

# PROGRAMMIER- HANDBUCH

---

## Avisaro Compact Flash Datenlogger

---

RS232 , I2C , SPI, CAN  
Modul und Box



Version 2.3

## Änderungshistorie

2004-10-01		Erstellung
2004-10-25		At-Kommandos
2004-11-04		Fehlermeldungen, Flash-Update, Quick-Start
2004-11-18		Erweiterungen der Befehle und Geschwindigkeit Serielle Schnittstelle
2004-12-13		Erweiterungen des Befehlssatzes. Insbesondere der Paketmodus ist hinzugekommen. Automatischer Lernmodus, ob bei der Eingabe <cr><lf> verwendet wird. Statusabfrage über AT+STATUS. Flusskontrolle (Hardware und Software)
2006-05-18		Pinbelegung WAGO Stecker hinzugefügt
2007-04-02		CAN hinzugefügt
2007-06-14		I2C Konfig Befehle aktualisiert

Kontakt:

Avisaro AG  
Vahrenwalderstr. 7 (tch)  
30165 HannoverTelefon:  
+49-511-7809390Telefax:  
+49-511-35319624eMail:  
support@avisaro.com

---

**INHALTSVERZEICHNIS**


---

Inhaltsverzeichnis .....	3
Dieses Dokument .....	5
Leichte Navigation .....	5
Beschriebene Produkte .....	5
Kommando~, Daten~, Packet~ und Automatikmodus.....	6
Kommandomodus .....	6
Datenmodus .....	6
Paketmodus .....	7
Datenstrom und Paketmodus .....	8
Automatikmodus .....	8
AT-Kommandos: Allgemeine Hinweise .....	9
Tabellarische Übersicht AT-Kommandos .....	10
Allgemeine AT-Befehle.....	12
at+status .....	12
at+commands .....	12
at+errorlist .....	13
at+readerror .....	13
at+restart .....	14
stopsequenz (“+ + +” / EOT) .....	14
Befehle zur Konfiguration.....	15
at+linepar .....	15
at+i2caddr .....	15
at+i2cflow .....	16
at+i2cspeed .....	16
at+i2cpull .....	17
at+spi .....	18
at+stopsequence .....	19
at+can .....	19
at+autolog .....	23
at+flushtimer .....	23
at+minlogfile .....	24
Befehle zum Schreiben und Lesen von Daten.....	25
at+datastream .....	25
at+writepacket .....	25
at+readpacket .....	26
at+repeatpacket .....	27
at+packetlength .....	27
Befehle zum Arbeiten mit Dateien und Verzeichnissen.....	28
at+open .....	28
at+create .....	28
at+append .....	29
at+createdir .....	30
at+close .....	30

at+delete	31
Befehle zum Abfragen von Informationen .....	32
at+freespace	32
at+diskspace	32
at+filesize	33
at+filedate	33
at+dir	34
at+cardpresent	34
at+mediainfo	35
Weitere Befehle .....	36
at+setdate	36
at+settime	36
at+dump	37
Programmierbeispiele.....	38
Quellcode CRC Berechnung	38
Quellcode „write packet“	39

---

## DIESES DOKUMENT

---

### LEICHTE NAVIGATION

Dieses PDF Dokument ist **verlinkt**. So können Sie schneller Navigieren. Wenn Sie z.B. auf einen Eintrag im Inhaltsverzeichnis klicken, springt Adobe Reader zur entsprechenden Seite. Ebenso führen Sie alle „siehe Kapitel ...“ Kommentare direkt zu weiterführenden Informationen.

Thumbnails (verkleinerte Seitendarstellung) und Lesezeichen sind aktiviert.

### BESCHRIEBENE PRODUKTE

Dieses Dokument beschreibt die Produkte „Avisaro WLAN Box“ und „Avisaro WLAN Modul“. Beide Produkte sind funktional sehr ähnlich.



*Avisaro WLAN Modul*



*Avisaro WLAN Box*

---

## KOMMANDO~, DATEN~, PACKET~ UND AUTOMATIKMODUS

---

Es werden drei Modi unterschieden:

- 1) Im Kommandomodus werden Befehle eingegeben. D.h. in dieser Modus sind die 'AT'-Befehl gültig und somit können alle Dateioperationen ausgeführt werden
- 2) Im Datenmodus werden die Daten übertragen. Der Datenmodus wird durch den Befehl at+datastream eingeschaltet und mit der Stop-Sequenz (drei Plus-Zeichen im zeitlichem Abstand) wieder ausgeschaltet.
- 3) Im Packetmodus werden ebenfalls Daten übertragen. Durch den Befehl at+writepacket wird für eine wohldefinierte Anzahl von Bytes in den Datenmodus geschaltet und danach automatisch wieder zurück in den Befehlsmodus.

Die Unterscheidung der drei Modi ist sinnvoll, da Daten nicht als Befehle interpretiert werden dürfen.

### KOMMANDOMODUS

- Im Kommandomodus können Befehle beginnend mit AT+ eingegeben werden.
- Ein Kommando endet entweder mit einem CR, einem LF oder zwei Zeichen, die entweder CR oder LF sein können. Das erste Kommando nach dem Einschalten oder Neustart hat eine Verzögerung von drei Sekunden, während der auf ein zweites Endekennzeichen gewartet wird (entweder CR oder LF). Dadurch merkt sich das CF-Modul, mit welchen Zeichen der Benutzer seine Kommandos abschliesst.
- Kommandos können gross- oder klein geschrieben werden.
- Kommandos mit einer Länge über 64 Zeichen (nach dem AT+) werden mit einem Fehler abgebrochen.
- Nach der Ausführung eines Kommandos wird ein OK<CR+LF> (Erfolgreich) oder ERROR<CR+LF> zurückgesendet.
- Nach der Ausführung eines Kommandos befindet sich das CF-Modul wieder im Kommandomodus.

### DATENMODUS

- In den Datenmodus wird AT+DATASTREAM gewechselt.
- In den Datenmodus kann nur gewechselt werden, wenn eine Datei geöffnet ist (entweder zum Lesen oder Schreiben).
- Das CF-Modul sendet OK<CR+LF> zurück.

- Ist eine Datei zum Schreiben geöffnet, dann können jetzt Rohdaten zum CF-Modul gesendet werden.
- Ist eine Datei zum Lesen geöffnet, dann wird augenblicklich der Inhalt der Datei zum Benutzer-Terminal gesendet.
- Beim Schreiben in eine Datei bricht der Datenmodus ab, wenn ein Filesystem-Fehler aufgetreten ist. In diesem Fall wird ERROR<CR+LF> ausgegeben und die Datei geschlossen.
- Beim Lesen aus einer Datei bricht der Datenmodus ab, wenn die Datei komplett übertragen wurde oder wenn ein Filesystem-Fehler aufgetreten ist. Im Fehlerfall wird ERROR<CR+LF> ausgegeben und die Datei geschlossen.
- Beim Schreiben in eine Datei kann der Datenmodus nur durch die Eingabe von +++ (mit mindestens 0.5 Sekunden Abstand) abgebrochen werden (wenn kein Fehler aufgetreten ist).
- Das CF-Modul befindet sich nach dem Abbruch des Datenmodus' wieder im Kommandomodus.

