1. DECLARATIONS

```
DECLARE SUB BswitchOn (unit%)
DECLARE SUB Bread (unit%, rval!, status%)
DECLARE SUB TBread (ts!, tr!, tsa!)
DECLARE SUB ShowTB ()
DECLARE SUB tempread (tsa!)
DECLARE SUB ShowTempLim ()
DECLARE SUB ShowTemp ()
DECLARE SUB ShowHubeLim ()
DECLARE SUB Sset (unit%, sval!)
DECLARE SUB showSlim ()
DECLARE SUB limSverif (unit%, newval!, warn%)
DECLARE SUB SsetSafe (unit%, newval!)
DECLARE SUB ShowSval (unit%)
DECLARE SUB CraneDispl (light%)
DECLARE SUB ShowDia ()
DECLARE FUNCTION DiaIn% ()
DECLARE SUB ShowAtten ()
DECLARE FUNCTION Attenin% ()
DECLARE SUB ShowWavl ()
DECLARE SUB ctime ()
DECLARE SUB tsetread (ts!, tr!, tsa!)
DECLARE SUB LPnarrow ()
DECLARE SUB LPnormal ()
DECLARE SUB SaveLim ()
DECLARE FUNCTION DruStatsKeith% (unit%)
DECLARE SUB ShowTset ()
DECLARE SUB tsetset (tset!)
DECLARE SUB Tclose ()
DECLARE SUB Topen ()
DECLARE SUB ShowMonLim ()
DECLARE SUB TsetSetSafe (newval!)
DECLARE SUB ShowTsetLim ()
DECLARE SUB SelRead (rval!, status%)
DECLARE SUB SelSet (selv!)
DECLARE SUB ShowSelLim ()
DECLARE SUB ShowSel ()
DECLARE SUB SelSetSafe (newval!)
DECLARE SUB ZeroSteps ()
DECLARE SUB countSuspend ()
DECLARE SUB countResume ()
DECLARE SUB CraneShow ()
DECLARE FUNCTION CraneIn% ()
DECLARE SUB EToShell ()
DECLARE SUB DoDos (aCom$)
DECLARE FUNCTION Bstatus% (unit%)
DECLARE SUB ShowHp ()
DECLARE SUB verify (restart%)
DECLARE SUB rates (t!, m!, d!, mrate!, drate!)
DECLARE SUB counting (tpreset!, time!, mon!, count1!, count2!, count3!)
DECLARE SUB BsetSafe (unit%, newval!)
DECLARE SUB MsetSafe (unit%, newval!)
DECLARE SUB MakeSteps ()
DECLARE SUB I3eErro r (unit%)
DECLARE SUB cStat ()
DECLARE SUB HPread (hpval!)
```

```
DECLARE SUB paintscreen ()
DECLARE SUB updscreen ()
DECLARE SUB cDos ()
DECLARE SUB Mset (unit%, sval!)
DECLARE SUB Mread (unit%, rval!)
DECLARE SUB ShowBval (unit%)
DECLARE SUB ShowMVal (unit%)
DECLARE SUB ShowMLim (unit%)
DECLARE SUB ShowBLim (unit%)
DECLARE SUB limBverif (unit%, newval!, warn%)
DECLARE SUB LimMVerif (unit%, newval!, warn%)
DECLARE SUB countstat (time!, monit!, c1!, c2!, c3!, done%)
DECLARE SUB countstart (pres!, t%)
DECLARE SUB init ()
DECLARE SUB dispatch (thecom%)
DECLARE SUB verbs (thecom%)
DECLARE SUB WaitTic (tic&)
DECLARE SUB msg (mess$, dobeep%)
DECLARE SUB nxtcomm ()
DECLARE SUB getcomm ()




REM $INCLUDE: 'ieeecom.bas'

'subs Commons  and constants we might need

REM $INCLUDE: 'manual2.bi'
CALL init

FOR i% = 1 TO ncmnds
  READ cmnds(i%)
NEXT i%


DO WHILE true                       'main loop
 IF comlen% = 0 THEN CALL getcomm
 CALL nxtcomm
 CALL verbs(thecom%)

 IF dorun% THEN
   IF jobflg%(job%) THEN            ' execute command only if job needed
          IF newjob% THEN GOSUB jobheader
          CALL dispatch(thecom%)
   ELSE
          IF thecom% = 4 THEN       ' if not needed still check for C
            job% = job% + 1         ' to increment job% count
            newjob% = true          ' and we have a new job
          END IF
   END IF
 ELSE
          CALL dispatch(thecom%)    ' not RUN-ing so do it
 END IF
LOOP


jobheader:
  PRINT "Job:"; job%
  IF lpton% THEN
    LPRINT "Job:"; job%, DATE$, TIME$
  END IF
```

```
  IF dodata% THEN
    PRINT #chdat, USING "J## \          \ \        \"; job%; DATE$; TIME$
  END IF
  newjob% = false

RETURN

DATA QUIT,LP,!,COUNT,EXECUT,RUN,QRUN,BOBINE,MOTOR,LIMIT,WATCH,DOS,ZERO,STATUS
DATA ECHO,TIME,PATH,HP,HELP,SETUP,DIR,TYPE,KEDIT,RATE,CRANE,STORE
DATA SELECT,TILL,WAVLEN,DIAPHR,SAMPLE,TEMP,HUBER,NET,REGUL,ATTENU
END

'--------- error trap ---------
errtrap:
 erflg% = ERR
' PRINT " QB error number:"; erflg%
 RESUME NEXT
RETURN

'--------- F1 trap ---------
TrapF1:
hitF1% = true
CALL ZeroSteps          'hitF1 can happen before reaching a C or E
RETURN

'--------- F2 trap ---------
TrapF2:
hitF2% = true
RETURN

'--------- F5 trap ---------
TrapF5:
hitF5% = true
RETURN

'------------------------------------------------------------------------------
```

