p.n: 990 00648

RCSL No: 52-AA1137

Edition: September, 1982

Author: Bo Bagger Laursen

Title:

RC3502 LOADER
Reference Manual

RCSL 42-i1905

RC International

Keywords:

PASCAL80, RC3502, LOADER.

Abstract:

The RC3502 LOADER reads binary relocatable PASCAL80 object programs, allocates memory for the programs and performs the necessary linkage editing.

(18 printed pages).

Printed by A/S Regnecentralen af 1979, Copenhagen

RCSL 42-i1905

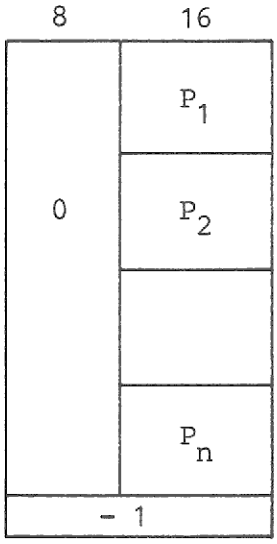
CONTENTS PAGE

1. INTRODUCTION 1.

The RC3502 LOADER reads binary relocatable PASCAL80 object programs from the external devices FPA100 and PTR (RC500) or from an application process simulating a load driver.

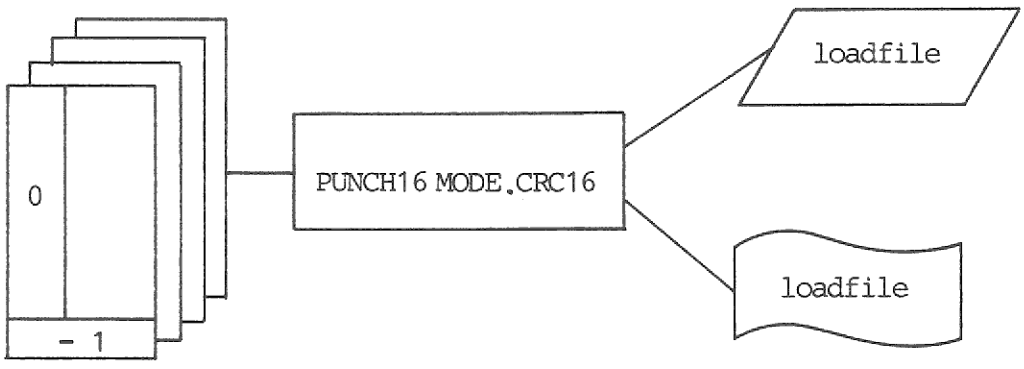
The LOADER allocates memory space for the loaded programs and performs the necessary linkage editing.

The PASCAL80 cross compiler on RC8000 produces one object file with the format (24 bits RC8000 words).

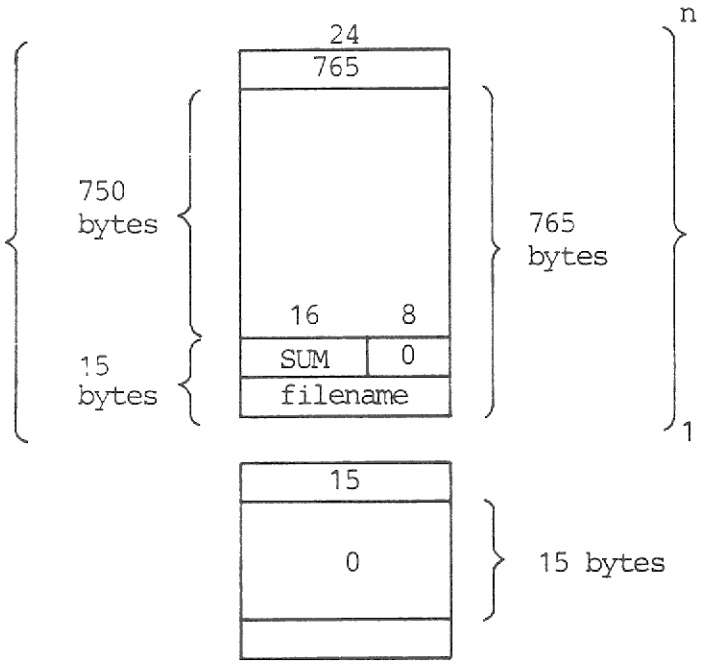


The file contains one or several PASCAL80 object programs (P_i).

PUNCH16 reformats PASCAL80 object files to a loadfile, either on disc or on a papertape.



