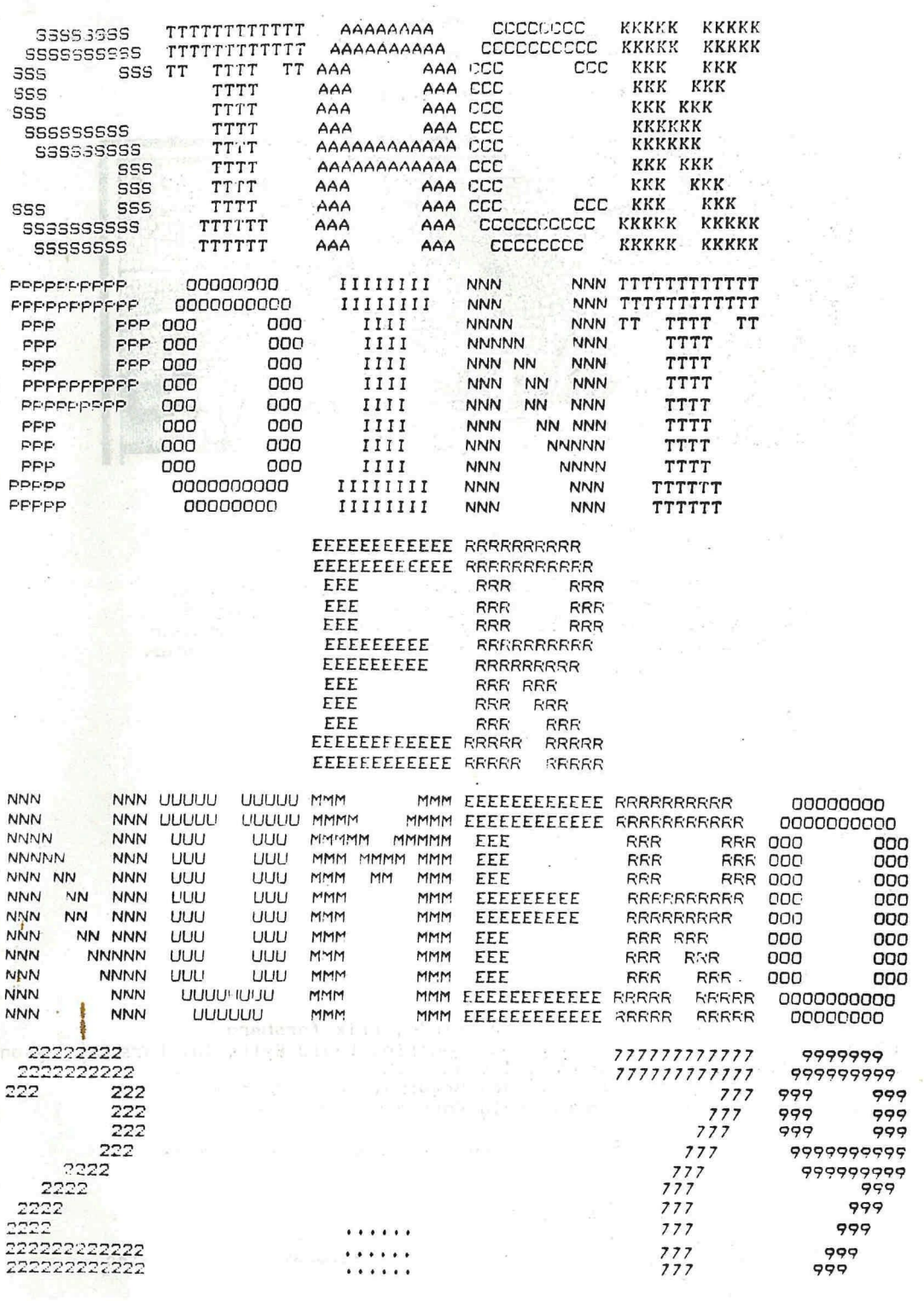
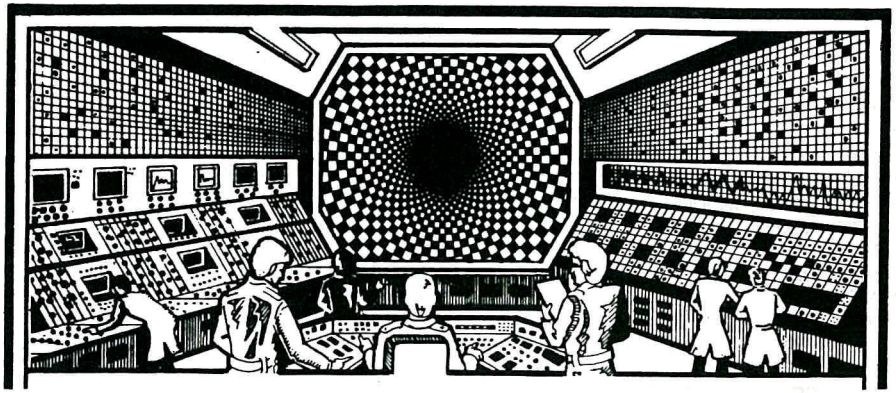


STACKPOINTER



OBS!!!! STACKMASKIN FRÅN ZILOG:

Under sommaren kommer STACKEN att disponera en maskin från Zilog, dvs vi får låna ett system av Scandia Metric. På detta kommer vi att kunna köra Assembler, PLZ, Pascal m.m. Ovanstående figur visar hur vi har tänkt oss att det skall se ut i STACKEN-lokalen i sommar.

Och så här ser den nya styrelsen ut:

Ordf	Nils Söderman
Vice ordf	Jan Gauffin
Sekreterare	Göran Skog
Kassör	Evald Koitsalu
Suppl	Per Lindberg
Hexmästare	JP Jynning
Redaktörer	Björn Ahlén Åke Fröjdh
Revisorer	Lars Ljungdahl Hans Nordström
Rev. Suppl	Rolf Andersson

De nya arbetsgrupperna:

Mjukvara	Björn Larsson, Björn Ahlén, Erik Forsberg
Hårdvara	Gunnar Eriksson, Jan Gauffin, Evald Koitsalu, Lars Bengtsson
Teknisk grupp	Gunnar Eriksson, Istvan Toth
Samköp	Rolf Andersson, Lars Brådinq, Kent Edgardh
Pitryckning	Nils Söderman, Evald Koitsalu, Erik Forsberg

Valberedning 1980: Rolf Andersson, Lars Ljungdahl, Björn Larsson

Stackpointer utges av mikrodatorföreningen STACKEN,
Box 5079, 141 05 Huddinge, postgiro 433 01 15-9.
Ansvarig utgivare: Nils Söderman, Fredrikssonsv. 12, Hägersten