### **PAKETMODUS**

- In den Paketmodus wird mit AT+WRITEPACKET gewechselt.
- In den Paketmodus kann nur gewechselt werden, wenn eine Datei zum Schreiben geöffnet ist.
- Bei aktivem Software Flow Control (XON/XOFF) wird Paketmodus abgelehnt.
- AT+WRITEPACKET gibt OK<CR+LF> zurück wenn in den Paketmodus gewechselt wurde.
- Nach dem Wechsel in den Paketmodus muß das Paket gesendet werden. Das Paket ist wie folgt aufgebaut: 2 Bytes Längenangabe, 1...1536 Bytes Daten, 2 Bytes CRC-Wert.
- Alle 2-Byte Werte sind 'Little Endian' (Low Byte zuerst)
- Die Längenangabe bezieht sich auf die Daten d.h. Länge und CRC werden nicht mitgezählt.
- Der CRC-Wert ist optional. Ein CRC-Wert von 0 bedeutet dass kein CRC-Wert angegeben wurde. In diesem Fall wird das Paket ohne CRC-Check akzeptiert.
- Nach dem CRC-Check wird OK<CR+LF> oder ERROR<CR+LF> zurückgesendet. Genauere Angaben bekommt man im Fehlerfall mit AT+READERROR.
- Nachdem das Paket gesendet wurde befindet sich das CF-Modul wieder im Kommandomodus d.h. für ein weiteres Paket muss erneut AT+WRITEPACKET und ein Paket gesendet werden.

## DATENSTROM UND PAKETMODUS

Es stehen zwei Möglichkeiten zur Verfügung Daten zur Karte zu schreiben bzw. Daten von der Karte zu lesen. Beim „Datastream“ werden die Daten kontinuierlich zur Avisaro Box geschrieben, während sie beim „Paketmodus“ in Blöcken übertragen werden.

Vorteil des Paketmodus ist insbesondere die garantierte Datenübertragung. Es kann kein Datenverlust durch einen Bufferüberlauf entstehen, ebenso kann kein Datenverlust durch Fehler auf z.B. dem seriellen Kabel entstehen.

Welcher Modus einfacher zu Verwenden ist, hängt von der Anwendung ab. Wenn immer eine bekannte Datenmenge (z.B. ein Datagramm) übertragen wird, dann bietet sich der Packetmodus an, da keine Stop-Sequenz (drei Plus-Zeichen mit Pause) generiert werden muss. Werden kontinuierlich Daten unterschiedlicher Menge geschrieben werden müssen, bietet sich der Datenstrom Modus an.

	<b>at+writepacket</b>	<b>at+datastream</b>
Vorteil	<ul style="list-style-type: none"> <li>• Datensicherung durch CRC</li> <li>• Datenverlust ausgeschlossen</li> </ul>	<ul style="list-style-type: none"> <li>• Einfache Handhabung bei variablen Datenmengen</li> </ul>
Nachteil	<ul style="list-style-type: none"> <li>• Die Datenmenge muss vor dem Schreiben bekannt sein</li> <li>• CRC Datensicherung erfordert Rechenaufwand</li> </ul>	<ul style="list-style-type: none"> <li>• Eine Stopsequenz muss erzeugt werden, um wieder in den Befehlsmodus zurück zu wechseln</li> <li>• Keine besondere Datensicherung</li> </ul>

## AUTOMATIKMODUS

Der Automatikmodus dient zum automatischen Loggen von Daten. Dieser Modus ist im Benutzerhandbuch beschrieben.

---

## AT-KOMMANDOS: ALLGEMEINE HINWEISE

---

Allgemeine Hinweise:

- Alle AT-Befehle werden mit „Carriage Return“ (<cr>) und „Line Feed“ (<lf>) abgeschlossen. Der Befehl zum Erzeugen einer Datei *test.txt* lautet also: `at+create test.txt <cr><lf>`. In der nachfolgenden Aufstellung wird auf das <cr><lf> verzichtet.
- Alle Antworten (z.B. OK, ERROR, 32000) werden ebenfalls mit <cr><lf> abgeschlossen. Antworten wie z.B. die Anzahl der freien Speicherplätze wird im ASCII – Format zurück gegeben ('3','2','0','0','0').
- Sie können Groß- und Kleinbuchstaben verwenden. Alle Antworten vom Modul werden immer in Großbuchstaben gegeben.
- Parameter werden mit „Leerzeichen“ von einander getrennt.

## TABELLARISCHE ÜBERSICHT AT-KOMMANDOS

Auf den Befehl klicken, um weitergeleitet zur werden.

Befehl	Parameter	Beschreibung
--------	-----------	--------------

### Allgemeine Befehle

<a href="#">at+status</a>	Keine	Listet Statusinformationen (Versionsnummer, ...)
<a href="#">at+commands</a>	Keine	Zeigt verfügbare Befehle an
<a href="#">at+errorlist</a>	Keine	Zeigt die Fehlercodes mit Bemerkung an
<a href="#">at+readerror</a>	Keine	Zeigt den letzten Fehler an
<a href="#">at+restart</a>	Keine	Startet das Modul neu
<a href="#">stopsequence</a>	+ + + / EOT	Bricht einen Vorgang ab

### Befehle zur Konfiguration der Schnittstellen und interner Parameter

<a href="#">at+linepar</a>	Beispiel : 9600 n 1 none	Setzt die Parameter für die RS232 Schnittstelle
<a href="#">at+i2caddr</a>	73	Setzt die Adresse auf dem I2C Bus
<a href="#">at+i2cflow</a>	On / Off	Schaltet I2C Flusskontrolle (Stretching) ein/aus
<a href="#">at+i2cspeed</a>	Normal / Fast	Definiert die I2C Busgeschwindigkeit
<a href="#">at+i2cpull</a>	None / SCL / SDA / Both	Schaltet Widerstände auf die I2C Leitung
<a href="#">at+spi</a>	polarity, phase, bit order, auto reset	Setzt die Parameter für die SPI Schnittstelle
<a href="#">at+stopsequence</a>	On / OFF / EOT	Definiert die Art der Stopsequence
<a href="#">at+can</a>	baudrate, extrx, exttx, stdrx, stdtx, rxid, txid, bandfilter, filterid, filtermask, filtermode, mode	Konfiguriert die CAN Schnittstelle
<a href="#">at+autolog</a>	On / Off	Direkter Start der Aufzeichnung im Automatikmodus
<a href="#">at+flushtimer</a>	Zeit	Schreibt periodisch alle gepufferten Daten
<a href="#">at+minlogfile</a>	1..1024	Größe einer einzelnen Datei im Loggermodus

### Befehle zum Schreiben und Lesen von Daten

<a href="#">at+datastream</a>	keine	Schaltet vom Befehlsmodus in den Datenmodus
<a href="#">at+writepacket</a>	Datenpaket	Schreibt ein Datenpaket
<a href="#">at+readpacket</a>	Keine	Liest ein Datenpaket

<a href="#">at+repeatpacket</a>	Keine	Wiederholt das letzte gelesene Datenpaket
<a href="#">at+packetlength</a>	Länge	Legt die Paketlänge beim Lesen fest

### Befehle zum Arbeiten mit Dateien und Verzeichnissen

<a href="#">at+open</a>	dateiname, \directory\dateiname	Öffnet eine Datei zum Lesen
<a href="#">at+create</a>	dateiname, \directory\dateiname	Erzeugt eine neue Datei zum Schreiben
<a href="#">at+append</a>	dateiname, \directory\dateiname	Öffnet eine vorhandene Datei um Daten anzuhängen
<a href="#">at+createdir</a>	\directory	Erzeugt ein neues Verzeichnis
<a href="#">at+close</a>	Keine	Schließt eine offene Datei
<a href="#">at+delete</a>	dateiname, \directory\dateiname, \directory	Löscht eine Datei oder ein Verzeichnis

