Title:

# BSSYSTEM

A/S REGNECENTRALEN

RC SYSTEM LIBRARY: FALKONERALLE 1 DK-2000 COPENHAGEN F

RCSL No: 31-D288

Edition: July 1972

Author: Lis Clement,
Vilhelm Rosenqvist

Keywords:

RC 4000, Disc-files, External Algol Procedure, sortbs, tapebs

Abstract:

The bssystem handles backing storage files of records. tapebs copies bssystem-files to/from magtape produced within as well as without MTS. Danish edition. 19 pages.



Printed by A/S Regnecentralen, Copenhagen

Indholdsfortegnelse



Litteraturhenvisninger


1. Magnetic Tape System MTS2, RCSL No. 31-D192
2. Mdsort, RCSL No. 28-D3
3. Sortbs under udarbejdelse *)


*) Sortbs virker som mdsort paanær flg:
   a) for input-fil skrevet af bs-system vil output-fil ogsaa blive skrevet med bs-system.
   b) parameteren <block> kan udelades, men kan evt. bruges til at ændre blocklength for output-filen.

Følgende fem ting har bevirket at discfiler med transaktioner er blevet mere almindelige

1) fremkomsten af procedurerne invar/outvar
2) større disckapacitet (bl.a. som følge af system 3)
3) discsortering
4) disc mere paalidelig end baand (stationer)
5) krav om bedre kørselsadministration (kørselsadministrations-problemet er proportionalt med kvadratet paa antal baand!!).

Der har derfor meldt sig et behov for et administrerende system omkring discfiler.

Af ønskede faciliteter kan nævnes

1) beskyttelse mod skrivning i filer aabnet til læsning (og omvendt)
2) fortsat skrivning i en fil
3) oplysninger om antal poster i en fil (til evt. sortering)
4) automatisk oprettelse og afkortning af filer (som fp-filer i system 3)
5) automatisk logudskrift med documentnavn, postantal m.m. (som ved heapsort)
6) bloklængdekontrol og updatemark test (se open_bs).

Med disse ønsker for øje blev følgende procedurer lavet

| | |
|---|---|
| open_bs | - til aabning for sekventiel behandling (læsning, skrivning og fortsat skrivning) |
| close_bs | - til lukning af filer aabnet med open_bs |
| sh_length_bs | - til beregning af sharelængde. |

Systemets administrative data gemmes i halen paa katalogindgangen, paa følgende form

| | |
|---|---|
| tail(1) | size (no of segment) |
| tail(2-5) | devicename |
| tail(6) | no of records in file |
| tail(7) | last block used |
| tail(8) | last byte used (in last block) |
| tail(9) | content.blocklength (segments) |
| tail(10) | updatemark |

ved at anbringe disse oplysninger i halen opnaas følgende

1) brugeren har nem adgang til disse ved hjælp af fp_programmerne lookup og set
2) no of records kan kommunikeres til sortbs via tail(6).

En standard integer variabel result_bs indeholder resultatet af et kald af en af procedurerne (s.d.).

Hvis en standard boolean variabel log_bs er sand vil open_bs og close_bs producere logudskrifter paa current output.

integer procedure open_bs proc. 1

aabner den angivne bs-fil

kald:

open_bs(z,doc,give_up,funktion)

| | |
|---|---|
| open_bs | (return value, integer)<br>antal poster i filen paa aabningstidspunktet<br>(tail(6))<br>(call and return value, zone) |
| z | efter kaldet beskriver z documentet |
| doc | (call value, string)<br>navnet paa et baggrundsareal |
| give_up | (call value, integer)<br>give up maske som i open |
| funktion | (call value, integer)<br>specificerer brugen af filen paa formen<br>postlængde shift 12 add mode<br>, hvor<br>postlængde=0 betyder invar/outvar anvendes<br>postlængde>0 betyder inrec/outrec med denne faste længde i bytes anvendes<br>(inrec/outrec med variabel længde forbydes, men kan ikke kontrolleres)<br>og<br>mode=0 læsning (hvis invar da med checksumkontrol)<br>=1 læsning (hvis invar da uden checksumkontrol)<br>=2 skrivning forfra (hvis outvar da med checksum)<br>=3 fortsat skrivning (hvis outvar da med checksum) |

