

---

**RCSL No:** 52-AA1029

**Edition:** February 1981

**Author:** Erik Bak Kristensen

---

**Title:**

PASCAL80 FPA100 Driver

---

---

**Keywords:**

PASCAL80, FPA100, 3502, Fpadriver.

---

**Abstract:**

This manual contains a description of the FPA100 driver written in PASCAL80.

(16 printed pages)

---

**Copyright © 1981, 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.**

CONTENTS	PAGE
1. INTRODUCTION .....	1
2. FUNCTIONS SUPPORTED BY THE FPA DRIVER .....	2
3. DRIVER INTERFACE .....	5
3.1 Sense .....	5
3.2 Reset .....	5
3.3 Autoload .....	6
3.4 Read data .....	6
3.5 Write data .....	7
3.6 Write and read data .....	8
4. PARENT RESPONSIBILITIES .....	9
A. REFERENCES .....	10



1      INTRODUCTION

1

The FPA driver is written in PASCAL80. The driver runs on a 3502-machine and controls a FPA100 device controller [1].

To support one FPA100 controller you need two incarnations of the driver code, a transmitter incarnation and a receiver incarnation.

The functions supported by the driver are formed according to the PASCAL80 Driver Conventions [2], except for the format of a databuffer in a write and read operation, see section 3.6.

The driver is capable of handling a databuffer stack.

2      FUNCTIONS SUPPORTED BY THE FPA DRIVER

2

The receiver and the transmitter incarnation of the driver supports the same functions:

- control:
  - sense the statusword of the hardware
  - send a controlword to the hardware:
    - reset: The hardware is reset.  
The transmitter transfers a reset status to the receiver in the other computer.
  
- autoload:
  - The transmitter transfers an autoload status to the receiver in the other computer. The control word is dummy for the receiver.
  
- read data
- write data
- write and read data

The driver returns the following type of results to the application:

- processed successfully (basic result 0)
- rejected (basic result 1)
- soft error, (basic result 2)
  - parity error (8)
  - process timeout (16)
  - combined write error (32)
  - block length error (64)
  - receiver\_not\_ready (128)

One or more of these 5 kinds of soft errors can be returned at the same time.

- hard error (basic result 3)

- disconnected (8)
- reset (16)
- combined write error (32)
- autoload received (64)

One or more of the 4 kinds of hard errors can be returned at the same time.

If the receiver receives an autoload request in its status word, a switch in the RAM memory is inspected. If the switch is set, the 3502 machine is autoloaded, else an autoload result is returned to the application.

- unintelligible (basic result 4)

- bad message (8)
- combined write error (32)

Zero or more of these 2 kinds of unintelligible errors can be returned at the same time.

If more than one of these 5 kinds of basic result occur during the treatment of one request the basic result with the highest priority is returned.

The priority of the basic result can be trimmed in the `CONST_` declaration part of the driver. At the

present time the priority is as follows:

rejected,	highest priority
unintelligible	
hard error	
soft error	
processed successfully,	lowest priority

The driver starts in rejecting mode. The driver enters processing mode, when a control function is processed successfully. The driver enters rejecting mode, when a hard error occurs.



3 DRIVER INTERFACE

3

Each request supported by the driver is described in one section.

3.1 Sense

3.1

The status of the hardware is returned.

Function code 0 (control + 0 \* 4).

Possible results:

0 processed successfully, the driver enters  
processing mode.  
19 (hard error + 2 \* 8), reset received  
(receiver only).  
67 (hard error + 8 \* 8), autoload received  
(receiver only).  
3 + x \* 8 (hard error combinations),  
where x is 10  
(receiver only).

3.2 Reset

3.2

A reset control command is sent to the hardware. The hardware will transfer a reset status to the other computer (transmitter only).

Function code 4 (control + 1 \* 4).

Possible results:

0 processed successfully, the driver enters  
processing mode.  
11 (hard error + 1 \* 8), disconnected.  
Maybe possible, depending  
on hardware reaction.  
19 (hard error + 2 \* 8), reset received  
(receiver only).  
67 (hard error + 8 \* 8), autoload received  
(receiver only).  
3 + x \* 8 (hard error combinations),  
where x is 10  
(receiver only).

3.3 Autoload

3.3

An autoload control command is sent to the hardware. The hardware will transfer the status to the other computer (transmitter only).

Function code 8 (control + 2 \* 4).

