

Commands

General Rules:

A command will be executed upon the reception of <LF>.

Any wrong command will return „E“.

A return value ends with <CR> <LF>.

A list of return values is separated by <CR> <LF>.

Exit

Terminate session. Keep state of master.

SetTimeout(timeoutSec)

Enables or disables Connection State supervision.

Will terminate connection after {timeoutSec} sec without traffic.

Parameters:

timeoutSec	n	timeout in seconds (default = 30, 0 = disabled)
------------	---	---

Return:

timeout=	n	in seconds
----------	---	------------

KeepAlive

prevents Connection Timeout

Return:

	1	
--	---	--

Commands

ListDevices		
Returns List of found USB FTDI devices		
<i>Return:</i>		
	n	Number of devices, list with devices follows
	dev0	Only if devices are found
	...	

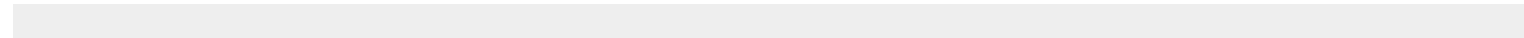
OpenSerial(serial)		
Open connection to selected FTDI Master by device serial number.		
<i>Parameters:</i>		
serial number	n	serial number of device
<i>Return:</i>		
connected=	[0 1]	

OpenList(listId)		
Open connection to selected FTDI Master by device list position.		
<i>Parameters:</i>		
devId	n	Number in device list, starting with 0
<i>Return:</i>		
connected=	[0 1]	

Close

Commands

Close connection to FTDI Master. Stop Periodic Transfers.		
<i>Return:</i>		
connected=	[0 1]	



SetBtrate(bitrate)		
Select Btrate		
<i>Parameters:</i>		
bitrate	n	in baud (default = 500000)
<i>Return:</i>		
bitrate=	n	

SetSync(sync)		
Select SYNC		
<i>Parameters:</i>		
sync	8 (default)	bit
	32	bit
<i>Return:</i>		
sync=	n	

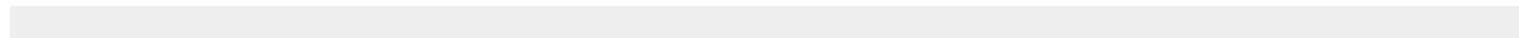
SetHeader(header)		
--------------------------	--	--

Commands

Select Header format		
<i>Parameters:</i>		
header	3 (default)	byte header
	4	byte header
<i>Return:</i>		
header=	n	

SetParity(parity)		
Select Parity		
<i>Parameters:</i>		
parity	0 (default)	Even
	1	Odd
	2	Space
	3	None
<i>Return:</i>		
parity=	n	

SetBreak(break)		
Select Break Length		
<i>Parameters:</i>		
break	x,y	double format (default = 13,5)
<i>Return:</i>		
break=	x,y	



Commands

SetNodeAddr(addr)		
Select address of node.		
<i>Parameters:</i>		
addr	n	
<i>Return:</i>		
addr=	n	

SendWakeup(symbol, ack)		
Send Wakeup, symbol if selected, ack if selected.		
<i>Parameters:</i>		
symbol	[0 1]	1: send wakeup symbol + sleep 25ms
ack	[0 1]	1: send wakeup ack
<i>Return:</i>		
	1	

SendSleepBroadcast		
Send Broadcast for Sleep.		
<i>Return:</i>		
	1	

Write(addr, words, data)		
---------------------------------	--	--

Commands

Write {words} data words starting with {addr} with the same {data}.
Address will be auto-incremented.

Parameters:

addr		address in decimal or hex
words		number of words in decimal
data		write address in decimal or hex

Return:

	[0 1]	
--	-------	--

Read(addr, words)

Read {words} data words starting with {addr}.
Address will be auto-incremented.

Parameters:

addr		address in decimal or hex
words		number of words in decimal

Return:

	[0 1]	1: list with read data follows
	data0	
	...	

Verify(addr, words, mask, expected)

Commands

Verify {words} data words starting with {addr}, checking that {expected} =? (rdata & {mask}). Address will be auto-incremented.

Parameters:

addr		address in decimal or hex
words		number of words in decimal
mask		verify mask in decimal or hex
expected		expected check value in decimal or hex

Return:

	[0 1]	0: list with read data follows
	data0	
	...	

SetPeriodicWrite(addr, data, words)

Enable periodic write of {words} data words starting with {addr} with the same {data}. Address will be auto-incremented.

Parameters:

addr		write address in decimal or hex
words		number of words in decimal (0 = disable)
data		write address in decimal or hex

Return:

	1	
--	---	--

SetPeriodicVerify(addr, words, mask, expected)

Commands

Enable periodic verify of {words} data words starting with {addr}, checking that {expected} =? (rdata & {mask}).
Address will be auto-incremented.

Parameters:

addr		write address in decimal or hex
words		number of words in decimal (0 = disable)
mask		verify mask in decimal or hex
expected		expected check value in decimal or hex

Return:

	1	
--	---	--

SetPeriodicIntervalMs(interval)

Set interval in ms between periodic transfers.

Parameters:

interval	n	in ms
----------	---	-------

Return:

interval=	n	
-----------	---	--

StartPeriodic

Start/Enable configured periodic transfers.

Return:

periodic=	[0 1]	
-----------	-------	--

StopPeriodic

Stop/Disable periodic transfers.

Commands

<i>Return:</i>		
periodic=	[0 1]	

GetStatus		
Returns and clears status flags		
<i>Return:</i>		
com_error=	[0 1]	Any error detected during transfer (readback, response, timeout, ...)
verify_error=	[0 1]	Verify error has occurred.

SetDebugHeaderCrcError(debugHeaderCrcError)		
Select injection of Header CRC Error		
<i>Parameters:</i>		
debugHeaderCrcError	[0 1]	
<i>Return:</i>		
debugHeaderCrcError=	[0 1]	

SetDebugWriteCrcError(debugWriteCrcError)		
Select injection of Write CRC Error		
<i>Parameters:</i>		
debugWriteCrcError	[0 1]	
<i>Return:</i>		
debugWriteCrcError=	[0 1]	

Commands

SetDebugUseLastLiveCounter(debugUseLastLiveCounter)		
Select injection of LiveCounter Error		
<i>Parameters:</i>		
debugUseLastLiveCounter	[0 1]	
<i>Return:</i>		
debugUseLastLiveCounter=	[0 1]	