2.SUB init

```
InRoutine$ = "init"

SCREEN scmode, , compage, compage
CLS
COLOR outcolor, 0



REDIM cmnds(ncmnds) AS STRING * 6
REDIM jobflg%(jobmax)
REDIM curset(nsupl), curread(nsupl), curstep(nsupl), curlast(nsupl)
REDIM curmaxh(nsupl), curminh(nsupl), curmaxs(nsupl), curmins(nsupl)
REDIM curdev%(nsupl)
REDIM curdac(nsupl), curadc(nsupl), cJustSet%(nsupl)
REDIM angleset(nmotor), angleread(nmotor), anglestep(nmotor), anglelast(nmotor)
REDIM anglemaxh(nmotor), angleminh(nmotor), anglemaxs(nmotor), anglemins(nmotor)

' error trap on
erflg% = 0
ON ERROR GOTO errtrap

' trap F1 key
ON KEY(1) GOSUB TrapF1
KEY(1) ON

' trap F2 key
ON KEY(2) GOSUB TrapF2
KEY(2) ON
```

```
' trap F5 key
ON KEY(5) GOSUB TrapF5
KEY(5) ON


'IEEE inits
CALL IEINIT
Counter% = 5            'counter address
curdev%(5) = 12         'Brukers 5-16 Dafi interface

FOR i% = 6 TO 16
  curdev%(i%) = curdev%(5)
NEXT i%
curdev%(17) = 11         'Brukers 17-20 Dafi interface
FOR i% = 18 TO 20
  curdev%(i%) = curdev%(17)
NEXT i%
curdev%(1) = 10         'Drusch 1 in fact the Keithley 576
curdev%(2) = 1          'Now Brucker no more 10 for Drusch 2
curdev%(3) = 10         'Drusch 3  pol anal
curdev%(4) = 10         'Drusch 4  reserva
HPdev% = 14             '#1 HP DVM
Dafodev% = 9            'Dafoldil position, TTLs ...
HP2dev% = 15            '#2 HP DVM address
SelDev% = 13            'selector address
baras% = 3              ' temperature control Barras
Treadev% = 25           ' T lecture
' variable inits
i3estatus% = 0
CraneWatch% = true      'start with Crane watch on
doexe% = false
dorun% = false
dodata% = false
lpton% = false
newjob% = false
hpon% = false
neton% = true           'start with copy network Net on
Tstatus% = 0
statusT% = 0
job% = 0
erflg% = 0
hitF1% = false
hitF2% = false
hitF5% = false
tfactor = 1
wtime = 10                  ' flop to watch state when counts for t>wtime
prompt$ = "M>"

ScreenForm1$ = "##########"
ScreenForm2$ = " ######.##"
ScreenForm3$ = "####.###(####.###)"
ScreenForm4$ = "#.######^^^^"


inpch% = 0
PRINT "Reading hard limits and calibration table..."

' read in hard limits and calibration table
OPEN tablefile FOR INPUT AS #chtmp
FOR i% = 1 TO nsupl
  INPUT #chtmp, tmp$, unit%, curminh(i%), curmaxh(i%), curdac(i%), curadc(i%)
NEXT i%
FOR i% = 1 TO nmotor
  INPUT #chtmp, tmp$, unit%, angleminh(i%), anglemaxh(i%)
NEXT i%
```

```
INPUT #chtmp, tmp$, selminh, selmaxh
INPUT #chtmp, tmp$, tsetminh, tsetmaxh
INPUT #chtmp, tmp$, tempminh, tempmaxh
INPUT #chtmp, tmp$, hubeminh, hubemaxh
CLOSE chtmp

PRINT "Reading soft limits ..."
' read in soft limits
OPEN limitfile FOR INPUT AS #chtmp
FOR i% = 1 TO nsupl
  INPUT #chtmp, tmp$, unit%, curmins(i%), curmaxs(i%)
  ' use this loop to init cJustSet
  cJustSet% = false
NEXT i%
FOR i% = 1 TO nmotor
  INPUT #chtmp, tmp$, unit%, anglemins(i%), anglemaxs(i%)
NEXT i%
INPUT #chtmp, tmp$, selmins, selmaxs
INPUT #chtmp, tmp$, tsetmins, tsetmaxs
INPUT #chtmp, tmp$, tempmins, tempmaxs
INPUT #chtmp, tmp$, hubemins, hubemaxs
INPUT #chtmp, tmp$, limmonit, MrateCal
INPUT #chtmp, tmp$, Phaseunit%, PhaseStep
CLOSE chtmp

' read last parameters for barras regulation temperature
PRINT "Reading Temperature Control BARRAS :";
OPEN barrasfile FOR INPUT AS #chtmp
FOR i% = 1 TO 7
  INPUT #chtmp, Tmax
  INPUT #chtmp, TminhighT
  INPUT #chtmp, TmaxlowT
  INPUT #chtmp, numlowT%
  INPUT #chtmp, numhighT%
  INPUT #chtmp, xxx%
  INPUT #chtmp, regfile$
NEXT i%
CLOSE chtmp
PRINT regfile$

' read in next numor
PRINT "Reading numor ...";
OPEN numorfile FOR INPUT AS #chtmp
INPUT #chtmp, numor&
CLOSE chtmp
PRINT "Next datafile:"; numor&

PRINT "Initializing the Keithley 576 and Brucker B2 "

KeithInit:
'********************* reset default on Brucker B2 ********
   PRINT "Cleaning up Brucker B2 ...";
   unit% = 2
   o$ = "R1"          ' reset default if possible
   action% = 1
   CALL iewrit(cur dev%(unit%), o$)     'and try to switch on
   SLEEP (2)
   CALL ieread(curdev%(unit%), f$)
   PRINT " done ", f$

'********** init Keithley 576 **********
            PRINT "Cleaning up Keithley buffer wait ...";
            CALL ieread(curdev%(1), l$)
            PRINT "done"; l$

            CALL iewrit(curdev%(1), "syst:term cr;")
```

```
            IF i3estatus% <> 0 THEN GOSUB AskGiveUp

            CALL iewrit(curdev%(1), "syst:idn?;")
            CALL ieread(curdev%(1), l$):
            PRINT : PRINT "Keithley 576 version:"; l$: PRINT
'       INPUT x
            o$ = "chan 5,0-1 :mode in;"             'configure chan 0-1 input
            CALL iewrit(curdev%(1), o$)
            o$ = "chan 5,2 :mode out;"            'configure chan 2 output
            CALL iewrit(curdev%(1), o$)
            o$ = "chan 5,3 :mode in;"             'configure chan 3 input sensing relay
            CALL iewrit(curdev%(1), o$)

"        buffer$ = "iwrite raw,5,2,85;"          'set all Drusch ref on
"        CALL iewrit(curdev%(1), buffer$)     ' diverses modifs 10/11 94....


            o$ = "chan 3,0-1 :range 10b;"             'DAC16 +10V mode
            CALL iewrit(curdev%(1), o$)
            o$ = "chan 4,0-1 :range 1b;"              'DAC13 +/- 1V mode
            CALL iewrit(curdev%(1), o$)

            o$ = "syst:amm 2k;"                     ' filter the signal
            CALL iewrit(curdev%(1), o$)

'***************** diff-global,local gain-range **********

            FOR i% = 0 TO 7
              o$ = "chan 1," + STR$(i%) + ":mode df:gain 1,10:range 1b;"
              CALL iewrit(curdev%(1), o$)
'       PRINT o$
'       INPUT x
            NEXT i%

'******************** init supply values and read status *******
SupInit:
PRINT " Initializing Suplies"

FOR unit% = 1 TO nsupl
  CALL Bread(unit%, rval, status%)
  IF status% <> 0 THEN
    CALL msg(" Supply " + STR$(unit%) + " is off", false)
    curset(unit%) = 0
  ELSE
    curset(unit%) = rval
    CALL ShowBval(unit%)
  END IF
  curread(unit%) = rval
NEXT unit%

' init motor values

FOR unit% = 1 TO nmotor
  CALL Mread(unit%, rval)
  angleset(unit%) = rval
  angleread(unit%) = rval
NEXT unit%

' init selector value

CALL SelRead(rval, status%)
  selsetv = rval
  selreadv = rval

comline$ = "EXE INIT"
```

```
comlen% = LEN(comline$)

EXIT SUB

AskGiveUp:
        SOUND 500, 2
        INPUT "S[kip] this unit or A[bort_all] def=A :", ans$
        IF UCASE$(LEFT$(ans$, 1)) <> "S" THEN STOP

END SUB

'---------------------------------------------------------------------------
```

3.SUB IEINI2

```
' replace IEINIT . Pb of initialization IEEE devices ; ex: ATNE ATSR100
'       CALL initialize(21, 0)   'controleur avec adresse 21
' a faire a l initialisation de branchement uniquement '
' sinon cela remet en mode local l atne
        CALL dmachannel(3)       'canal de DMA Nø 3
        CALL settimeout(10000)   'time out de 10 sec
        CALL setoutputEOS(13, 0) 'sortie IEEE avec return
        CALL setinputEOS(13)     'entree IEEE avec return


END SUB

'---------------------------------------------------------------------------
```

4.SUB cCrane

```
tmp$ = cutitem$(onecom$, onelen%, " ")
SELECT CASE MID$(tmp$, 1, 2)
 CASE "ON"
   CraneWatch% = true
 CASE "OF"
   CraneWatch% = false
END SELECT
CALL CraneShow

END SUB
```

5.FUNCTION CraneIn%

```
' returns TRUE if Crane is in the dangerous zone
InRoutine$ = "CraneIn"

buffer$ = "D6"
action% = 1
CALL iewrit(Dafodev%, buffer$)
CALL I3eError(Dafodev%)
action% = 2
CALL ieread(Dafodev%, buffer$)
CALL I3eError(Dafodev%)
flags% = ASC(MID$(buffer$, 4, 1)) - 48
IF flags% > 9 THEN flags% = flags% - 7        'hexa conversion to integer

IF (flags% AND CranePattern) THEN      'if bit1 is present Crane is OK !!
  CraneIn% = false
ELSE
  CraneIn% = true
END IF
'tmp% = NOT (flags% AND CranePattern)       'if bit1 is present Crane is OK !!
'PRINT tmp%, true, (flags% AND CranePattern)
END FUNCTION
```

6.SUB CraneDispl (light%)

```
InRoutine$ = "CraneDispl"

CALL iewrit(Dafodev%, "R")
   action% = 1
```

```
   CALL I3eError(Dafodev%)
CALL ieread(Dafodev%, buffer$)
   action% = 2
```

7.SUB CraneShow

```
IF CraneIn% THEN
  tmp1$ = " IN"
ELSE
  tmp1$ = "OUT"
END IF
IF CraneWatch% THEN
  tmp2$ = " ON"
ELSE
  tmp2$ = "OFF"
END IF
buffer$ = "Crane is: " + tmp1$ + "   Crane watching is: " + tmp2$
PRINT buffer$
IF lpton% THEN LPRINT buffer$

END SUB
```

8.SUB cDia

```
InRoutine$ = "cDia"

tmp$ = cutitem$(onecom$, onelen%, " ")
SELECT CASE MID$(tmp$, 1, 2)
 CASE "IN"
   CALL iewrit(Dafodev%, "V10")
   action% = 1
   CALL I3eError(Dafodev%)
   setin% = 1
 CASE "OU"
   CALL iewrit(Dafodev%, "V08")
   action% = 2
   CALL I3eError(Dafodev%)
   setin% = 2
 CASE ELSE
   CALL ShowDia
   EXIT SUB
END SELECT
i% = 0
DO
  WaitTic (20)          'give him some time to settle
  IF setin% <> DiaIn% THEN
          i% = i% + 1
  ELSE
          i% = 12
  END IF
LOOP WHILE (i% < 12)

CALL ShowDia

END SUB

'------------------------------------------------------------------------
```

9.FUNCTION DiaIn%

```
InRoutine$ = "DiaIn"

CALL iewrit(Dafodev%, "R")
   action% = 1
   CALL I3eError(Dafodev%)
CALL ieread(Dafodev%, buffer$)
   action% = 2
   CALL I3eError(Dafodev%)

SELECT CASE MID$(buffer$, 3, 2)
 CASE "70"
```

```
    DiaIn% = 1
  CASE "B0"
    DiaIn% = 1
  CASE "D0"
    DiaIn% = 1
  CASE "68"
    DiaIn% = 2
  CASE "A8"
    DiaIn% = 2
  CASE "C8"
    DiaIn% = 2

  CASE ELSE
    DiaIn% = 0
END SELECT

END FUNCTION
```

10.SUB ShowDia

```
posi% = DiaIn%
SELECT CASE posi%
  CASE 1
    posi$ = "in"
  CASE 2
    posi$ = "out"
  CASE ELSE
    posi$ = "unknown"
    BEEP
END SELECT
form$ = " Diaph : "
PRINT form$; posi$
IF lpton% THEN
   LPRINT form$; posi$
END IF
IF dodata% THEN
            PRINT #chdat, form$; posi$
END IF

END SUB
```

11.SUB ccomment

```
CALL msg(lastcom$, false)
END SUB
```

12.SUB cDir

```
CALL DoDos("dir")
END SUB

'------------------------------------------------------------------------
```

13.SUB cDos

```
'SCREEN scmode, , stapage, stapage
SHELL onecom$
'SCREEN scmode, , compage, compage
END SUB
```

14.SUB cHelp

```
PRINT "Known commands:"
FOR i% = 1 TO ncmnds
    PRINT cmnds(i%),
NEXT i%
PRINT " See syntax on booklet "
PRINT
END SUB
```

15.SUB cKedit

```
CALL DoDos("kedit")
END SUB

'------------------------------------------------------------------------
```

16.SUB cpath

```
tmp$ = cutitem$(onecom$, onelen%, " ")
IF onelen% = 0 THEN GOTO showpath

tmp$ = Rpad$(tmp$, 4)
path$ = cutitem$(onecom$, onelen%, " ")

SELECT CASE tmp$
  CASE "DATA"
    datapath$ = path$

  CASE "SETS"
    setspath$ = path$
END SELECT

showpath:
  PRINT "Data path: "; datapath$; "  Sets path: "; setspath$

IF lpton% THEN
  LPRINT "Data path: "; datapath$; "  Sets path: "; setspath$
END IF

END SUB
```

