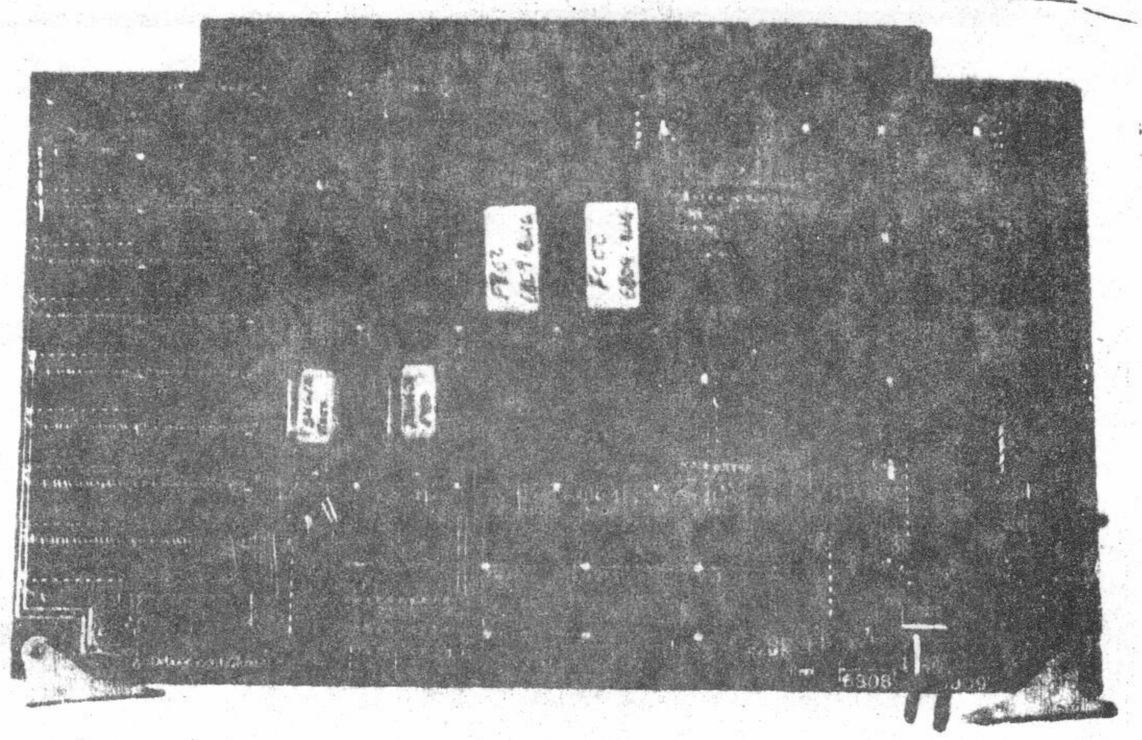
8200 ON J GOTO 9000,9010,9020,9030,9040
8210 ON J GOTO 9050,9060,9070,9080,9090
8220 ON J GOTO 9100,9110,9120,9130,9140
8230 ON J GOTO 9150,9160,9170,9180,9190
8240 ON J GOTO 9200,9210,9220,9230,9240
8250 ON J GOTO 9250,9260,9270,9280,9290
8260 ON J GOTO 9300,9310,9320,9330,9340
8270 ON J GOTO 9350,9360,9370,9380,9390
8280 ON J GOTO 9400,9410,9420,9430,9440
8290 ON J GOTO 9450,9460,9470,9480,9490
8400 IF T<>N THEN 8500
8410 PRINT
8420 PRINT " SKATTEN ÄR HÄR!"
8500 IF T<>T2 THEN 8600
8510 IF T1=T2  THEN 8600
8520 IF T1<>N  THEN 8600
8530 PRINT
8540 PRINT "PÅ VÄGGEN STÅR ATT"
8550 PRINT "PIRATER ALDRIG LÄMNAR SINA SKATTER"
8560 PRINT "TVÅ GÅNGER PÅ SAMMA STÄLLE!"
8600 RETURN
9000 REM ==========================================
9001 REM FÖRSTA DATASATSEN ÄR ANTALET NODER OCH DE TVÅ
9002 REM GÖMSTÄLLENA FÖR SKATTEN
9003 DATA 42,23,12
9004 REM N E U D W S   < RIKTNINGARNA >
9010 DATA 1,0,0,0,0,0,0
9011 PRINT "DU HAR HAMNAT I EN ÅTERVÄNDSGRÄND"
9012 GOTO 8400
9020 DATA 2,-2,101,-2,0,0,0
9021 PRINT "DU KAN INTE GÅ I DEN RIKTNINGEN."
9022 PRINT
9023 GOTO 8400
9030 DATA 3,33,2,1,10,106,4
9031 PRINT "EN TUNNEL GÅR I NORD-SYDLIG RIKTNING."
9032 PRINT "ÅT VÄSTER ÄR EN ÖPPNING."
9037 GOTO 8400
9040 DATA 4,3,30,2,11,2,1
9041 PRINT "DU ÄR PÅ KANTEN TILL ETT DJUPT HÅL."
```