### Befehle zum Abfragen von Datenträger- und Dateiinformatoren

<a href="#">at+freespace</a>	Keine	Ermittelt den freien Speicherplatz
<a href="#">at+diskspace</a>	Keine	Ermittelt die Größe des Datenträgers
<a href="#">at+filesize</a>	Keine	Ermittelt die Größe einer Datei
<a href="#">at+filedate</a>	Keine	Ermittelt das Dateidatum
<a href="#">at+dir</a>	\directory	Zeigt den Inhalt eines Verzeichnis an
<a href="#">at+cardpresent</a>	Keine	Überprüft, ob eine Speicherkarte eingelegt ist
<a href="#">at+mediainfo</a>	Keine	Gibt Detailinfos zur eingelegten Speicherkarte

### Weitere Befehle

<a href="#">at+setdate</a>	Datum	Setzt das interne Datum (erlischt bei Neustart)
<a href="#">at+settime</a>	Zeit	Setzt die interne Uhrzeit (erlischt bei Neustart)
<a href="#">at+dump</a>	Sector	Liest einen Sector aus (nur für Diagnose)

---

**ALLGEMEINE AT-BEFEHLE**


---

AT+STATUS		
<b>Funktion:</b>	Status abfragen	
<b>Beschreibung:</b>	Listet diverse interne Statusinformationen:	
<b>Parameter:</b>	keine	
<b>Antworten:</b>	OK + Informationen	
<b>Beispiel:</b>	at+status	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
keine	Statusinformationen: - Copyrightmeldung, Rev.Nr. etc. - Eingabeerkennung zwei/ein Zeichen - Art der aktiven Flow Control Methode - Auslastung des Eingangspuffers - Ob Flow-Control wirksam war - Ob Bytes verschluckt wurden	

[<Zurück zur Übersicht>](#)

AT+COMMANDS		
<b>Funktion:</b>	Verfügbare Befehle anzeigen	
<b>Beschreibung:</b>	Listet alle verfügbaren Befehle und die Anzahl der erwarteten Parameter auf. Die Ausgabe erfolgt ohne das führende „at+“.	
<b>Parameter:</b>	keine	
<b>Antworten:</b>	OK + Befehlsliste	
<b>Beispiel:</b>	at+commands	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
keine	Befehlsliste	

[<Zurück zur Übersicht>](#)

<b>AT+ERRORLIST</b>		
<b>Funktion:</b>	Verfügbare Fehlermeldungen anzeigen	
<b>Beschreibung:</b>	Listet alle verfügbaren Rückgabemeldungen des Befehls at+readerror.	
<b>Parameter:</b>	keine	
<b>Antworten:</b>	OK + Fehlerliste	
<b>Beispiel:</b>	at+errorlist	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
keine	Fehlerliste	

[<Zurück zur Übersicht>](#)

<b>AT+READERROR</b>		
<b>Funktion:</b>	Fehlerbeschreibung des letzten Fehlers ausgeben	
<b>Beschreibung:</b>	Liest Detailinformationen zu einer Fehlermeldung aus. Es wird eine Fehlernummer und eine Fehlermeldung im Klartext ausgegeben.	
<b>Parameter:</b>	keine	
<b>Antworten:</b>	Fehlernummer Fehlertext	
<b>Beispiel:</b>	at+readerror	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
keine	Fehlernummer Fehlertext	

[<Zurück zur Übersicht>](#)

<b>AT+RESTART</b>		
<b>Funktion:</b>	Neustart des Moduls	
<b>Beschreibung:</b>	Startet das Modul neu	
<b>Parameter:</b>	Keine	
<b>Antworten:</b>	Keine	
<b>Beispiel:</b>	at+restart	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
keine		

[<Zurück zur Übersicht>](#)

<b>STOPSEQUENZ (“+ + +” / EOT)</b>		
<b>Funktion:</b>	Die aktuell ausgeführte Anweisung wird beendet.	
<b>Beschreibung:</b>	Mit der Stop-Sequenz wird eine Anweisung beendet. Der Haupeinsatz ist das Zurückschalten vom Datenmodus in den Befehlsmodus. Dabei wird die geöffnete Datei geschlossen.	
<b>Parameter:</b>	Keine	
<b>Antworten:</b>	OK	
<b>Beispiel:</b>	<p>Wenn “at+module stopsequence on” eingestellt ist:</p> <pre>'+' .. Pause &gt; 0.5 sec .. '+ .. Pause &gt; 0.5 sec .. '+</pre> <p>Wenn “at+stopsequence eot” eingestellt ist</p> <p>Eot (= Ascii Zeichen 4)</p> <p>Schaltet vom Datenmodus in den Kommandomodus</p>	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
keine	<p>Mögliche Antworten: OK</p> <p>Mögliche Antworten von at+readerror: AT_ERR_HAPPY</p>	

[<Zurück zur Übersicht>](#)

---

**BEFEHLE ZUR KONFIGURATION**


---

<b>AT+LINEPAR</b>		
<b>Funktion:</b>	Stellt die Parameter für die Serielle Schnittstelle ein	
<b>Beschreibung:</b>	Setzt die Parameter für die serielle Schnittstelle. Um diesen Befehl zu senden, muss mit den gegenwärtig eingestellten Parametern gearbeitet werden. Die Einstellung wird dauerhaft bis zum nächsten at+linepara gespeichert. Nach dem Ein-/Ausschalten bzw. nach „at+restart“ werden die neuen Parameter aktiv.  Befehl ist nur bei Modulen mit RS232 Firmware verfügbar	
<b>Parameter:</b>	<speed><parity><stop><fluss>	
<b>Antworten:</b>	OK, ERROR	
<b>Beispiel:</b>	at+linepar 115200 n 1 none Stellt die Geschwindigkeit auf 115kBaudein.	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
<speed>	Baudrate: 1200, 2400,4800, 9600, 19200,38400,57600,115200	
<parity>	Parität für none, even, odd: n, e, o	
<stop>	Anzahl der Stopbits: 1, 2	
<fluss>	Flusskontrolle für none, software, hardware: n, hw, sw	

[<Zurück zur Übersicht>](#)

<b>AT+I2CADDR</b>		
<b>Funktion:</b>	Stellt die Adresse für die I2C Schnittstelle ein	
<b>Beschreibung:</b>	Mit diesem Befehl wird die Adresse des Moduls auf dem I2C Bus konfiguriert. Der Parameter wird als Dezimalwert angegeben.	
<b>Parameter:</b>	Address	
<b>Antworten:</b>	OK, ERROR	
<b>Beispiel:</b>	at+i2caddr 73      Stellt die Adresse des Moduls auf 73 (dezimal)	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
<adresse>	Stellt die I2C Busadresse auf <adresse>	

[<Zurück zur Übersicht>](#)

<b>AT+I2CFLOW</b>		
<b>Funktion:</b>	Die Flusskontrolle mittels „Clockstretching“ wird aktiviert.	
<b>Beschreibung:</b>	<p>Mit dem Befehl AT+I2CFLOW ON bzw. AT+I2CFLOW OFF wird für die I2C-Schnittstelle das sogenannte Clock Stretching aktiviert oder deaktiviert.</p> <p>Clock Stretching ist eine Methode zur Flußkontrolle, mit der ein I2C-Slave den Busmaster ausbremsen kann, falls dieser Daten senden sollte, die temporär noch nicht verarbeitet werden können. Der Slave zieht dazu die SCK-Leitung auf 0-Pegel, um dem Master zu signalisieren, daß eine kurze Pause nötig ist. Der Master muß nach jedem Taktimpuls den Zustand von SCK abfragen und gegebenenfalls warten, bis dieser wieder auf 1-Pegel gesprungen ist. Der Befehl speichert eine Einstellung im EEPROM. Um den Befehl wirksam werden zu lassen, muß das Modul neu gestartet werden.</p> <p>Im Defaultzustand ist das Clock Stretching ausgeschaltet, d.h. die AVISARO-Modul arbeiten ohne Flusskontrolle.</p> <p>Befehl ist nur bei Modulen mit I2C Firmware verfügbar</p>	
<b>Parameter:</b>	ON, OFF	
<b>Antworten:</b>	OK, ERROR	
<b>Beispiel:</b>	<p>at+i2cflow on</p> <p>Schaltet die I2C Flusskontrolle ein.</p>	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
<on, off >	<p>on: aktiviert Clockstretching, off: deaktiviert Clockstretching</p>	