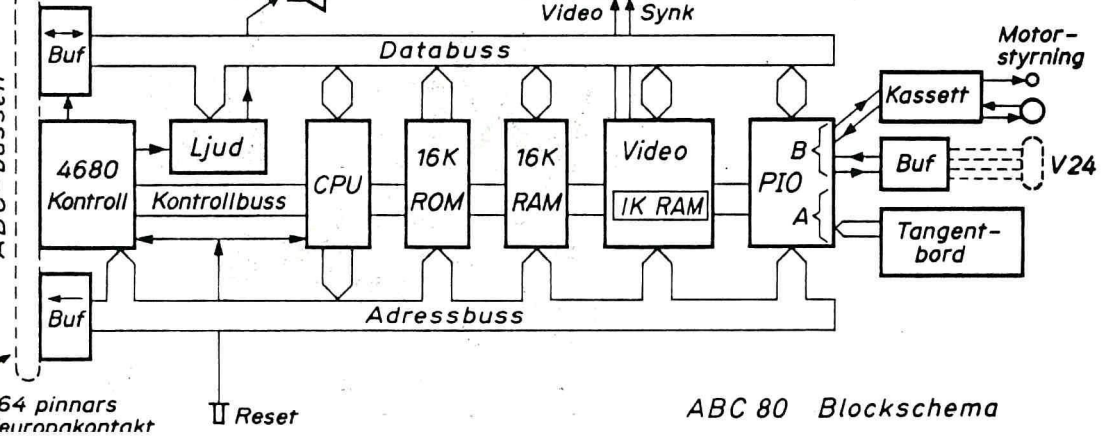
Den som regelbundet läser BYTE eller någon annan pervers datortidskrift kan inte undgå att lägga märke till en trend: de gamla fina S-100 datorerna förlorar mark till vad som lite löst brukar kallas "hemdatorer", dvs APPLE-II, TRS-80 å dom. Varför det? En förklaring är kanske, att det är andra människor som börjar vara med och handla computtrar. Det rör sig antagligen om s.k. "vanligt folk", som är mer intresserade av att köra STARTREK-program, än att felsöka floppyinterface. Marknaden för såna datorer är enorm: det har sålts n st. TRS-80!
Som de flesta säkert vet, finns det en svensk "hemdator" som har ungefär samma prestanda som TRS-80 och (ugh!) commodore PET. Jag menar alltså ABC80.

Grundversionen av en ABC80 består av en bildskärmsenhet (en lätt kamoflerad TV) en pyts med tangentbord och datorkort, och en kassettbandenhet för 800 baud. Ett lämpligt tillbehör är en dubbel minifloppy (Pertec, softsector). Vidare finns en 40 kolumners printer, plotter, digitizer, modem och lite annat smott och gott.

Hur ser då en ABC80 ut inuti? Jo, lyfter man på locket på'n, hittar ett tränat öga raskt en Z-80, en PIO-port, 16 kbyte dynamiska RAM, en BASIC i ROM, och lite annat junk.
Av RAMmarna har man som användare tillgång till 15-13 kbyte, beroende på om man kör floppy eller kassett. Minnet är expanderbartmed 16 k, men en "liten miss" i konstruktionen gör att man måste använda statiska RAM på ett externt kort.

I ROMmarna i botten ligger en 16 k BASIC-interpretator. Det är en riktigt bra "extended BASIC" med det mesta man kan önska sig, inklusive en s.k. "ASCII-aritmetik" som är aritmetiska operationer mellan numeriska strängar. Det enda man saknar är ett DELETE- kommando för att ta bort rader.
Som var och en som sett en ABC80 i levande livet vet, har den en textskärm med 40x24 tecken, fullt jämförbart med de amerikanska plagiaten, alltså. Det finns både svenska karaktärer och små boktsäver. Dessutom finns grafik: varje teckenposition är indelad i 2x6 "pixel", små rutor som kan hanteras från BASIC med satserna SETDOT X,Y CLRDOT X,Y och funktionen DOT(X,Y), (ex: IF DOT(I,J) THEN PRINT "Här var'e dott!")

Grafiken rymmer allstå 77x71 pixel, kanske lite i minsta laget om man vill rita mjuka kurvor och quadropeder, men fullt tillräckligt i de flesta fall. Grafikformatet är inte tillkommet av en slump: det är kompatibelt med Wiewdata-standarden, och det kan kanske ha vissa fördelar i framtiden tycks man ha resonerat.



ABC 80 Blockschema

Lite "shop talk": ABC:n har en "realtidsklocka", som fungerar så här: var 20:e millisekund får CPU:n ett interrupt från någonstans, och uppdaterar då tre bytes i minnet. (Dessutom lär interruptrutinen innehålla avkodning av tangentbordet.) Man kan alltså från BASIC läsa och skriva dessa bytes. Logiska värden lagras som hetalet -1% (%-tecknet anger heltal).
Man kan även ha logiska uttryck i tilldelningssatser:
satsen A%=A%=0% ger alltså A% värdet -1 om A%=0, annars blir A%= 0
Man kan naturligtvis komma åt enstaka bytes i bildminnet, men adresserna till varje tecken är inte organiserade radvis, som man kanske skule kunna tro. Adressen kan dock beräknas så här:

```
Z% = 31744% + Y% * 128% - Y% / 8% * 984% + X%
```

Där Z% blir RAM-aressen till tecknet X% på rad Y%

Baktill på tangentbordet/datorn sitter en 64-polig europakontakt. Där finns de flesta signalerna i datorn tillgängliga enligt 4680-bussens standard (dock inte RFRSH). Här anslutes paralellinterface till t.ex. floppy och printer.

Om man ska försöka summera intrycken av ABC80, kan man konstatera att det rör sig om en relativt stark BASIC-dator. Den som tycker om att programme i BASIC finner i ABC80 en kombinerad leksak och lekkamrat, som klarar det mesta. Den interna uppbyggnaden gör dock att det är svårt att köra annat än BASIC och lite assembler.
(Se min artikel i nästa STACKPOINTER: PASCAL på ABC80 -kan de' va' nåt?)

Om man är intresserad av insidan på ABC80, bör man läsa G. Markesjö's bok "ABC om mikrodatorer", som även innehåller en del matnyttigt för den som håller på med Z-80 i allmänhet.

Per Lindberg / The mad programmer strikes again !

"Which one of you does the bird imitations?"

Mjuka varupaket

ALGOL 68 vs PASCAL .
==========

Vad har ALGOL 68 , som inte finns i PASCAL ?

1) Det finns dynamisk minnesallokering , och en inbyggd garbage collector . På det sättet slipper man hålla reda på vilka areor i minnet man måste spara och vilka man kan göra sig av med när heapen börjar bli full och programmet behöver mer minnesutrymme . Det är datorns sak och inte programmerarens att hålla reda på minnesadresser och sånt krafs , när man programmerar i ett så kallat högnivåspråk . I gängäld får man stå ut med att programmet kanske drar lite mer CPU-tid , men det gör oftast detsamma i ett mikrodatorsystem , och i de fall där exekveringshastigheten är betydelsefull skriver man normalt hela programmet , eller i alla fall de mest tidskrävande delarna av det i assembler .