```
9047 GOTO 8400
9050 DATA 5,8,8,15,10,8,16
9051 PRINT "DU ÄR UTANFöR GROTTAN"
9052 PRINT "GÅ SöDER UT FöR ATT KOMMA IN IGEN"
9053 GOTO 8400
9060 DATA 6,16,3,2,10,2,2
9061 PRINT "DU ÄR I BERGAKUNGENS HEM."
9062 PRINT "SOM TUR ÄR HAR HAN GÅTT FöR DAGEN"
9067 GOTO 8400
9070 DATA 7,-2,101,-2,0,0,0
9071 PRINT "BERGAKUNGEN ÄR HÄR!"
9072 PRINT "BÄST DU GÅR UT!"
9073 PRINT
9077 GOTO 8400
9080 DATA 8,18,18,15,10,18,9
9081 PRINT "DU ÄR FöRLORAD I SKOGEN."
9087 GOTO 8400
9090 DATA 9,-2,33,5,1,0,-2
9097 GOTO 8400
9100 DATA 10,-2,101,-2,0,0,0
9101 PRINT "DU KOMMER INTE ATT KOMMA LÅNGT OM DU"
9102 PRINT "TÄNKER GRÄVA I BERG"
9103 PRINT
9107 GOTO 8400
9110 DATA 11,1,13,4,2,1,2
9111 PRINT "DU ÄR PÅ BOTTEN AV ETT HÅL. EN LITEN"
9112 PRINT "STRöM FLYTER öVER STENARNA HÄR."
9117 GOTO 8400
9120 DATA 12,36,2,1,2,1,2
9121 PRINT "DU HAR HAMNAT I EN ÅTERVÄNDSGRÄND!"
9127 GOTO 8400
9130 DATA 13,2,37,2,1,11,14
9131 PRINT "DU ÄR PÅ EN öPPEN PLATS.HÄR ÄR EN SOTIG"
9132 PRINT "FLÄCK DÄR NÅGON NYSS HAR VILAT"
9133 PRINT "EN FACKLA MOT VÄGGEN.OSTADIGA STENAR"
9134 PRINT "VILAR OVANFöR DITT HUVUD"
9137 GOTO 8400
9140 DATA 14,13,1,19,2,31,31
9141 PRINT "DU ÄR I EN KLYFTA.PÅ VÄGGEN HöGT OVANFöR"
9142 PRINT "DIG ÄR FöLJANDE MEDDELANDE RISTAT:"
9143 PRINT "          'BILBO WAS HERE'"
9147 GOTO 8400
9150 DATA 15,-2,101,-2,0,0,0
9151 PRINT "DU ÄR INGEN FÅGEL. INSE ATT"
9152 PRINT "DU INTE KAN FLYGA."
9153 PRINT
9157 GOTO 8400
9160 DATA 16,5,33,2,10,1,106
9161 PRINT "DU ÄR I EN MYCKET LÅG KAMMARE. EN TRÅNG"
9162 PRINT "TUNNEL LEDER MOT öSTER OCH DU KAN"
9163 PRINT "GÅ MOT SYD ELLER VÄST. I NORR KAN"
9164 PRINT "MAN SE LJUS."
9165 PRINT
9167 GOTO 8400
9170 DATA 17,-2,101,-2,0,0,0
9171 PRINT "TUNNELN SMALNAR AV SÅ ATT DU EJ KAN"
9172 PRINT "KOMMA FöRBI MED SKATTEN"
9173 PRINT
```