[<Zurück zur Übersicht>](#)

<b>AT+I2CSPEED</b>		
<b>Funktion:</b>	Stellt das Timingverhalten des I2C Bus ein	
<b>Beschreibung:</b>	<p>Mit den Befehlen AT+I2CSPEED NORMAL und AT+I2CSPEED FAST wird das Timingverhalten der I2C-Schnittstelle im AVISARO-Modul beeinflusst.</p> <p>Im Modus NORMAL arbeitet das AVISARO-Modul im Standard I2C Modus bis 100kHz Taktfrequenz. Dieser Modus erlaubt die größtmögliche Kompatibilität mit vorhandener I2C-Hardware. Der Modus FAST erlaubt höhere Datenraten. Dabei darf der Master eine Taktfrequenz von 400kHz (I2C Fast-Mode) oder mehr benutzen, um mit dem AVISAROModul zu kommunizieren.</p> <p>Der Befehl speichert eine Einstellung im EEPROM. Um den Befehl wirksam werden zu lassen, muß das Modul neu gestartet werden.</p> <p>Per Default befindet sich das AVISARO-Modul im Kompatibilitätsmodus (LOW).</p> <p>Befehl ist nur bei Modulen mit I2C Firmware verfügbar</p>	
<b>Parameter:</b>	NORMAL, FAST	

<b>Antworten:</b>	OK, ERROR	
<b>Beispiel:</b>	at+i2cspeed fast Schaltet die I2C Bussgeschwindigkeit auf 400kbit/s ein.	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
<normal, fast >	normal: I2C Bussgeschwindigkeit liegt um die 100kBit/s fast: I2C Bussgeschwindigkeit liegt um die 400kBit/s	

[<Zurück zur Übersicht>](#)

<b>AT+I2CPULL</b>		
<b>Funktion:</b>	Schaltet Pull Up Widerstände am I2C Bus ein	
<b>Beschreibung:</b>	<p>Weil der I2C-Bus nach dem Wired-AND Prinzip arbeitet, sind für den Betrieb Pull-Up Widerstände nötig, die den High-Pegel erzeugen. Durch aktivieren der internen Pull-Ups im AVISARO-Modul, ist es möglich, ein AVISARO-Modul und einen I2C-Busmaster nur durch zwei Leitungen, also ohne sonstige externe Beschaltung, zu verbinden.</p> <p>Mit AT+I2CPULL können auf einer, keiner oder beiden I2C-Leitungen Pull-Up Widerstände eingeschaltet werden. Mit dem Parameter NONE, SCL, SDA oder BOTH kann bestimmt werden, welche der beiden Leitungen einen Widerstand erhalten soll.</p> <p>NONE Beide Pull-Up Widerstände sind deaktiviert.                      SCL Ein Pull-Up Widerstand befindet sich nur an der SCL-Leitung.                      SDA Ein Pull-Up Widerstand befindet sich nur an der SDA-Leitung.                      BOTH Beide Leitungen sind mit einem Pull-Up Widerstand versehen.</p> <p>Der Befehl speichert eine Einstellung im EEPROM. Um den Befehl wirksam werden zu lassen, muß das Modul neu gestartet werden.</p> <p>Im Defaultzustand sind beide internen Pull-Ups eingeschaltet. Sollte ein AVISARO-Modul in ein System integriert werden, daß bereits über Pull-Ups verfügt, dann sollten die internen Pull-Ups mit dem Befehl AT+I2CPULL NONE abgeschaltet werden.</p> <p>Befehl ist nur bei Modulen mit I2C Firmware verfügbar</p>	
<b>Parameter:</b>	NONE, SCL, SDA, BOTH	
<b>Antworten:</b>	OK, ERROR	
<b>Beispiel:</b>	at+i2cpull none  Schaltet die I2C Pull-Upwiderstände aus auf 400kbit/s ein.	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>

<none, scl, sda, both >	<p>NONE Beide Pull-Up Widerstände sind deaktiviert.                  SCL Ein Pull-Up Widerstand befindet sich nur an der SCL-Leitung.                  SDA Ein Pull-Up Widerstand befindet sich nur an der SDA-Leitung.                  BOTH Beide Leitungen sind mit einem Pull-Up Widerstand versehen.</p>
-------------------------	---

[<Zurück zur Übersicht>](#)

<b>AT+SPI</b>		
<b>Funktion:</b>	Stellt die Parameter für die SPI Schnittstelle ein	
<b>Beschreibung:</b>	Setzt die Parameter für die SPI Schnittstelle. Um diesen Befehl zu senden, muss mit den gegenwärtig eingestellten Parametern gearbeitet werden. Die Einstellung wird dauerhaft bis zum nächsten at+spi gespeichert. Nach dem Ein-/Ausschalten bzw. nach „at+restart“ werden die neuen Parameter aktiv.	
<b>Parameter:</b>	polarity, phase, bit order, auto reset	
<b>Antworten:</b>	OK, ERROR	
<b>Beispiel:</b>	at+spi polarity high  Stellt die Clock-Flanke auf positiven Übergang ein	
Parameter	Antworten / Rückgabe	Beschreibung
polarity <high, low>		Teilt dem AVISARO-Modul die Polarität des Taktsignals mit, welches der Master verwenden möchte. Erlaubt sind die Schlüsselwörter HIGH und LOW. HIGH sagt dem Modul, dass der Master positive Taktsignale verwendet. Bei LOW dagegen reagiert das Modul auf negative Taktsignale des Masters.
phase <odd, even>		<p>Dieser Parameter teilt dem AVISARO-Modul mit, welche Flanke des Taktsignals es als Änderung desselben erkennen soll. Erlaubt sind hierbei die Schlüsselwörter ODD und EVEN. Mit ODD wird das Modul so konfiguriert, dass bei jeder ungeradzahligem Flanke des Taktsignals ein Datenaustausch (ein Bit) stattfindet. Das heißt, daß die Flanken 1, 3, 5, 7... 15 (bei acht Taktimpulsen) ausschlaggebend sind. Bei EVEN werden die Flanken 2, 4, 6, 8, ... 16 verwendet.</p> <p>ACHTUNG/BESONDERHEIT: Ist der Parameter &lt;phase&gt; EVEN, dann kann der Master nach dem herunterziehen der Slave-Select Leitung (/SS) kontinuierliche Taktimpulse erzeugen (immer ein Vielfaches von Acht). Dies ist die bevorzugte Betriebsart. Ist &lt;phase&gt; ODD, dann muß der Master nach jedem übertragenen Byte die Slave-Select Leitung wieder hochziehen. Das Hochziehen der /SS-Leitung setzt auch den "Byte Stuffer" und den Status der Flußkontrolle (sofern auto reset aktiv ist, siehe unten) zurück. Diese Übertragungsart sollte deshalb nur für langsamen Betrieb und reine Textdaten benutzt werden.</p>
order <msb, lsb >		Dieser Parameter (bit order) bestimmt die Arbeitsrichtung der Schieberegister. Erlaubt sind die Schlüsselwörter LSB und MSB. Wird LSB angegeben, dann muss der Master das jeweils niederwertigste Bit (LSB) als erstes auf den Ausgang (MOSI Leitung) legen, bevor er ein Taktsignal ausgibt. Parallel dazu bekommt er vom Slave dessen niederwertigstes Bit zuerst gesendet. Ist der Parameter MSB, dann arbeiten beide Schieberegister anders herum.
autoreset <on, off >		Der letzte und vierte Parameter (auto reset) legt fest, welchen Einfluß das Hochziehen der /SS-Leitung auf den internen Status des Moduls hat. Ist er ON, dann führt das Hochziehen von /SS dazu, daß Byte Stuffer und Flußkontrollstatus zurückgesetzt werden. Eine eventuell angefangene Byte-stuffed Message wird abgebrochen und die SPI-Software im Modul geht in

	seinen Grundzustand. ist auto reset OFF, dann bleiben alle internen Zustände erhalten und das Hochziehen von /SS führt nur dazu, daß MOSI und MISO hochohmig werden.
--	--

