
RCSL No: 31-D602

Edition: June 1980

Author: Edith A. Rosenberg

Title:

RC8000 SQ-SYSTEM

Keywords:

RC8000, algol, backing storage package, sequential files, manual.

Abstract:

The SQ-SYSTEM is a set of external algol procedures, which facilitate sequential access to a backing storage file in connection with the record i/o procedures from the algol library.

(24 printed pages)

Copyright © 1980, A/S Regnecentralen af 1979
RC Computer A/S

Printed by A/S Regnecentralen af 1979, Copenhagen

Users of this manual are cautioned that the specifications contained herein are subject to change by RC at any time without prior notice. RC is not responsible for typographical or arithmetic errors which may appear in this manual and shall not be responsible for any damages caused by reliance on any of the materials presented.

FOREWORD (RCSL No 31-D 561):

The SQ-SYSTEM replaces the BS-SYSTEM (RCSL No 31-D 288), which is non-compatible with RC8000 standard software. Procedure formats are not changed, but an SQ-file holds a file head and the use of the catalog entry differs from that of a BS-file.

The system may be used at an RC4000, having SYSTEM 3 monitor, an RC6000, or at any RC8000 model.

A/S Regnecentralen, April 1979

Inge Borch

Second edition (RCSL No 31-D 589):

This edition contains corrections of some essential typographical errors (page 8 and 9), and an extended description of results from opensq.

Third edition (RCSL No 31-D 602):

This edition contains corrections of minor typographical errors, and an explanation of the word block.

A/S Regnecentralen af 1979, june 1980

Edith Rosenberg.

THE UNIVERSITY OF CHICAGO

1965

...

...

...

...

...

...

...

...

...

TABLE OF CONTENTS	PAGE
1. INTRODUCTION	1
1.1 Sequential Record Access within the Algol System .	1
1.2 The SQ-SYSTEM's Facilities	1
2. SQ-FILE FORMAT	3
2.1 SQ-Catalog Entry	3
2.2 SQ-Area	3
3. FILE HANDLING PROBLEMS	5
3.1 Creation	5
3.2 Record Processing	5
3.3 Sorting	6
4. PROGRAMMING GUIDE	7
4.1 Integer Procedure Closesq	7
4.2 Integer Procedure Opensq	8
4.3 Integer Resultsq	10
4.4 Integer Procedure Shlengthsq	10
A. REFERENCES	12
B. MESSAGES	13
C. FILE HEAD FORMAT	15
C.1 Layout of SQ-File Head Version 0-12	15
D. PROGRAMMING EXAMPLE	16

1. The first part of the report
 2. The second part of the report
 3. The third part of the report
 4. The fourth part of the report
 5. The fifth part of the report
 6. The sixth part of the report
 7. The seventh part of the report
 8. The eighth part of the report
 9. The ninth part of the report
 10. The tenth part of the report
 11. The eleventh part of the report
 12. The twelfth part of the report
 13. The thirteenth part of the report
 14. The fourteenth part of the report
 15. The fifteenth part of the report
 16. The sixteenth part of the report
 17. The seventeenth part of the report
 18. The eighteenth part of the report
 19. The nineteenth part of the report
 20. The twentieth part of the report
 21. The twenty-first part of the report
 22. The twenty-second part of the report
 23. The twenty-third part of the report
 24. The twenty-fourth part of the report
 25. The twenty-fifth part of the report
 26. The twenty-sixth part of the report
 27. The twenty-seventh part of the report
 28. The twenty-eighth part of the report
 29. The twenty-ninth part of the report
 30. The thirtieth part of the report
 31. The thirty-first part of the report
 32. The thirty-second part of the report
 33. The thirty-third part of the report
 34. The thirty-fourth part of the report
 35. The thirty-fifth part of the report
 36. The thirty-sixth part of the report
 37. The thirty-seventh part of the report
 38. The thirty-eighth part of the report
 39. The thirty-ninth part of the report
 40. The fortieth part of the report
 41. The forty-first part of the report
 42. The forty-second part of the report
 43. The forty-third part of the report
 44. The forty-fourth part of the report
 45. The forty-fifth part of the report
 46. The forty-sixth part of the report
 47. The forty-seventh part of the report
 48. The forty-eighth part of the report
 49. The forty-ninth part of the report
 50. The fiftieth part of the report
 51. The fifty-first part of the report
 52. The fifty-second part of the report
 53. The fifty-third part of the report
 54. The fifty-fourth part of the report
 55. The fifty-fifth part of the report
 56. The fifty-sixth part of the report
 57. The fifty-seventh part of the report
 58. The fifty-eighth part of the report
 59. The fifty-ninth part of the report
 60. The sixtieth part of the report
 61. The sixty-first part of the report
 62. The sixty-second part of the report
 63. The sixty-third part of the report
 64. The sixty-fourth part of the report
 65. The sixty-fifth part of the report
 66. The sixty-sixth part of the report
 67. The sixty-seventh part of the report
 68. The sixty-eighth part of the report
 69. The sixty-ninth part of the report
 70. The seventieth part of the report
 71. The seventy-first part of the report
 72. The seventy-second part of the report
 73. The seventy-third part of the report
 74. The seventy-fourth part of the report
 75. The seventy-fifth part of the report
 76. The seventy-sixth part of the report
 77. The seventy-seventh part of the report
 78. The seventy-eighth part of the report
 79. The seventy-ninth part of the report
 80. The eightieth part of the report
 81. The eighty-first part of the report
 82. The eighty-second part of the report
 83. The eighty-third part of the report
 84. The eighty-fourth part of the report
 85. The eighty-fifth part of the report
 86. The eighty-sixth part of the report
 87. The eighty-seventh part of the report
 88. The eighty-eighth part of the report
 89. The eighty-ninth part of the report
 90. The ninetieth part of the report
 91. The ninety-first part of the report
 92. The ninety-second part of the report
 93. The ninety-third part of the report
 94. The ninety-fourth part of the report
 95. The ninety-fifth part of the report
 96. The ninety-sixth part of the report
 97. The ninety-seventh part of the report
 98. The ninety-eighth part of the report
 99. The ninety-ninth part of the report
 100. The hundredth part of the report