2) Man kan ha variabla gränser i matriser . Det kan vara bra att ha , även om det alltid går bra att använda länkade listor i stället .

3) Parametrarna till en procedur kan tillhöra vilken variabeltyp som helst , likaså procedurens värde . En procedur med parametrar och värde av givna variabeltyper är också en variabeltyp .
Ex : sin är en PROC (REAL) REAL , d v s en procedur med en parameter av typ flyttal och som returnerar ett värde av samma typ . Värdet av proceduren är sinus för värdet av parametern .
Ex : PROC (REF FILE , STRING , CHANNEL) INT open , försöker öppna en fil med det namn som anges av textsträngen , som är parameter nummer två , på den kanal som är tredje parameter till open , och om det går att öppna filen sätts den REF FILE som är första parameter att peka på ett minnesblock som beskriver på vilket device filen är öppen , pekare till I/O-rutiner som ska användas och andra maskin- och operativsystemsberoende grejor . Proceduren returnerar värdet noll om det gick att öppna filen och en felkod om det inte gick . Felkoden anger vad som gick snett , t ex disk full , write protect ...

4) Man kan ha pekare på variabler av alla typer .

5) Precis som i PASCAL kan man definiera flerdiminsionella matriser av alla variabeltyper . En matris av en viss variabeltyp med ett visst antal indices är själv en egen variabeltyp .

6) Om man känner för det kan man definiera nya operatorer eller definiera om de som redan finns . Man kan definiera prioritetsordningen mellan operatorerna också . En operator är egentligen (nästan) detsamma som en procedur med en eller två parametrar . Procedurer är inte alls nödvändiga (som i LISP) , men programmen brukar bli lättare att läsa om man inte har så fördömt mycket paranteser .
OBS !!! En operator är ingen variabeltyp . Gissa varför !

7) Det finns operation-and-assignment-operatorer , som består av en normal operator hopbuntad med assignment-operatorn . Ex : counter +:= 1 , som är samma sak som : counter := counter + 1 , fast kortare att skriva . Dessutom kan kompilatorn se att det går att generera snabbare maskinkod , utan att den behöver behöver testa en massa specialfall , optimera i flera pass , och annat som förlänger kompileringstiden avsevärt .

8) Eftersom man ofta använder konstruktioner som : THEN IF och ELSE IF , så finns det speciella ord för det : THEF och ELIF . Det är ju som oftast så att man vill ha flera satser efter THEN och ELSE och då måste man ha ett BEGIN-END-block kring dem i PASCAL och andra ALGOL-språk . Lat ska man vara . I ALGOL 68 har man helt enkelt plockat bort kravet på att det bara får stå en sats efter THEN och ELSE , och satt dit ett litet FI , som slut på IF-satsen . Om man råkat glömma ett IF eller något kan kompilatorn upptäcka att man parat ihop BEGIN med FI eller CASE med END eller så och säga : aja baja ! Om man bara har BEGIN-END som satsparanteser är det svårare att upptäcka sådana här skeva block . En gång satt jag i 10-15 minuter med ett långt SIMULA-program och prickade av BEGIN-END-par som hörde ihop , inifrån innersta blocknivån och ut , tills jag hittade det ställe där ett extra END skulle petas in , jämte de fyra som redan stod där . När jag gjorde motsvarande fel i ett ALGOL 68-program på DEC-10 , rättade kompilatorn själv felet genom att stoppa in ett extra FI och skrev ut en varning .

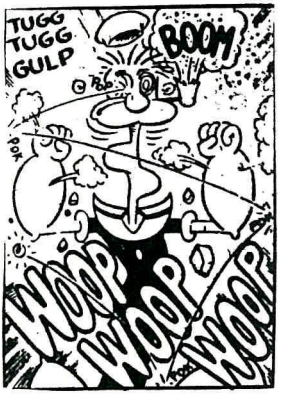
9) I en loop-sats i PASCAL måste man antingen ha testen först (WHILE villkor DO sats) eller sist (DO satser UNTIL villkor) . I ALGOL 68 kan man ha testvillkoret var som helst i loopen . Loop-satsen i ALGOL 68 kan dessutom innehålla en FOR-del , med styrande variabel , och valfritt steg (inte bara 1 eller -1) . En fullständig loop-sats ser ut som :

```
FOR variabel FROM startvärde BY stegvärde TO slutvärde
WHILE
   sats ; sats ... ; logiskt värde
DO
   sats ; sats .....
OD ;
```

Enligt en notis i senaste numret av BYTE finns det en ALGOL 68-kompilator för Z-80 . Vi ska undersöka saken närmare . Spännande fortsättning i nästa nummer ...

```
FROM birth TO death WHILE NOT mad
DO
  write your programs in algol 68
OD ;
UP gremlins
```

/Jan Michael Rynning - one of the sane programmers



PASCAL -Kan de' va' nåt ?

Som du säkert har märkt, pratar alla mikrodatorfans om ett språk som heter PASCAL, och som enligt sägnen lär besitta magiska egenskaper! vi ska nu titta lite på PASCAL, och vad det förmår.

Vi börjar med ett litet löjligt program, som illustrerar in-och utmatning.

```
PROGRAM cirkel;
CONST pi=3.1415926535;
VAR diameter,area : real;
BEGIN
  write('Was ist diametern? '); Break;
  read(diameter);
  area:=pi*diameter*diameter/4;
  writeln('Arealen=',area);
END.
```

Det första man lägger märke till, är likheten med andra språk av 'ALGOL'-typ. (ALGOL, SIMULA, ALGOL68, PL/Z PL/I etc.), och som de som var med på PASCAL-föredraget säkert kommer ihåg, har PASCAL anor från ALGOL W.
Ett program i PASCAL börjar alltid med PROGRAM namn;
Därefter kommer alla deklarationer, (CONST, VAR, etc)
Sen kommer ett BEGIN
åssså kommer alla satserna i programmet
och sist ett END. (med punkt efter.....)

Inmatning sker allstå med read( );
och utmatning med write( ); eller writeln( );
Skillnaden mellan write och writeln är att writeln bjuder på en radframmatning medan write skriver ut, och stannar på samma rad.
Satsen Break; används till att forcera utmatning av I/O bufferten.
Vi ser också att man kan ha ett godtyckligt antal argument i read och write.
Vad gör nu programmet? Tjaa..

PASCAL innehåller en hel del trevliga finesser. Man kan t.ex. definiera egna variabeltyper med TYPE, så här:

```
PROGRAM LATMASK;
TYPE day =(MONDAY,TUESDAY,WEDNESDAY,THURSDAY,FRIDAY,SATURDAY,SUNDAY);
VAR today,tomorrow : day;
    number : integer;
BEGIN
  for today:=MONDAY to WEDNESDAY do
  BEGIN
    tomorrow:=succ(today);
    number:=ord(FRIDAY)-ord(today);
    write('Today is ',today,', tomorrow is ',tomorrow);
    writeln(' and it is',number,' days until friday!!');
  END;
END.
```

Funktionen succ( ) ger alltså efterföljande element i mängden day.