17.SUB cType

```
CALL DoDos("type")
END SUB

'------------------------------------------------------------------------
```

18.SUB ctime

```
t$ = "The time is: " + DATE$ + " " + TIME$
PRINT t$
IF lpton% THEN LPRINT t$
END SUB
```

19.SUB cLim

```
l$ = LEFT$(onecom$, 1)
onelen% = LEN(onecom$)
IF LEN(onecom$) > 1 THEN test% = ASC(MID$(onecom$, 2, 1))
SELECT CASE l$
  CASE "B"
        IF test% > 47 AND test% < 58 THEN onecom$ = l$ + "OBI " + RIGHT$(onecom$, onelen% - 1)
  CASE "M"
        IF test% > 47 AND test% < 58 THEN onecom$ = l$ + "OTO " + RIGHT$(onecom$, onelen% - 1)
  CASE "S"
        IF test% > 47 AND test% < 58 THEN onecom$ = l$ + "AMP " + RIGHT$(onecom$, onelen% - 1)

END SELECT
onelen% = LEN(onecom$)
tmp$ = cutitem$(onecom$, onelen%, " ")

l% = imax(LEN(tmp$), 1)
l% = imin(l%, 4)

tmp$ = LEFT$(tmp$, l%)

IF tmp$ = LEFT$("BOBI", l%) THEN tmp$ = "BOBI"
```

```
IF tmp$ = LEFT$("MOTO", l%) THEN tmp$ = "MOTO"
IF tmp$ = LEFT$("MONI", l%) THEN tmp$ = "MONI"
IF tmp$ = LEFT$("TILL", l%) THEN tmp$ = "TILL"
IF tmp$ = LEFT$("SELE", l%) THEN tmp$ = "SELE"
IF tmp$ = LEFT$("SAMP", l%) THEN tmp$ = "SAMP"
IF tmp$ = LEFT$("HUBE", l%) THEN tmp$ = "HUBE"
IF tmp$ = LEFT$("TEMP", l%) THEN tmp$ = "TEMP"


SELECT CASE tmp$
 CASE "BOBI"
   GOSUB ulh
   IF unit% < 1 OR nsupl < unit% THEN
    CALL msg(" Invalid supply: " + tmp$ + STR$(unit%), true)
    EXIT SUB
   END IF
   IF haslim% THEN
     lowtmp = max(lowval, curminh(unit%))
     hightmp = min(highval, curmaxh(unit%))
     IF lowtmp <> lowval OR hightmp <> highval THEN
            CALL msg(" Out of hard limits !", true)
            haslim% = false
     ELSE
            curmins(unit%) = lowval
            curmaxs(unit%) = highval
     END IF
   END IF
   CALL ShowBLim(unit%)

 CASE "MOTO"
   GOSUB ulh
   IF unit% < 1 OR nmotor < unit% THEN
    CALL msg(" Invalid motor: " + tmp$ + STR$(unit%), true)
    EXIT SUB
   END IF
   IF haslim% THEN
     lowtmp = max(lowval, angleminh(unit%))
     hightmp = min(highval, anglemaxh(unit%))
     IF lowtmp <> lowval OR hightmp <> highval THEN
            CALL msg(" Out of hard limits !", true)
            haslim% = false
     ELSE
            anglemins(unit%) = lowval
            anglemaxs(unit%) = highval
     END IF
   END IF
   CALL ShowMLim(unit%)



CASE "MONI"
   IF onelen% > 0 THEN
    haslim% = true
    limmonit = VAL(cutitem$(onecom$, onelen%, " "))
   ELSE
    haslim% = false
    CALL ShowMonLim
   END IF

 CASE "SELE"
   GOSUB ul
   IF haslim% THEN
     lowtmp = max(lowval, selminh)
     hightmp = min(highval, selmaxh)
     IF lowtmp <> lowval OR hightmp <> highval THEN
            CALL msg(" Out of hard limits !", true)
```

```
            haslim% = false
        ELSE
            selmins = lowval
            selmaxs = highval
        END IF
    END IF
    CALL ShowSelLim

CASE "TILL"
    GOSUB ul
    IF haslim% THEN
        lowtmp = max(lowval, tsetminh)
        hightmp = min(highval, tsetmaxh)
        IF lowtmp <> lowval OR hightmp <> highval THEN
            CALL msg(" Out of hard limits !", true)
            haslim% = false
        ELSE
            tsetmins = lowval
            tsetmaxs = highval
        END IF
    END IF
    CALL ShowTsetLim

CASE "TEMP"
    GOSUB ul
    IF haslim% THEN
        lowtmp = max(lowval, tempminh)
        hightmp = min(highval, tempmaxh)
        IF lowtmp <> lowval OR hightmp <> highval THEN
            CALL msg(" Out of hard limits !", true)
            haslim% = false
        ELSE
            tempmins = lowval
            tempmaxs = highval
        END IF
    END IF
    CALL ShowTempLim

CASE "HUBE"
    GOSUB ul
    IF haslim% THEN
        lowtmp = max(lowval, hubeminh)
        hightmp = min(highval, hubemaxh)
        IF lowtmp <> lowval OR hightmp <> highval THEN
            CALL msg(" Out of hard limits !", true)
            haslim% = false
        ELSE
            hubemins = lowval
            hubemaxs = highval
        END IF
    END IF
    CALL ShowHubeLim

CASE "SAMP"
    GOSUB ul
    IF haslim% THEN
        '  moteur 3 sample position
        lowtmp = max(lowval, angleminh(3))
        hightmp = min(highval, anglemaxh(3))
        IF lowtmp <> lowval OR hightmp <> highval THEN
            CALL msg(" Out of hard limits !", true)
            haslim% = false
        ELSE
            anglemins(3) = lowval
            anglemaxs(3) = highval
        END IF
```

```
    END IF
    CALL showSlim



  CASE ELSE
    FOR unit% = 1 TO nsupl
      CALL ShowBLim(unit%)
    NEXT unit%
    FOR unit% = 1 TO nmotor
      CALL ShowMLim(unit%)
    NEXT unit%
    CALL ShowSelLim
    CALL ShowTsetLim
    CALL ShowTempLim
    CALL ShowHubeLim
    CALL ShowMonLim
    CALL showSlim

END SELECT

'    Save new limits

IF haslim% THEN CALL SaveLim

EXIT SUB

ulh:
    unit% = VAL(cutitem$(onecom$, onelen%, " "))
    GOSUB ul
    RETURN

ul:
    IF onelen% > 0 THEN
      haslim% = true
      lowval = VAL(cutitem$(onecom$, onelen%, " "))
      highval = VAL(cutitem$(onecom$, onelen%, " "))
      IF highval < lowval THEN
            CALL msg(" high limit < low limit !", true)
            EXIT SUB
      END IF
    ELSE
      haslim% = false
    END IF
    RETURN


END SUB
```

20.SUB clpset

```
tmp$ = cutitem$(onecom$, onelen%, " ")
IF LEFT$(tmp$, 2) = "ON" THEN
  lpton% = true
  WIDTH LPRINT 85
END IF
IF LEFT$(tmp$, 2) = "OF" THEN lpton% = false
END SUB
```

21.SUB cWatch

```
IF onelen% <> 0 THEN
   tmp$ = cutitem$(onecom$, onelen%, " ")
   wtime = max(VAL(tmp$), 5)
ELSE
   CALL paintscreen
   LOCATE 24, 1
   PRINT "<Hit F2 to leave>";
```

```
  hitF2% = false
  DO WHILE NOT hitF2% AND NOT hitF1%
            FOR unit% = 1 TO nsupl
              CALL Bread(unit%, curread(unit%), status%)
'         CALL WaitTic(2)
              IF status% <> 0 AND curset(unit%) <> 0 THEN
                CALL msg("B" + STR$(unit%) + " found off", true)
                CALL BswitchOn(unit%)
              END IF
            NEXT unit%
    CALL updscreen
  LOOP
  CLS
  SCREEN scmode, , compage, compage
END IF
END SUB

'---------------------------------------------------------------------------
```

22.SUB cexe

```
IF dorun% THEN
  CALL msg(" EXE from a RUN is not allowed", true)
  EXIT SUB
END IF
IF doexe% THEN          ' we allow chaining
  CLOSE chexe
  exefile$ = ""
END IF
'  we are tricky and kept doexe%=true so it will not continue
'  an exe file if it finds an occasional exe command without file spec
IF onelen% = 0 AND NOT doexe% THEN
  INPUT " EXE file:"; exefile$
ELSE
  exefile$ = cutitem$(onecom$, onelen%, " ")
END IF
doexe% = false
ppos% = INSTR(1, exefile$, ".")
IF ppos% = 0 AND exefile$ <> "" THEN exefile$ = exefile$ + ".exb"
erflg% = 0
OPEN setspath$ + exefile$ FOR INPUT AS #chexe
IF erflg% <> 0 THEN
  CALL msg(" file not found:" + setspath$ + exefile$ + " err=" + STR$(erflg%), true)
  exefile$ = ""
ELSE
  doexe% = true
END IF
erflg% = 0

restlexe$ = comline$
restlnexe% = comlen%
comline$ = ""
comlen% = 0

END SUB
```