1. INTRODUCTION

1.

1.1 Sequential Record Access within the Algol System

1.1

A backing storage file consists of a number of physical blocks, called segments. When the file is accessed from a program, an integral number of segments, the logical block, is transferred to a zone in the program from the file, or reversed, by the internal algol i/o system, which is explained in details in ref. 1. The high level zone procedures for record i/o, inrec6, outrec6, changerec6, invar, outvar, changevar, and checkvar, may be used for sequential record access, and are usually enclosed by the zone control procedures open and close. (See ref. 1 and 2). The logical block length is implicitly defined by the zone declaration, and the logical length of the file is also settled by the user's program, when it differs from the physical one.

When the word block is used alone in this manual, it always means logical block.

1.2 The SQ-SYSTEM's Facilities

1.2

While the above mentioned procedures are designed for sequential record access on any kind of document, the SQ-SYSTEM is made especially for backing storage to handle some trivial problems, the user otherwise will meet.

For that purpose the system maintains a file head and the catalog entry with information about the user's definition of the file structure and the use of the file.

The most important kinds of informations are:

- .logical blocklength
- .logical length of the file
- .number of records
- .update mark

These are utilized for

- .transferring of file definitions between programs
- .extended writing in 'old' files
- .parameter for an effective sorting strategy
- .releasing of unused segments in the end of a file
- .security against restart on a 'broken' file.

The system holds three procedures and a standard variable, which is used to test the result of the procedure calls. The procedure `shlengthsq` yields the logical blocking of the file. The procedure `opensq` replaces the algol procedure `open` (ref. 2), settles the file definitions, and defines the access mode (reading, or writing from the beginning or from the logical end of the file). The procedure `closesq` replaces the algol procedure `close` (ref. 2), updates the logical length of the file and releases unused segments.

2. SQ-FILE FORMAT

2.

2.1 SQ-Catalog Entry

2.1

The file is described as an area in the catalog of the disc, but the catalog entry tail holds more information than that of a standard area. Fig. 1.

tail word	
1	no of segments in area
2-5	device name
6	shortclock
7	0
8	0
9	21. segsperblock
10	no of records

Figure 1: SQ-area entry

Segsperblock is the blocklength expressed in segments and this quantity is packed with the contents key 21, which is unique for SQ-files. The number of records in the file is shown in tail word 10.

Shortclock will be updated when the file head is created and when the file is closed after writing.

2.2 SQ-Area

2.2

Figure 2 shows an area holding an SQ-file.

file head	first block	last block	unused
1	segsperblock	segsperblock	rest (≥ 0)

Figure 2: SQ-Area

3.3 Sorting

3.3

The records of an SQ-file may easily be sorted by the mdsort system. (Ref. 3). The system may fetch the blocklength and the number of records from the file and deliver the sorted records in an SQ-file.

4. PROGRAMMING GUIDE

4.