Om du vill studera PASCAL på lite närmare håll, kan du titta i någon av de här böckerna: (som finns på Almqvist & Wiksell på g:la Brogatan)

A PRACTICAL INTRODUCTION TO PASCAL av Wilson och Addyman
PASCAL USER MANUAL AND REPORT av Jensen och Wirth (som gjorde PASCAL)

Dessutom finns en ny bok på svenska att köpa i försöksupplaga på bl.a. teknis' kompendieförmedling.

Per Lindberg / The mad programmer strikes again!

UCSD PASCAL

Den här artikeln är tänkt att kortfattat beskriva UCSD-implementationen av språket PASCAL på Z-80.

PASCAL utvecklades av Niklaus Wirth i Zürich i slutet av 60-talet för få ett språk lämpat för undervisning i strukturerad programmering och för att möjliggöra effektiva implementationer på alla datorsystem. Denna standard-PASCAL blev pga dessa skäl något litet för abstrakt för praktisk användning.

University of California, San Diego, UCSD har därför utvecklat ett Pascal-system lämpat för mikroprocessorer. Första systemet blev DEC's LSI-11 serie. Fr.o.m. V1.4 som kom 1977 finns också Z-80 m.fl. implementerade.

UCSD Pascal grundar sig på begreppet P-kod som innebär att kompilatorn genererar maskinkod för en tänkt P-maskin. Denna P-maskin kan antingen finnas i hårdvara (WD:s Pascal engine) eller i form av en interpretator skriven i assemblerspråket för den maskin som ska exekvera P-kod. På Z-80 är interpretatorn skriven i assembler och är på ca 8K.

UCSD Pascal är emellertid ej bara en Pascal-kompilator, utan också ett helt operativsystem.
Man får när man köper UCSD Pascal:

P-kodsinterpretator
Pascalkompilator
Operativsystem
Editorer
File handler utilities
Linkers
Assembler

Av dessa program är endast interpretatorn skriven i assembler, resten är skrivet i Pascal.

Detta medför att programmen lätt kan överföras till olika maskiner.

Man behöver därför inga andra programsystem för att köra Pascal utan UCSD Pascal innehåller allt som behövs för att tillämpa språket i praktiken på sin dator.

Vad behövs då för att köra Pascal?
Man behöver en dator med
- minst 48K RAM
- minst en 8" floppy-drive
- en terminal av något slag

UCSD levererar systemet på IBM-disketter.
Om man har CP/M igång på sin dator är det speciellt lätt att få igång Pascal. UCSD skickar nämligen med en CP/M-disk med programvara för att starta upp Pascal första gången.

All I/O för Pascalen sker genom BIOS-delen av CP/M som då kan föras över till Pascaldisken tillsammans med en bootstrap-loader (sourcen finns på CP/M-disken). Efter detta behöver man bara Pascaldisken för att starta upp.

Editorn finns i två versioner, en för snabba videoterminaler och en för teletypeliknande terminaler.
Videoeditorn kräver vissa funktioner ss cursoradressering m.m. Olika terminaler har ju olika koder för detta, men systemet kan lätt anpassas för dessa.

/Erik Forsberg

Programmeringsspråket PLZ

PLZ är ett högnivåspråk till Zilogs MCZ datorer. Språket är i
förtsta hand avsett att ersätta assemblerprogrammering i speciellt
olika systemprogram som kompilatorer, etc.
PLZ är egentligen en hel familj språk.
PLZ/ASM är ett assemblerspråk som förutom vanliga assemblerfaciliteter
ger möjligheter till vissa strukturerande uttryck.
PLZ/SYS är ett generellt högnivåspråk.
Som illustration kommer nu ett par exempel på PLZ/SYS.
Först ett sorteringsprogram (bubblesort).

```
bubble_sort module

CONSTANT
    false  := 0
    true   := 1
EXTERNAL
    print_array procedure (first ^word count byte)

INTERNAL
    a  array [ 10 word ] := [ 33 10 2000 400 410 3 3 33 500 1999 ]

    sort procedure (n byte)
    local
       i,j     byte
       switched  byte
       temp     word
    entry
    do
        switched := false
        i := 0
        do
           if i > n-2 then exit fi
           j += 1
           if a[i] > a[j] then
              switched := true
              temp := a[i]
              a[i] := a[j]
              a[j] := temp
           fi
           i += 1
        od
        if switched = false then return fi
     od
     end sort

GLOBAL
     main procedure
     entry
        sort (10)
        print_array (## a[0], 10)
     end main

end bubble_sort
```

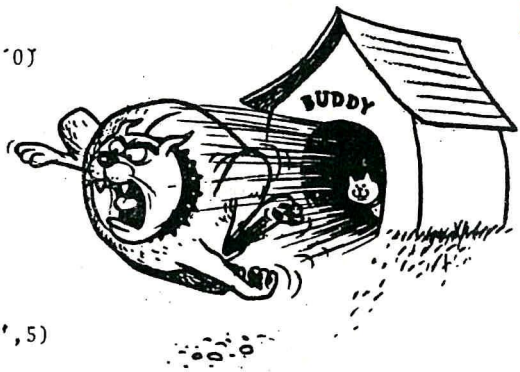
De rutiner som ej finns i programmet är deklarerat under EXTERNAL-
rubriken och är således separatkompilerat och ska sedan länkas ihop
med huvudprogrammet.

Sedan ett exempel på en enkel rekursiv procedur.

```
EXTERNAL
    putch  procedure (char byte)
    putstring procedure (start ^byte count byte)

INTERNAL
    nprint procedure (n integer) !prints non-negative decimal
                                  integer with leading zeroes suppressed!
    entry
       if n > 0 then
          nprint (n/10)
          putch (byte (n mod 10) + '0')
       fi
    end nprint

GLOBAL
    printdec procedure (n integer)
    entry
       if n > 0 then nprint (n)
       else
          if n=0 then putch ('0')
          else putch ('-')
             if n=-32768 then
                putstring (#'32768',5)
             else
                nprint (abs(n))
             fi
          fi
       fi
    end printdec
```



ROS

Operativsystemet ROS är ett intressant stycke mjukvara till Z-80-systemen. ROS:en innehåller bl.a. Editor, Assembler och systemkommandon som DUMP, PUNCH, LOAD m.m. Minnesbehovet är 8K ROM eller RAM från A000H till BFFFH + 4K system-RAM på adress D000H. Användarprogrammen kan läggas varsomhelst i övrigt minne.

