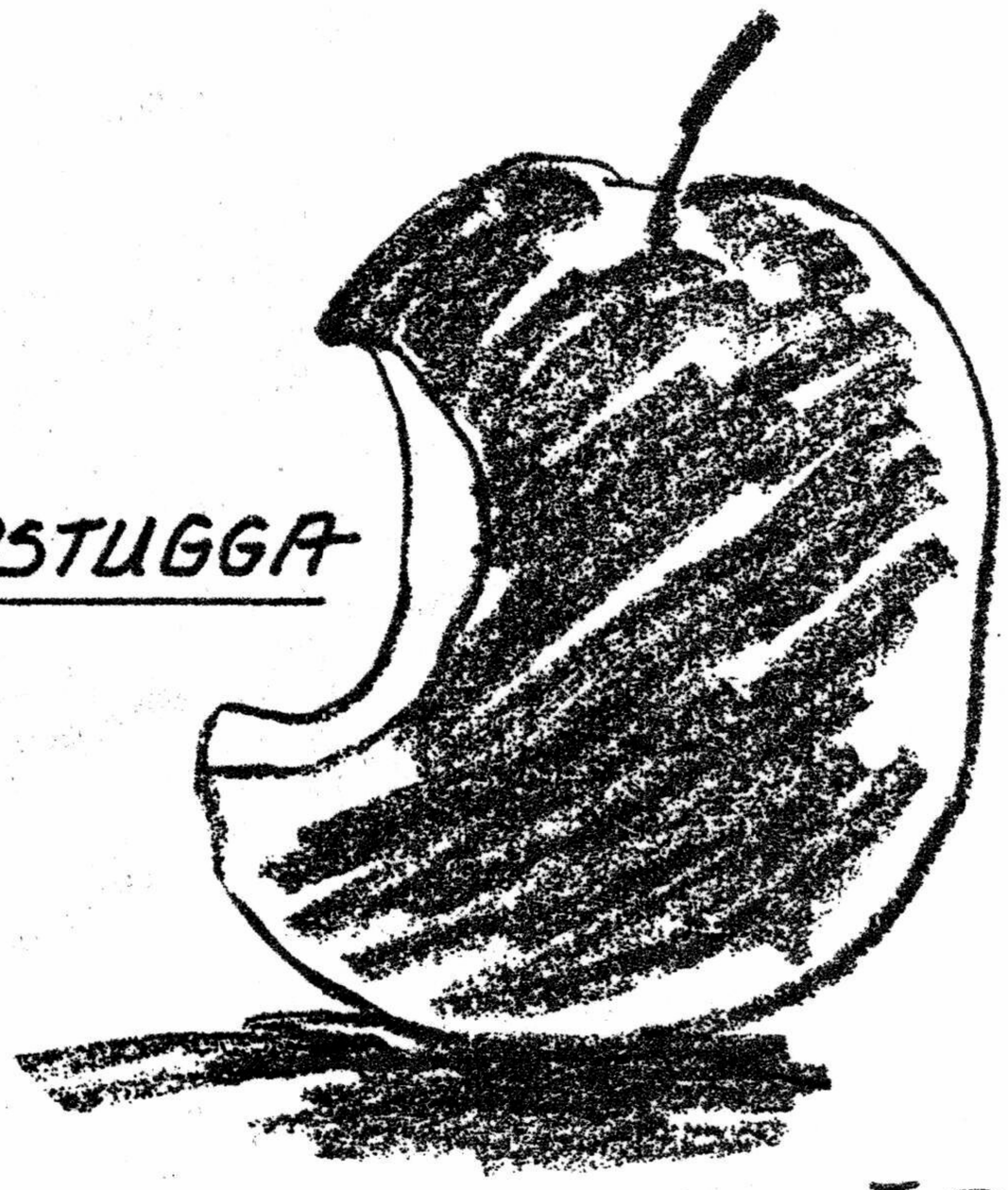
MPU-laren

TIDSKRIFT FÖR PRIVATDATAKLUBBEN PD68



EN KUNSKAPSTUGGA I DATAVÄRLDEN!