[<Zurück zur Übersicht>](#)

<b>AT+STOPSEQUENCE</b>		
<b>Funktion:</b>	Definiert die Stopsequenz für den Abbruch von Befehlen	
<b>Beschreibung:</b>	Mit dem Befehl AT+STOPSEQUENCE kann die Stopsequenz des AVISARO Moduls geändert werden. Die Stopsequenz (oder Escapesequenz) ist eine Folge von Zeichen, mit der das AVISARO Modul wieder in den Eingabemodus versetzt werden kann, wenn es sich im Datenmodus befindet, oder auf die Abarbeitung eines Befehls wartet. Nur im Eingabemodus können AT+-Befehle vom Modul erkannt und ausgeführt werden. Im Grundzustand ist eine spezielle Stopsequenz aktiv: Eine Folge von drei '+' -Zeichen, mit mindesten 500mS und höchsten 2S langen Pausen dazwischen.	
<b>Parameter:</b>	ON, OFF, EOT	
<b>Antworten:</b>	OK	
<b>Beispiel:</b>	at+stopsequence on Schaltet die „+++“ Stopsequenz ein	
Parameter	Antworten / Rückgabe	Beschreibung
<on, off, eot >	AT+STOPSEQUENCE ON Dieser Befehl aktiviert die oben beschriebene Stopsequenz. Dies ist die Default-Einstellung.  AT+STOPSEQUENCE EOT Dieser Befehl aktiviert die Stopsequenz EOT (End of Transmission). EOT ist ein einzelnes Byte mit dem ASCII-Wert 4. Dieses Zeichen ist im standardisierten ASCII-Code als Ende der Übertragung definiert und wird von vielen Terminals als Ende der Sitzung bzw. Logout interpretiert. Die Stopsequenz EOT sollte nur bei der Übertragung von reinen Textdaten verwendet werden. Sie ist nicht geeignet für Binärdaten, die ein EOT als gültiges Datenwort enthalten können.  AT+STOPSEQUENCE OFF Mit diesem Befehl wird jegliche Erkennung von Stopsequenzen im AVISARO-Modul abgeschaltet. Diese Einstellung sollte nur in seltenen Sonderfällen gemacht werden, in denen die anderen beiden Stopsequenzen sich störend auf den Übertragung auswirken Laufende Aktionen im Modul können in diesem Modus nicht mehr abgebrochen werden.	

[<Zurück zur Übersicht>](#)

<b>AT+CAN</b>	
<b>Funktion:</b>	Stellt die Parameter für die CAN Schnittstelle ein

<b>Beschreibung:</b>	<p>Setzt die Parameter für die CAN Schnittstelle. Um diesen Befehl zu senden, muss mit den gegenwärtig eingestellten Parametern gearbeitet werden. Die Einstellung wird dauerhaft bis zum nächsten at+can gespeichert. Nach dem Ein-/Ausschalten bzw. nach „at+restart“ werden die neuen Parameter aktiv.</p> <p>Der Übersicht halber, sind alle CAN Befehle hier in dieser Tabelle zusammengefasst.</p>	
<b>Parameter:</b>	baudrate, extrx, exttx, stdrx, stdtx, rxid, txid, bandfilter, filterid, filtermask, filtermode, canmode	
<b>Antworten:</b>	OK, ERROR	
<b>Beispiel:</b>	<p>at+can baudrate 1000</p> <p>Stellt die Baudrate auf 1 Mbit/s</p>	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
at+canbaudrate <16...1000>		<p>Mit dem Befehl AT+CANBAUDRATE &lt;x&gt; kann die Baudrate des Moduls auf die des CAN-Buses eingestellt werden. &lt;x&gt; kann zwischen 16 und 1000 liegen und legt die Baudrate in Kilobits pro Sekunde fest.</p> <p>Werte unterhalb von 16 und oberhalb von 1000 werden vom Modul abgelehnt (Rückgabe: ERROR) . Ein darauffolgendes AT+READERROR würde 04 PARAMTER OUT OF RANGE zurüchliefern.</p> <p>Folgende Baudraten sind einstellbar: 16...35, 37, 38, 40, 41, 43, 45, 47, 50, 52, 55, 58, 62, 65, 71, 76, 83, 90, 100, 111, 125, 142, 166, 200, 250, 333, 500 und 1000 kBits/s</p> <p>Dazwischen liegende Werte stellen die nächst höhere Baudrate ein, z.B. würde man mit AT+CANBAUDRATE 121 die Baudrate auf 125 kBits/s einstellen.</p>
at+canextrx <0 ...1FFFFFFF>		<p>Mit dem Befehl AT+CANEXTRX &lt;x&gt; wird festgelegt, welche Extended Message-ID das Modul als “an sich gerichtet” akzeptieren soll. Alle Extended CAN-Messages mit dieser Message-ID werden vom Modul verarbeitet, falls dieses für den Empfang von Extended-IDs konfiguriert ist (Siehe auch AT+CANRXID und AT+CANTXID).</p> <p>AT+CANEXTRX &lt;x&gt; verlangt nach einer Hexadezimalzahl zwischen 0 und 1FFFFFFF (einschließlich). Damit ist der gesamte Wertebereich von CAN2.0B wählbar. Werte oberhalb von 1FFFFFFF als Parameter von AT+CANEXTRX werden vom Modul abgelehnt.</p>
at+canexttx <0 ..1FFFFFFF >		<p>Mit dem Befehl AT+CANEXTTX &lt;x&gt; wird festgelegt, welche Extended Message-ID das Modul zum Senden eigener Messages verwenden soll, falls dieses für das Senden von Extended-IDs konfiguriert ist (Siehe auch AT+CANRXID und AT+CANTXID).</p> <p>AT+CANEXTTX &lt;x&gt; verlangt nach einer Hexadezimalzahl zwischen 0 und 0x1FFFFFFF (einschließlich). Damit ist der gesamte Wertebereich von CAN2.0B wählbar. Werte oberhalb von 0x1FFFFFFF als Parameter von AT+CANEXTTX werden vom Modul abgelehnt.</p>
at+canstdrx <0 ... 7FF >		<p>Mit dem Befehl AT+CANSTDRX &lt;x&gt; wird festgelegt, welche Standard Message-ID das Modul als “an sich gerichtet” akzeptieren soll. Alle Standard CAN-Messages mit dieser Message-ID werden vom Modul verarbeitet, falls dieses für den Empfang von Standard-IDs konfiguriert ist (Siehe auch AT+CANRXID und AT+CANTXID).</p> <p>AT+CANSTDRX &lt;x&gt; verlangt nach einer Hexadezimalzahl zwischen 0 und 7FF (einschließlich). Damit ist der gesamte Wertebereich von CAN2.0A Message-IDs wählbar. Werte oberhalb von 7FF als Parameter von AT+CANSTDRX werden vom Modul abgelehnt.</p>