Originalversionen är avsedd för serie-I/O med status på port 0 och data på port 1. Det är dock lätt att ändra detta till parall-ell I/O, t.ex. för CD:s bildskärm.

Vid assembleringen specificerar man var maskinkoden skall ligga och hur listfilen skall se ut, t.ex. standard, endast felaktiga rader, standard + symboltabell eller standard+ korsreferenslista. Man kan också läsa/skriva mot t.ex. kassettinterface, skrivare m.m.

Flera källprogram kan finnas inne i minnet samtidigt under olika namn. Dessa kan man sedan editera och assemblera efter behag, puncha på remsa eller kassett etc.

Som stöd för programmerandet finns det i ROS 48 st användbara subrutiner. Man kan t.ex. utföra systemkommandon från användar-program, skriva ut hextal, text m.m., läsa in tal, plocka parametrar öka i tabeller, jämföra strängar m.m. m.m.

Tack vare en mycket omfattande lista över entrypoints och parameter-adresser till rutinerna är det någorlunda enkelt att disassemblera hela ROS:en. Intresserad person kan påräkna tillgång till Stackens lånade Zilog-system under sommaren. Hör av dig till någon i styrelsen.

SNABB MULTIPLIKATION 8 * 16 BITAR FÖR Z-80

Ofta vill man snabbt kunna multiplicera ett 16-bits tal med någon måttlig konstant. Här är en bra lösning, i stort sett stulen ur Electronics March 15, 1979.

```
MULT: ; HL:=A*DE    (Both unsigned)

        LD      HL,0          ; CLEAR TARGET
MLOOP   OR      A             ; CLEAR CARRY & CHECK IF Ø
        RET     Z             ; ZERO WHEN DONE
        RRA                   ; MULTIPLIER LSB -> CY
        JR      NC,SHIFT      ; IF LSB=Ø, NO ADD
        ADD     HL,DE         ; ADD IN MULTIPLICAND
SHIFT   SLA     L             ; SHIFT MULTIPLICAND
        RL      H             ; LEFT
        JR      MLOOP         ; KEEP ON RUNNING
```

Obs att exekveringstiden blir kortare ju färre signifikanta bitar det finns i multiplikatorn.

/Björn Ahlén

Lite datorslang

@ Commercial-at, kallas oxo kanelbulle, alfaslang, snabel-a eller öra.
# Numbersign, eller pound sign, kallas oxo Brädhög (efter kartttecknet med samma namn), skräpa eller stege.
() Parentes "Parre", mjuka parenteser
{} Klammer "fiskmåsar"
[] hakparenteser, (vänsterhake, högerhake)
/ Slash
\ Backslash
<> mindre än, större än. Används oxo i en del datortext som parenteser. Kallas även vänster resp. högerpil
& Ampersand. Latinets "Et"
" Citationstecken, eller flarp, dubbelflärp, dubbelblipp, fnutt-fnutt, flugskit, gäsfot.
' Accent eller flärp, blipp, fnutt
` Grav accent eller backblipp
| Vertical line
_ Underscore, understrykningstecken.
~ Tilde
^ Cirkumflex, Pil-upp, tak, up-arrow.


1


2


3


4

1K MONITOR FÖR Z-80

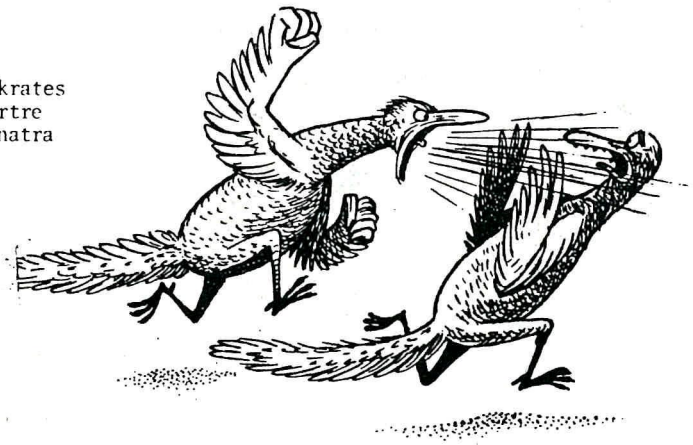
Här en liten orientering om kommandona i 1K-monitorn för våra Z-80-system:

DM	Display Memory	- visar minnesinnehåll hexadecimalt
DR	Display Registers	- visar registrens innehåll i hex
G	Go	- börja exekvera på viss adress
G /	Go w. Breakpoints	- d:o men med upp till 5 brytpunkter
I	In	- läser en I/O-port
J	Jump	- hoppa till PROM nr 2 (adr 400H)
K		- hoppa till prom nr 3 (adr 800H)
L		- hoppa till prom nr 4 (adr C00H)
M	Move	- flytta data från adr till adr
N	Nulls	- skriv NULls till remsa/kassett
O	Out	- skicka data till utport
R	Read	- ladda remsa/kassett
SM	Substitute Memory	- skriv in data/program hexadecimalt
S	Substitute register	- ändra registerinnehåll
V	Verify memory	- jämför två minnesareor
W	Write	- lagra på remsa/kassett
<	"After" operator	- separerar flera kommandon på en rad

I övriga PROM kan man t.ex. ha CD Tiny BASIC..., floppy-rutiner etc.

PRESS-STOPP! Pascal-manualerna klara. Skicka 50:- + namn,adress till Pg 27 01 15 - 9 "Chalmers Datorförening" så kommer den på posten. De som ej beställt tidigare får oxo så långt lagret räcker. /BA

"To do is to be" -Sokrates
"To be is to do" -Sartre
"Do be do be do" -Sinatra



PRESS-STOPP!
BYTE:s Z-80-assembler
finns i lokalen/Björn A.

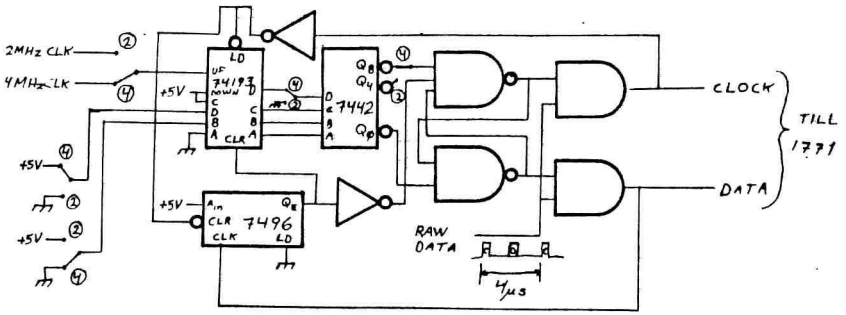
Hårda varupaket

DATASEPARERING I ETT FLOPPY DISK INTERFACE

ATT ANSLUTA EN FLOPPY DISK DRIVE TILL ETT MIKRODATORSYSTEM GÖR MAN IDAG ENKLAST MED EN FLOPPY CONTROLLER KRETS. VI VALDE WESTERN DIGITAL 1771 (TILLVERKAS ÄVEN AV NATIONAL) EN VÄL-BEPRÖVAD KRETS , KANSKE DEN VANLIGASTE. MEN ATT ANSLUTA DEN TILL EN SHUGART 800 DRIVE (SOFT SECTORING) VISADE SIG INTE VARA SÅ ENKELT SOM VI TROTT FRÅN BÖRJAN. ALLTING VERKADE FUNGERA NORMALT SÅ NÄR SOM SEPARERINGEN AV DATA OCH KLOCKPULSER SOM LIGGER BLANDADE PÅ SKIVAN.