krav:

| | fejlnr |
|---|---|
| 1) zonestate=4 (just after declaration) | 2 |
| 2) doc i overensstemmelse med monitorens krav | 4,5 |
| 3) overensstemmelse mellem programmet og halen paa følgende maade | |
| a) parameteren postlængde<=sharelængde ∧ sharelængde mindst 1 segment | 3 |
| b) sharelængde=tail(9) extract 12 medmindre skrivning forfra (incl. fortsat skrivning i tom fil) i dette tilfælde indsættes sharelængden i halen | 8 |
| 4) content=20 medmindre skrivning forfra (incl. fortsat skrivning i tom fil) i dette tilfælde accepteres content=0 og content ændres i halen til 20 | |

resultat:

zonen aabnet til doc, i tilfælde af fortsat skrivning er der positioneret til positionen opgivet i tail(7) og tail(8) (current record er tom)

zonestate:=if mode<2 then 5 else 6;

updatemark sat, hvis mode>1

zone_descr(11)=free param (se get_zone) indeholder oplysninger til brug ved beregninger af recordantal og checksumkontrol, dvs. zone_descr(11) ikke maa ændres af brugeren

result_bs=1 ok
=2 updatemark fandtes
=3 baggrundsarealet fandtes ikke, men er oprettet (fortrinsvis paa disc) hvis mode>1

alarmer:

1,2,3,4,5,6,7,8

integer procedure close_bs proc. 2

lukker den specificerede zone

kald:

close_bs(z,release,cut)

| | |
|---|---|
| close_bs | (return value, integer)<br>hvis læsning da antal læste poster (<>tail(6)<br>hvis ikke alle poster læst)<br>ellers antal poster i filen (=tail(6)) |
| z | (call and return value,zone)<br>specificerer documentet og mode (via zonestate) |
| release | (call value, boolean)<br>som ved close |
| cut | (call value, boolean)<br>hvis sand afkortes documentet til et helt<br>antal blokke |

krav:

| | fejlnr |
|---|---|
| 1) zonestate= 5 V zonestate=6 | 2 |
| 2) documentet skal eksistere | 5 |
| 3) content=20 | 7 |

resultat:

zonen lukkes, dvs. zonestate=4 (just after declaration)

hvis skrivning:
ajourføres halen, dvs.
1) updatemark slettes
2) aktuel position og postantal indsættes i tail(6-8)
og den sidste blok fyldes op med -8388608 (det mindste integer),
er den sidste blok fyldt helt op af brugeren dannes en blok, som
fyldes op med -8388608 (pseudo-eof, se side 9.3))

eventuelt afkortes documentet

result_bs=1 ok
=2 updatemark paa trods af læsning

alarmer:

2,5,6,7

integer procedure sh_length_bs proc. 3

beregner en korrekt sharelængde til det aktuelle document

kald:

```
sh_length_bs(doc)

sh_length_bs (return value,integer)
              hvis doc findes og bloklængden (i tail) *128
                   kan accepteres af open_bs som sharelængde
                   antager proceduren den værdi
                   (tail(9) extract 12*128)
              ellers 0
doc           (call value, string)
              navnet paa et baggrundsareal
```

krav:

| | fejlnr |
|---|---|
| 1) doc i overensstemmelse med monitorens krav | 5 |

NB! content testes ikke.

resultat:

```
result_bs=1  ok
         =2  uacceptabel bloklængde i tail(9) (sh_length_bs=0)
         =3  ukendt documentnavn (sh_length_bs=0)
```

alarmer:

5

Alarmer:

| fejlnr | tekst | integer | betydning | proc. |
|---|---|---|---|---|
| 1 | ill.mode | mode | mode>3 | 1 |
| 2 | z.state | z.state | illegal zonestate | 1,2 |
| 3 | s.length | zonens sh.l. | illegal sharelængde | 1 |
| 4 | create | monitor *) resultat | fejl under oprettelsen af et document | 1 |
| 5 | lookup | monitor *) resultat | fejl i forbindelse med et lookup-kald | 1,2,3 |
| 6 | change | monitor *) resultat | fejl ved ændring af tail | 1,2 |
| 7 | content | content 1 tail | illegal content | 1,2 |
| 8 | illegal bloklængde | b.længde, ant. shares, b.længde i tail | sharelængde i zone <> bloklængde i tail | 1 |

*) se forklaring i multiprogrammering-bogen.