Nr 2 1989
ÅRGÅNG 12

Privatdataklubben PD68

Box 1098
122 21 ENSKEDE
Pg 96 04 68 - 7

STYRELSE & REDAKTION

Funktion	Namn	Adress	Postadress	Telefon
Ordf	Bo Sandholm	Lingvägen 217	123 59 FARSTA	08-93 32 66
V ordf	Göran Johansson	Östbergahöjden 51	125 37 ÄLVSJÖ	08-81 81 37
Kassör	Lars Wester	Isjaktsgränd 24	126 60 HÄGERSTEN	08-46 54 16
Sekr	Ulf Holm	Lysviksgatan 7	123 42 FARSTA	08-713 98 74
V sekr	Lars Lundström	Fasanstigen 73	144 00 RÖNNINGE	0753-539 24
Suppl	P O Ryeng	Vilhelmsrovägen 1	144 00 RÖNNINGE	0753-548 38
-"-	Leif Hansson	Svampstigen 26	144 00 RÖNNINGE	0753-529 39
-"-	Lars Romare	Vintertullstorget 10	116 43 STOCKHOLM	08-41 82 19

Klubbdatorn

Klubbdatorn är en AT med 80 Mbyte hårddisk.

För kontakt med maskinen finns två modem anslutna.

Modem 1 är ett modem för trafik med
300 och 75/1200 baud samt 1200 och 2400 full duplex telefon 0753-503 48.

Modem 2 är ett AT-modem med 300 och 75/1200 baud: telefon 0753-503 88.

Nya medlemmar

Du som vill bli medlem i PD68 går till väga enligt följande:
Sätt in 150.- på klubbens postgironummer 96 04 68 - 7. Meddela samtidigt
ditt namn, adress och telefonnummer.

Annonser

Helsida	500.-	Priserna gäller tryckfärdigt material.
Halvsida	300.-	Medlemmar får sätta in privata rad-annonser gratis.
Kvartssida	200.-	Bilaga enligt överenskommelse.

Privatdataklubben PD68
Box 1098
122 21 ENSKEDE
Pg 96 04 68 - 7

Innehåll :

Sekreteraren har ordet.

När detta skrivs lackar det mot jul. Vi vill med denna magra tidning ändå få ut en julhälsning till er ute i stugorna.

GOD JUL på er.

För övrigt är det illa ställt med medlemmarnas intresse för klubbverksamhet. Vi i styrelsen har diskuterat ett byggprojekt med en processor som har beteckningen 68HC11. Bygget är tänkt som ett modulsystem. Det betyder att du kan bygga den där hus-styr-datorn som du aldrig byggde. Se förklaring på annan plats i tidningen.

Det ena modemet på klubbdatorn har bytts mot ett AT-modem för att bättre passa dem som kör med detta modem. Klubbdatorn innehåller program för FLEX, IDRIS, PC samt snart även OS9.

Många av medlemmarna verkar ha gått över till PC-världen. PC-datorerna är trots allt bra som verktyg för olika applikationer (jag tänker faktiskt inte på bojstenar när jag skriver detta). Man kan faktiskt göra en hel del skojigt och nyttigt såsom.....??????????? Vi hörs.

BRINNANDE INTRESSE.

Då våra kära medlemmar visar ett så brinnande intresse för klubbverksamheten i PD68 kan det kanske vara på plats med lite goda råd så här inför julen.