<p>at+canstdtx &lt;0 ... 7FF &gt;</p>	<p>Mit dem Befehl AT+CANSTDTX &lt;x&gt; wird festgelegt, welche Standard Message-ID das Modul zum Senden eigener Messages verwenden soll, falls dieses für das Senden von Standard-IDs konfiguriert ist (Siehe auch AT+CANRXID und AT+CANTXID).</p> <p>AT+CANSTDTX &lt;x&gt; verlangt nach einer Hexadezimalzahl zwischen 0 und 7FF (einschließlich). Damit ist der gesamte Wertebereich von CAN2.0A Message-IDs wählbar. Werte oberhalb von 7FF als Parameter von AT+CANSTDTX werden vom Modul abgelehnt.</p>
<p>at+canrxid &lt;standard / extended&gt;</p>	<p>Dieser Befehl dient zur Auswahl des Typs von Message-ID, den das Modul als "an sich gerichtet" akzeptieren soll. Mit den Befehlen AT+CANSTDRX und AT+CANEXTRX werden beide Message-IDs, also eine Standard- und eine Extended-MessageID im Modul gespeichert. Mit AT+CANRXID STANDARD oder AT+CANRXID EXTENDED wird entweder die eine oder die andere der beiden Message-IDs aktiviert.</p>
<p>at+cantxid &lt;standard / extended&gt;</p>	<p>Dieser Befehl dient zur Auswahl des Typs von Message-ID, den das Modul für eigene Messages verwenden soll. Mit den Befehlen AT+CANSTDTX und AT+CANEXTTX werden beide Message-IDs, also eine Standard- und eine Extended-MessageID im Modul gespeichert. Mit AT+CANTXID STANDARD oder AT+CANTXID EXTENDED wird entweder die eine oder die andere der beiden Message-IDs aktiviert.</p>
<p>at+canbandfilter &lt;on / off &gt;</p>	<p>Mit diesem Befehl kann für den Empfang von CAN Messages ein zusätzlicher Bandfilter ein- oder ausgeschaltet werden. Der Bandfilter ermöglicht den Empfang einer Menge von CAN-Messages, die das Modul im Normalfall ignorieren würde.</p> <p>AT+CANBANDFILTER OFF: Das Modul akzeptiert nur Messages, die exakt die Message-ID haben, welche mit AT+CANEXTRX, AT+CANSTDRX und AT+CANRXID festgelegt wurde.</p> <p>AT+CANBANDFILTER ON: Das Modul empfängt und verarbeitet zusätzlich zur "eigenen" Message-ID auch noch solche, die der Bandfilter durchgelassen hat.</p> <p>Mit den Befehlen AT+CAN FILTERMODE, AT+CAN FILTERMASK und AT+CAN FILTERID kann der Bandfilter konfiguriert werden. Der Filter arbeitet nach dem folgenden Prinzip:</p> <pre>// msg_id: ID der empfangenen Message // filtermask: Maske, die mit AT+FILTERMASK gesetzt wird // filter_id: Filter-ID, die mit AT+FILTERID gesetzt wird if ((msg_id   filtermask) == (filter_id   filtermask)){ // Message wird durchgelassen}</pre> <p>In Worten: Ist das Ergebnis der bitweisen ODER-Verknüpfung einer empfangenen Message-ID mit der Filtermaske gleich der bitweisen ODER-Verknüpfung der Filter-ID mit der selben Filtermaske, dann wird die Message durchgelassen.</p> <p>Die Filtermaske enthält also "don't care" Bits, was bedeutet, daß alle Bits die in der Filtermaske auf '1' stehen, bei der Überprüfung der ID der empfangenen Message keine Bedeutung haben.</p> <p>Beispiele:</p> <ul style="list-style-type: none"> <li>• Wenn die Filter-ID auf 0x100 eingestellt wird und die Filtermaske den Wert 0x03 enthält, werden alle Messages mit den IDs 0x100, 0x101, 0x102 und 0x103 durchgelassen.</li> <li>• Steht die Filter-ID auf 0x100, aber die Filtermaske hat den Wert 0, dann wird vom Bandfilter nur die ID 0x100 durchgelassen.</li> <li>• Enthält die Filtermaske den Wert 0x1FFFFFFF, dann werden alle Messages vom Bandfilter durchgelassen, unabhängig vom Wert der Filter-ID.</li> </ul>
<p>at+canfilterid &lt;0 ... 1FFFFFFF &gt;</p>	<p>Mit diesem Befehl kann die FilterID, d.h. die Basis-ID für den Bandfilter eingestellt werden. Dies ist die Message-ID, welche vom Bandfilter immer durchgelassen wird, auch wenn die Filtermaske 0 ist.</p>

	<p>Filtermaske 0 ist.                  Als Parameter für AT+CANFILTERID &lt;x&gt; ist der gesamte Wertebereich der CAN2.0B Message-IDs (0...1FFFFFFF) erlaubt und muß als Hexadezimalwert eingegeben werden.                  Werte außerhalb dieses Bereichs werden vom Modul zurückgewiesen.                  Für eine genauere Beschreibung der Bandfilterfunktionalität siehe: AT+CANBANDFILTER weiter oben.</p>
<p>at+canfiltermask                  &lt;0 ... 1FFFFFFF &gt;</p>	<p>Mit diesem Befehl kann die Filtermaske für den Bandfilter gewählt werden. Die Filtermaske enthält die "don't care" Bits und bestimmt daher die Bandbreite des Filters.</p> <p><u>ACHTUNG/BESONDERHEIT:</u> Ist dieser Wert 0x1FFFFFFF und der Filtermodus auf "Extended" eingestellt (siehe AT+CANFILTERMODE), dann werden sämtliche CAN Messages durchgelassen, unabhängig von deren Format (Standard oder Extended) .</p> <p>Als Parameter für AT+CANFILTERMASK &lt;x&gt; ist der gesamte Wertebereich der CAN2.0B Message-IDs (0...1FFFFFFF) erlaubt und muß als Hexadezimalwert eingegeben werden.                  Werte außerhalb dieses Bereichs werden vom Modul zurückgewiesen.</p> <p>Für eine genauere Beschreibung der Bandfilterfunktionalität siehe: AT+CANBANDFILTER weiter oben.</p>
<p>at+canfiltermode                  &lt;0 ... 1FFFFFFF &gt;</p>	<p>Der Befehl AT+CANFILTERMODE dient zur Auswahl des Adressraums für den Bandfilter.</p> <p>Gibt man AT+CANFILTERMODE EXTENDED an, dann gelten die Einstellungen für den Bandfilter, also AT+CANFILTERID und AT+CANFILTERMASK für CAN2.0B Messages. Bei AT+CANFILTERMODE STANDARD werden vom Bandfilter nur CA2.0A Messages durchgelassen. Ist der Adressraum "Standard" ausgewählt, dann werden von den Parametern des Bandfilters (Mask und ID) jeweils nur die unteren 11 Bits für die Filterung benutzt.</p> <p>Die Kombination aus AT+CANFILTERMODE EXTENDED und AT+CANFILTERMASK 1FFFFFFF sorgt dafür, daß alle Messages durchgelassen werden, gleich ob CAN2.0A oder CAN2.0B (siehe auch die Beschreibung von AT+CANFILTERMASK). Der Bandfilter ist damit maximal geöffnet.</p>
<p>at+canmode                  &lt;data, message&gt;</p>	<p>Der Befehl AT+CANMODE dient zur Umschaltung zwischen den beiden Betriebsarten "Datenmodus" und "Messagemodus".</p> <p>Im Datenmodus (AT+CANMODE DATA) verarbeitet das Modul ausschließlich den Inhalt des Nutzdatenfeldes von CAN Datenframes. Eingangsseitig wird der Inhalt aller Datenframes, die von den Filtern durchgelassen werden (siehe dazu auch AT+CANFILTERMODE, AT+CANRXID und verwandte Kommandos), in den Eingangspuffer eingelesen und alle anderen Informationen werden verworfen. Ausgangsseitig generiert das Modul aus einem unformatierten Datenstrom selbstständig CAN-Datenframes mit der eingestellten Message-ID (siehe dazu auch AT+CANTXID und verwandte Kommandos).</p> <p>Im Messagemodus (AT+CANMODE MESSAGE) werden vollständige CAN-Messages vom Modul verarbeitet. Auf dem CAN-Bus empfangene Messages werden, falls von den Filtern durchgelassen, vom Modul in ein spezielles Format umgewandelt (siehe unten: Format zur Speicherung und Übermittlung von CAN-Frames) und in diesem Format weiterverarbeitet. Auf dem umgekehrten Weg, also über TCP/UDP oder von der CF-Karte gelesene Datenschnipsel in diesem Format, werden vom Modul in CAN-Frames umgewandelt und auf den Bus gesendet.</p>