Angaaende logudskrifter:

Logudskrifterne, der udskrives fra procedurerne open_bs og close_bs saafremt log_bs er sand, har normalt følgende udseende

<procedurenavn> on <doc> for <action>
tail is
size <tail(1)> device <tail(2-5)> no of records <tail(6)>
last block used <tail(7)> last byte used <tail(8)>
content <tail(9).1> blocklength <tail(9).2> updatemark <tail(10)>

Ting at tage vare paa:

1) positionering
systemet omfatter pt. ingen procedure til positionering, fordi man kan opstille flere rimelige positioneringsønsker (segmentnr, bloknr, postnr eller logisk position (dvs. postindhold)).

Hvis en positioneringsprocedure fremstilles skal den virke paa følgende maade for at garantere korrekt indhold i halen:

a) close_bs(z,false,cut)
b) open_bs(z,doc,give_up,læsning)
c) positioneringen
d) læsning herfra

eller

a) positioneringsproceduren maa selv ajourføre halen og zonen paa korrekt vis
b) læsning eller skrivning herfra

2) inrec/outrec kontra invar/outvar
systemet kan ikke kontrollere om der benyttes postlængder ved inrec/outrec forskellig fra den til open_bs specificerede værdi. Vi har alligevel medtaget inrec/outrec-muligheden, idet vi regner med at ingen nu da invar/outvar findes vil benytte variabel postlængde i forbindelse med inrec/outrec.

3) pseudo-eof
naar en bs fil har været brugt til skrivning, vil den ubenyttede del af den sidste blok altid være fyldt op med integer-værdien -8388608 (af close_bs). Dette kan af brugeren udnyttes som et logisk slutmærke paa filen.
Logisk slutmærke kan testes i en blokprocedure, idet invar vil kalde denne med bit 12 sat i statusordet, hvis

a) recordchecksumfejl (testes kun hvis mode=0 open_bs)
b) recordlængdefejl
I) recordlængde>remaining *)
II) recordlængde<0

kun i tilfælde bII, hvor recordlængde (og resten af blokken) er lig -8388608 kan man med en til vished grænsende sandsynlighed regne med, at man har mødt det logiske slutmærke.

NB!!
da open_bs afleverer antal poster i filen, anbefales det at benytte sig af dette, fremfor at lave en blokprocedure, som den ovenfornævnte!

*)
det antal bytes, der endnu ikke er benyttet i aktuel blok.

programeksempel:
Paa efterfølgende sider er gengivet et program, til kopiering af transaktionsfiler, skrevet med outvar, mellem baand og disc.

Discfilerne er bsfiler, mens magnetbaandsfilerne kan være skrevet med eller uden MTS. Med programmet kan man derfor konvertere til/fra MTS-filer ved at bruge disc som mellemled. Hovedformaalet med at lave programmet er ønsket om at kunne køre disc-sortering selvom output afleveres paa baand.

Programkald:
tapebs <funktion>.<tapespec> <bsspec>
<funktion> ::= fromtape|frommts|frmul|
totape|tomts|tomul|
<tapespec> ::= <tapename> {.<første fil> {.<sidste fil>}$_0^1$}$_0^1$
<bsspec> ::= <bsfilname> {.continue {.cut}$_0^1$}$_0^1$
<tapename> ::= et baandnavn (af formen mtxxxxxx)
<bsfilname> ::= et baggrundsarealnavn