This section holds the user description of the SQ-SYSTEM, viz. the procedure opensq, closesq, and shlengthsq, and the standard integer resultsq.

The user must be familiar with the algol i/o system (ref. 1) in reading this section.

Note that if 'requirements' are violated, the system will abort with an alarm message, (see app. B). A primitive programming example is shown in app. D.

4.1 Integer Procedure Closesq

4.1

Terminate the use of an SQ-file.

Call: closesq (z, release, cut)

closesq (return value, integer). If close after reading, then the number of records accessed since opening, else the number of records in the file.

z (call and return value, zone). As for the algol procedure close (ref. 2).

release (call value, boolean). As for the algol procedure close.

cut (call value, boolean). If cut=true and zone-state = 6 (writing) the file will be cut to an integral number of blocks, else this parameter is ignored.

Functions:

If zonestate is 5 (after reading), the closing is performed as for the algol procedure close. If zonestate is 6 (after writing), the last block (possibly a whole block) is filled with all bits set on, the block is transferred to the file, maybe the file is cut, the

file head and the catalog entry tail are updated and finally normally closing is performed.

Requirements:

1. zonestate must be 5 or 6.
2. the file must be an SQ-file.

Results:

resultsq = 1 ok,
 2 updatemark was sensed after
 reading.
 zonestate = 4 (after declaration).

4.2 Integer Procedure Opensq

4.2

Makes an sq-file available for recordprocessing by the standard i/o procedures invar, outvar, changevar, inrec6, and outrec6.

Call: opensq (z, doc, giveup, function)

opensq (return value, integer). The number of records in the file.

z (call and return value, zone). As for the algol procedure open. (Ref. 2).

doc (call value, string). As for the algol procedure open.

giveup (call value, integer). As for the algol procedure open.

function (call value, integer). This integer parameter is used as two halfwords, which specify the use of the file:

function:= recordsize shift 12 add mode;

Recordsize = 0 indicates that the procedures invar, outvar, or changevar will be used after the opening.

Recordsize > 0 indicates that the procedures inrec6 or outrec6 will be used after the opening and recordsize is the number of half-words in each record.

Mode = 0, means reading inclusive checking of the checksum.

Mode = 1, reading exclusive checking of the checksum. This mode is always used in connection with inrec6.

Mode = 2, writing from the beginning of the file and with checksum, if outvar/changevar is used.

Mode = 3, writing from the logical end-of-file and with checksum, if outvar/changevar is used.

Functions:

The parameters are checked. The file is looked up in the catalog, and if no entry is found and mode is 2, an area creation is attempted.

The catalog entry tail is interpreted. The filehead is checked or created, if writing in a new file is wanted. Finally the file is positioned, and the zone prepared for record access.

Requirements:

1. Zonestate = 4 (after declaration).
2. The filename (doc) must fulfil the requirements of the monitor.
3. If mode <> 2 the tail and the filehead must agree with the SQ-system conventions.
4. The sharelenght of z must be able to hold any occuring record and the block.

Results:

```

resultsq = 1  ok,
          = 2  updatemark was found,
          = 3  file not found in the catalog,
                but if mode >1, it is created
                and ready for use.
zonestate = if mode <2 then 5 else 6.

```

zonedescrptor(11), free parameter, holds information about the number of records and checksum. (It is used by closesq, so it should not be changed by the user).

If opensq is called with recordsize >0, the recordsize is stored in zonedescrptor(11). In this case outvar must not be used, as this would destroy be recordsize. The SQ-system does not check that outvar is not used in this case.

The update mark is set if mode >1.

4.3 Integer Resultsq

4.3

Yields the result of the latest call of one of the sqprocedures, and should always be checked by the user.

4.4 Integer Procedure Shlengthsq

4.4

Calculates the shrelength for an SQ-file.

Call: shlengthsq (doc)

shlengthsq (return value, integer).

If a file named doc exists, and the quantity segsperblock in the catalog entry tail is reasonable, the value will be the blocklength expressed in doublewords, otherwise it will be zero.

doc (call value, string). The name of a backing storage area, holding an SQ-file.

Functions: The name 'doc' is looked up in the catalog, and if it exists, the sharelength (block-length) is calculated from tail (9).

Requirements: 'doc' must describe a backing storage area if it exists.

Results:

- resultsq = 1 ok,
- = 2 error in tail(9)
- = 3 unknown documentname.

A. REFERENCES

A.

