RCSL No: 31-D634

Edition: February 1981

Author: Edith Rosenberg

Title:

Corrections to RCSL No 31-D600 RC8000 Indexed Sequential Files (ISQ).

A/S REGNECENTRALEN af 1979

Keywords:

RC8000, Backing Storage Package, Indexed Sequential Files, corrections.

Abstract:

This paper describes the manual opdates due to changes in the software package and misprintings in the original manual (RCSL No 31-D600).

(18 printed pages)

RCSL 42-i1592

FOREWORD

First edition: RCSL No 31-D634
This set of corrections can be used as single-leaf cancels in RCSL No 31-D600: RC8000 Indexed Sequential Files (ISQ).

The changes are indicated by correction lines in the left margin.

The main reason for these changes is the new parameter possibilities for the procedures getparamsi and setparamsi.

Edith Rosenberg
A/S Regnecentralen af 1979, February 1981

CONTENTS	PAGE	CHANGED IN
		edition 1
	25	–
	26	x
	33	x
	34	–
	37	x
	38	–
	39	–
	40	x
	45	x
	46	x

6. PROCEDURE SPECIFICATIONS 6.

This chapter contains, in alphabetic order, the specifications of all the procedures offered by the system. To each file processing procedure is assigned a number, procno_i, by which the procedure is identified in the use of the test facilities (see section 6.15).

A survey of the procedures, in procno_i order, is given in Appendix A together with the possible result_i values, their meaning, and the corresponding values of available record.

6.1 Integer Procedure buflengthi 6.1

Call: buflength_i (filename, full_insert)

buflength_i	(return value, integer). Number of double-worditems needed in the zone buffer for processing the indexed-sequential file given by filename.
filename	(call value, string). The name of a backing storage area containing an indexed-sequential file.
full_insert	(call value, boolean). True if a buffer with room for general insertions is wanted.

Function: Reads the first segments of the document given by filename into a local zone and computes the needed buflength. The area is not released.

Errors: Uses stderror and giveup = 0. If the needed parameters in the file head do not conform to an indexed-sequential file buflength_i will yield the value zero.

6.2 Procedure deletereci

Call: delete_rec_i (z)

z	(call and return value, zone). Specifies the file.

Function: Deletes the available record from the file and makes the successor available.

Requirements: zonestate = update_i or put_i.

Results: zonestate: unchanged.
procno_i : 9

result_i :	Available record:
1 Deleted	The successor to the available.
2 Deleted, end of file	The first in the file.
3 Not deleted, only one record left	The one.

6.3 Integer Procedure getparamsi

Call: get_params_i (z) One or more pairs:(paramno, val)

get_params_i	(return value, integer). Overall result of call: 0 : All parameters processed. > 0: Exit on error in parameter pair number get_params_i.
z	(call value, zone). Specifies the file.
paramno	(call value, integer or long). Identifies the wanted value.
val	(return value, integer or long). Receives the value of a parameter in the zone buffer identified by paramno. If the value in the zone buffer is of type long, but val is an integer, only the rightmost 24 bits of the value are returned.

3	Not inserted, too expensive, see below.	The successor to the specified
4	Not inserted, file is full.	The successor to the specified
5	Not inserted, improper length	The successor to the specified
6	Not inserted, there was no room for the record in the block to which it belonged and the zone buffer is too small for a more complicated insertion, see below.	The successor to the specified

6.8.1 Insertion Strategy 6.8.1

If there is room for the record in the block to which it belongs, it can be inserted without further trouble; otherwise a more complicated strategy is used. This requires an extra block in the zone buffer. Unless this block is present it is therefore pure luck if the insertion succeeds.

The following describes the full insertion strategy, it may be skipped unless you want to modify it.

The organization of the file requires that records are stored in keyorder. This means that the insertion of a new record in general will involve a reorganization of some parts of the file in order to get room for the record in the proper block.

The cost of an insertion, in terms of segment transports and other use of resources, depends strongly on how this reorganization is done. The insertion algorithm implements the following scheme which, by taking prices imposed on the involved resources into account, tries to strike a reasonable balance between a fully automatic and a user controlled strategy.

The file head holds a list of relative prices imposed on resources and with initial values assigned by head_file_i:

Name, initial value:	Meaning:
emptybuckprice,	The value of having an empty bucket.
emptyblockprice,	The value of having an empty block.
compressprice,	The initial cost of compressing, i.e. of the pushing together of records in consecutive blocks.
priceperblock,	The cost of (two block transports + central processor time) for one block involved in compressing.
priceperbuck,	The cost of (two block transports + two block table transports + central processor time) for moving an empty block over one bucket.
pricelimit,	The maximum price accepted for an insertion. If the total cost, as computed below, exceeds pricelimit then the insertion will not be done.

These prices are used to compute the total cost of an insertion in step 2, 3, and 4 of the following 7 steps which the algorithm goes through:

1: There is room for the record in the block in which it belongs: The insertion is done without further analysis.
Otherwise the insertion will push one or more records out of the block and thus create an overflow, and:

2: A pushing together of records in at most n (key-) consecutive blocks will absorb the overflow:
cost: n * priceperblock + compressprice.
and/or:

written back to the document before a new block is read or the mode is changed.

Requirements:	zonestate = update_i or put_i.		
Results:	zonestate:	unchanged	
	procno_i:	11	
	result_i:		Available record:
	1 Done		Unchanged

6.11 Integer resulti 6.11

Yields the result of the latest call of one of the processing procedures (see Appendix A.2).

