Title:

FORMAT 8000 ALGOL PROCEDURES
wait_trans, open_trans, close_trans, next_field

A/S REGNECENTRALEN

RC SYSTEM LIBRARY: FALKONERALLE 1 DK-2000 COPENHAGEN F

RCSL No: 52-AA491
Edition: June, 1976
Author: Jørgen Zachariassen

Keywords:

RC 4000/RC 8000, Algol 6, Format 8000, Transaction Oriented Systems, Online Systems.

Abstract:

The procedures facilitate the processing of format 8000 transactions, by converting transaction heads and field designators according to the format 8000 transaction syntax. Processing of the real data of a transaction is not performed by these procedures, but is done by the usual procedures for character reading and writing.

(16 pages).

Printed by A/S Regnecentralen, Copenhagen

CONTENTS Section

pp. 1 : 16.

This paper describes 4 Algol 6 procedures intended for use in transaction oriented applications written in Algol. Together with the FORMAT 8000 DISPLAY SYSTEM and the context concept in Algol 6, the procedures complete the first version of a nucleus for an RC transaction oriented system. The procedures facilitate processing of formatted transactions by recognizing and converting transaction heads and field designators according to the syntax of FORMAT 8000 transactions, leaving the processing of the contents of the transaction (data) to the usual Algol 6 procedures for character reading and printing.

The formats of the transactions are compatible with the IBM 3270 System, and transaction based communication with IBM 370 is possible in the format 8000 system.

Ref. 1: FORMAT 8000 DISPLAY SYSTEM, RCSL: 52-AA494

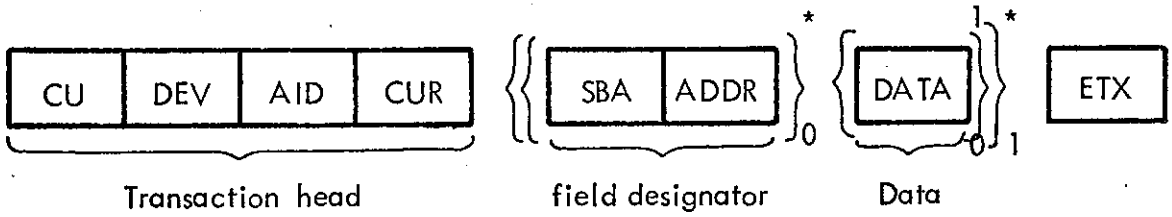
Ref. 2: Context Blokke i ALGOL 6, RCSL: 52-AA490

Ref. 3: IBM 3270 Information, Display system, Component Description, GA27-2749-4

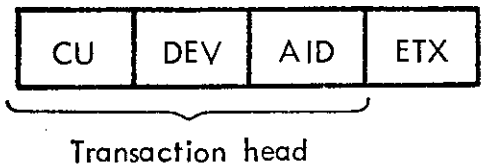
A format 8000 transaction consists of a transaction head and a number of fields. A field consists of a field designator and the data of the field. A field designator consists of a field type followed by one or more characters defining characteristics (attribute, char positions etc.). For further details, please consult ref. 1 and ref. 3.

The format 8000 transaction system distinguishes between 4 formats, identical with the IBM 3270 formats (see ref. 3):