1. RCSL No 42-i 0781, January 1979, Bodil Larsen
ALGOL7, User's Manual, Part 1.
2. RCSL No
ALGOL8, User's Manual, Part 2.
If it is not yet available, please use:
RCSL No 21-D 322:
ALGOL6, User's Manual, Chapter 9.
3. RCSL No 31-D 562, April 1979, Jørgen Winther
RC8000 Backing Storage Area Sorting.
4. RCSL No 31-D 477, January 1978, Tove Ann Aris
RC8000 Monitor, Part 2, Reference Manual.

B. MESSAGES

B.

In the case that a requirement of an SQ-procedure is violated, a message is printed at current output, and the system exits via system (9,..... (See ref. 2).

The format of the messages is:

```
***<SQ-procedure> on: <filename>
<text> <n>
```

The values of <text> and the associated <n> are:

<text>	explanation
change	Change entry impossible. (See ref. 2 and ref. 4, monitor function 44). <n> is the monitor result.
contents	Illegal contents value in catalog entry. <n> is the value found.
create	Create entry impossible. (See ref. 2 and ref. 4, monitor function 40). <n> is the monitor result.
cut	Change entry impossible, when the file should be cut. (See ref. 2 and ref. 4, monitor function 44). <n> is the monitor result.
ht-conf1	Segsperblock in the entry tail differs from segsperblock in the file head. <n> is segsperblock from the entry.
ill.mode	Mode > 3 or (mode = 0 and recordsize > 0). <n> is the mode value.
lookup	Lookup of the file without success. (See ref. 2 and ref. 4, monitor function 42). <n> is the monitor result.

<text>	explanation
reccount	No of records exceeds 4194303 or is negative. <n> is no of records from the file head.
recsize	Recordsize in the call differs from the record size in the file head. <n> is the recordsize from the file head.
s.length	The sharelength is not an integral number of segments or recordsize > sharelength. <n> is the sharelength calculated in double words.
spb-head	Segsperblock in the file head differs from segsperblock defined by the zone. <n> is segsperblock from the file head.
sq-sum	An internal SQ-sumcheck tells that the file head has been violated. <n> is irrelevant.
sqvers	The version of the SQ-SYSTEM used is older than the version, which has written the file.
z.state	Zonestate is illegal for the procedure. <n> is the zonestate value.

C. FILE HEAD FORMAT

C.

The file head is a record of variable length kept in the n first segments of the SQ-area. ($n = 1$ in version 12). It consists of several parts of variable length, which may be addressed by a link. Each part will have a maximum length of 40 halfwords.

C.1 Layout of SQ-File Head Version 0-12

C.1

part no	rel. word	Contents
0	1	recordlength
	2	checksum
1	1	field address of next part
	2	sq-sum
	3	sq-version
	4	no of segments reserved for file head
2	1	field address of next part
	2	
	.	
	.	entry tail at file head creation
	.	
	11	
	12	recordsize at file head creation
3	1	field address of next part
	2	file extension count
	3	no of records
	4	segment no of last block
	5	last halfword in last block
	6	update mark
	7	shortclock
4	1	field address of next part (= 0)

D. PROGRAMMING EXAMPLE

D.

```

begin
comment example: continued writing in the sq-file
    <:outsq:>, which may have been created thus:
        outsq = set <segs><device> 0 0 0 21. <s.p.b>
    the new records are read from the sq-file
    <:insq:>

;
integer iblock, oblock;
    iblock:= sh_length_sq(<:insq:>);
    oblock:= sh_length_sq(<:outsq:>);
    if iblock = 0 or oblock = 0 then
        write(out,<:<10> bad start:>)
    else
    begin
        integer irecs, orecs, i;
        zone zi(2*iblock, 2, stderr), zo(2*oblock, 2, stderr);
        irecs:= open_sq(zi, <:insq:> 0, 0);
        if result_sq <> 1 then
            system(9, result_sq, <:<10>open i:>);
        orecs:= open_sq(zo, <:outsq:>, 0, 3);
        if result_sq <> 1 then
            system(9, result_sq, <:<10>open o:>);
        for i:= 1 step 1 until irecs do
        begin
            invar (zi);
            outvar (zo, zi);
        end;
        if close_sq(zo, true, true) <> irecs+orecs then
            write(out, <:<10>mystery:>);
        close_sq(zi, true, false);
        if result_sq <> 1 then
            write(out, <:<10> updatemark in insq:>);
    end;
    write (out, <:<10>:>);
end example

```

RETURN LETTER

Title: RC8000 SQ-SYSTEM

RCSL No.: 31-D602

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

..... Fold here

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

Affix
postage
here

 **REGNECENTRALEN**
af 1979

Information Department
Lautrupbjerg 1
DK-2750 Ballerup
Denmark