[<Zurück zur Übersicht>](#)

<b>AT+AUTOLOG</b>		
<b>Funktion:</b>	Definiert das Start-Verhalten im Automatikmodus	
<b>Beschreibung:</b>	Mit dem Befehl AT+AUTOLOG ON wird der automatische Logging-Modus aktiviert. Befindet sich auf der CF-Karte eine Datei namens indata.log, dann wird sofort nach dem Einschalten des Moduls die Datei geöffnet und eingehende Daten an diese angehängt. Die Datei kann durch Druck auf den Taster, geschlossen werden. Der automatische Logging-Modus kann durch den Befehl AT+AUTOLOG OFF wieder aufgehoben werden. Ist der automatische Logging-Modus abgeschaltet, dann muß die Aufzeichnung von Daten in die Datei indata.log manuell, also durch Druck auf den Taster, gestartet werden. Das automatische Logging funktioniert nur, wenn sich die Datei indata.log auf der CF-Karte befindet.	
<b>Parameter:</b>	ON, OFF	
<b>Antworten:</b>	OK	
<b>Beispiel:</b>	at+autolog on : Beim Neustart direkt mit der Aufzeichnung gegonnen.	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
<on, off >		

[<Zurück zur Übersicht>](#)

<b>AT+FLUSHTIMER</b>		
<b>Funktion:</b>	Definiert das periodische Speichern gepufferter Daten	
<b>Beschreibung:</b>	Der Befehl AT+FLUSHTIMER t sorgt dafür, dass zum Schreiben geöffnete Dateien periodisch geschlossen und dann wieder geöffnet werden. Das Schließen und Öffnen der Datei bewirkt, daß in regelmäßigen Abständen alle intern gepufferten Daten auf die Karte geschrieben und alle dazugehörigen FAT-Einträge erneuert werden. AT+FLUSHTIMER kann vor Datenverlusten schützen, die entstehen, wenn das Modul ausgeschaltet wird ohne die Datei vorher zu schließen. Der Flushtimer wird aktiviert durch Eingabe von AT+FLUSHTIMER t, wobei "t" die Zeit in Sekunden ist, nach der die Datei periodisch geschlossen und wieder geöffnet werden soll. Das kleinste Intervall ist eine Sekunde, das Größte 65534 Sekunden (etwa alle 18.2 Stunden). Durch Eingabe des Befehls AT+FLUSHTIMER 0 wird der Flushtimer deaktiviert. Zum Schreiben geöffnete Dateien bleiben dann dauerhaft geöffnet, bis sie auf gewöhnlichen Wege geschlossen werden.	
<b>Parameter:</b>	<0...65534>	
<b>Antworten:</b>	OK	

<b>Beispiel:</b>	at+flushtimer 5 : Alle 5 Sekunden werden alle Daten im Eingangsbuffer geschrieben	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
<0 ... 65534 >		

[<Zurück zur Übersicht>](#)

<b>AT+MINLOGFILE</b>		
<b>Funktion:</b>	Größe einer Datei beim automatischen Loggen (in Mbyte)	
<b>Beschreibung:</b>	<p>Mit dem Befehl AT+MINLOGFILE wird die Mindestgröße der einzelnen Dateien für fragmentiertes Datenlogging festgelegt. Der Befehl erwartet einen Parameter, welcher diese Dateigröße in kBytes (1kByte = 1024 Bytes) angibt.</p> <p>Erlaubte Werte sind 1 bis 1024 kBytes. Werte außerhalb dieses Bereichs werden mit einer Fehlermeldung vom Modul abgelehnt.</p> <p>Siehe „Fragmentiertes Datenlogging“ im Benutzerhandbuch</p>	
<b>Parameter:</b>	<0...1024>	
<b>Antworten:</b>	OK	
<b>Beispiel:</b>	at+minlogfile 5 : Wenn die Datei mindestens 5 Mbyte groß ist, wird eine neue Datei angelegt.	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
<0 ... 1024 >		

[<Zurück zur Übersicht>](#)

## BEFEHLE ZUM SCHREIBEN UND LESEN VON DATEN

AT+DATASTREAM		
<b>Funktion:</b>	Schaltet vom Kommandomode in den Datamode	
<b>Beschreibung:</b>	Dieses Kommando schaltet vom Kommandomodus in den Datenmodus. Dies gelingt aber nur, wenn eine Datei zum Schreiben oder Lesen geöffnet ist.  Um vom Datamodus wieder zum Kommandomodus zurückzuwechseln kann die Stopsequenz (,+ + +?) verwendet werden. Ist eine Datei komplett gelesen worden, wechselt das Modul automatisch in den Kommandomodus.	
<b>Parameter:</b>	Keine	
<b>Antworten:</b>	OK, ERROR	
<b>Beispiel:</b>	at+datastream  Wechselt in den Datamodus. Antwort: OK	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
keine	Mögliche Antworten: OK, ERROR	

[<Zurück zur Übersicht>](#)

AT+WRITEPACKET		
<b>Funktion:</b>	Daten blockweise als Paket schreiben	
<b>Beschreibung:</b>	Anstelle eines kontinuierlichen Datenstroms werden Daten packetweise zum Schreiben übermittelt. Eine Datei muss vorher zum Schreiben geöffnet worden sein. Das Modul antwortet auf at+writepacket mit OK oder ERROR. Dann kann das Paket übertragen werden. Das Format ist: <ul style="list-style-type: none"> <li>• 2 Bytes: Datenlänge (nur Daten, ohne CRC). Das niederwertige Byte wird zuerst übertragen.</li> <li>• x Bytes: Daten (Anzahl gemäß Länge, Maximal: 1536 Bytes.)</li> <li>• 2 Bytes: CRC Prüfsumme. Die CRC Überprüfung kann ausgeschaltet werden, indem zwei Bytes mit einer 0 übertragen werden. Das Avisaro Modul antwortet mit OK oder ERROR. Bei ‚ERROR‘ (z.B. falsche Prüfsumme) wurde das Paket nicht auf das Speichermedium geschrieben. Die Übertragung kann mit der Stop Sequenz abgebrochen werden. Die maximale Paketlänge ist durch das Modul vorgegeben (meist 512 Bytes).</li> </ul>	
<b>Parameter:</b>	<i>packet (length, data, crc)</i>	

<b>Antworten:</b>	OK, ERROR	
<b>Beispiel:</b>		
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
<i>packet (length, data, crc)</i>	<p><b>Voraussetzung:</b></p> <p>Funktioniert nicht, wenn XON/XOFF aktiv ist. Eine Datei muss vorher geöffnet worden sein.</p> <p><b>Mögliche Rückgabewerte von at+readerror:</b></p> <ul style="list-style-type: none"> <li>- IMPROPER FLOW CONTROL METHOD</li> <li>- FILE NOT OPEN</li> <li>- I AM HAPPY</li> </ul>	