23.SUB crun (pstat%)

```
IF dorun% THEN
  CALL msg(" no run nesting is allowed", true)
  EXIT SUB
END IF

IF onelen% = 0 AND NOT doexe% THEN      ' run file was not specified
  INPUT " run file:"; runfile$
ELSE
  runfile$ = cutitem$(onecom$, onelen%, " ")
```

```
END IF
ppos% = INSTR(1, runfile$, ".")
IF ppos% = 0 AND runfile$ <> "" THEN runfile$ = runfile$ + ".run"

erflg% = 0
OPEN setspath$ + runfile$ FOR INPUT AS #chrun
IF erflg% <> 0 THEN
  CALL msg(" file not found:" + setspath$ + runfile$ + " err=" + STR$(erflg%), true)
  runfile$ = ""
  EXIT SUB
  erflg% = 0
END IF
hasjob% = false
job% = 1
FOR j% = 1 TO jobmax%
  jobflg%(j%) = false
NEXT j%


DO WHILE onelen% <> 0
  tmp$ = cutitem$(onecom$, onelen%, " ")
  tmplen% = LEN(tmp$)
  j1% = VAL(cutitem$(tmp$, tmplen%, "-"))
  j2% = VAL(tmp$)
  j1% = imax%(j1%, 1)
  j1% = imin%(j1%, jobmax%)
  j2% = imin%(j2%, jobmax%)
  j2% = imax%(j1%, j2%)   'j2% should be <= then j1%
                          ' as a consequence if j2% invalid it takes the value
                          ' of j1% eg: if "-" was not found
  FOR j% = j1% TO j2%
    jobflg%(j%) = true
    hasjob% = true
  NEXT j%
LOOP

IF NOT hasjob% THEN     ' if did not have any specified do it all
  FOR j% = 1 TO jobmax%
    jobflg%(j%) = true
  NEXT j%
END IF

tmp$ = prompt$
prompt$ = "Data-file time-scale:"
IF comlen% = 0 THEN CALL getcomm
CALL nxtcomm
prompt$ = tmp$
tmp$ = cutitem$(onecom$, onelen%, " ")
SELECT CASE tmp$
  CASE "NEXT"
    datfile$ = LTRIM$(STR$(numor&))
    numor& = numor& + 1
    dodata% = true
  CASE "NO"
    dodata% = false
  CASE ELSE
    datfile$ = tmp$
END SELECT
ppos% = INSTR(1, datfile$, ".")
IF ppos% = 0 AND datfile$ <> "" THEN datfile$ = datfile$ + ".dat"

tfactor = VAL(cutitem$(onecom$, onelen%, " "))


dorun% = true
erflg% = 0
```

```
IF dodata% THEN OPEN datapath$ + datfile$ FOR OUTPUT AS #chdat
IF erflg% <> 0 THEN
  CALL msg(" error" + STR$(erflg%) + " in open datafile:" + datfile$, true)
  datfile$ = ""
  dodata% = false
END IF

PRINT "RUN:"; runfile$; "   DATA:", datfile$
IF dodata% THEN PRINT #chdat, "RUN "; runfile$; "   DATA "; datfile$
IF lpton% THEN

   LPRINT CHR$(28); "V"; CHR$(1); CHR$(28); "E"; CHR$(1)
   LPRINT " RUN:"; runfile$; CHR$(27); "$"; CHR$(0); CHR$(1); "DATA:"; datfile$;
   LPRINT CHR$(28); "V"; CHR$(0); CHR$(28); "E"; CHR$(0)
END IF
IF pstat% THEN CALL cStat

job% = 1
newjob% = true

restlrun$ = comline$
restlnrun% = comlen%
comline$ = ""
comlen% = 0


END SUB

'---------------------------------------------------------------------------
```

24.SUB cEcho

```
DIM cnt(-1 TO 2, 1 TO 4)
DIM aver(1 TO 4), phi(1 TO 4), echo(1 TO 4), phcorr(1 TO 4), echoerr(1 TO 4)
ehead$ = "Det  Aver    Echo    error    phase   min@B##  correction"
eform$ = "#   #######  #######+/-#####.#  ####.#  ###.###  ###.###"
ef1form$ = "#######(##### ##### #####)"
ef2form$ = "EC # Av=#######  E=#######+/-#####.#  ph ####.#  min  B##=###.###  corr ###.###"

IF onelen% = 0 THEN
    GOSUB ShowEchoDef
    EXIT SUB
END IF

tmp$ = cutitem$(onecom$, onelen%, " ")

IF tmp$ = "DEF" THEN
   Phaseunit% = VAL(cutitem$(onecom$, onelen%, " "))
   PhaseStep = VAL(cutitem$(onecom$, onelen%, " "))
   GOSUB Uverif
   GOSUB ShowEchoDef
   CALL SaveLim
ELSE
   GOSUB Uverif
   Etime = VAL(tmp$)
   Erep% = VAL(cutitem$(onecom$, onelen%, " "))
   Etime = max(Etime, 0)
   Erep% = imax(Erep%, 1)
   tpreset = Etime * tfactor
   FOR i% = 1 TO Erep%
      IF i% > 1 THEN CALL MakeSteps
      StartVal = curset(Phaseunit%)

    FOR d% = 1 TO 4
              aver(d%) = 0
```

```
    NEXT d%
    PRINT USING "E-time:####  "; tpreset
    IF lpton% THEN LPRINT USING "E-time:####  "; tpreset;
    CALL LPnarrow
    timeav = 0
    monav = 0
    cntnum = 0
    FOR e% = -1 TO 2
            IF hitF1% THEN EXIT FOR
            tmpval = StartVal + PhaseStep * e%
            CALL BsetSafe(Phaseunit%, tmpval)
            CALL counting(tpreset, time, mon, count1, count2, count3)
            IF hitF1% THEN EXIT FOR
            cnt(e%, 1) = count1
            cnt(e%, 2) = count2
            cnt(e%, 3) = count3
            cnt(e%, 4) = count1 + count2 + count3
            FOR d% = 1 TO 4
              aver(d%) = aver(d%) + cnt(e%, d%)
            NEXT d%
            PRINT USING ef1form$; cnt(e%, 4); cnt(e%, 1); cnt(e%, 2); cnt(e%, 3);
            IF e% = 0 THEN PRINT
            IF lpton% THEN LPRINT USING ef1form$; cnt(e%, 4); cnt(e%, 1); cnt(e%, 2); cnt(e%, 3);
            timeav = timeav + time
            monav = monav + mon
            cntnum = cntnum + 1
    NEXT e%
    PRINT : IF lpton% THEN LPRINT " "
    CALL LPnormal
    IF NOT hitF1% THEN
    PRINT USING ehead$; Phaseunit%
    IF lpton% THEN LPRINT USING ehead$; Phaseunit%
    FOR d% = 1 TO 4
            aver(d%) = aver(d%) / 4
            IF cnt(2, d%) <> cnt(0, d%) THEN
              phi(d%) = ATN((cnt(-1, d%) - cnt(1, d%)) / (cnt(2, d%) - cnt(0, d%)))
            ELSE
              phi(d%) = pi / 2
            END IF
            IF cnt(0, d%) > cnt(2, d%) THEN phi(d%) = phi(d%) + pi
            echo(d%) = SQR((cnt(-1, d%) - cnt(1, d%)) ^ 2 + (cnt(0, d%) - cnt(2, d%)) ^ 2)
            phcorr(d%) = PhaseStep / pi * 2 * phi(d%)
            echoerr(d%) = SQR((cnt(-1, d%) - cnt(1, d%)) ^ 2 * (cnt(-1, d%) + cnt(1, d%)) + (cnt(0, d%) - cnt(2, d%)) ^ 2 *
(cnt(0, d%) + cnt(2, d%))) / echo(d%)
            echo(d%) = echo(d%) / 2: echoerr(d%) = echoerr(d%) / 2
            phi(d%) = phi(d%) * 180 / pi
            PRINT USING eform$; d%; aver(d%); echo(d%); echoerr(d%); phi(d%); StartVal + phcorr(d%); phcorr(d%)
            IF lpton% THEN LPRINT USING eform$; d%; aver(d%); echo(d%); echoerr(d%); phi(d%); StartVal + phcorr(d%);
phcorr(d%)
     NEXT d%
            IF hpon% THEN CALL ShowHp
            IF dodata% THEN
              PRINT #chdat, "ED Mon:######## Time:######.##"; monav / cntnum; timeav / cntnum;
              FOR e% = -1 TO 2
                PRINT #chdat, USING ef1form$; cnt(e%, 4); cnt(e%, 1); cnt(e%, 2); cnt(e%, 3);
              NEXT e%
              PRINT #chdat, " "
              FOR d% = 1 TO 4
                      PRINT #chdat, USING ef2form$; d%; aver(d%); echo(d%); echoerr(d%); phi(d%); Phaseunit%; StartVal +
phcorr(d%); phcorr(d%)
              NEXT d%
            END IF
    END IF
    CALL BsetSafe(Phaseunit%, StartVal)
    IF hitF1% THEN EXIT FOR
  NEXT i%
```

```
  CALL ZeroSteps
  IF hitF1% THEN CALL msg(" Echo stopped!", true)

END IF
EXIT SUB

ShowEchoDef:
  PRINT USING "E def B## step ###.####"; Phaseunit%; PhaseStep
  IF lpton% THEN LPRINT USING "E def B## step ###.####"; Phaseunit%; PhaseStep
RETURN

Uverif:
  IF Phaseunit% > nsupl OR Phaseunit% < 4 THEN
    GOSUB ShowEchoDef
    CALL msg(" Invalid Echo Supply !", true)
    Phaseunit% = 0
    PhaseStep = 0
    EXIT SUB
  END IF
RETURN

END SUB

'------------------------------------------------------------------------
```