När ni till exempel skall smälta stearin för att stöpa jul-ljus så är det lämpligast att placera kärlet med stearinmassan i en kastrull med vatten. På så sätt riskerar ni inte att få så hög temperatur på stearinet att det tar eld.
Goda råd är ju till för att inte följas tycker vissa besserwisser's som naturligtvis lyckas med bravaden att få brand i stearingrytan. Denna brand släcker vår käre wisser genom att hälla vatten i stearingrytan. Tror han. Vad som händer är att det uppstår en ångexplosion, eftersom vattnet när det övergår i ånga utvidgar sig med en faktor 1700. Till och med en wisser torde nu begripa att all denna ånga inte får plats i kastrullen. Resultatet blir att kastrullens innehåll kastas ur densamma och träffar de övriga familjemedlemmarna i ansiktet. Det blir inte någon GOD JUL.
Täck i stället över kastrullen med ett lock vilket ni naturligtvis har till hands.

Sedan kommer turen till alla fina ljusstakar.

Ljusstakarna skall vara av obrännbart material såsom glas, sten och metall.
Ljusmanschetterna och mossan i adventsstaken är ofta av olämpligt material. Lämna aldrig levande ljus utan tillsyn.

Säljes: CTC 40 MB Winchester 8" med
DTC kontroller.
Pris: 1200:-

Hans Persson
Ringduvegatan 115
724 70 VÄSTERÅS

GENERELL TERMINALRUTIN

Efter att ha knåpat ihop Microemacs och fått den att fungera i OS9 kände jag mig rätt nöjd med mig själv och satte mig ner och småhackade korta program (20-raders, de enda som fungerar vid första försöket) ända till den dag jag blev tvungen att köra datorn från en terminal. Självfallet hade terminalen helt andra styrkoder för skärmen än min dator (jag har ingen aning om vad min dator emulerar, men terminalen var en VT100). Efter envist plöjande i all outnyttjad källkod till emacs fann jag en fil som hette tcap.c och som i sin tur skulle använda några rutiner i ett paket som hette termcap. Någon sådan hade jag inte, men skam den som ger sig. Jag luskade ut vad emacs ville ha av termcap och skrev ett program som utifrån en fil skickade rätt saker till emacs. Resultatet ser ni i listan "nedan".

Rutinerna är mycket generellt och kan användas direkt till nästan alla terminaler. Med små ändringar kan man göra om rutinerna till en generell printerrutin, som läser olika styrkoder till olika skrivare. Problemen är att jag inte vet om detta är "rätt" (spelar mindre roll) och att jag inte vet vilka olika styrkoder som ingår i "riktiga" termcap. De få som används i emacs visar bara i princip hur det ska gå till. Dessa koder har jag hittat:

li<num>	- anger antal textrader på terminalen
co<num>	- anger antal kolumner på terminalen
cl<str>	- clear to end of page
ce<str>	- clear to end of line
cm<str>	- cursor move
se<str>	- set normal video
so<str>	- set reverse video

Finns det någon som känner till fler koder? Skicka isåfall ett brev till MPUlaren, och gärna ett till mig också (det går snabbare). I annat fall kan vi väl definiera en "PD68-standard" för termcap med (t ex) dessa koder:

in<str>	- string to initialize terminal
ex<str>	- string to close terminal
hp<str>	- home and page clear
il<str>	- insert line
dl<str>	- delete line
ic<str>	- insert char at cursor position
dc<str>	- delete char at cursor position
sd<str>	- set dim video
sf<str>	- set flashing video
su<str>	- set underline video
mu<str>	- move up one line
md<str>	- move down one line
ml<str>	- move left one character (non-destructive)
mr<str>	- move right one character (non-destructive)
f1<str>	- set foreground color 1
:	
f9<str>	- set foreground color 9
b1<str>	- set background color 1
:	
b9<str>	- set background color 9

En fråga till: Hur gör vi med tangentbordet? Ska vi definiera piltangenter och dylikt i samma fil och göra en inmatningsrutin som översätter dessa sekvenser (vissa terminaler sprutar ur sig en mindre

uppsats när man trycker på en tangent) till (t ex) negativa tal, där varje tal nger en speciell funktion (t ex "pil upp") eller har ni några andra förslag?