betydning

| | |
|---|---|
| fromtape: | der læses fra baand til disc. der aabnes med 'open' til baand og med open_bs til disc |
| frommts: | som fromtape, blot aabnes til baand med open_adp |
| frmul: | som frommts, men foruden det opgivne baandnavn forsynes mts med to extra navne kaldet mthelp1 og mthelp2. Dette er en enkel (men daarlig) løsning paa multivolume problematiken. |
| totape: | modsat fromtape idet der læses til baand fra disc |
| tomts: | modsat frommts |
| tomul: | modsat frmul |
| <første fil>: | et tal der angiver filnummeret paa baandet (standard er 1) |
| <sidste fil>: | hvis der læses fra baand kan der angives en sidste fil (et tal), og programmet vil da læse alle filerne fra første - til sidste - fil ned i samme bs-fil |
| continue: | kan angives hvis der læses fra baand, der aabnes da med mode = 3 til bsfilen i stedet for mode = 2, og der closes med cut = false. (betydningen heraf: se open/close_bs) |
| cut: | kan angives hvis der køres med continue, og bevirker at der alligevel closes med cut=true. |

continue kan bruges hvis man vil flette flere filer paa forskellige magnetbaand sammen i en bs_fil.

cut anvendes i det sidste af flere kald, hvor filer er flettet sammen, for at faa afkortet filen, der maaske er sat kæmpestor fra starten (især under system2, hvor udvidelse af filer ikke er mulig).

Programudskriften fylder flere sider, hvor de første er indlæsning af fp_parametre, og kun den sidste (side) indeholder kopieringen, hvor brugen af bs_systemet ses.

Det vedlagte output er fra en kørsel, hvor en magnetbaandsfil (MTS) læses ned paa disc, sorteres og derefter udskrives som en MTS-fil.