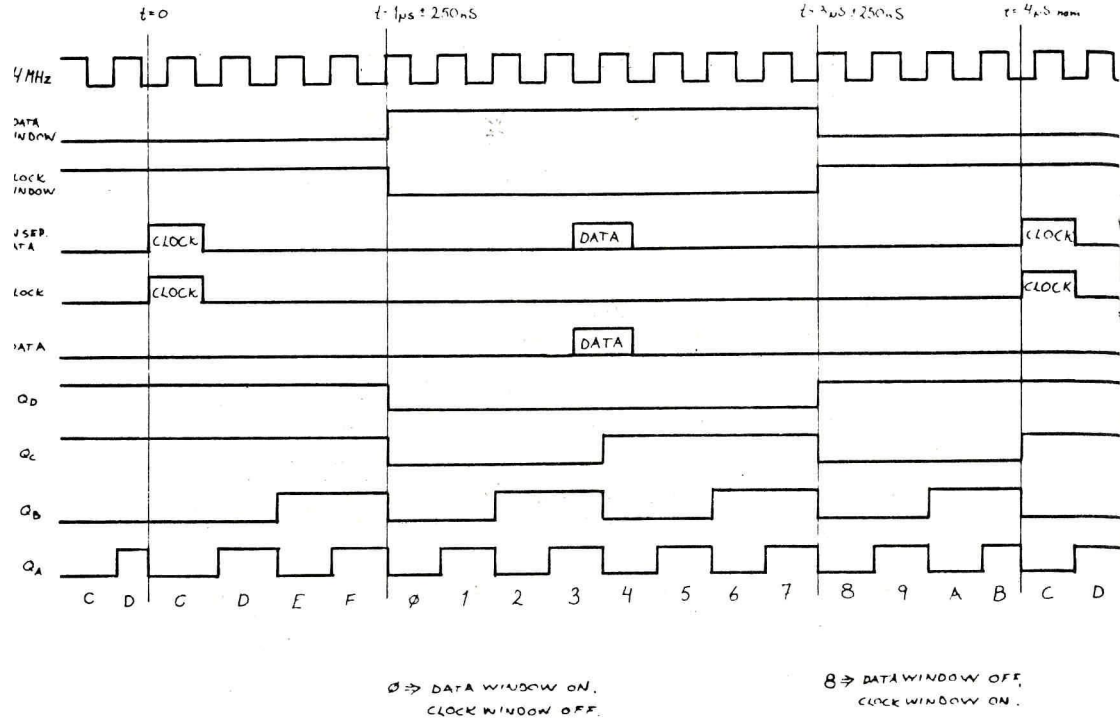
I VÅRAN FLOPPY DRIVE FANNS EN SEPARATOR INBYGGD MEN MED HJÄLP AV DEN LYCKADES VI INTE ATT LÄSA NÅGOT FRÅN SKIVAN ÖVER HUVUD TAGET. TROTS ATT ALLA SPECIFIKATIONER ENLIGT DATA-BLADEN VAR UPPFYLLDA SÅ VITT VI KUNDE MÄTA PÅ ETT OSCILLOSKOP. 1771:AN HAR EN INBYGGD DATASEPARATOR SOM MAN KAN ANVÄNDA, MEN SOM DET MYCKET RIKTIGT PÅPEKAS I DATA BLADEN BÖR MAN INTE ANVÄNDA DEN MER ÄN I NÖDFALL. VI FICK MÅNGA FEL LÄSNINGAR MED DEN SEPARATORN. DET ÄNDA SOM ÅTERSTOD VAR ATT BYGGA EN EGEN DATASEPARATOR.

I ETT TILLÄGGSBLAD TILL DATABLADET FÖR KRETSEN KALLAT "APPLICATION NOTE" FANN VI EN VETTIG LÖSNING MEN DEN VAR TÄNKT ATT ANVÄNDAS I ETT 4MHZ SYSTEM VILKET TYVÄRR INTE PASSADE OSS. VI KONSTRUERADE OM DET HELA TILL 2MHZ OCH DEN FUNGERADE SEDAN UTMÄRKT. VI VISAR HÄR BÅDE 2MHZ OCH 4MHZ SEPARATORERNA.



④ ⇒ KOPPLAT FÖR 4MHz KLOCKA (INRITAT FÖR 4MHz)

② ⇒ —"— —"— 2MHz —"—



KORT BESKRIVNING AV DATA SEPARATORNS FUNKTIONSSÄTT

I. KOPPLINGEN LIGGER I RÄTT FAS.

OM VI FÖRUTSÄTTER ATT KOPPLINGEN LIGGER I RÄTT FAS, DVS DÅ KLOCKPULSERNA FINNS PÅ CLOCK-UTGÅNGEN OCH DATA PULSERNA FINNS PÅ DATA-UTGÅNGEN SÅ KOMMER EN KLOCKPULS DELS ATT NOLLSTÄLLA ALLA BITAR I SKIFTREGISTRET OCH DELS LADDA RÄKNAREN MED EN KONSTANT (2MHZ => 6 HEX, 4MHZ => C HEX). VARJE DATA PULS KOMMER ATT KLOCKA SKIFTREGISTRET SÅ ATT EN ETTA SKIFTAS ETT STEG.

II. KRETSEN LIGGER UR FAS.

OM KRETSEN LIGGER UR FAS SÅ KOMMER KLOCKPULSERNA UT PÅ DATA-UTGÅNGEN OCH DATAPULSERNA UT PÅ CLOCK-UTGÅNGEN.
OM EN BYTE BESTÅENDE AV ENDAST NOLLOR LÄSES FRÅN SKIVAN SÅ KOMMER ÅTTA SKIFTPULSER TILL SKIFTREGISTRET, UTAN ATT NÅGON PULS HAR LADDAT RÄKNAREN ELLER NOLLSTÄLLT SKIFTREGISTRET. REDAN EFTER FEM SKIFTPULSER KOMMER EN ETTA IGENOM REGISTRET VILKET NOLLSTÄLLER RÄKNAREN OCH STÄLLER VIPPAN I ETT LÄGE SÅ ATT NÄSTA PULS HAMNAR PÅ CLOCK-UTGÅNGEN. DÅ LIGGER VI I FAS IGEN.
SUMMAN AV HELA BALETTEN BLIR ATT KRETSEN SÄTTER SIG SJÄLV I FAS SÅ FORT FEM NOLLOR I FÖLJD LÄSES FRÅN SKIVAN. DET FINNS FLERA 00-BYTES INNAN VARJE SECTOR SÅ KAN MAN VARA SÄKER PÅ ATT SEPARATOR LIGGER I FAS.