25.SUB cStat

```
staform$ = "!## ###.###(###.###)! "

  IF dodata% THEN PRINT #chdat, "Head ";
  FOR unit% = 1 TO nmotor
    CALL Mread(unit%, m):
    angleread(unit%) = m
    IF ABS(angleset(unit%) - m) > .02 THEN
      COLOR errcolor, 0
      warnstr$ = "*"
    ELSE
      COLOR outcolor, 0
      warnstr$ = " "
    END IF
    IF lpton% THEN LPRINT USING staform$; "M"; unit%; angleset(unit%); angleread(unit%); warnstr$;
    PRINT USING staform$; "M"; unit%; angleset(unit%); angleread(unit%); warnstr$;
    IF dodata% THEN PRINT #chdat, USING staform$; "M"; unit%; angleset(unit%); angleread(unit%); warnstr$;
  NEXT unit%
  IF lpton% THEN LPRINT " "
  PRINT " "
  IF dodata% THEN
            PRINT #chdat, ""
            PRINT #chdat, "Head ";
  END IF
  FOR unit% = 1 TO nsupl
    CALL Bread(unit%, b, status%)
    IF ABS((curset(unit%) - b) / curmaxh(unit%)) > CurTolerance THEN
      COLOR errcolor, 0
      warnstr$ = "*"
    ELSE
      COLOR outcolor, 0
      warnstr$ = " "
    END IF
    curread(unit%) = b
    IF lpton% THEN LPRINT USING staform$; "B"; unit%; curset(unit%); curread(unit%); warnstr$;
    PRINT USING staform$; "B"; unit%; curset(unit%); curread(unit%); warnstr$;
    IF dodata% THEN PRINT #chdat, USING staform$; "B"; unit%; curset(unit%); curread(unit%); warnstr$;
    IF unit% = 3 * INT(unit% / 3) THEN
            IF lpton% THEN LPRINT " "
            PRINT " "
            IF dodata% THEN
```

```
            PRINT #chdat, ""
            IF unit% <> nsupl THEN PRINT #chdat, "Head ";
          END IF
    END IF
  NEXT unit%
  COLOR outcolor, 0
  IF dodata% THEN
     PRINT #chdat, ""
  END IF
  IF hpon% THEN CALL ShowHp
  IF dodata% THEN PRINT #chdat, "Head ";
  CALL ShowSel
  IF dodata% THEN PRINT #chdat, "Head ";
  CALL ShowDia
END SUB
```

26.SUB counting (tpreset, time, mon, count1, count2, count3)

```
DoAgain:

  IF tpreset > 0 THEN
    GOSUB Cstart
    finished% = false
  ELSE
    finished% = true
  END IF

  IF tpreset > wtime THEN
    CALL paintscreen
  ELSE
    CALL rates(-1, 0, 0, mrate, drate)          'reset rates
  END IF


  DO UNTIL finished% OR hitF1%

    IF (CraneWatch% AND CraneIn%) THEN GOSUB WaitCrane

    CALL countstat(time, mon, count1, count2, count3, finished%)
    IF time > .5 THEN
      csum = count1 + count2 + count3
      CALL rates(time, mon, csum, mrate, drate)
      IF mrate < limmonit THEN GOSUB beamclosed
    END IF

    IF tpreset > wtime THEN
          FOR unit% = 5 TO nsupl                ' Drusch reading perturbes?
            CALL Bread(unit%, curread(unit%), status%)
            CALL WaitTic(5)
            IF status% <> 0 AND curset(unit%) <> 0 THEN
              CALL msg("B" + STR$(unit%) + " found off", true)
              CALL BswitchOn(unit%)
              restart% = true
            END IF
          NEXT unit%
          CALL updscreen
    ELSE
          CALL WaitTic(10)
    END IF
  IF hitF5% THEN CALL EToShell
  LOOP

  IF tpreset > wtime THEN
     SCREEN scmode, , compage, compage
     COLOR outcolor, 0
  END IF
```

```
 IF NOT hitF1% THEN
   CALL verify(restart%)
   IF restart% THEN GOTO DoAgain
 END IF

EXIT SUB

Cstart:
   IF MrateCal > 0 THEN
      mpreset = tpreset * MrateCal
     CALL countstart(mpreset, false)
   ELSE
     CALL countstart(tpreset, true)
   END IF
RETURN

beamclosed:
            CALL msg("Beam is off....", true)
            DO WHILE mrate < limmonit AND NOT hitF1%
               SOUND 500, 2
               CALL rates(-1, 0, 0, mrate, drate)
               IF MrateCal > 0 THEN
                  mpreset = tpreset * MrateCal
                  CALL countstart(mpreset, false)
               ELSE
                  CALL countstart(tpreset, true)
               END IF
               CALL WaitTic(40)
               CALL countstat(time, mon, count1, count2, count3, finished%)
               CALL rates(time, mon, csum, mrate, drate)
               IF hitF5% THEN CALL EToShell
            LOOP
            RETURN

WaitCrane:
            blink% = true
            CALL msg("Crane is too close....", true)
            CALL countSuspend
            DO WHILE CraneIn% AND NOT hitF1%
               SOUND 500, 2
'           CALL CraneDispl(blink%)
'           blink% = NOT blink%
               WaitTic (40)
               IF hitF5% THEN CALL EToShell
            LOOP
'       CALL CraneDispl(false)
            IF NOT hitF1% THEN CALL msg("Crane moved out....", true)
            CALL countResume
            RETURN

END SUB
```

27.SUB ccount

```
sform$ = "Mon:########. Time:####.## M/T:###### Det:######.(######. ######. ######.)"
asform$ = "aver:#######.## +/- #####.##"
fform$ = "C ########. #######.## ########. ######. ######. ######."
afform$ = "av#########. #######.#"
nostep% = true
gsum = 0
repdone% = 0

tpreset = max(VAL(cutitem$(onecom$, onelen%, " ")), 0)
tpreset = tpreset * tfactor
rep% = max(VAL(cutitem$(onecom$, onelen%, " ")), 1)         ' repeat at least once
FOR i% = 1 TO rep%
```

```
 IF i% <> 1 THEN CALL MakeSteps

 CALL counting(tpreset, time, mon, count1, count2, count3)

 csum = count1 + count2 + count3
 IF nostep% AND NOT hitF1% THEN
    gsum = gsum + csum
    repdone% = repdone% + 1
 END IF
 IF time <> 0 THEN
   mrate = mon / time
 ELSE
   mrate = 0
 END IF

 IF time > 0 THEN
    PRINT USING sform$; mon; time; mrate; csum; count1; count2; count3
    IF dodata% THEN
      PRINT #chdat, USING fform$; mon; time; csum; count1; count2; count3;
    END IF
    IF lpton% THEN
      LPRINT USING sform$; mon; time; mrate; csum; count1; count2; count3
      IF 5 * INT(i% / 5) = i% THEN LPRINT ""
    END IF

    IF Tstatus% > 0 THEN CALL ShowTset
    IF statusT% = 1 THEN CALL ShowTemp  ' lecture T sur HP
    IF statusT% = 2 THEN CALL ShowTB    ' lectures Ts sur Barras
    IF Tstatus% < 1 AND statusT% < 1 AND dodata% THEN PRINT #chdat, ""
    IF hpon% THEN CALL ShowHp

 END IF

IF hitF1% THEN EXIT FOR
NEXT i%

IF nostep% AND repdone% > 1 THEN
  PRINT USING asform$; gsum / repdone%; SQR(gsum) / repdone%
  IF dodata% THEN
    PRINT #chdat, USING afform$; gsum / repdone%; SQR(gsum) / repdone%
  END IF

  IF lpton% THEN LPRINT USING asform$; gsum / repdone%; SQR(gsum) / repdone%
END IF

CALL ZeroSteps

IF dorun% THEN job% = job% + 1
newjob% = true
  IF hitF1% THEN CALL msg(" Count aborted ", true )

END SUB
```

28.SUB countResume

```
InRoutine$ = "countResume"
buffer$ = "G"
action% = 1
CALL iewrit(Counter%, buffer$)
CALL I3eError(Counter%)
END SUB

'------------------------------------------------------------------------
```

29.SUB countstart (pres, t%)

```
' starts the counting with preset=pres
' t%=true preset is time in sec
' t%=false preset is in monitor
```

```
'PRINT " counting starts"

InRoutine$ = "countstart"

IF t% THEN
  lead$ = "T"
  prese = INT(pres * 100)
ELSE
  lead$ = "N"
  prese = INT(pres / 10)
END IF
ttime$ = STR$(prese)
ttime$ = lead$ + RIGHT$(ttime$, LEN(ttime$) - 1)
action% = 1
CALL iewrit(Counter%, ttime$)
CALL I3eError(Counter%)

END SUB

'------------------------------------------------------------------------
```

30.SUB countstat (time, monit, c1, c2, c3, done%)

```
' return the actual status of the counters
' and done%=true if finished, =false if not

InRoutine$ = "countstat"

d$ = "D"
x$ = "X"
action% = 1
CALL iewrit(Counter%, d$)
CALL I3eError(Counter%)
action% = 2
CALL ieread(Counter%, ansd$)
CALL I3eError(Counter%)
'PRINT "ansd$="; ansd$
action% = 3
CALL iewrit(Counter%, x$)
CALL I3eError(Counter%)
action% = 4
CALL ieread(Counter%, ansx$)
CALL I3eError(Counter%)
'PRINT "ansx$="; ansx$
IF MID$(ansd$, 2, 1) = "1" THEN
  done% = true
ELSE
  done% = false
END IF
time = VAL(MID$(ansx$, 1, 8)) / 100
monit = VAL(MID$(ansx$, 10, 8)) * 10
c1 = VAL(MID$(ansx$, 28, 8))
c2 = VAL(MID$(ansx$, 37, 8))
c3 = VAL(MID$(ansx$, 46, 8))

END SUB
```

31.SUB countSuspend

```
InRoutine$ = "countSuspend"
buffer$ = "B"
action% = 1
CALL iewrit(Counter%, buffer$)
CALL I3eError(Counter%)

END SUB
```

32. CALL I3eError(Dafodev%)

```
stat = VAL(MID$(buffer$, 1, 1))
'if it becomes real hexa this line will brake!!!
IF light% THEN
   stat = stat OR 2
ELSE
   stat = stat AND 1   ' here agan we keep only the bit 1
END IF
setback$ = "V" + HEX$(stat) + MID$(buffer$, 2, 1)
CALL iewrit(Dafodev%, setback$)
   action% = 3
   CALL I3eError(Dafodev%)

END SUB
```