Jag har tänkt sätta ihop ett paket med funktioner som använder termcap och emulerar de funktioner som terminalen själv inte har, så att t ex "clear to end of line" skriver över slutet av raden med mellanslag och "delete line" skriver om skärmen från nuvarande rad och nedåt, men det projektet ligger långt in i framtiden (det finns alldeles för många som vill att jag ska göra alldeles för mycket...). Sedan ska jag göra liknande paket för skrivare, plottrar, ...(pust!)

Tyvärr har jag inget modem, så ni som vill ha programmet får nog skriva av det ur tidningen.

Till sist kommer ett exempel på hur termcap används i ett program:

```
§define TCAPSLEN 315
char tcapbufÄTCAPSLENÅ;
char *UP, PCc, *CM, *CE, *CL, *SO, *SE;

tcapopen()

ä
        char *getenv();
        char *p, *tgetstr();
        char tcbufÄ1024Å;
        char *tv_stype;
        char err_strÄ72Å;

        if ((tv_stype = getenv("TERM")) == NULL)
        ä
                puts("Environment variable TERM not defined!");
                exit(1);
        å

        if ((tgetent(tcbuf, tv_stype)) != 1)
        ä
                sprintf(err_str, "Unknown terminal type %s!", tv_stype);
                puts(err_str);
                exit(1);
        å


      if ((t_nrow=tgetnum("li")-1) == -1)ä
                puts("termcap entry incomplete (lines)");
                exit(1);
      å

      if ((t_ncol=tgetnum("co")) == -1)ä
                puts("Termcap entry incomplete (columns)");
                exit(1);
      å

        p = tcapbuf;

        CL = tgetstr("cl", &p);
        CM = tgetstr("cm", &p);
        CE = tgetstr("ce", &p);
        SE = tgetstr("se", &p);
```

```
SO = tgetstr("so", &p);
if (SO != NULL)
        revexist = TRUE;

if(CL == NULL öö CM == NULL)
ä
        puts("Incomplete termcap entryÖn");
        exit(1);
å

if (CE == NULL)          /* will we be able to use clear to EOL? */
        eolexist = FALSE;

if (p >= &tcapbufÄTCAPSLENÅ)
        ä
                puts("Terminal description too big!Ön");
                exit(1);
        å
        ttopen();       /* other initializations (operative system) */
å

tcapmove(row, col)
register int row, col;
ä
        tputs(tgoto(CM, col, row), 1, ttputc);
å

tcapeeol()
ä
        tputs(CE, 1, ttputc);
å

ttputc(c);   /* cannot use putchar directly, it is a macro! */
int c;
ä
        putchar(c);
å
```

Om ni har synpunkter direkt till mig så finns jag här:

Jens Hansson
Dag Hammarskjöjds väg 1D
22364 Lund
tel 046-112295 (sällan hemma)

```
/*
 *
 *  termcap -- a wild guess on how the "real" termcap works...
 *
 * termcap is a set of routines that handle the strings used by a terminal
 * to perform different actions like cursor positioning and blanking the
 * screen. It also handles numerical parameters like screen size. The
 * package includes the following routines:
 *
 * int tgetent(buf, type)   Reads information about specified terminal
 * char *buf;               type into space pointed to by buf. See below
 * char *type;              for the file characteristics. Returns 1 on
 *                  success and 0 if it fails.
 *
 * int tgetnum(code)        Finds the numeric value of the two-character
 * char *descr;             parameter code. Returns -1 on failure.
 *
 * char *tgetstr(code, p)   Reads the string value of the two-character
 * char *descr;             parameter code and updates *p to the next
 * char **p;                free byte after the string. Returns *p on
 *                  success and NULL if it fails. It is used to
 *                  pack several strings in an array of chars
 *                  with individual pointers to the strings.
 *
 * char *tgoto(descr, col, row) Interprets the string descr and builds the
 * char *descr;             sequence to position cursor at specified
 * int col, row;            row and column. Returns a pointer to that
 *                  sequence. Upper left corner = (0, 0)
 *
 * tputs(str, n, tputc)     Outputs string str n times thru routine
 * char *str;               tputc. This routine must be used to output
 * int n;               control-sequences, while they are not stored
 * int (*tputc)();          as normal null-terminated C strings. The
 *                  special "raw" output tputc is used because
 *                  many operating systems filters control-
 *                  characters.
 *
 * The terminal descriptions must be located in the file defined by TERMFILE
 * and be in the following format:
 *
 *   One line with terminal name (eg "vt100") starting in column 1
 *   one or more lines starting with some spaces/tabs followed by
 *      a two-character code (eg "cm" for cursor movement) and
 *      the parameter value for that code, decimal number or string
 *   Another line with (new) terminal name
 *   descriptions for that terminal
 *   etc...
 *
 * Values must start immediately after the parameter code (no whitespace).
 *
 * Numerical values are ordinary decimal values.
 *
 * String values consist of characters that are copied directly, and two-
 * digit hexadecimal codes, copied in as characters.
 *
 * A special case is the cursor-move string,
 * the first byte contains the offset to add to col and row
 * the following characters are the command string, where:
 *    "%x" is the character code col+offset
 *    "%y" is character code row+offset
```

```
*     "ØxØx" is col+offset printed in decimal
*     "ØyØy" is row+offset printed in decimal
*     "ØØ" is replaced with "Ø", so "ØØx" will output as "Øx"
*
* Example: a (short) description of vt100 and vt52 (each line in quotes).
*
* "vt100"
* "     li24"
* "     co80"
* "     cm$01$1bAØyØy;ØxØxH"
* "vt52"
* "     li24"
* "     co80"
* "     cm$20$1bYØyØx"
* EOF
*
*/

§define TERMFILE "/dd/sys/terminals"

§include <stdio.h>
§include <ctype.h>

static char *terminfo;

/*
 *
 * itoa  -- not found in OS9 system??!!
 *
 */

static itoa(n,s)
int n;
char *s;
ä
    int i,j,sign;

    if ((sign = n) < 0) n=-n;
    i = 0;
    do ä
        sÄi++Å = n % 10 + '0';
    å while ((n /= 10) > 0);
    if (sign < 0) sÄi++Å = '-';
    sÄiÅ = 0;
    for (j = 0, i--; j < i; j++, i--) ä
        sign = sÄiÅ;
        sÄiÅ = sÄjÅ;