/A

MATS FREDRIKSSON

Build a *FAST* Cassette Interface

ÄR DU INTRESSERAD AV ETT BILLIGT ,SNABBT (FÖR ATT VARA KASSETT) OCH PÅLITLIGT SÄTT ATT LAGRA DATA OCH PROGRAM ? DÅ KAN DU LÄSA OM EN METOD HÄR .ARTIKELN ÄR KOPIERAD UR "BEST OF BYTE - VOL. 1 ". VI HAR ANVÄNT INTERFACET I TVÅ ÅRS TID , OCH HAR ENDAST FÅTT ETT LÄSFEL VAR. INTERFACET GÅR MED 1100 BAUDS FART . VI HAR SNÅLAT IN PÅ EN UART-KRETS GENOM ATT ANVÄNDA OSS AV EN SOFTVARUMÄSSIG UART . PROGRAMMET FINNS I ARTIKELN.

MAN KAN EVENTUELLT ANVÄNDA INTERFACET SOM ENKELT MODEM . VI HAR PROVAT DENNA IDE' MED 600 BAUDS FART. VID TESTNINGEN SÅ KOPPLADE VI AKUSTISKT GENOM ATT HÅLLA TELEFONMICKEN MOT EN HÖRLUR OCH I ANDRA ÄNDAN AV TELEFONLEDNINGEN SÅ HÖLLS BANDSPELARMICKEN MOT TELEFONLUREN .TROTS DENNA HÖGST OSTABILA KOPPLING SÅ ÖVERFÖRDES TVÅ LÄNGRE PROGRAM HELT FELFRITT.

MATS FREDRIKSSON (OBS TELEFONNUMRET ÄR 08/357306 ,FELTRYCK I SP-1-79)
PER GIBSON

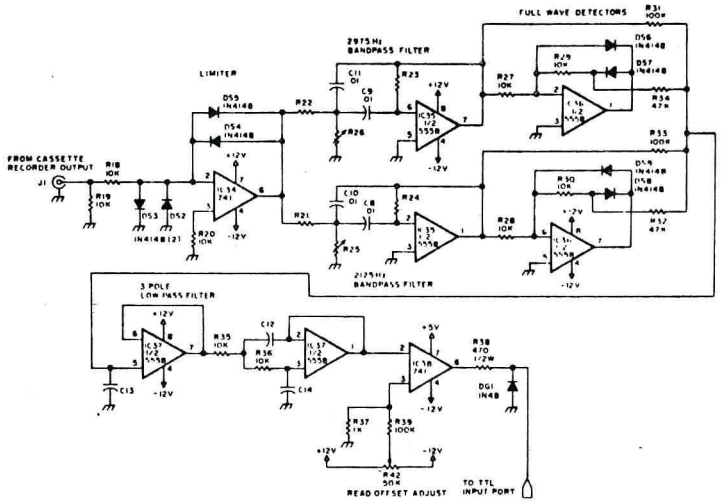


Figure 1: The schematic of the Suding cassette input interface as found in the Digital Group systems. This interface amplifies and clips the cassette output with limiting amplifier IC34, discriminates the two data frequencies (see table 1) with bandpass filters followed by full wave detectors, passes the detected signal through a 3 pole active low pass filter, then converts the result to a TTL level which is read by a single bit input port. One example of software (see listing 1) to drive this input interface uses a programmed simulation of UART input algorithm; an actual UART or ACIA device could be substituted if desired.

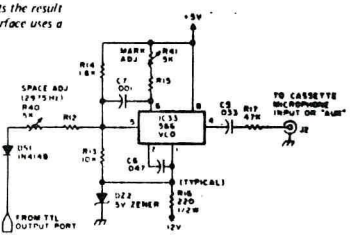


Figure 2: The schematic of the Suding cassette output interface as found in the Digital Group systems. The output interface is a simple audio frequency shift keyer made up of a 566 voltage controlled oscillator with two frequency states controlled by a single TTL data line. The TTL level which drives the output modulator is a single bit derived from an output port. The software (see listing 2) to drive this output interface is shown as a programmed simulation of a UART output algorithm; an actual UART or ACIA device could be substituted if desired.

ANNONSSIDAN

Insända bidrag från enskilda medlemmar till denna sida införes utan kostnad. För regelrätt annonsering-kontakta redaktionen.

SÄLJES

1 st MATRON VIDEORAM MTX 1632E
16 rader à 32 tecken i 7*9 matris, små och stora bokstäver + grekiska alfabetet.

Nypris 1400:- säljes för 700:-

ASCII-tangentbord CP CLARE
i vacker blå metallåda (pultform) med sladd och kontakt.
säljes för 200:-

Björn Ahlén
Råsundav. 125
171 30 SOLNA
Tel 08/83 45 41 hem
08/82 04 00 jobb

SÄLJES

EN NÄSTAN NY

ABC80

RING LASSE 0758-11164

Experimentkort: Ett alldeles utmärkt virkort för framtagning av prototyper i Europakortformat har framtagits an Nanoprodukter. Till kortet hören fullständig dokumentation för beskrivning av konstruktionen. Se närmare anslag i klubbens lokaler.
Pris ca 60:- Anteckna er i samköpspärmen.

KORTA NOTISER

Ny litteratur: Programmering i PASCAL av Anders Haraldsson Studentlitteratur, Pris ca 40:-, Finns på Kompendieförmedlingen, Kåren, Teknis

Ny programvara: Manual till ALGOL 68 (!!!) för Z-80 efterskänd. Presenteras av Jan Rynning i nästa STACK=>.

Efterlysning: Följande böcker har fått vingar och lämnat klubbens lokaler:
1 ex av Programmering i PASCAL av A Haraldsson
1 ex av Technical Aspects of Data Communication av McNamara
Alla klubbens medlemmar hälsar flyttfåglarna välkomna tillbaka snarast.

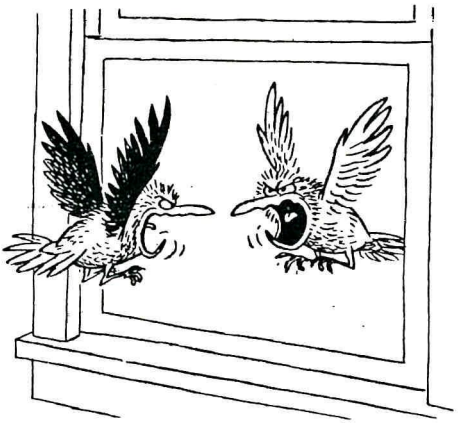
Öra: Det finns i dag en s k VOICE RECOGNITION EQUIPMENT, som kan känna igen 32 olika talade ord. Jämför dock med en indisk arbetselefant, som kan känna igen 500 (!!!) talade ord. Some way to go ...