Possible results:

- 0 processed successfully, the driver enters processing mode.
- 11 (hard error + 1 \* 8), disconnected.  
Maybe possible, depending on hardware reaction.
- 19 (hard error + 2 \* 8), reset received (receiver only).
- 67 (hard error + 8 \* 8), autoload received (receiver only).
- 3 + x \* 8 (hard error combinations), where x is 10 (receiver only).

3.4 Read data

3.4

Data is transferred from the hardware into the data buffer(s). The first byte received is stored in u3 of the topmost header message.

Function code 1 (read + 0 \* 4).

Possible results:

- 0 processed successfully.
- 1 rejected.
- 4 unintelligible, the size of one of the databuffers is not supported by the driver.
- 10 (soft error + 1 \* 8), parity error.
- 18 (soft error + 2 \* 8), driver process timed out.
- 66 (soft error + 8 \* 8), block length error: data buffer(s) too small.
- 2 + x \* 8 (soft error combinations), where x is 3, 9, or 10.
- 11 (hard error + 1 \* 8), disconnected.

19 (hard error + 2 \* 8), reset received  
(receiver only).  
67 (hard error + 8 \* 8), autoloading received  
(receiver only).  
3 + x \* 8 (hard error combinations),  
where x is 10  
(receiver only).

### 3.5 Write data

3.5

Data is transferred from the data buffer(s) to the hardware. The first byte transferred is taken from u3 in the topmost header message.

Function code 2 (write + 0 \* 4).

Possible results:

0 transferred successfully.

1 rejected.

4 uninterpretable, the size of one of the  
databuffers  
is not supported by the driver  
or first in one of the databuffers  
is less than 6  
or last in one of the databuffers  
is greater than the size of the  
buffer.

10 (soft error + 1 \* 8), parity error.

18 (soft error + 2 \* 8), process timeout.

130 (soft error + 16 \* 8), receiver  
not ready.

2 + x \* 8 (soft error combinations),  
where x is  
3, 17, or 18.

11 (hard error + 1 \* 8), disconnected.

19 (hard error + 2 \* 8), reset received  
(receiver only).

66 (hard error + 8 \* 8), autoloading received  
(receiver only).

3 + x \* 8 (hard error combinations),  
where x is 10  
(receiver only).

### 3.6 Write and read data

3.6

Data is transferred from the databuffer(s) to the hardware. The first byte transferred is u3 of the topmost header message.

Note that there is a violation upon the 'driver conventions' [2] for a databuffer. The conventions suggest that the write part of a buffer is from FIRST to NEXT, both included. But the driver uses NEXT as a top so that the datapart is from FIRST to NEXT-1, both included.

Then data is transferred from hardware to the databuffer(s). The first byte transferred is put into u3 of the topmost header message.

Function code 3 (writeread + 0 \* 4).

Possible results:

0 proceed successfully.

1 rejected.

4 unintelligible, see section 3.4

36 (unintelligible + 4 \* 8),  
unintelligible write,  
see section 3.5

34 + x \* 8 (soft error in write),  
where x is 1, 2, 3,  
16, 17, or 18.

2 + x \* 8 (soft error in read),  
where x is 1, 2, 3,  
8, 9, or 10.

35 + x \* 8 (hard error in write),  
where x is 1 or  
2,8,10 (receiver only).

3 + x \* 8 (hard error in read),  
where x is 1 or  
2,8,10 (receiver only).

PARENT RESPONSIBILITIES

The PASCAL80 process heading of the driver is as follows:

```
PROCESS fpadriver (var sem: semaphore;  
                  level, blocktime: integer  
                  receiver: boolean);
```

where sem is the driver's request semaphore, level is the hardware interface level number. Blocktime is the maximum number of seconds used to process a single driver request.

Receiver is true for the receiver incarnation of the driver, else false.

An incarnation of the driver uses max. 500 b stack-storage. The driver code occupies approximately 3 Kb of storage. This amount can be considerably reduced when dynamic types has been implemented in PASCAL80.

The driver will then be more flexible, too. At the time being the driver only supports three different buffer sizes (small\_size, medium\_size, and large\_size). These sizes can be trimmed in the CONST\_declaration part of the driver.

A      REFERENCES

A

- [1] RCSL No. 52-AA359  
FPA100 FRONT PROCESSOR ADAPTOR  
for RC3500, Reference Manual.
- [2] RCSL No. 31-D617  
PASCAL80 Driver Conventions

**RETURN LETTER**

Title: PASCAL80 FPA100 Driver

RCSL No.: 52-AA1029

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