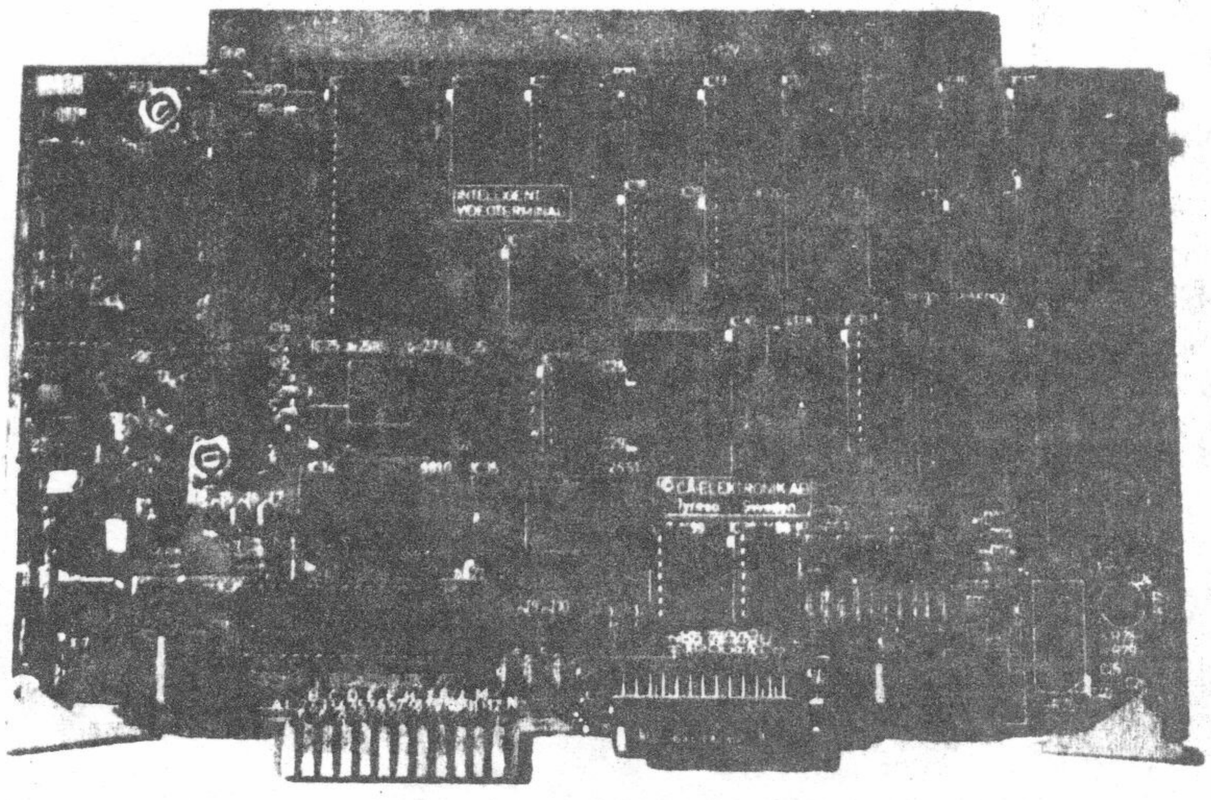
```
9177 GOTO 8400
9180 DATA 18,-2,101,8,0,0,0
9181 PRINT "JAG TROR INTE DU KAN FINNA GROTTAN"
9187 GOTO 8400
9190 DATA 19,224,2,2,14,1,42
9191 PRINT "DU ÄR HöGST UPP PÅ EN AVSATS."
9192 PRINT "UNDER DIG KAN DU LÄSA ETT MEDDELANDE"
9193 PRINT "          'BILBO WAS HERE'"
9197 GOTO 8400
9200 DATA 20,226,1,2,2,25,2
9201 PRINT "DU ÄR PÅ NORRA SIDAN AV EN SPRICKA,"
9204 PRINT "RINGANDE EKON FRÅN DJUPET ÄR"
9205 PRINT "DET ENDA SÄTTET ATT MÄTA SPRICKANS DJUP."
9207 GOTO 8400
9210 DATA 21,1,226,2,2,38,25
9211 PRINT "DU HAR KOMMIT TILL XANADU. UNDER DIG"
9212 PRINT "RINNER DEN HELIGA FLODEN ALPH GENOM"
9213 PRINT "GROTTVINDLINGAR OMÄTBARA FÖR"
9214 PRINT "MÄNNISKAN, NED TILL ETT HAV UTAN SOL."
9217 GOTO 8400
9220 DATA 22,-2,33,13,50,29,30
9227 GOTO 8400
9230 DATA 23,2,1,2,31,2,2
9231 PRINT "DU ÄR PÅ HYLLAN ÖVER GILJOTINRUMMET"
9237 GOTO 8400
9240 DATA 24,-2,101,19,0,0,0
9241 PRINT "JAG HöR JÄTTEN DÄR!!!"