Betydningen af udskrifterne fra programmet kan, hvis de ikke er umiddelbart forstaaelige, udledes fra programudskriften.

```
 1 begin
 2   boolean  fromtape,continue,cut,mts,multi;
 3   integer  bsblock, file1, file2, fpnumber, fpparam, i, i1, i2,
 4     paramnumber,error, pointname, pointnumber, spacename,
 5     tapeblock, tapeshares;
 6   real  fpname;
 7   real array  bsname,tapename, rarr(1:2), ident(1:18);
 8
 8   procedure  fpstop;
 9     system(9, paramnumber, <:<10>fp param:>);
10
10
10   procedure  nextfp;
11   begin
12     paramnumber:= paramnumber + 1;
13     fpparam:= system(4, paramnumber, rarr);
14
14     if fpparam <> 0 then
15     begin
16       write(out, if fpparam shift (-12) = 4 then  <:  :>
17         else  <:.:>);
18
18       if fpparam extract 12 = 4 then
19       begin
20         fpnumber:= rarr(1);
21         write(out, <<d>, fpnumber);
22       end
23       else
24       begin
25         fpname:= rarr(1) shift (-8) shift 8;
26         i:= 1;
27         write(out, string rarr(increase(i)));
28       end;
29     end  fpparam <> 0;
30   end  nextfp;
31
31   pointname:=    8 shift 12 add 10;
32   pointnumber:=  8 shift 12 add  4;
33   spacename:=    4 shift 12 add 10;
34
34
34 comment    standard values;
35   tapeshares:= 2;
36   tapeblock:= bsblock:= 512;
37   file1:= file2:= 1;
38   fromtape := mts := multi := false ; error := 0 ;
39
39   system(2, i, rarr);
40   i:= 1; write(out, <:<10>:>, string rarr(increase(i)));
41
41   paramnumber:= 0;
42   nextfp;
43
43   if fpname = real<:fromt:> then  fromtape:= true
44   else
45   if fpname = real<:totap:> then  fromtape:= false
46   else
47   if fpname = real<:fromm:> then mts := fromtape := true
48   else
49   if fpname = real<:tomts:> then mts := true
50   else
51   if fpname = real<:frmul:> then mts := multi := fromtape := true
52   else
53   if fpname = real<:tomul:> then mts := multi := true
54   else
55     fpstop;
56
```

```
56
56   nextfp;
57   if fpparam <> pointname then  fpstop;
58   ident(17):=ident(18):=real<::>;
59   ident(16):= rarr(1) shift 16 add
60               (rarr(2) shift (-32) extract 16);
61   tapename(1) := rarr(1) ; tapename(2) := rarr(2) ;
62
62   nextfp;
63   if fpparam = pointnumber then
64   begin
65     file1:= fpnumber;
66     if file1 < 1 then  fpstop;
67     nextfp;
68     if fpparam = pointnumber then
69     begin
70       if -, fromtape then fpstop ;
71       file2:= fpnumber;
72       if file2 < file1 then  fpstop;
73       nextfp;
74     end;
75   end;
76
76 if file1=1 and file2=1 and multi then
77 begin
78   comment multivolumenfile mulig ;
79   ident(17):=real<:help1:>;
80   ident(18):=real<:help2:>;
81 end ;
82
82   if fpparam <> spacename then  fpstop;
83   for i:= 1, 2 do  bsname(i):= rarr(i);
84
84   continue:= false ;
85   cut:= true;
86   nextfp;
87   if fpparam = 0                                        then        else
88   if fpparam extract 12 = pointnumber extract 12  then fpstop else
89   if fpname <>real<:conti:>                             then fpstop else
90   if -, fromtape                                        then fpstop else  continue := true;
91
91   if continue then
92   begin
93   nextfp;
94     if fpparam = 0                                       then cut := false else
95     if fpparam extract 12 = pointnumber extract 12 then fpstop       else
96     if fpname <>  real<:cut:>                            then fpstop       else cut:=true;
97   end ;
98
98   logbs:= true;
99   i1:= i2:= 1;
100
100  if shlengthbs(string bsname(increase(i1))) > 0 then
101    bsblock:= shlengthbs(string bsname(increase(i2)));
102
```

```
102
102 begin
103   zone  bs(2*bsblock, 2, bsproc);
104
104   zone  tape((if multi then 18 else if mts then 16 else 0) +
105                tapeshares * tapeblock, tapeshares, tapeproc);
106
106   integer  bytes, file, i, recs;
107   integer  field  varlength;
108
108
108   procedure  bsproc(z, s, b);
109   zone  z; integer  s, b;
110   begin
111     own boolean  latercall;
112     if -,latercall then
113     begin
114       latercall:= true;
115       write(out, <:<10>bs status, bytes: :>, << ddddd>, s, b,
116         <:<10>recs, bytes, segments: :>, recs, bytes,
117         (bytes - 1)//512 + 1);
118       if s extract 1=1 then error := error + 1000 ;
119       if -, fromtape then goto endfile ;
120     end;
121     b:= 4*bsblock;
122   end  bsproc;
123
123
123   procedure  tapeproc(z, s, b);
124   zone  z; integer  s, b;
125 if b > 0 or mts or s extract 1 = 1 then
126   begin
127     write(out, <:<10>tape status, bytes: :>, << ddddd>, s, b,
128       <:<10>file, recs, bytes: :>, file, recs, bytes);
129     if mts and s shift (-1) <> 0 or
130     -, mts and s extract 1=1 then error := error + 1 ;
131     goto endfile;
132   end  tapeproc;
133
```

```
133
133   varlength:= 2;
134   i1:= i2:= 1;
135
135   recs:= openbs(bs, string bsname(increase(i1)), 0,
136         if continue then 3 else if fromtape then  2 else  1);
137
137   if fromtape then  recs:= 0;
138   ident(1):= 0;
139   ident(2):= ident(3):= if file1=1 and file2=1 and multi then 3 else 1 ;
140   ident(4):= 1 + 1 shift 1 + 1 shift 4 + 1 shift 5
141            + 1 shift (-12) + 1 shift (-15) + 1 shift 16;
142
142   i:= 1 ;
143   if mts then openadp(tape, ident, true) else
144    open(tape,18,string tapename(increase(i)),if fromtape then 1 shift 16 else 0) ;
145   file:= file1;
146   bytes:= 0;
147
147 postape:
148   if mts then positionadp(tape, file, if fromtape then  1 else  2) else
149   setposition(tape,file,0) ;
150
150   if fromtape then
151   begin
152
152 readtape:
153     invar(tape);
154     outvar(bs, tape);
155     recs:= recs + 1;
156     bytes:= bytes + bs.varlength;
157     goto readtape;
158
158 endfile:
159     if file < file2 then
160     begin
161       file:= file + 1;
162       goto postape;
163     end;
164   end  fromtape
165   else
166
166   begin
167     for i:= 1 step 1 until recs do
168     begin
169       invar(bs);
170       outvar(tape, bs);
171       bytes:= bytes + tape.varlength;
172     end;
173   end  totape;
174
174   closebs(bs, true, cut);
175   if mts then closeadp(tape, 0) else close(tape,false) ;
176   write(out, <:<10><10>ready, recs, bytes, segments: :>, << ddddd>,
177     recs, bytes, (bytes - 1)//512 + 1);
178   if error <> 0 then system(9,error,<:<10>hard_err:>) ;
179 end
180 end

algol end 97
```