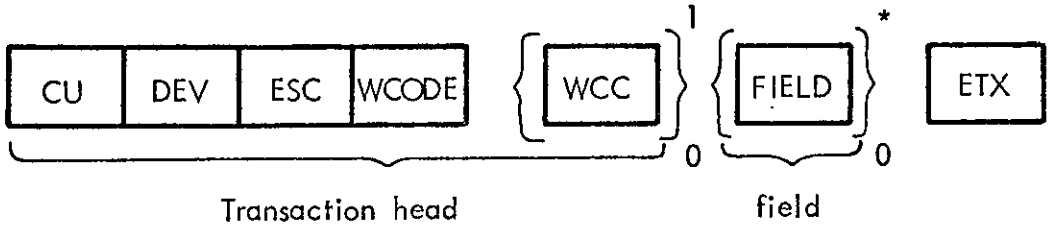
Read Modified Format. 3.1



Short Read Format. 3.2

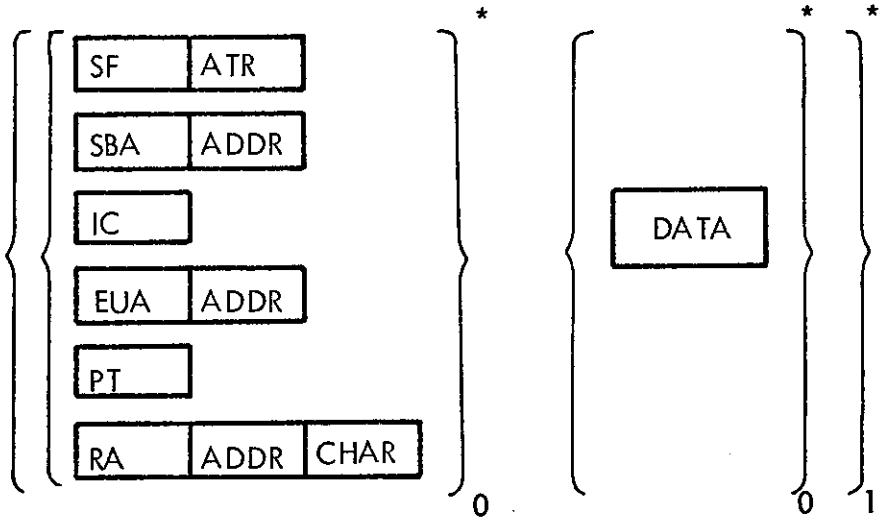


Write Format. 3.3

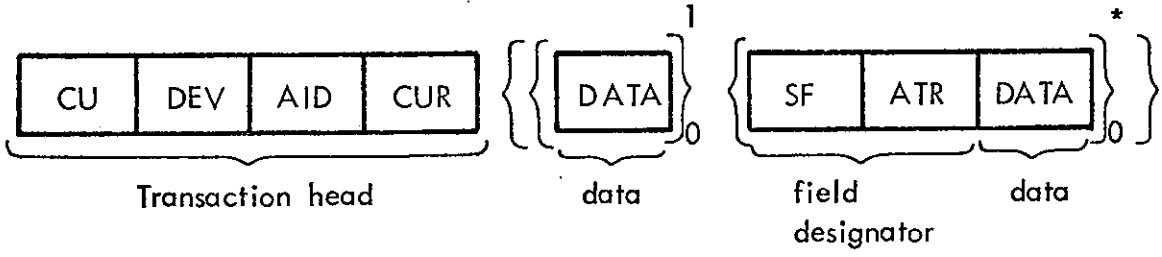


where:

field =



Read Buffer Format. 3.4



The abbreviations used in the formats above are:

CU	Control unit identification	1	ISO	char
DEV	Device identification	1	-	-
AID	Attention key identification	1	-	-
CUR	Cursor position	2	-	-
SBA	Set buffer address command	1	-	-
ADDR	Character position	2	-	-
DATA	Alfanumeric or numeric data	n	-	-
ETX	End of text	1	-	-
ESC	Escape character	1	-	-
WCODE	Write command	1	-	-
WCC	Write Control character	1	-	-
SF	Start field	1	-	-
ATR	Attribute character	1	-	-
IC	Insert CURSOR command	1	-	-
EUA	Erase unprotected to addr command	1	-	-
PT	Program tab command	1	-	-
RA	Repeat to address command	1	-	-
CHAR	ISO character	1	-	-

In this section 4 ALGOL procedures for processing of format 8000 transactions are described; the procedures are:

wait_trans	awaits the arrival of next transaction, recognizes the format and reads the transaction head
next_field	reads or writes a field designator
open_trans	writes a transaction head
close_trans	terminates an output transaction

The sketch of normal transaction processing might be:

```
Input:  next:     wait_trans (in, ...);
                  ...
        infield:  next_field (in, ...);
                  read, read_string or alike
                  ...
                  if more_fields then goto infield;
                  ...
                  goto next;

Output:           open_trans (out, ...);
                  ...
        outfield: next_field (out, ...);
                  write, outtext or alike
                  ...
                  if more_fields then goto outfield;
                  ...
                  close_trans (out);
```

Note that the procedures mentioned above do not read data of transactions, but only transaction heads or field designators - and vice versa for output. The processing of transaction data must be done by the Algol 6 library procedures for character reading and printing.

Procedure wait_trans. 4.1

Awaits the arrival of a transaction. On arrival the transaction head is input and converted to integers, which on return are assigned to return parameters of the procedure.

Call: wait_trans (z, format, destination, aux1, aux2).

z (call and return value, zone). Specifies the document from which transactions are arriving.

format (return value, integer). Defines the format of the transaction. The value is an integer in the range: 0, 1, ..., 4, which defines the format of the transaction: 0 (unknown format), 1 (read modified format), 2 (short read format), 3 (write format), and 4 (read buffer format).

destination (return value, integer). Designates the originator (sender) of the transaction, i.e. display terminal, computer, or RC 8000 application.

aux1 (return value, integer). Depending on the format of the transaction, aux1 specifies the attention type, write command code, or is undefined.

aux2 (return value, integer). Depending on the format of the transaction, aux2 specifies the cursorposition, write control character, or is undefined.

The originator of the transaction is a pair of integers:

destination = cu shift 12 + device

where cu is the communication unit, and device is a display terminal, computer, or an RC 8000 application. cu and device are ISO characters converted to integers according to the table shown in section 5.1; the range for cu and device is: 0, 1, 2, ..., 31. The routing of transactions and connection between integers and process names is done by the environment.