The loadfile is formatted as a single logical file, partitioned in blocks of size 765 bytes according to the format demanded by the RC8000 AUTOLOAD utility program [ ]:



The loadfile is terminated with a stop block of size 15 bytes.

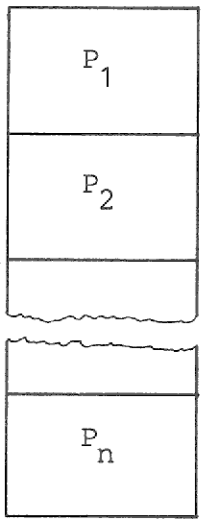
If load is from FPA, the transfer from RC8000 is done by the AUTO-LOAD program and 765 bytes are transferred per block.

If the loadfile is punched, all 768 bytes per block are loaded.

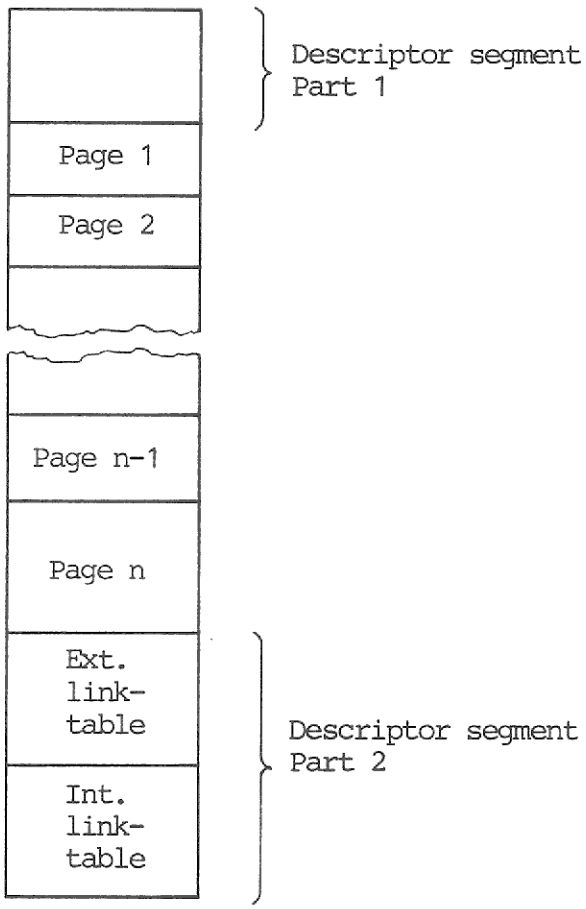
When load is from a simulated driver, the blocksize may be variable and no 'length' and SUM are included in the loaded data.

2. FUNCTIONS SUPPORTED BY THE LOADER 2.

The LOADER loads all or specified programs in a loadfile. The actual block reading is done in a format described in chapter 1. Internally the LOADER reads the loadfile byte per byte (the function 'getbyte') according to the following logical format of the loadfile:



The format of program P_i is:



The load is performed in a number of scans.

In scan 1 all requested programs are loaded. The internal linkage editing is performed during load.

External linkage editing (call of external routines) is then executed. If an external reference cannot be defined, the loadfile is logically rewound and a new scan is performed.

Note, that in this version of the LOADER a scan actually reads the loadfile from "loadpoint" until the last requested program is read.

If all the requested programs are loaded and the linkage editing were successful, the programs are included in the LINKER catalog.

LOADER requests ADAM to remove 'LOADER', so if LOADER is a child of ADAM, it dies automatically.

3. LOADER INTERFACE 3.

The LOADER has two software interfaces:

1. An interface which defines the requests it accepts from other processes.

2. An interface which defines how the LOADER requires the actual data to be received.

3.1 LOADER Messages 3.1

Requests are taken from the semaphore

sv(loadersem)↑

(see chapter 4 concerning 'sv') and interpreted in the following way:

	message
u1	function
u2	notused
u3	notused
u4	notused

u1: function

0 - load the specified programs and deliver an immediate answer

<> 0 - load the specified programs and deliver an answer after loading

The request is supposed to hold a variable 'buffer' of type:

```
buffertype = RECORD
               kind     ,
               loaderchannel,
               number         : integer;
               names          : ARRAY(1..number) OF alfa;
             END;
```

The maximum value, 'number' may take, is at present 50.

buffer.kind

0 - (EXT) the LOADER will signal requests for data to the predefined semaphore

sv(loaddriversem)↑

(see chapter 4 concerning 'sv').