33.SUB DoDos (aCom$)

```
tmp$ = "DOS  " + aCom$ + " "
IF onelen% > 0 THEN
  tmp$ = tmp$ + onecom$
ELSE
  tmp$ = LEFT$(tmp$, LEN(tmp$) - 1)
END IF
comline$ = tmp$ + "," + comline$
comlen% = LEN(comline$)

END SUB
```

34.FUNCTION DruStatsKeith% (unit%)

```
' returns the bit pattern actually written to the output relays
InRoutine$ = "DruStatsKeith"

        iedev% = curdev%(unit%)
        bitp% = 2 ^ (unit% - 1)
        action% = 1
        CALL iewrit(iedev%, "iread raw,5,3;")
        CALL I3eError(iedev%)
        action% = 2
        CALL ieread(iedev%, tmp$)
        CALL I3eError(iedev%)
        DruStatsKeith% = VAL(tmp$) AND (bitp%)        'bitp% = 0 (false) if on

END FUNCTION
```

35.SUB ShowBLim (unit%)

```
form$ = " Lim B## Min-Max hard ###.## ###.## soft ###.## ###.##"
PRINT USING form$; unit%; curminh(unit%); curmaxh(unit%); curmins(unit%); curmaxs(unit%)
IF lpton% THEN
  LPRINT USING form$; unit%; curminh(unit%); curmaxh(unit%); curmins(unit%); curmaxs(unit%)
END IF
END SUB

'------------------------------------------------------------------------
```

36.SUB cZ

```
PRINT " B5 - B16 zeroing"
IF lpton% THEN LPRINT " B5 - B20 zeroing"
FOR unit% = 5 TO nsupl
  newval = 0
  curlast(unit%) = curset(unit%)
'   INPUT " avant Bset safe", x
  CALL BsetSafe(unit%, newval)
'   INPUT " apres Bset safe", x
'   PRINT unit%
  IF newval <> 0 THEN CALL ShowBval(unit%)
NEXT unit%
END SUB
```

37.SUB ShowBval (unit%)

```
form$ = "B## ####.###(####.###)!"

CALL Bread(unit%, rval, status%)
curread(unit%) = rval

IF (ABS(curset(unit%) - curread(unit%)) / curmaxh(unit%) > CurTolerance) THEN
  COLOR errcolor, 0
  warn$ = "*"
ELSE
  warn$ = " "
END IF

PRINT USING form$; unit%; curset(unit%); curread(unit%); warn$;
IF trial% > 1 THEN
  COLOR errcolor, 0
  PRINT "  "; trial%; "<==="
  SOUND 500, 1
ELSE
  PRINT
END IF

'PRINT USING form$; unit%; curset(unit%); curread(unit%);warn$;
'------------- while in test-------
'CALL HPread(hpval)
'PRINT USING " Hp= #.######^^^^  read/Hp=#.######^^^^"; hpval; curread(unit%) / hpval

IF lpton% THEN
  LPRINT USING form$; unit%; curset(unit%); curread(unit%); warn$;
  IF trial% > 1 THEN
        LPRINT "  "; trial%; "<==="
  ELSE
        LPRINT " "
  END IF
trial% = 1




  'lprint USING form$; unit%; curset(unit%); curread(unit%); warn$;
        '------------- while in test-------
  'lprint USING " Hp= #.######^^^^  read/Hp=#.######^^^^"; hpval; curread(unit%) / hpval

END IF
COLOR outcolor, 0
IF dodata% THEN PRINT #chdat, USING form$; unit%; curset(unit%); curread(unit%)
IF status% <> 0 THEN CALL msg("Supply is OFF", true)
END SUB
```

38.SUB limBverif (unit%, newval, warn%)

```
' verifies hard and soft limits
' set back to last value if it was out of limits
' prints warning only if warn%=true
testval = newval
testval = max(curmins(unit%), testval)
testval = min(curmaxs(unit%), testval)
IF testval <> newval THEN
 newval = curset(unit%)
 IF warn% THEN
   CALL msg(" Out of limits!", true)
   CALL ShowBLim(unit%)
 END IF
END IF

END SUB
```

```
'------------------------------------------------------------------------
```

39.SUB EToShell

```
 IF tpreset > wtime THEN
    SCREEN scmode, , compage, compage
    COLOR outcolor, 0
 END IF

onecom$ = ""
CALL cDos
hitF5% = false

IF tpreset > wtime THEN CALL paintscreen

END SUB

'------------------------------------------------------------------------
```

40.SUB I3eError (OnUnit%)

```
IF i3estatus% <> 0 THEN
            ermess$ = "IEEE error =" + STR$(i3estatus%)
            CALL msg(ermess$, true)
            ermess$ = "In sub:" + InRoutine$ + " at action:" + STR$(action%) + "dev%=" + STR$(OnUnit%)
            CALL msg(ermess$, false)
            IF ((i3estatus% = 8) AND (NOT hitF1%)) THEN
'         STOP
            IF (OnUnit% = 10) THEN
              PRINT "cleaning up Keithley buffer wait ...";
              CALL ieread(OnUnit%, l$)
              PRINT "done"
            ELSE
              CALL msg("program suspended for 30 sec...", false)
              FOR i = 1 TO 30
                SOUND 500, 2
                SLEEP 1
'           PRINT hitF1%
                IF hitF1% THEN EXIT FOR
              NEXT i
            END IF

  IF NOT hitF1% THEN CALL msg("... try to continue", false)
END IF

END IF

END SUB
```

41.SUB MakeSteps

```
 FOR n% = 1 TO nsupl
  IF curstep(n%) <> 0 THEN
    nostep% = false
            curlast(n%) = curset(n%)
            newval = curset(n%) + curstep(n%)
            CALL BsetSafe(n%, newval)
            CALL ShowBval(n%)
  END IF
 NEXT n%
 FOR n% = 1 TO nmotor
  IF anglestep(n%) <> 0 THEN
    nostep% = false
            anglelast(n%) = angleset(n%)
            newval = angleset(n%) + anglestep(n%)
```

```
            CALL MsetSafe(n%, newval)
            CALL ShowMVal(n%)
    END IF
  NEXT n%
END SUB
```

42.SUB cM

```
tmp$ = cutitem$(onecom$, onelen%, " ")
unit% = VAL(tmp$)
IF unit% < 1 OR nmotor < unit% THEN
  CALL msg(" Invalid motor: " + tmp$, true)
  EXIT SUB
END IF

IF onelen% = 0 THEN
  CALL ShowMVal(unit%)
  EXIT SUB
END IF

tmp$ = cutitem$(onecom$, onelen%, " ")

' special case for M3 Smaple position . Exist OUT case position 4 .
IF unit% = 3 AND MID$(tmp$, 1, 2) = "OU" THEN tmp$ = "4"

newval = VAL(tmp$)
CALL MsetSafe(unit%, newval)
CALL ShowMVal(unit%)

IF onelen% > 0 THEN anglestep(unit%) = VAL(cutitem$(onecom$, onelen%, " "))
END SUB
```

43.SUB LimMVerif (unit%, newval, warn%)

```
' verifies hard and soft limits
' set back to last value if it was out of limits
' prints warning only if warn%=true
ok% = true
testval = newval
testval = max(anglemins(unit%), testval)
testval = min(anglemaxs(unit%), testval)
IF testval <> newval THEN
  newval = angleset(unit%)
  IF warn% THEN
    CALL msg(" Out of limits!", true)
    CALL ShowMLim(unit%)
  END IF
END IF
END SUB
```

44.SUB Mread (unit%, rval)

```
InRoutine$ = "Mread"

unit$ = STR$(unit%)
' motor 1 2 angles . Motor 3 Position or relative codeur
IF unit% = 3 THEN
aCom$ = "N" + RIGHT$(unit$, LEN(unit$) - 1)     ' cut off leading space
ELSE
aCom$ = "A" + RIGHT$(unit$, LEN(unit$) - 1)     ' cut off leading space
END IF
action% = 1
CALL iewrit(Dafodev%, aCom$)
CALL I3eError(Dafodev%)
action% = 2
CALL ieread(Dafodev%, pos$)
CALL I3eError(Dafodev%)
IF unit% = 3 THEN
```

```
rval = VAL(pos$)
ELSE
rval = VAL(pos$) / 100
END IF
END SUB
'---------------------------------------------------------------------------
```

45.SUB Mset (unit%, sval)

```
' here should come IEEE stuff

CALL Mread(unit%, rval)
IF ABS(rval - sval) <= .01 THEN EXIT SUB

InRoutine$ = "Bset"
unit$ = STR$(unit%)

IF unit% <> 3 THEN
setv$ = STR$(CLNG(sval * 100))
mcom$ = "G" + RIGHT$(unit$, LEN(unit$) - 1) + "," + RIGHT$(setv$, LEN(setv$) - 1)
ELSE
setv$ = STR$(sval)
mcom$ = "P" + RIGHT$(unit$, LEN(unit$) - 1) + "," + RIGHT$(setv$, LEN(setv$) - 1)
END IF
action% = 1
CALL iewrit(Dafodev%, mcom$)
action% = 2
f$ = "00"
DO UNTIL f$ = "01" OR hitF1%
  CALL WaitTic(2)
  CALL iewrit(Dafodev%, "F")
  CALL ieread(Dafodev%, f$)
  CALL I3eError(Dafodev%)
LOOP

IF hitF1% THEN
  action% = 3
  CALL iewrit(Dafodev%, "B")
  CALL I3eError(Dafodev%)
  CALL Mread(unit%, sval)      ' if stopped set sval to actual value
END IF

action% = 4
CALL iewrit(Dafodev%, "L")
CALL ieread(Dafodev%, l$)
CALL I3eError(Dafodev%)
IF l$ <> "00" THEN
 CALL msg(" Motor status:" + l$, true)
END IF

END SUB
```

46.SUB MsetSafe (unit%, newval)

```
'similar to Mset but includes the
'       limit verification
'       updating last value
'       updating current value

  CALL LimMVerif(unit%, newval, true)     ' set back to old value if out
  CALL Mset(unit%, newval)
'  IF newval = 0 AND angleset(unit%) <> 0 THEN anglelast(unit%) = angleset(unit%)
  angleset(unit%) = newval

END SUB

'---------------------------------------------------------------------------
```

