

parallel

COLLABORATORS

	<i>TITLE :</i> parallel		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY		March 28, 2025	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	parallel	1
1.1	parallel.doc	1
1.2	parallel.device/CMD_CLEAR	1
1.3	parallel.device/CMD_FLUSH	1
1.4	parallel.device/CMD_READ	2
1.5	parallel.device/CMD_RESET	2
1.6	parallel.device/CMD_START	2
1.7	parallel.device/CMD_STOP	3
1.8	parallel.device/CMD_WRITE	3
1.9	parallel.device/OpenDevice	4
1.10	parallel.device/PDCMD_QUERY	5
1.11	parallel.device/PDCMD_SETPARAMS	5

Chapter 1

parallel

1.1 parallel.doc

CMD_CLEAR	CMD_START	PDCMD_QUERY
CMD_FLUSH	CMD_STOP	PDCMD_SETPARAMS
CMD_READ	CMD_WRITE	
CMD_RESET	OpenDevice()	

1.2 parallel.device/CMD_CLEAR

NAME

Clear -- clear the parallel port buffer

FUNCTION

This command just RTS's (no buffer to clear)

IO REQUEST

io_Message	mn_ReplyPort initialized
io_Device	set by OpenDevice
io_Unit	set by OpenDevice
io_Command	CMD_CLEAR (05)

1.3 parallel.device/CMD_FLUSH

NAME

Flush -- clear all queued I/O requests for the parallel port

FUNCTION

This command purges the read and write request queues for the parallel device.

IO REQUEST

io_Message	mn_ReplyPort initialized
io_Device	set by OpenDevice
io_Unit	set by OpenDevice
io_Command	CMD_FLUSH (08)

1.4 parallel.device/CMD_READ

NAME

Read -- read input from parallel port

FUNCTION

This command causes a stream of characters to be read from the parallel I/O register. The number of characters is specified in `io_Length`.

The `parallel.device` has no internal buffer; if no read request has been made, pending input (i.e. handshake request) is not acknowledged.

IO REQUEST

<code>io_Message</code>	<code>mn_ReplyPort</code> initialized
<code>io_Device</code>	set by <code>OpenDevice</code>
<code>io_Unit</code>	set by <code>OpenDevice</code>
<code>io_Command</code>	<code>CMD_READ (02)</code>
<code>io_Flags</code>	If <code>IOF_QUICK</code> is set, driver will attempt Quick IO
<code>io_Length</code>	number of characters to receive.
<code>io_Data</code>	pointer where to put the data.

RESULTS

`io_Error` -- if the Read succeeded, then `io_Error` will be null.
If the Read failed, then `io_Error` will contain an error code.

SEE ALSO

`parallel.device/PDCMD_SETPARAMS`

1.5 parallel.device/CMD_RESET

NAME

Reset -- reinitializes the parallel device

FUNCTION

This command resets the parallel device to its freshly initialized condition. It aborts all I/O requests both queued and current and sets the devices's flags and parameters to their boot-up time default values.

IO REQUEST

<code>io_Message</code>	<code>mn_ReplyPort</code> initialized
<code>io_Device</code>	set by <code>OpenDevice</code>
<code>io_Unit</code>	set by <code>OpenDevice</code>
<code>io_Command</code>	<code>CMD_RESET (01)</code>

RESULTS

Error -- if the Reset succeeded, then `io_Error` will be null.
If the Reset failed, then the `io_Error` will be non-zero.

1.6 parallel.device/CMD_START

NAME

Start -- restart paused I/O over the parallel port

FUNCTION

This command restarts the current I/O activity on the parallel port by reactivating the handshaking sequence.

IO REQUEST

io_Message	mn_ReplyPort initialized
io_Device	set by OpenDevice
io_Unit	set by OpenDevice
io_Command	CMD_START (07)

SEE ALSO

parallel.device/CMD_STOP

1.7 parallel.device/CMD_STOP

NAME

Stop -- pause current activity on the parallel device

FUNCTION

This command halts the current I/O activity on the parallel device by discontinuing the handshaking sequence.

IO REQUEST

io_Message	mn_ReplyPort initialized
io_Device	set by OpenDevice
io_Unit	set by OpenDevice
io_Command	CMD_STOP (06)

SEE ALSO

parallel.device/CMD_START

BUGS

Using any other parallel.device command will restart IO.