        sÄjÅ = sign;
    å
å

/*
 *
 *  tgetent -- reads info on specified terminal type from file;
 *             if successful returns 1 and buffer contains terminal info
 *             otherwise returns 0
 *
 */

int tgetent(buf, type)
```

```
char *buf;
char *type;
ä
    FILE *f;
    char strÄ80Å;
    char *cp;
    int i, j, found = 0;

    terminfo = buf;
    if ((f = fopen(TERMFILE,"r")) == NULL) ä
        fprintf("termcap: cannot open /dd/sys/terminalsÖn");
        exit(1);
å
while (found == 0 && fgets(str,80,f) != NULL) ä
    for (cp = str; *cp != 'Ön'; cp++);
    *cp = 0;
    found = (strcmp(type,str) == 0);
å
if (found) ä
    while (fgets(str,80,f) != NULL && isspace(*str)) ä
        for(cp = str; isspace(*cp); cp++);
        j = 1;
        while (*cp != 'Ön' && *cp) ä
            if (*cp) ä
                if (*cp == '$' && isxdigit(*(cp+1)) &&
                isxdigit(*(cp+2))) ä
                    cp++;
                    bufÄjÅ = 0;
                    for (i = 1; i <= 2; i++) ä
                        *cp = tolower(*cp);
                        bufÄjÅ *= 16;
                        bufÄjÅ += (*cp < 'a') ?
                            *cp -'0':*cp -'a'+10;
                        cp++;
                    å
                    j++;
                å else bufÄj++Å = *cp++;
            å
        å
        *buf = j-1;
        buf += j;
    å
    *buf = 0;
    return(1);
å else return(0);
å

/*
 *
 *  findparam  -- internal function to find position of specified
 *                parameter value
 *
 */

static char *findparam(param)
char *param;
ä
    char *start;

    start = terminfo;
```

```
    while (*start) ä
        if (*param == startÄ1Å && *(param+1) == startÄ2Å) ä
            return(start);
        å
        start += startÄ0Å+1;
    å
    return(NULL);
å

/*
 *
 *  tgetnum  -- returns numerical value of specified parameter
 *              on failure: returns -1
 *
 */

int tgetnum(str)
char *str;
ä
    int i;
    char convÄ6Å;

    if ((str = findparam(str)) == NULL) return(-1);
    for (i=3; i <= strÄ0Å; i++) convÄi-3Å = strÄiÅ;
    convÄi-3Å = 0;
    return(atoi(conv));
å

/*
 *
 *  tgetstr  -- copies parameter string to specified address and updates
 *              pointer to next free address. Returns parameter address.
 *
 */

char *tgetstr(str,p)
char *str;
char **p;
ä
    char *t;
    int i;

    t = *p;
    if ((str = findparam(str)) == NULL) return(NULL);
    for (i = 1; i <= strÄ0Å - 2; i++) (*p)ÄiÅ = strÄi + 2Å;
    **p = i-1;
    *p += i;
    return(t);
å

/*
 *
 *  tgoto  -- returns string to position cursor
 *
 */

char *tgoto(descr,col,row)
char *descr;
int col,row;
ä
```

```
int ofs = descrÄ1Å;
 static char bufÄ80Å;
 char *p = buf + 1;
 int i = 2;

 while(i <= descrÄ0Å) ä
     if (i <= descrÄ0Å-1 && descrÄiÅ == 'ÖÖ' && descrÄi+1Å == 'x') ä
         i += 2;
         if (i <= descrÄ0Å-1 && descrÄiÅ == 'ÖÖ' &&
                 descrÄi+1Å == 'x') ä
             i += 2;
             itoa(col+ofs,p);
             while (*++p);
         å else *p++ = col+ofs;
     å else
     if (i <= descrÄ0Å-1 && descrÄiÅ == 'ÖÖ' && descrÄi+1Å == 'y') ä
         i += 2;
         if (i <= descrÄ0Å-1 && descrÄiÅ == 'ÖÖ' &&
                 descrÄi+1Å == 'y') ä
             i += 2;
             itoa(row+ofs,p);
             while (*++p);
         å else *p++ = row+ofs;
     å else
     if (descrÄiÅ == 'ÖÖ') ä
         i++;
         *p++ = descrÄi++Å;
     å else *p++ = descrÄi++Å;
å
bufÄ0Å = p - buf - 1;
return(buf);
å

/*
 *
 *  tputs  -- prints string to terminal specified number of times
 *
 */

tputs(str,n,tputc)
char *str;
int n;
int (*tputc)();
ä
    int i;

    while (n-- > 0)
        for (i = 1; i <= strÄ0Å; i++) (*tputc)(strÄiÅ);
å
```