47.SUB ShowMVal (unit%)

```
form$ = "M ## #####.###(#####.###)!"
```

```
CALL Mread(unit%, rval)
angleread(unit%) = rval

IF ABS(angleset(unit%) - angleread(unit%)) > .01 THEN
  COLOR errcolor, 0
  warn$ = "*"
ELSE
  warn$ = " "
END IF

PRINT USING form$; unit%; angleset(unit%); angleread(unit%); warn$
IF lpton% THEN LPRINT USING form$; unit%; angleset(unit%); angleread(unit%); warn$
IF dodata% THEN PRINT #chdat, USING form$; unit%; angleset(unit%); angleread(unit%)
COLOR outcolor, 0
END SUB
```

48.SUB ShowMLim (unit%)

```
form$ = " Lim M## Min-Max hard ###.## ###.## soft ###.## ###.##"
PRINT USING form$; unit%; angleminh(unit%); anglemaxh(unit%); anglemins(unit%); anglemaxs(unit%)
IF lpton% THEN
 LPRINT USING form$; unit%; angleminh(unit%); anglemaxh(unit%); anglemins(unit%); anglemaxs(unit%)
END IF

END SUB
```

49.SUB SaveLim

```
lform1$ = "! _, ##_,######.##_,######.##"
lform3$ = "! _, ######.##_,######.##"
 erflg% = 0
 OPEN limitfile FOR OUTPUT AS #chtmp
 FOR i% = 1 TO nsupl
   PRINT #chtmp, USING lform1$; "B"; i%; curmins(i%); curmaxs(i%)
 NEXT i%
 FOR i% = 1 TO nmotor
   PRINT #chtmp, USING lform1$; "M"; i%; anglemins(i%); anglemaxs(i%)
 NEXT i%
 PRINT #chtmp, USING lform3$; "S"; selmins; selmaxs
 PRINT #chtmp, USING lform3$; "T"; tsetmins; tsetmaxs
 PRINT #chtmp, USING lform3$; "T"; tempmins; tempmaxs
 PRINT #chtmp, USING lform3$; "H"; hubemins; hubemaxs
 PRINT #chtmp, USING "Monitor and Rate _, ######## _, ########"; limmonit; MrateCal
 PRINT #chtmp, USING "Echo Def _, ## _, ###.#####"; Phaseunit%; PhaseStep

 CLOSE chtmp
 IF erflg% <> 0 THEN
   tmp$ = " Error=" + STR$(erflg%) + " in saving LIMITS2.NOW"
   CALL msg(tmp$, true)
 END IF
END SUB
```

50.SUB cTset

```
IF onelen% = 0 THEN
 CALL ShowTset
 EXIT SUB
END IF

tmp$ = cutitem$(onecom$, onelen%, " ")
IF tmp$ = "" THEN
  CALL ShowTset
  EXIT SUB
END IF
SELECT CASE MID$(tmp$, 1, 2)
 CASE "ON"
```

```
   IF (Tstatus% = 0) THEN CALL Topen
   Tstatus% = 1
   statusT% = 0
   CALL ShowTset
  CASE "OF"
   Tstatus% = 0
   CALL Tclose
  CASE ELSE
   newval = VAL(tmp$)
   CALL TsetSetSafe(newval)
   CALL ShowTset
END SELECT
END SUB
```

51.SUB ShowTset

```
   IF (Tstatus% = 0) THEN
    PRINT "TILL is off"
    IF lpton% THEN LPRINT "TILL is off"
   ELSE
    form$ = "Tsre: ####.## ####.## ####.##"
'    form$ = "Ts:####.## Tr:####.## Te:####.##"
    CALL tsetread(ts, trg, tsa)
    PRINT USING form$; ts; trg; tsa
    IF lpton% THEN LPRINT USING form$; ts; trg; tsa
    IF dodata% THEN PRINT #chdat, USING form$; ts; trg; tsa
   END IF

END SUB
```

52.SUB ShowTsetLim

```
form$ = " Lim TILL Min-Max hard ###### ###### soft ###### ######"
PRINT USING form$; tsetminh; tsetmaxh; tsetmins; tsetmaxs
IF lpton% THEN
 LPRINT USING form$; tsetminh; tsetmaxh; tsetmins; tsetmaxs
END IF

END SUB

'---------------------------------------------------------------------------
```

53.SUB cHP

```
tmp$ = cutitem$(onecom$, onelen%, " ")
SELECT CASE MID$(tmp$, 1, 2)
 CASE "ON"
   hpon% = true
 CASE "OF"
   hpon% = false
 CASE ELSE
   CALL ShowHp
END SELECT

END SUB
```

54.SUB ShowHp

```
   form$ = " Hp= #.######^^^^"
   CALL HPread(hpval)
   PRINT USING form$; hpval
   IF lpton% THEN
    LPRINT USING form$; hpval
   END IF
END SUB

'---------------------------------------------------------------------------
```

55.SUB HPread (hpval)

```
InRoutine$ = "HPread"

' reads the HP whatever is on the display
'action% = 1
  CALL setinputEOS(10)
  CALL ieread(HPdev%, buffer$)
  CALL setinputEOS(13)
  CALL I3eError(HPdev%)
hpval = VAL(buffer$)
END SUB
```

56.SUB TsetSetSafe (newval)

```
  testval = newval
  testval = max(tsetmins, testval)
  testval = min(tsetmaxs, testval)
  IF testval <> newval THEN
          CALL msg(" Out of limits!", true)
          CALL ShowTsetLim
          EXIT SUB
  END IF
  IF (Tstatus% = 0) THEN
    CALL Topen
    Tstatus% = 1
    ' cancel other temperature device
    statusT% = 0
  END IF
  CALL tsetset(newval)

 END SUB

'------------------------------------------------------------------------
```

57.SUB ShowSel

```
form$ = "Sel #####(#####)!"

CALL SelRead(selreadv, status%)

IF (ABS(selsetv - selreadv) > selsetv * .01) THEN
  COLOR errcolor, 0
  warn$ = "*"
ELSE
  warn$ = " "
END IF

PRINT USING form$; selsetv; selreadv; warn$
IF dodata% THEN
          PRINT #chdat, USING form$; selsetv; selreadv; warn$
END IF

IF lpton% THEN
  LPRINT USING form$; selsetv; selreadv; warn$
END IF
COLOR outcolor, 0
IF status% <> 0 THEN CALL msg("Selector Error = " + STR$(status%), true)

END SUB

'------------------------------------------------------------------------
```

58.SUB cSelector

```
IF onelen% = 0 THEN
 CALL ShowSel
 EXIT SUB
END IF

tmp$ = cutitem$(onecom$, onelen%, " ")
```

```
newval = VAL(tmp$)
CALL SelSetSafe(newval)
CALL ShowSel

END SUB
```

59.SUB cWavl

```
tmpval = VAL(cutitem$(onecom$, onelen%, " "))
IF tmpval <> 0 THEN
  selspeed = SelectorConstant / tmpval
  CALL SelSetSafe(selspeed)
END IF

CALL ShowWavl

END SUB

'------------------------------------------------------------------------
```

60.SUB cRate

```
mr$ = "Monitor rate calibration: ######"


tmpval = VAL(cutitem$(onecom$, onelen%, " "))
IF tmpval <> 0 THEN
  MrateCal = tmpval
  CALL SaveLim
END IF

PRINT USING mr$; Mrat eCal
IF lpton% THEN LPRINT USING mr$; MrateCal

END SUB

'------------------------------------------------------------------------
```

61.SUB SelSet (selv)

```
form$ = "Sel #####(#####)  Status%=#"

 CALL SelRead(rval, status%)
 IF status% AND 1 THEN EXIT SUB        'on local
 InRoutine$ = "SelSet"
 action% = 1
 CALL iewrit(SelDev%, "s0=0")
 CALL I3eError(SelDev%)

 setv% = selv * .992
 buffer$ = "v1=" + STR$(setv%)
 action% = 2
 CALL iewrit(SelDev%, buffer$)
 CALL I3eError(SelDev%)

 iloop% = 0
 chgsgn% = 0
 oval = rval
 osgn% = SGN(selv - oval)
 looplimit% = ABS(selv - oval) + 120

 CALL ctime
 IF lpton% THEN
          LPRINT " Changing selector speed"
 END IF
 CALL LPnarrow
 atline% = CSRLIN
' wait loop
 DO WHILE (chgsgn% < 5 AND iloop% < looplimit%)
   CALL WaitTic(55)
```

```
    IF hitF1% THEN EXIT DO

    CALL SelRead(rval, status%)
    LOCATE atline%, 1
    PRINT USING form$; selv; rval; status%;
    IF lpton% THEN
            LPRINT rval;
            IF INT(iloop% / 20) = iloop% / 20 THEN LPRINT
    END IF
    nsgn% = SGN(rval - oval)
    oval = rval

   IF osgn% * nsgn% <= 0 THEN
            osgn% = nsgn%
            chgsgn% = chgsgn% + 1
    END IF
    iloop% = iloop% + 1
  LOOP
  CALL LPnormal
  CALL ctime

  action% = 3
  CALL iewrit(SelDev%, "s0=1")
  CALL I3eError(SelDev%)


END SUB
```

62.SUB SelSetSafe (newval)

```
  testval = newval
  testval = max(selmins, testval)
  testval = min(selmaxs, testval)
  IF testval <> newval THEN
            CALL msg(" Out of limits!", true)
            CALL ShowSelLim
            EXIT SUB
  END IF
  CALL SelSet(newval)
  selsetv = newval

END SUB

'------------------------------------------------------------------------
```