The interpretation of aux1 depends on the transaction format as follows:

format	aux1
1, 2, and 4	Attention key identification (aid). The value is an ISO character converted to an integer in the range: 1, 2, ..., 16, 17, according to the table shown in section 5.2.
3	Write command (WCODE). The value is an integer defining the type of a write-operation: 49 write 53 Erase/write 63 Erase all unprotected
0	Undefined

The interpretation of aux2 depends on the transaction format as follows:

format	aux2
1 and 4	Cursor position. The value is an integer in the range: 0, 1, ..., 1919, which is a character position on a display terminal ("buffer address"). The value is a conversion of the 2-character ISO character representation, according to the algorithm described in section 5.3.
3	Write control character (WCC), which specifies a bitpattern defining modes of display terminal output operations: 1 shift 0: RESET MDT bits 1 shift 1: KBD RESTORE 1 shift 2: SOUND ALARM
0 and 2	Undefined

In communication with display terminals, the transaction format will always be 1, 2, or 4. The format 3 or 0 can be actual in communication with other computers or other RC 8000 applications.

Zone state. The zone state must be open and ready for character reading (state 0, 1, or 2), i.e. since the latest call of open, setposition or character reading on that zone. To ensure proper working of the procedure next_field, and character reading procedures, the block procedure of the zone must be: transerror; transerror works as stderror, but influences termination of character reading in fields of the transaction. If the application using "wait_trans" is a BOSS job, the zone must be opened:

```
open (z, 10, <:reader:>,<give up>);
```

Processing of data of the transaction is done by means of the procedure next_field and the Algol 6 library procedures for character reading. After a call of wait_trans, the zone is positioned to the first character following the transaction head.

Alphabet. The character input table used by Algol character reading procedures is changed, so that later calls of these procedures translate the characters: SBA, SF, IC, EUA, PT, and RA to:

char	class	value
SBA	8	25
SF	8	25
IC	8	25
EUA	8	25
PT	8	25
RA	8	25

causing an EM-reaction, when character reading procedures attempt exceeding a transaction field.

Procedure open_trans. 4.2

Outputs a transaction head and initiates a zone for character writing. Open_trans may be regarded as the reverse operation of wait_trans. Note that no waiting takes place.

Call: open_trans (z, format, destination, aux1, aux2).

z (call and return value, zone). Specifies the document to which transactions are transferred.

format (call value, integer). Defines the format of the transaction to be sent. For further details, see: wait_trans.

destination (call value, integer). Designates the receiver of the transaction. For further details, see: wait_trans.

aux1, aux2 (call value, integer). Same explanation as the corresponding parameters in: wait_trans.

Note that in communication with display terminals, only format 3 (write format) can be used. Format 1, 2, and 4 can only be sent to computers or other RC 8000 applications.

Zone state: The zone must be open and ready for character printing (state 0 or 3), i.e. since the latest call of open, setposition or character-output procedures (incl. next_field).

After the call of open_trans, the zone is ready for output of characters, by means of the procedures: next_field, write, outtext etc.

Procedure close_trans. 4.3

Terminates the current transaction, i.e. writes the ETX character and empties the current character buffer to the zone.

Call: close_trans (z);

z (call and return value, zone). Specifies the document, to which transactions are transferred.

Zone state. The zone state must be 3, after character printing. It is not checked that a call of close_trans is preceeded by a call of open_trans.

Procedure next_field.

Inputs or outputs a transaction field designator.

Call: next_field (z, field_type, aux).

z — (call and return value, zone). Specifies the document from which current transactions is input or to which current transaction is output.

field_type — (call and return value, integer). If the zone z is in input state, field_type= the transaction field just recognized; field_type = 0 means that end of transaction (ETX) is met. If the zone z is in output state, field_type defines the transaction field to be output; field_type = 0 means that the current transaction is terminated as for close_trans.

aux — (call and return value, integer). The interpretation depends on field_type, and defines a char position or an attribute character.

A transaction field designator is the pair of integers: (field_type, aux), which defines how the field data should be interpreted by the application (if the zone is in input state) or by the receiver (if the zone is in output state).

The connection between field_type and aux is as follows:

	field_type	aux
field_type unknown	0	undefined
SBA (ISO 17)	1	char position
SF (ISO 29)	2	attribute char
IC (ISO 19)	3	undefined
EUA (ISO 18)	4	char position
PT (ISO 9)	5	undefined
RA (ISO 20)	6	char shift 12 + char position

Zone state. The zone state must be ready for character reading or printing (state 0, 1, 2, or 3). The function of "next_field" depends on zone state:

Input: The document, from which the current transaction is input, is scanned until an ISO character defining a field_type or until an ETX character is met. A field_type character is converted to an integer in the range 1, 2, ..., 6 according to the table above. If the transaction head is not followed immediately by a field designator, a first call of "next_field" after a call of "wait_trans" will return the value 0 to the parameter: field_type. The attribute character or char position characters following the field_type character are returned to the parameter: aux, corresponding to the table values above.

Output: The values of field_type and aux are converted to ISO characters and transferred to the document specified by the zone z. Field data can now be output by Algol 6 library procedures for character printing.

CONVERSION TABLES 5

cu/device addressing 5.1

see ref. 1, appendix C.

aid configurations 5.2

see ref. 1, appendix D.

2-character address translation 5.3

see ref. 1, appendix E.