9242 PRINT "DET ÄR BÄST DU GÅR TILLBAKA!"
9243 PRINT
9247 GOTO 8400
9250 DATA 25,21,20,2,2,1,19
9251 PRINT "DU ÄR I JÄTTENS GROTTA.DET ÄR NOG BÄST"
9252 PRINT "ATT INTE VARA HÄR NÄR JÄTTEN KOMMER!"
9257 GOTO 8400
9260 DATA 26,-2,65,-2,50,11,14
9261 PRINT "DU ÄR JUST NU I AREAN FÖR FORSKNING OCH"
9262 PRINT "UTVECKLING AV QUEST"
9263 PRINT
9264 PRINT "JAG ÄR LEDSEN, MEN BESÖKARE ÄR INTE"
9265 PRINT "TILLÅTNA HÄR. DU MÅSTE TYVÄRR"
9266 PRINT "AVLÄGSNA DIG."
9267 PRINT
9268 GOTO 8400
9270 DATA 27,2,40,2,2,21,20
9271 PRINT "DU ÄR NU I KRISTALLPALATSET. VÅGORNA"
9272 PRINT "HÄR VIBRERAR FRAM EN SÅNG AV ÄNGSLAN."
9277 GOTO 8400
9280 DATA 28,-2,60,221,50,14,19
9281 PRINT "- ARRRGHHHH....."
9282 PRINT
9287 GOTO 8400
9290 DATA 29,2,42,2,13,1,1
9291 PRINT "DU ÄR PÅ TOPPEN AV EN JÄTTE-STALAGTIT."
9292 PRINT "DU KAN HASA NER MEN DET GÅR INTE ATT"
9293 PRINT "BESTIGA DEN IGEN."
9297 GOTO 8400
9300 DATA 30,34,34,2,1,4,2
9301 PRINT "DU ÄR I EN LITEN GROTTA.HÄR FINNS EN"
```