1.8 parallel.device/CMD_WRITE

NAME

Write -- send output to parallel port

FUNCTION

This command causes a stream of characters to be written to the parallel output register. The number of characters is specified in `io_Length`, unless `-1` is used, in which case output is sent until a zero byte in the data: note that this is independent of setting `EOFMODE` in `io_ParFlags` and using the `PTermArray` to terminate the write.

IO REQUEST

io_Message	mn_ReplyPort initialized
io_Device	set by OpenDevice
io_Unit	set by OpenDevice
io_Command	CMD_WRITE (03)
io_Flags	If IOF_QUICK is set, driver will attempt Quick IO
io_Length	number of characters to transmit, or if set to -1 send until zero byte encountered
io_Data	pointer to block of data to transmit

RESULTS

io_Error -- If the Write succeeded, then io_Error will be null.
If the Write failed, then io_Error will contain an error code.

SEE ALSO

parallel.device/PDCMD_SETPARAMS

1.9 parallel.device/OpenDevice

NAME

Open -- a request to open the parallel port

SYNOPSIS

```
error = OpenDevice("parallel.device", unit, ioExtPar, flags)
D0          A0          D0    A1    D1
```

FUNCTION

This is an exec call that starts up the parallel.device.

This function allows the requestor software access to the parallel device. Unless the shared-access bit (bit 5 of io_ParFlags) is set, exclusive use is granted and no other access is allowed until the owner closes the device. The PTermArray of the ioExtPar is initialized only if the EOFMODE bit is set in io_ParFlags.

INPUTS

"parallel.device" - a pointer to literal string "parallel.device"
unit - Must be zero for future compatibility
ioExtPar - pointer to an IO Request block of structure IOExtPar to be initialized by the Open routine. (see devices/parallel.h for definition)
The io_ParFlags field must be set as desired (see shared-access description, above). Note that this is not a standard IO Request structure.
flags - Must be zero for future compatibility

RESULTS

d0 -- same as io_Error
io_Error -- if the Open succeeded, then io_Error will be null.
If the Open failed, then io_Error will be non-zero.

SEE ALSO

exec/CloseDevice

1.10 parallel.device/PDCMD_QUERY

NAME

Query -- query parallel port/line status

FUNCTION

This command return the status of the parallel port lines and registers.

IO REQUEST

io_Message	must have mn_ReplyPort initialized
io_Device	set by OpenDevice
io_Unit	set by OpenDevice
io_Command	PDCMD_QUERY (09)

RESULTS

io_Status	BIT	ACTIVE	FUNCTION
	0	high	printer busy toggle (offline)
	1	high	paper out
	2	high	printer selected on the A1000 printer selected & serial "Ring Indicator" on the A500/A2000 Use care when making cables.
	3	-	read=0,write=1
	4-7		reserved

BUGS

In a earlier version of this AutoDoc, BUSY and PSEL were reversed.
The function has always been correct.

1.11 parallel.device/PDCMD_SETPARAMS

NAME

SetParams -- change parameters for the parallel device

FUNCTION

This command allows the caller to change parameters for the parallel port device. It will disallow changes if any reads or writes are active or queued. The PARB_EOFMODE bit of io_ParFlags controls whether the io_PTermArray is to be used as an additional termination criteria for reads and writes. It may be set directly without a call to SetParams, setting it here performs the additional service of copying the PTermArray into the device default array which is used as the initial array for subsequent device opens. The Shared bit can be changed here, and overrides the current device access mode set at OpenDevice time.

IO REQUEST

io_Message	mn_ReplyPort initialized
io_Device	preset by OpenDevice
io_Unit	preset by OpenDevice
io_Command	PDCMD_SETPARAMS (0A)

NOTE that the following fields of your IORequest

are filled by Open to reflect the parallel device's current configuration.

`io_PExtFlags` must be set to zero, unless used

`io_ParFlags` see definition in `parallel.i` or `parallel.h`
NOTE that `x00` yields exclusive access, `termarray` inactive.

`io_PTermArray` ASCII descending-ordered 8-byte array of termination characters. If less than 8 chars used, fill out array w/lowest valid value. Terminators are used only if `EOFMODE` bit of `io_Parflags` is set. (e.g. `x512F040303030303`) This field is filled on `OpenDevice` only if the `EOFMODE` bit is set.

RESULTS

`io_Error` -- if the `SetParams` succeeded, then `io_Error` will be null.
If the `SetParams` failed, then `io_Error` will be non-zero.