RCSL No: 52-AA1093

Edition: February 1982

Author: Harald Villemoes

Title:

RC3502 CPUSE

RCSL 42-i1905

A/S REGNECENTRALEN
af 1979

Keywords:

RC3502, Real Time Pascal, CPU-load.

Abstract:

CPUSE is a small REAL TIME PASCAL process, which can be used to measure the CPU load in an RC3502 computer.
This manual describes the function and usage of CPUSE. In addition, it contains examples of usage and a complete process listing.

(14 printed pages)

Printed by A/S Regnecentralen af 1979, Copenhagen

RCSL 42-i1905

CPUSE is a small PASCAL80 process which can be used to measure the CPU-load in an RC3502 computer.

The CPU-load, which is output in per cent, is calculated from the number of activations of the dummy process in the microprogram per time unit. The period of measurement is user-selected in the interval 5-60 sec. and the accuracy is approx. 6% using 5 sec. period and approx. 0.5% using 60 sec.

The measurement may be repeated automatically up to 32767 times.

CPUSE is pre-calibrated for an RC3502 CPU with microprogram revision 5, but the user may command the program to recalibrate itself on a CPU with different speed.

CPUSE should be run with high priority in order to obtain reasonable accurate measurement period on a machine with high load.

Rev. 5 program size: 1126 bytes
Rev. 5 stack size: 300 words (default)

App. A: The program text
App. B: Example of use

```
job hlv 2 600 time 30 size 100000 perm mini 25 2
(
mode list.yes
bcpuse=SET 20 mini
scope project bcpuse
bcpuse=pascal80 stack.300 ioenvir
finis
)
PROCESS cpuse(VAR sv: system←vector);

CONST

  version= coded←date(81,8,10);

    (*  this process is used to measure the amount of free cpu-time
    in % derived from the number of times the schedule instruc-
    tion ceased to find a non-empty activequeue.

    programmed by harald villemoes, tele, rc79                    *)

  countreg= #h3e6; (* scheduler dummy-process msb counter *)
  defaultfull= 225; (* initial value of fullcount *)

TYPE

  opbuf= RECORD
         ! f, l, n: integer;
         ! name:    alfa;
         ! chars:   ARRAY (6+12..6+12+80-1) OF char;
         END;

VAR

  oppool: pool 3 OF opbuf;
  timerpool: pool 1;
  zi, zo: zone;
  kbsem,
  timersem: semaphore;
  r: reference;
  period, repeats, startcount, endcount, fullcount: integer;

  PROCEDURE getregister(VAR value: integer; index: integer); EXTERNAL;

  FUNCTION msub(a,b: integer): integer; EXTERNAL;
```

```
 47          BEGIN (* main program *)
 48     1    !
 49     2    ! openopzone(zi,sv(operatorsem),ref(kbsem),1,oppool,1,1,1,1);
 50     3    ! openopzone(zo,sv(operatorsem),ref(zo.free),2,oppool,2,0,0,0);
 51     4    ! outtext(zo,"CPU LOADMETE"); outtext(zo,"R - VERS: #");
 52     5    ! outdate(zo,version); outnl(zo);
 53     6    !
 54     7    ! alloc(r,timerpool,timersem);
 55     8    !
 56     9    ! fullcount:= defaultfull;
 57    10    !
 58    11    ! REPEAT (* main loop *)
 59    12    ! !
 60    13    ! ! REPEAT (* read params *)
 61    14    ! ! !
 62    15    ! ! ! opin(zi);
 63    16    ! ! ! outtext(zo,"period (5-60"); outtext(zo," sec) : #");
 64    17    ! ! ! outend(zo);
 65    18    ! ! ! opwait(zi,oppool); ininteger(zi,period); opin(zi);
 66    19    ! ! ! outtext(zo,"repeats (-1 "); outtext(zo,"means calibr");
 67    20    ! ! ! outtext(zo,"ate) : #"); outend(zo);
 68    21    ! ! ! opwait(zi,oppool); ininteger(zi,repeats);
 69    22    ! ! !
 70    23    ! ! UNTIL ((period>=5) AND (period<=60)) AND (repeats>=-1);
 71    24    ! !
 72    25    ! ! outtext(zo,"CPUSE starts"); outnl(zo);
 73    26    ! !
```

```
 74   27    ! ! WHILE repeats <> 0 DO
 75   28    ! !   BEGIN (* measurement *)
 76   29    ! !     !
 77   30    ! !     ! rt.u3:= period; rt.u4:= 10;
 78   31    ! !     ! getregister(startcount,countreg);
 79   32    ! !     ! sendtimer(r); (* delay *)
 80   33    ! !     ! wait(r,timersem);
 81   34    ! !     ! getregister(endcount,countreg);
 82   35    ! !     !
 83   36    ! !     ! IF repeats < 0 THEN
 84   37    ! !     !   BEGIN (* calibration *)
 85   38    ! !     !     !
 86   39    ! !     !     ! fullcount:= msub(endcount,startcount) * 60 DIV period;
 87   40    ! !     !     ! outtext(zo,"count normal"); outtext(zo,"ized for 60 ");
 88   41    ! !     !     ! outtext(zo,"sec = #"); outinteger(zo,fullcount,5); outnl(zo);
 89   42    ! !     !     ! repeats:= 0;
 90   43    ! !     !     !
 91   44    ! !     !   END (* calibration *)
 92   45    ! !     !
 93   46    ! !     ! ELSE
 94   47    ! !     !
 95   48    ! !     !   BEGIN (* normal measurement *)
 96   49    ! !     !     !
 97   50    ! !     !     ! outtext(zo,"cpu free : #");
 98   51    ! !     !     ! outinteger(zo,msub(endcount,startcount) * 60 DIV period
 99   52    ! !     !     ! * 100 DIV fullcount,4);
100   53    ! !     !     ! outtext(zo," %#"); outnl(zo);
101   54    ! !     !     ! repeats:= repeats - 1;
102   55    ! !     !     !
103   56    ! !     !   END; (* normal measurement *)
104   57    ! !     !
105   58    ! !   END; (* while repeats <> 0 *)
106   59    ! !
107   60    ! UNTIL 5 = 7;
108   61    !
109   62    END; (* CPUSE *)
110         .
```

```
          0    1*   50    50    50    74    83    99
          1   29*   35*   49    49    49    49    49    70   101
          2    1*    1*   50    50
          3   34*
          4   99
          5   70    88   107
          6    1*   29*   29*
          7  107
          8   13*
         10   13*   77
         12   29*   29*
         20    4*
         25    1*
         60   70    86    98
         80   29*
         81   13*
        100   99
        225   22*
        300    6*
        600    1*
     100000    1*
a               44*
alfa            28*
alloc           54
b               44*
bcpuse           4*    5*    6*
char            29*
chars           29*
codedtdate      13*
countreg        21*   78    81
cpuse            9*
defaultfull     22*   56
endcount        40*   81    86    98
external        42*   44*
f               27*
finis            7*
free            50
fullcount       40*   56=   86=   88    99
getregister     42*   78    81
h3e6            21*
hlv              1*
index           42*
ininteger       65    68
integer         27*   40*   42*   42*   44*   44*
```

```
ioenvir          6*
job              1*
xbsem           37*   49
                27*
ist              3*
nini             1*    4*
node             3*
nsub            44*   86    98
                27*
name            28*
opbuf           26*   34*
openopzone      49    50
operatorsem     49    50
pin             62    65
ppool           34*   49    50    65    68
pwait           65    68
utdate          52
utend           64    67
utinteger       88    98
utnl            52    72    88    100
uttext          51    51    63    63    66    66    67    72    87    87    88    97    100
ascal80          6*
eriod           40*   65    70    70    77    86    98
erm              1*
ool             34*   35*
rocess           9*
roject           5*
                39*   54    77    77    79    80
ef              49    50
eference        39*
epeats          40*   68    70    74    83    89=   101=  101
cope             5*
emaphore        38*
endtimer        79
ize              1*
tack             6*
tartcount       40*   78    86    98
v                9*   49    50
ystem+vector
                 9*
ime              1*
imerpool        35*   54
imersem         38*   54    80
3               77=
```