Privatdataklubben PD68
Box 1098
122 21 ENSKEDE
Pg 96 04 68 - 7

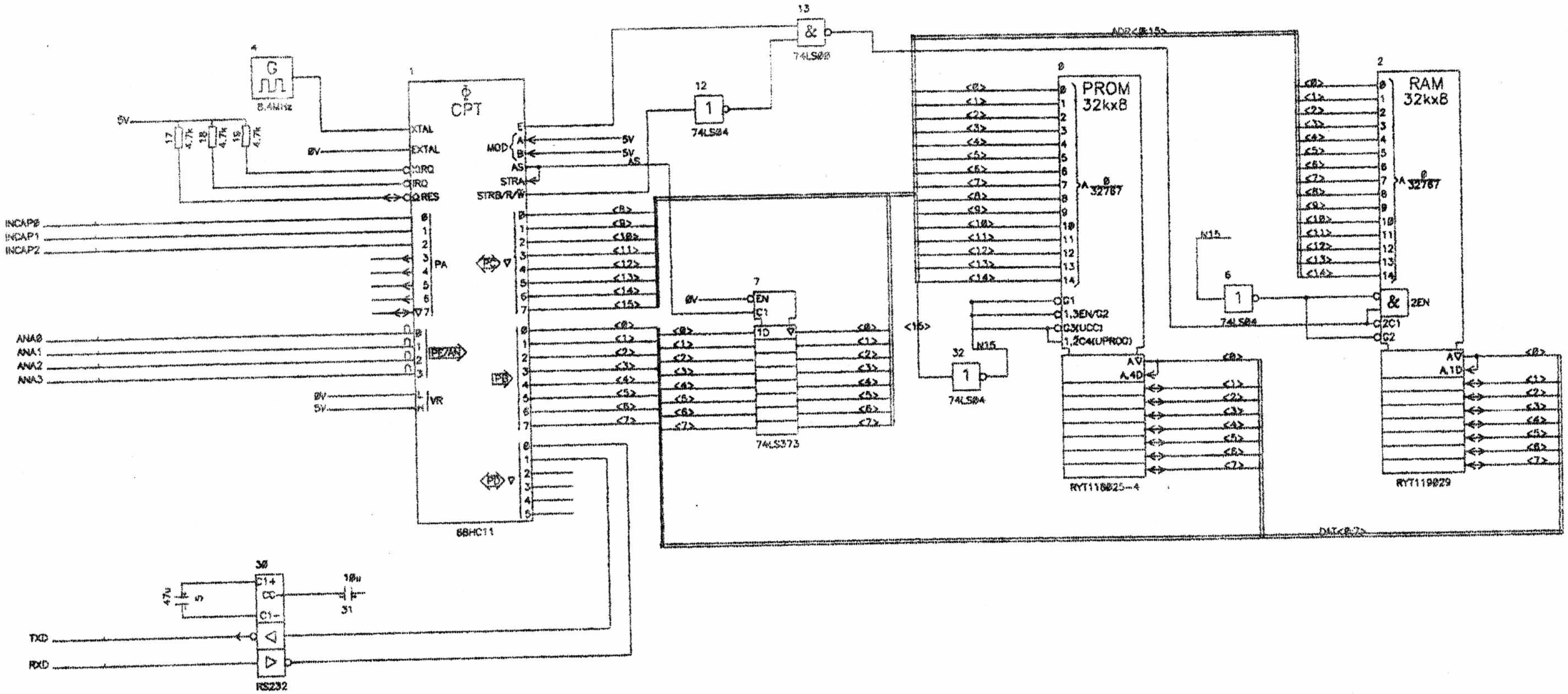
Vi tänker försöka starta ett projekt med en enkortsdator.

Denna är uppbyggd kring Motorolas processor 68HC11. Kortet är tänkt att bestyckas med 32k Prom och 32k Ram. Dessutom en MAX 232 för att ansluta till en terminal.

Vi har fått tag på en public domain monitor motsvarande CBUG med inbyggd radassembler den kallas för BUFFALO. Dessutom har vi fått tag på en OS-kärna som körs i realtid som kallas MCX11.

Datorn har 4 analoga ingångar. Dessa kan användas för temeraturmätningar eller liknande.

Detta kort kan kanske bli början till en husdator.



CPT
68HC11
8.4MHz
PROM
32kx8
RAM
32kx8
74LS373
74LS04
74LS00
RYT118025-4
RYT119029
RS232
TXD
RXD
INCAP0
INCAP1
INCAP2
ANA0
ANA1
ANA2
ANA3
ADR<0:15>
DAT<0:7>