File: **INSENDAR.TXT**

Vad ska vi anvanda STACKENs futtiga kassa till? Jag tycker en viktig post ar prenumerationer pa diverse tidskrifer. Det sager sig sjalvt att det ar billigare med EN prenumeration an om varje medlem ska prenumerera. Atminstone BYTE, Dr.Dobbs Journal, Kilobaud, Interface och Creative Computing borde finnas i vart bibliotek. (Magazinotek?)
Finns det fler bra tidnigar som ar varda att satsa pa? Ar nagon av de ovanstaende dalig? Om ingen protesterar, tycker jag isafall att vi ger var hog arade kassor uppdraget att snarast ordna med prenumerationerna.

PL/TMPSA

PS Ursakta, men jag har inte hunnit branna om prommarna i printern annu, sa a,a,o finns inte.



För att i någon mån underlätta redaktionens arbete, bör insänt artikelmaterial vara maskinskrivet och i stående A4-format (alltså ej förminskat, det fixar vi själva). Lämna gärna plats för figurer o.dyl.

OBS! Du som har upptäckt fel angivet telefonnummer i förra numret av STACKPOINTER alternativt har flyttat, skriv ner din rätta adress samt telefonnummer på ett postkort för flyttningsanmälan (Dylika finns portofria på Posten!) och skicka till STACKEN.

Då det finns ett starkt intresse för arbetsområdet datorgrafik inom klubbarna STACKEN och Chalmers Datorförening (CD) så skall jag här i korthet presentera arbetsområdet.

Datorgrafik omfattar all information som inte är alfanumerisk. De mest generella metoderna för generering av datorgrafik är DOT-GRAFIK och VEKTOR-GRAFIK.
Dotgrafik lämpar sig väl för presentation på vanlig TV-skärm, medan vektorgrafik lämpar sig bäst för X-Y rör.
I dotgrafik finns all information tillgänglig "statiskt", medan man i vektorgrafik genererar data dynamiskt.
Det senare begränsar den mängd information som kan presenteras medan så icke är fallet för dotgrafik.

Jag föreslår att vi standardiserar följande format:
256x256 och 512x512 bildpunkter.
De kvadratiska formaten är lämpligast då både linjäritet och upplösning faller starkt åt sidorna på en normal bildskärm.
Med 16k dynamiska minnen så räcker 4 resp. 16 kapslar för binär information med 16 gråskalenivåer alt 16 färger får antalen multipliceras med 4. Här uppstår ett visst utrymme för samköp.

Det svåra är inte att åstadkomma ett bra bildminne med hyfsad skarp TV-bild utan den nödvändiga PROGRAMVARAN som behövs för att generera bilden.

Jag föreslår att vi standardiserar en TURTLE-GRAFIK med TEKTRONIX-standard för överföring av bild-information.
Jag lämnar nu den mest generella nivån och övergår till näst mest generella datorgrafik -den SEMIGRAFISKA:

Den semigrafiska grafiken bygger i allmänhet på det ordinarie minnet för alfanumerisk information i en TV-terminal.
Man låter de sex bitarna i ASCII-koden sätta en till sex rutor i en 2x3-matris. Åtskilligt skojigt går att åstadkomma i denna väg och billigare alternativ finns inte.
Jag föreslår att CD-bildskärmen får bilda standard här.

Den absolut lägsta nivån på grafik är den för ett och endast ett ändamål tillskapadegrafiken där varje symbol är ett oförändeligt tecken. t.ex. ett schackbräde med tillhörande pjäser.
En grafik som bara kan rita kopplingsscheman eller logikscheman.
Dessa senare grafiker blir programmässigt väldigt enkla och effektiva men kräver istället utveckling av speciell hårdvara.

Avslutningsvis hoppas jag att detta skall bli ett upprop så att alla intresserade får lära känna varandras respektive speciella intresseområde. Schack t.ex. Fyll i talongen på näst sista sidan i STACKPOINTER och skicka till redaktörn, så kommer det in en lista i nästa SP. Jag hoppas att DATORGRAFIK skall bli en stående rubrik i alla kommande STACKPOINTER -så fatta pennan, broder (och dyra syster)!!

Hälsn LARS B:

Datorgrafik

Namn:

Adress:

Telefon:

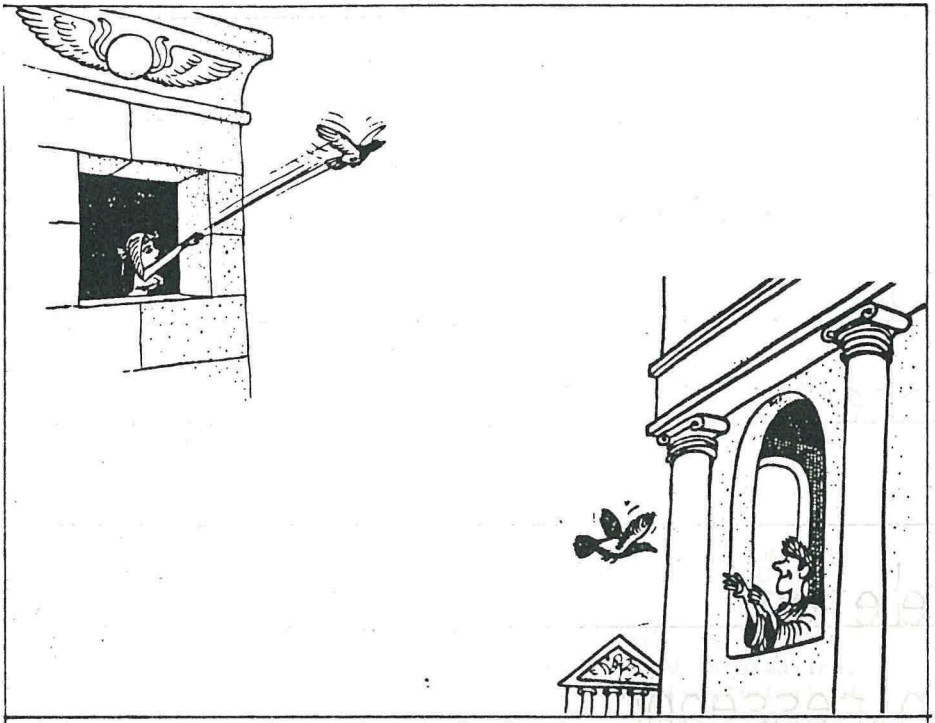
Intresseomr:

O Dotgrafik

O Semigrafik

O Spec.grafik

Anm:



Mikrodatorföreningen Stacken
Box 5079
141 05 Huddinge