63.SUB rates (t, m, d, mrate, drate)

```
' calculats the monitor rate based on the last t>=1 sec
' t<0 resets rates to zero

    IF t <= oldtime1 OR t <= oldtime2 THEN
      oldtime1 = 0
      oldtime2 = 0
      oldmon1 = 0
      oldmon2 = 0
      olddet1 = 0
      olddet2 = 0
       IF t < 0 THEN
            mrate = 0
            drate = 0
            EXIT SUB
       END IF
    END IF

    dt = t - oldtime2
    dm = m - oldmon2
   dd = d - olddet2
    IF dt <> 0 THEN
```

```
    mrate = dm / dt
    drate = dd / dt
  ELSE
    mrate = 0
    drate = 0
  END IF

  IF t - oldtime1 > 1 THEN
        oldtime2 = oldtime1
        oldmon2 = oldmon1
        olddet2 = olddet1
        oldtime1 = t
        oldmon1 = m
        olddet1 = d
  END IF


END SUB
```

64.SUB ShowSelLim

```
form$ = " Lim Sel Min -Max hard ###### ###### soft ###### ######"
PRINT USING form$; selminh; selmaxh; selmins; selmaxs
IF lpton% THEN
  LPRINT USING form$; selminh; selmaxh; selmins; selmaxs
END IF

END SUB
```

65.SUB ShowWavl

```
form$ = "Wavl ##.##(##.##)!"

CALL SelRead(selreadv, status%)

IF (ABS(selsetv - selreadv) > selsetv * .01) THEN
   COLOR errcolor, 0
   warn$ = "*"
ELSE
   warn$ = " "
END IF
wvset = SelectorConstant / selsetv
wvread = SelectorConstant / selreadv
PRINT selsetv, selreadv

PRINT USING form$; wvset; wvread; warn$
IF dodata% THEN
        PRINT #chdat, USING form$; selsetv; selreadv; warn$
END IF

IF lpton% THEN
   LPRINT USING form$; wvset; wvread; warn$
END IF
COLOR outcolor, 0
IF status% <> 0 THEN CALL msg("Selector Error = " + STR$(status%), true)

END SUB
```

66.SUB SelRead (rval, status%)

```
  InRoutine$ = "SelRead"
  action% = 1
  CALL iewrit(SelDev%, "v2")
  CALL I3eError(SelDev%)
  action% = 2
  CALL ieread(SelDev%, buff$)
  CALL I3eError(SelDev%)
  rval = VAL(buff$) / 2
```

```
  action% = 3
  CALL iewrit(SelDev%, "e")
  CALL I3eError(SelDev%)
  action% = 4
  CALL ieread(SelDev%, buff$)
  CALL I3eError(SelDev%)
  buff$ = RIGHT$(buff$, 2)
  status% = VAL(LEFT$(buff$, 1)) * 2 + VAL(RIGHT$(buff$, 1))

END SUB
```

67.SUB ShowMonLim

```
mform$ = " limit MONITOR: ##########"

            PRINT USING mform$; limmonit
            IF lpton% THEN
              LPRINT USING mform$; limmonit
            END IF

END SUB

'------------------------------------------------------------------------
```

68.SUB updscreen

```
   COLOR 2, 0
   LOCATE 2, 6: PRINT DATE$;
   LOCATE 2, 22: PRINT TIME$;
   LOCATE 3, 5: PRINT exefile$;
   LOCATE 3, 21: PRINT runfile$;
   LOCATE 3, 37: PRINT datfile$;
   LOCATE 3, 53: PRINT job%

'detector stuff
   CALL countstat(t, m, c1, c2, c3, done%): csum = c1 + c2 + c3
   LOCATE 5, 4: PRINT USING ScreenForm1$; c1;
   LOCATE 6, 4: PRINT USING ScreenForm1$; c2;
   LOCATE 7, 4: PRINT USING ScreenForm1$; c3;
   LOCATE 5, 22: PRINT USING ScreenForm1$; csum;
   LOCATE 5, 41: PRINT USING ScreenForm1$; m;
   CALL rates(t, m, csum, mrate, drate)
   IF MrateCal > 0 THEN
      dexpect = csum * (tpreset * MrateCal) / m
   ELSE
      dexpect = csum / t * tpreset
   END IF
   LOCATE 6, 41: PRINT USING ScreenForm2$; mrate
   LOCATE 7, 41: PRINT USING ScreenForm2$; drate
   LOCATE 5, 57: PRINT USING ScreenForm2$; t;
   LOCATE 6, 57: PRINT USING ScreenForm2$; tpreset;
   LOCATE 7, 57: PRINT USING ScreenForm1$; dexpect;

'selector
   CALL SelRead(sel, ssta%)
   LOCATE 7, 22: PRINT USING ScreenForm1$; sel;
'Diaphragm
   posi% = DiaIn%
   col% = liveDcolor
   SELECT CASE posi%
     CASE 1
       posi$ = "     in"
     CASE 2
       posi$ = "    out"
     CASE ELSE
       posi$ = "unknown"
       col% = errcolor
   END SELECT
   COLOR col%, 0
```

```
  LOCATE 6, 22: PRINT posi$;


'Attenuator
  posi% = Attenin%
  col% = liveDcolor
  SELECT CASE posi%
    CASE 1
      posi$ = "     1 "
    CASE 2
      posi$ = "    out"
    CASE 3
      posi$ = "     2 "
    CASE ELSE
      posi$ = "unknown"
      col% = errcolor
  END SELECT
  COLOR col%, 0
  LOCATE 8, 22: PRINT posi$;

'motors
  FOR unit% = 1 TO nmotor
    CALL Mread(unit%, m):
    y% = 10: x% = 4 + 24 * (unit% - 1)
    LOCATE y%, x%
    IF ABS(angleset(unit%) - m) > .02 THEN
      col% = errcolor
    ELSE
      col% = liveDcolor
    END IF
    COLOR col%, 0
    PRINT USING ScreenForm3$; angleset(unit%); m;
    angleread(1) = m
  NEXT unit%

' currents
  FOR unit% = 1 TO nsupl
    t% = INT((unit% - 1) / 3)
    y% = 12 + t%
    x% = 5 + 24 * (unit% - 3 * t% - 1)
    LOCATE y%, x%
    IF ABS((curset(unit%) - curread(unit%)) / curmaxh(unit%)) > CurTolerance THEN
      col% = errcolor
    ELSE
      col% = liveDcolor
    END IF
    COLOR col%, 0
    PRINT USING ScreenForm3$; curset(unit%); curread(unit%);
  NEXT unit%
  COLOR liveDcolor%, 0

'   CALL HPread(hpval): LOCATE 19, 9: PRINT USING ScreenForm4$; hpval
'* temperature ill printed ; part not checked in manual2
  IF Tstatus% = 1 THEN
  CALL tsetread(tst, trg, tsa)
  LOCATE 21, 5: PRINT USING ScreenForm2$; tst;
  LOCATE 21, 29: PRINT USING ScreenForm2$; trg;
  LOCATE 21, 53: PRINT USING ScreenForm2$; tsa;
  END IF
'* temperature via hp address  15
  IF statusT% = 1 THEN
  CALL tempread(tsa)
  LOCATE 21, 53: PRINT USING ScreenForm2$; tsa;
  END IF
'* temperature via Barras address 3
  IF statusT% = 2 THEN
```

```
  CALL TBread(tst, trg, tsa)
  LOCATE 21, 5: PRINT USING ScreenForm2$; tst;
  LOCATE 21, 29: PRINT USING ScreenForm2$; trg;
  LOCATE 21, 53: PRINT USING ScreenForm2$; tsa;
  END IF

  COLOR 2, 0
  'SOUND 500, 1
  EXIT SUB

END SUB
```

69.SUB paintscreen

```
 OPEN "screen.sta" FOR INPUT AS #chtmp
 SCREEN scmode, , stapage, stapage
 CLS
 COLOR 11, 0
 LOCATE 1, 1
 erflg% = 0
 DO WHILE true
   LINE INPUT #chtmp, l$
   IF erflg% <> 0 THEN EXIT DO
   PRINT l$
 LOOP
 CLOSE chtmp
 erflg% = 0
CALL rates( -1, 0, 0, a, b)          'reset rates
END SUB
```

70.SUB verify (restart%)

```
' necessary verification during counts
InRoutine$ = "verify"

restart% = false

FOR unit% = 1 TO nsupl
 status% = Bstatus(unit%)
 IF status% <> 0 AND curset(unit%) <> 0 THEN
   CALL msg("B" + STR$(unit%) + " found off", true)

   CALL BswitchOn(unit%)
   restart% = true
 END IF
NEXT unit%

END SUB
```

71.SUB ZeroSteps

```
FOR n% = 1 TO nsupl
 curstep(n%) = 0
NEXT n%
FOR n% = 1 TO nmotor
 anglestep(n%) = 0
NEXT n%

END SUB
```

72.SUB cStore

```
form$ = "B ## ####.####"

IF onelen% = 0 AND (NOT doexe%) AND (NOT dorun%) THEN
 INPUT " File to store currents:"; tmp$
ELSE
 tmp$ = cutitem$(onecom$, onelen%, " ")
```

```
END IF


IF tmp$ = "" THEN EXIT SUB        ' changed our mind

ppos% = INSTR(1, tmp$, ".")
IF ppos% = 0 AND tmp$ <> "" THEN tmp$ = tmp$ + ".exb"    ' default extension

OPEN setspath$ + tmp$ FOR OUTPUT AS #chtmp

PRINT #chtmp, "! stored at: "; DATE$, TIME$
FOR unit% = 1 TO nsupl
        PRINT #chtmp, USING form$; unit%; curset(unit%)
NEXT unit%
PRINT #chtmp, USING "RATE #######"; MrateCal
PRINT #chtmp, USING "Echo Def ## ##.####"; Phaseunit%; PhaseStep

CLOSE chtmp

tmp$ = "Stored currents in: " + tmp$ + " " + DATE$ + " " + TIME$

PRINT tmp$
IF lpton% THEN LPRINT tmp$

END SUB
```

73.SUB mexit

```
CLOSE

STOP
END SUB

'------------------------------------------------------------------------
```