```
9302 PRINT "BOK TITULERAD JANE'S KRIGSFARTYG,"
9303 PRINT "DATERAD 1763."
9307 GOTO 8400
9310 DATA 31,14,14,23,2,1,2
9311 PRINT "DU ÄR I GILJOTIN-RUMMET.EN MYCKET SKARP"
9312 PRINT "OCH TUNG STEN BALANSERAR PÅ HYLLAN EXAKT"
9313 PRINT "OVANFöR DIG!!!"
9317 GOTO 8400
9320 DATA 32,-2,101,516,0,0,0
9321 PRINT "DU ÄR I ETT SCHAKT OCH FALLER NEDFöR"
9322 PRINT "KLIPPORNA ! DU HAR INGEN CHANS ATT"
9323 PRINT "STANNA! HÅLL HUVUDET KALLT!"
9327 GOTO 8400
9330 DATA 33,2,1,2,1,116,3
9331 PRINT "DEN TRÅNGA TUNNELN SVÄNGER"
9332 PRINT "RUNT ETT HöRN"
9337 GOTO 8400
9340 DATA 34,1,35,2,1,30,30
9341 PRINT "DU ÄR I EN LITEN KRÅNGLIG LABYRINT"
9347 GOTO 8400
9350 DATA 35,2,1,2,37,34,36
9351 PRINT "DU ÄR I EN LITEN KRINGELKROKANDE LABYRINT"
9357 GOTO 8400
9360 DATA 36,35,2,1,37,34,12
9361 PRINT "DU ÄR I EN LITEN KRINGLANDE LABYRINT"
9367 GOTO 8400
9370 DATA 37,2,1,35,2,13,2
9371 PRINT "DU ÄR I EN SNÄRIG LITEN LABYRINT"
9377 GOTO 8400
9380 DATA 38,2,21,2,116,1,2
9381 PRINT "DU ÄR I EN FöRHISTORISK BOPLATS. PÅ"
9382 PRINT "VÄGGARNA SER MAN TECKNINGAR AV MYSK-"
9383 PRINT "OXAR MÅLADE I RöD LERA. PÅ GOLVET"
9384 PRINT "ÄR FULLT AV UTSTRöDDA BEN, "
9385 PRINT "LÄMNINGAR AV FöRGÅNGNA RITUALER."
9387 GOTO 8400
9390 DATA 39,2,40,2,32,21,26
9391 PRINT "DU ÄR I ETT SVART HÅL."
9392 PRINT "GRAVITATIONSKRAFTEN ÄR öVERVÄLDIGANDE."
9397 GOTO 8400
9400 DATA 40,40,40,2,2,40,41
9401 PRINT "DU ÄR I LABYRINTEN."
9407 GOTO 8400
9410 DATA 41,40,40,40,2,40,39
9411 PRINT "DU ÄR I LABYRINTEN."
9412 PRINT "DET ÄR MYCKET MöRKT HÄR."
9417 GOTO 8400
9420 DATA 42,28,28,28,28,28,28
9421 PRINT "DU ÄR I NÄRHETEN AV HELVETET!"
9422 PRINT "RöKELSE LIGGER TUNG I LUFTEN OCH"
9423 PRINT "DU TYCKER ATT ALLA RIKTNINGAR SER"