2 - (FPA) the load will be performed from FPA100 by a dedicated internal FPA driver

4 - (PTR) the load will be performed from a papertape reader (e.g. RC500, RC2500) by a dedicated internal PTR driver

buffer.loaderchannel

specifies the io-channel, where the load device is connected, if PTR or FPA is specified as kind. Otherwise the information is not used.

buffer.number

0 - all programs in the load file are loaded regardless of their names

n > 0 - the request buffer contains the names of 'n' programs to be loaded

buffer.name(i)

The name of program no 'i', which is requested to be loaded.

3.2 LOADER Answers

LOADER returns the requests according to:

	answer
u1	unchanged
u2	result
u3	unchanged
u4	unchanged

and the buffer is unchanged.

u2: result

0 - ok, the load request is processed successfully

2 - warning, the load is completed, but:

a. a program had a versionerror, it was loaded anyway

b. a program was already in the LINKER catalog - the loaded program was deleted

c. a requested program was not in the load file

3 - loaderror, all loading is aborted because:

a. the LOADER could not get resources to run

b. an external reference among the loaded programs could not be defined

4 - unintelligible request

3.3 Loaddriver Messages

3.3

When load with kind = 0 is requested, the LOADER requests data at the semaphore

sv(loaddriversem)↑

(see chapter 4 concerning 'sv').

The request is a datamessage of size 774 bytes. The buffer follows the driver conventions and may thus contain a datablock of max. 768 bytes.

The header is:

	drivermessage
u1	function
u2	7
u3	control
u4	not used

u1: function

1 - read. Read data to the buffer after having performed the control function specified by u3.

u3: control

0 - transmit next block. Move the load file pointer to the first byte of the next block and read this block to the buffer.

1 - retransmit block. Read the block defined by the current load file pointer to the buffer. The load file pointer is not moved.

2 - rewind. Move the load file pointer to the first byte, block, and file in the loadfile and read a status block to the buffer.

4 - upspace block. Move the loadfile pointer to the first byte of the next block, and read a status block to the buffer.

8 - upspace file. Move the loadfile pointer to the first byte in the first block of the next file, and read a status block to the buffer.

12 - finis. Terminate reading. No data is read to the buffer.

128 - sense. Read a status block to the buffer.

Note: Only control = 0, 2, 12 are used in this version of the LOADER.

3.4 Loaddriver Answers

3.4

The LOADER interprets answers in the following way:

	driver answer
u1	not used
u2	result
u3	block type
u4	not used

u2: result

0 - ok, the buffer contains a valid data block or status block.

<> 0 - not ok, the buffer contains no valid data. The LOADER will repeat the request.

u3: block type

249 - the buffer contains a status block.

251 - the buffer contains a data block.

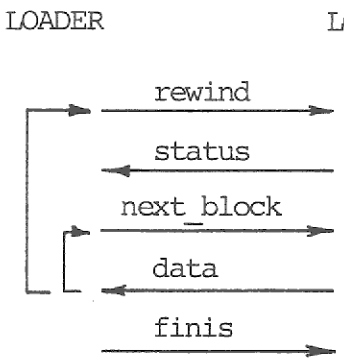
If u2 = 0 the buffer contains a variable 'buffer' of either data_type or status_type:

```
data_type   = RECORD
                first,
                last,
                next : integer;
                data : ARRAY(6..773) OF byte;
              END;

status_type = RECORD
                first,
                last,
                next : integer;
                data : RECORD
                  status,
                  size,
                  filenumber,
                  blocknumber : integer;
                END;
              END;
```

3.5 Loaddriver Protocol

The communication between LOADER and the loaddriver, when kind = 0 is specified, may be visualized in the following way:



4. PARENT PROCESS RESPONSIBILITIES 4.

The LOADER has the following header:

```
PROCESS loader (VAR sv : system_vector)
```

where 'system_vector' is a predefined type.

The LOADER may be run by a process itself by appropriate declarations and routine calls, or by requesting ADAM to do the job.

The code occupies approx. 8K bytes.

The stack must be approx. 700 words.

Default create_size may be used (= 0) in the create call.

Requests to the LOADER are signalled to a predefined semaphore in the parameter 'sv':

```
signal (request, sv(loadersem)↑);
```

When load with kind = 0 (external) is requested, the LOADER requests data at the predefined semaphore:

```
sv (loaddriversem)↑
```

An application must process the request and return the message with data.

RETURN LETTER

Title: RC3502 LOADER
Reference Manual

RCSL No.: 52-AA1137

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

RC International

Information Department
Lautrupbjerg 1
DK-2750 Ballerup
Denmark