Kørselseksempel

```
*OPMESS MT281375 UDEN RING
*FNUDFIL=SET 8000 DISC 0 0 0 20.8
*IF OK.NO
*TAPEBS FROMMTS.MT281375.1 FNUDFIL

TAPEBS  FROMMTS.MT281375.1  FNUDFIL
OPENBS ON FNUDFIL FOR OUTPUT
TAIL IS
SIZE 8028 DEVICE DISC NO OF RECORDS 0
LAST BLOCK USED 0 LAST BYTE USED 0
CONTENT 20 BLOCKLENGTH 8 UPDATEMARK 0

TAPE STATUS, BYTES:  65536    56
FILE, RECS, BYTES:       1 87022 5178858
BEFORE CLOSEBS ON FNUDFIL FOR OUTPUT
TAIL IS
SIZE 10152 DEVICE DISC NO OF RECORDS 0
LAST BLOCK USED 0 LAST BYTE USED 0
CONTENT 20 BLOCKLENGTH 8 UPDATEMARK 1

AFTER CLOSEBS ON FNUDFIL
TAIL IS
SIZE 10188 DEVICE DISC NO OF RECORDS 87022
LAST BLOCK USED 1269 LAST BYTE USED 2462
CONTENT 20 BLOCKLENGTH 8 UPDATEMARK 0


READY, RECS, BYTES, SEGMENTS:  87022 5178858 10115
HARD ERR     1  LINE 178-180
*RELEASE MT281375
*SORTBS IN.FNUDFIL.CLEAR OUT.SORTFNUD VAR.100 WORD.6,
   LONG.10 LONG.14 LONG.18 LONG.22 LONG.26
```

```
SORTBS  IN.FNUDFIL.CLEAR  OUT.SORTFNUD,
        VAR.100,
        WORD.6,
        LONG.10,
        LONG.14,
        LONG.18,
        LONG.22,
        LONG.26
BS SYSTEM   CONTENT:  20          BLOCK:   8   8

SORT START:

***SORTPROCBS WARNING:    ONLY BUFS   8

SORT 1.  22 MIN. LEFT

SORT 2.  18 MIN. LEFT

SORT OK:
OUTPUT  RECORDS:   87022       SEGMENTS:   10184
MINUTES    REAL:   30.33            CPU:   17.67
END   218
*IF OK.NO
*OPMESS MT281306 MED RING
*TAPEBS TOMTS.MT281306.1 SORTFNUD

TAPEBS  TOMTS.MT281306.1  SORTFNUD
OPENBS ON SORTFNUD FOR INPUT
TAIL IS
SIZE 10188 DEVICE KIT11 NO OF RECORDS 87022
LAST BLOCK USED 1272 LAST BYTE USED 1890
CONTENT 20 BLOCKLENGTH 8 UPDATEMARK 0

BEFORE CLOSEBS ON SORTFNUD FOR INPUT
TAIL IS
SIZE 10188 DEVICE KIT11 NO OF RECORDS 87022
LAST BLOCK USED 1272 LAST BYTE USED 1890
CONTENT 20 BLOCKLENGTH 8 UPDATEMARK 0

POSITION ON SORTFNUD
 NO OF RECORDS 87022 LAST BLOCK USED 1272 LAST BYTE USED 1890


READY, RECS, BYTES, SEGMENTS:  87022 5178858 10115
END    79
```