9424 PRINT "LIKADANA UT. DU KÄNNER DIG VILLRÅDIG."
9427 GOTO 8400
9430 REM DET FINNS INGA NODER FöR FöLJANDE RADNUMMER.OM DU HAMNAR
9440 REM HÄR, HAR DU ETT FEL I EN DATA SATS.
9450 REM
9460 REM
9470 REM
9480 REM
9490 REM
9500 REM
9510 REM
9520 REM
9530 REM
9999 END
```

NYHET 6809-CPU-KORT



Ett nytt CPU-kort med både 6809 och 6808 (=6800 med inbyggd klocka). På kortet finns även 8 k RAM samt 4 k EPROM till varje CPU. Val av CPU sker med en omkopplare. Kommer att beskrivas i RT nr 12, 1979. Prisex. bestyckad med 6808 och 1 k RAM (ingen eprom) kr 1.140,-. Komplett med 8 k RAM, 4 k eprom och 6809 kr 3.130,- inkl. moms. Enbart kretskort kr 355,-

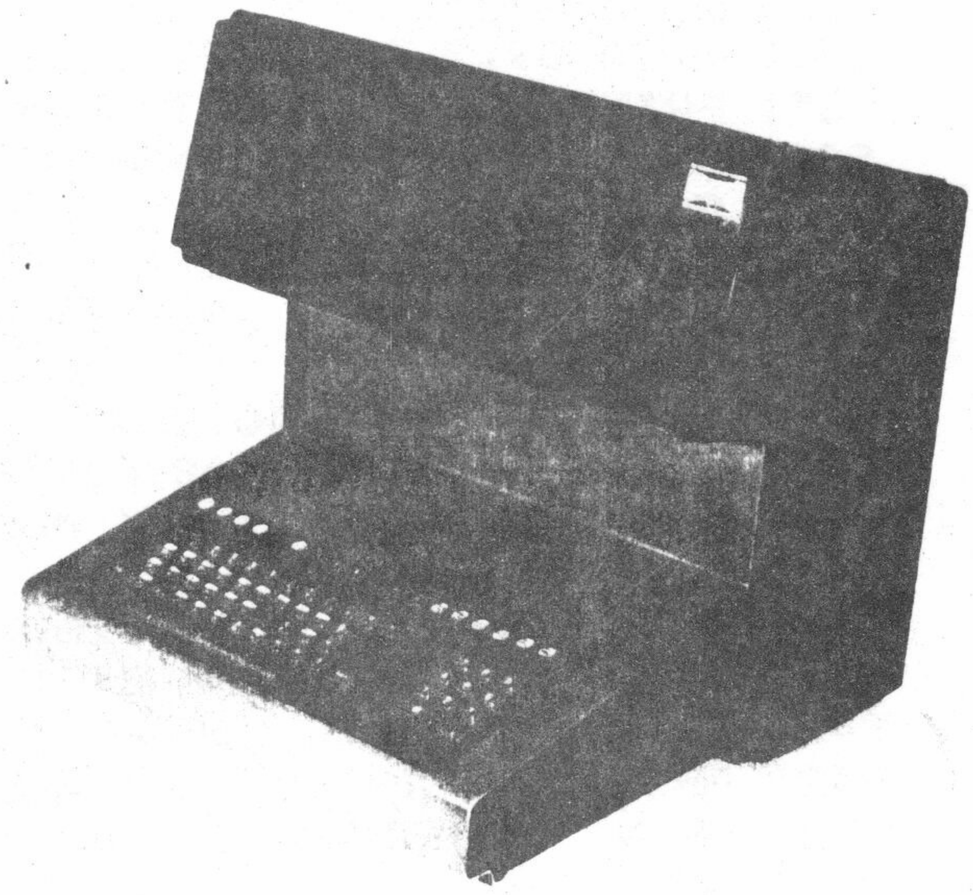


Intelligent videoterminal CÅ-790. Beskriven i RT nr 8 och 9, 1979. Nu ännu bättre med bl.a. in/urkopplingsbart KCS-modem. Alla funktioner styrs från tangentbordet. Lämpligt bord är Keytronics Sverigebord.

Specialpris till PD68-medlemmar: Terminal med KCS-modem, tangentbord och låda med hål. 2.275,- inkl.moms.

ETT TILLFÄLLE ....

En ombyggd Olivetti stansterminal innehållande kraftig nätdel på +5V 12 A, +12V/5A, -12V/5A och +24V/1A. 9-tums videomonitor, Modifierat D2-kit med 1 MHz-klocka, separat baudrateosc. 110-9600 b. Extra ACIA med V24-utgång för printer. Videominne CÅ-6806. Tangentbord Keytronic L1740 och KCS-modem. Fem lediga kortkont. (86-pol) för minneskort mm. TBUG2-monitor ingår. Allt i prima skick. Plats finns för två minifloppy.
Pris utan minifloppy: 3.900,-
Tillägg för en minifloppy med kontrollkort och FLEX 2.0 2.100,-
Tillägg för två minifloppy 3.700,-
(minneskort ingår ej)
(Nypris med två floppy ca 11.000,-)



INGENJÖRSFIRMA CÅ-ELEKTRONIK AB, Box 2010, 135 02 TYRESÖ
Tel. 08-742 34 01 kl. 12.30-16.30