<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

u4	77=													
version	13*	52												
wait	80													
yes	3*													
zi	36*	49	62	65	65	65	68	68						
zo	36*	50	50	51	51	52	52	63	63	64	66	66	67	67	72
	72	87	87	88	88	88	97	98	100	100				
zone	36*													

AND	2
ARRAY	1
BEGIN	4
CONST	1
DIV	3
DO	1
ELSE	1
END	5
FUNCTION	1
IF	1
OF	2
PROCEDURE	1
RECORD	1
REPEAT	2
SET	1
THEN	1
TYPE	1
UNTIL	2
VALUE	1
VAR	3
WHILE	1

B. EXAMPLE OF USE

```
>opsys
lookup sqrttest cpuse
PROCESS      sqrttest       1981.08.12 10.11  1 [illegible]
PROCESS      cpuse          1981.08.10 14.44  3 [illegible]
prio 2 run cpuse
>cpuse
CPU LOADMETER - VERS:  1981.08.10
period (5-60 sec) : 5
repeats (-1 means calibrate) : 100
CPUSE starts
>opsys
prio -3 size 5000 run sqrttest
>cpuse
cpu free :    96 %
cpu free :   101 %
>sqrttest
sqrt table:
>cpuse
cpu free :    58 %
>sqrttest
eps:             0.010000
>cpuse
cpu free :     5 %
>sqrttest
  1:      1.000
>cpuse
cpu free :    10 %
>sqrttest
  2:    1.50000

                          3
                          4
>cpuse
cpu free :     0 %
>sqrttest
                        -1
                         2
>opsys
stop sqrttest
>sqrttest
  -0.50000
>cpuse
cpu free :     0 %
cpu free :    64 %
cpu free :   101 %
>opsys
stop cpuse
>cpuse
cpu free :    96 %
```

RETURN LETTER

Title: RC3502 CPUSE RCSL No.: 52-AA1093

A/S Regnecentralen af 1979/RC Computer A/S maintains a continual effort to improve the quality and usefulness of its publications. To do this effectively we need user feedback, your critical evaluation of this manual.

Please comment on this manual's completeness, accuracy, organization, usability, and readability:

Do you find errors in this manual? If so, specify by page.

How can this manual be improved?

Other comments?

Name: _____ Title: _____

Company: _____

Address: _____

Date: _____

Thank you

42-i 1288

. . . . . Fold here . . . . .

. . . . . Do not tear - Fold here and staple . . . . .

Affix
postage
here

REGNECENTRALEN
af 1979

Information Department
Lautrupbjerg 1
DK-2750 Ballerup
Denmark