6.12 Integer Procedure setparamsi 6.12

Call: set_params_i (z) One or more pairs:(paramno, val)

set_params_i	(return value, integer). Overall result of the call: 0: All parameters processed. > 0: Exit on error in parameter pair number set_params_i.
z	(call and return value, zone). Specifies the file.
paramno	(call value, integer or long). Identifies the parameter in the zone buffer to which val is assigned.
val	(call value, integer or long). The value to be assigned to the parameter identified by paramno.

Function: Assigns values to a selected set of parameters in the zone buffer of an indexed-sequential file. The possible values of paramno and their meanings are listed in Appendix B.

Requirements: zonestate = any file_i state.

Results: Affects only the parameters assigned to.
procno_i: 13

6.13 Procedure setputi 6.13

Call: set_put_i (z)

z (call and return value, zone). Specifies the file.

Function: Terminates the current mode and sets put-mode.

Requirements: zonestate = any file_i state.

Results: zonestate: put_i.
procno_i: 5

result_i:	Available record:
1 Normal mode change	Unchanged.
2 Initialization terminated	The first in the file.

6.14 Procedure setreadi 6.14

Call: set_read_i (z)

z (call and return value, zone). Specifies the file.

Function: Terminates the current mode and sets readonly-mode.

Requirements: zonestate = any file_i state.

Results:	zonestate: read_only_i	
	procno_i: 4	
	result_i:	Available record:
	1 Normal mode change	Unchanged.
	2 Initialization terminated	The first in the file.

6.15 Integer Procedure settesti

Call:	set_test_i	(z) Optional parameter:(test_proc) one or more pairs:(procno_i, results)

set_test_i	(return value, integer). Overall result of call: - 1: Exit on error in first parameter. 0: All parameters processed. > 0: Exit on error in parameter pair number set_test_i.
z	(call and return value, zone). Specifies the file.
test_proc	(call value, procedure). The name of a procedure which must be declared at the same level as the zone or at an outer level. It must conform to the declaration: procedure test_proc (z, record, procno_i); zone z; array record; integer procno_i; It will, when specified, see below, be called just before the exit from a file_i proc with the following parameters: z: The zone of the file_i proc call. record: The array of the file_i proc call or, if not present, the zone z.

procno_i: The identification of the file_i proc.

The parameter test_proc may be left out if it already has been given in a previous call of set_test_i.

procno_i (call value, integer). Specifies the result_i values for which test_proc should be called upon exit from the file_i proc identified by procno_i. Any number of result_i values can be specified in one parameter by representing each result_i value as one digit in the decimal representation of results.

Function: Specifies a procedure to be called upon exit from certain file_i procs with certain result_i values.

The parameter pairs, procno_i - results, are processed in order and only specified changes in the situation will be effectuated but with the following additional conventions:

procno_i = 0 denotes all file_i procs.
results = 0 denotes clearing of all previously specified result_i values for procno_i.
Non-existing result_i values are ignored.

The procedures startfilei and initfilei reset the test values (corresponding to the call settesti (z, 0, 0)), so the result values from startfilei and initfilei can never be caught by any test procedure.

Requirements: zonestate = any file_i state.

Results: Affects only the test situation.
procno_i: 14

6.15.1 Examples

The call
set_test_i (z, 0, 0)
will prevent any further calls of the current test_proc.

A.3 Alphabetic List of Alarm Causes

The system adds the messages below to the list of possible alarm causes from the standard procedures of RC8000 ALGOL.

head i p <i>	Parameter error in call of head_file_i: i = 1: Not room for two records in a block. 2: Not room for at least one block in the first bucket. 0: Other illegal parameter values.
prep i <i>	Error during init_file_i, init_rec_i, extendi, start_file_i, set_put_i, or set_update_i: i = 1: Too few or many segments in the document. 2: The bucket head is not consistent. 3: Too small a zone buffer. 4: The file head is not consistent. 5: Not three shares. 6: Zone state <> 0. 7: Empty file after start_file_i or mode change. 8: Contents field of catalog entry <> 22. 9: Updatemark found.
recdescr <i>	Error or inconsistency in the record description in the call of head_file_i. i < 2044: Error in field i. i >= 2044: Key too big.
state i <i>	Zonestate error in call of any file_i proc: i = zonestate * 100 + procno_i.

B. PARAMETERS IN THE ZONE BUFFER B.

The lists below define the values of paramno to be used in calls of get_params_i or set_params_i.

The lists may be extended when it appears that more parameters are of interest to the user.

B.1 Parameter Values to getparamsi B.1

paramno	name	meaning
1	recsinfile (long)	number of records in the file
2	recbytes (long)	number of halfwords used for records
3	transports	number of input or output operations performed since the processing was started
4	pricelimit	for 4-9, see section 6.8, insertreci
5	emptybuckprice	
6	emptyblockprice	
7	compressprice	
8	priceperblock	
9	priceperbuck	
10	computed cost	the cost computed in the last call of insert_rec_i

B.2 Parameter Values to setparamsi B.2

The following of the parameters above may also be assigned to by set_params_i with values in the intervals shown:

paramno	name	legal values
4	pricelimit	0 <= val <= upper limit for integers
5	emptybuckprice	0 <= val < 2048
6	emptyblockprice	0 <= val < 2048
7	compressprice	0 <= val < 2048
8	priceperblock	0 <= val < 2048
9	priceperbuck	0 <= val < 2048

RETURN LETTER

Title: Corrections to RCSL 31-D600
RC8000 Indexed Sequential Files (ISQ). RCSL No.: 31-D634

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

A/S REGNECENTRALEN
af 1979

Information Department
Lautrupbjerg 1
DK-2750 Ballerup
Denmark