[<Zurück zur Übersicht>](#)

<b>AT+READPACKET</b>		
<b>Funktion:</b>	Daten blockweise als Paket lesen	
<b>Beschreibung:</b>	<p>Anstelle eines kontinuierlichen Datenstroms werden Daten paketweise gelesen. Eine Datei muss vorher zum Lesen geöffnet worden sein.</p> <p>Das Modul antwortet mit Daten im Paketformat:</p> <ul style="list-style-type: none"> <li>2 Bytes: Datenlänge (ohne CRC)</li> <li>x Bytes: Daten (Anzahl gemäß Länge)</li> <li>2 Bytes: CRC Prüfsumme</li> </ul> <p>Mit einem wiederholten at+readpackt Befehl werden die nächsten Daten aus der geöffneten Datei angefordert.</p> <p>Die maximale Paketgröße lässt sich konfigurieren.</p>	
<b>Parameter:</b>	<i>keine</i>	
<b>Antworten:</b>	OK, ERROR	
<b>Beispiel:</b>		
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
<i>keine</i>	<i>packet (length, data, crc)</i>	

[<Zurück zur Übersicht>](#)

AT+REPEATPACKET		
<b>Funktion:</b>	Das letzte gelesene Paket wiederholen	
<b>Beschreibung:</b>	Das letzte Datenpaket wird wiederholt. Dies wird verwendet, um z.B. bei einer fehlerhaften CRC Prüfsumme die Daten erneut anzufordern. Das Modul antwortet mit Daten im obigen Paketformat.	
<b>Parameter:</b>	<i>keine</i>	
<b>Antworten:</b>	OK, ERROR	
<b>Beispiel:</b>		
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
<i>keine</i>	<i>packet (length, data, crc)</i>	

[<Zurück zur Übersicht>](#)

AT+PACKETLENGTH		
<b>Funktion:</b>	Definiert die Paketlänge für den at+readpacket Befehl.	
<b>Beschreibung:</b>	Beim Lesen einer Datei mit at+readpacket werden die Daten blockweise ausgegeben. Wie lang ein solcher Block ist, wird über at+packetlength angegeben.	
<b>Parameter:</b>	<0.. 512>	
<b>Antworten:</b>	OK, ERROR	
<b>Beispiel:</b>		
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
<0.. 512>		

[<Zurück zur Übersicht>](#)

---

**BEFEHLE ZUM ARBEITEN MIT DATEIEN UND VERZEICHNISSEN**


---

<b>AT+OPEN</b>		
<b>Funktion:</b>	Datei zum Lesen öffnen	
<b>Beschreibung:</b>	<p>Öffnet eine vorhandene Datei zum Lesen. Ist die Datei nicht vorhanden, wird ‚ERROR‘ zurück gegeben.</p> <p>Um die Datenübertragung zu starten, wird der Befehl at+datastream verwendet.</p> <p>Um den Vorgang abzuschließen, z.B. um eine andere Datei zu lesen, wird der Befehl at+close verwendet.</p> <p><b>TIP:</b> Der at+open Befehl kann auch verwendet werden, um zu testen ob eine Datei vorhanden ist: ‚OK‘: Datei ist da, ‚ERROR‘: Datei ist nicht da. Es muss nur nach einem ‚OK‘ der at+close Befehl geschickt werden, wenn die Datei nicht weiter verwendet werden soll.</p>	
<b>Parameter:</b>	<i>dateiname, \directory\dateiname</i>	
<b>Antworten:</b>	OK, ERROR	
<b>Beispiel:</b>	at+open test.txt  Öffnet die Datei test.txt zum Lesen	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
<i>dateiname, \directory\dateiname</i>	<p><b>Voraussetzung:</b> Es darf keine Datei geöffnet sein.</p> <p><b>Mögliche Rückgabewerte von at+readerror:</b></p> <ul style="list-style-type: none"> <li>- FILE IS OPEN</li> <li>- I AM HAPPY</li> <li>- Filesystem Fehler (Nummer 0x40 bis 0x4d)</li> </ul>	

[<Zurück zur Übersicht>](#)

<b>AT+CREATE</b>	
<b>Funktion:</b>	Erzeugt eine neue Datei und öffnet sie zum schreiben
<b>Beschreibung:</b>	<p>Erzeugt eine Datei auf dem Datenträger mit dem Namen „Dateiname“. Wird nur ein Dateiname angegeben, wird die Datei im Wurzelverzeichnis angelegt. Mit „\directory\dateiname“ wird eine Datei im angegebenen Verzeichnis angelegt. Dies muss vorhanden sein. Ist eine Datei mit dem selben Namen bereits vorhanden, so wird diese gelöscht.</p> <p>Nach dem Erzeugen ist die Datei geöffnet. Soll sie nicht weiter verwendet werden, muss sie mit „at+close“ geschlossen werden.</p>
<b>Parameter:</b>	<i>dateiname, \directory\dateiname</i>

<b>Antworten:</b>	OK, ERROR	
<b>Beispiel:</b>	at+create test.txt Die Datei test.txt wird angelegt	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
<i>dateiname, \directory\dateiname</i>	<b>Voraussetzung:</b> Es darf keine Datei geöffnet sein.  <b>Mögliche Rückgabewerte von at+readerror:</b> - FILE IS OPEN - I AM HAPPY - Filesystem Fehler (Nummer 0x40 bis 0x4d)	

[<Zurück zur Übersicht>](#)

<b>AT+APPEND</b>		
<b>Funktion:</b>	Öffnet eine vorhandene Datei zum schreiben. Daten werden angehängt.	
<b>Beschreibung:</b>	Es wird die vorhandene Datei "Dateiname" zum schreiben geöffnet. Alle Daten werden an das Ende der Datei angehängt. Um Daten zu senden, kann der Befehl at+datastram verwendet werden.	
<b>Parameter:</b>	<i>dateiname, \directory\dateiname</i>	
<b>Antworten:</b>	OK, ERROR	
<b>Beispiel:</b>	at+append test.txt Die Datei test.txt wird geöffnet – neue Daten werden angehängt	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
<i>dateiname, \directory\dateiname</i>	<b>Voraussetzung:</b> Es darf keine Datei geöffnet sein.  <b>Mögliche Rückgabewerte von at+readerror:</b> - FILE IS OPEN - I AM HAPPY - Filesystem Fehler (Nummer 0x40 bis 0x4d)	

[<Zurück zur Übersicht>](#)

AT+CREATEDIR		
<b>Funktion:</b>	Erzeugt ein neues Verzeichnis.	
<b>Beschreibung:</b>	Es wird ein Verzeichnis mit dem Namen "Directory" angelegt. Ist das Verzeichnis bereits vorhanden, so wird eine Fehlermeldung ausgegeben.	
<b>Parameter:</b>	\directory	
<b>Antworten:</b>	OK, ERROR	
<b>Beispiel:</b>	at+createdir sensor  Es wird ein Verzeichnis mit dem Namen „sensor“ erzeugt.	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
\directory	<b>Voraussetzung:</b> Es darf keine Datei geöffnet sein.  <b>Mögliche Rückgabewerte von at+readerror:</b> - FILE IS OPEN - I AM HAPPY - Filesystem Fehler (Nummer 0x40 bis 0x4d)	

[<Zurück zur Übersicht>](#)

AT+CLOSE		
<b>Funktion:</b>	Eine offene Datei wird geschlossen	
<b>Beschreibung:</b>	Schließt eine geöffnete Datei. Dieser Vorgang ist wichtig, um die Integrität des Dateisystems zu gewährleisten. Der Befehl bezieht sich immer auf die aktuelle Datei, es werden also keine Parameter benötigt.	
<b>Parameter:</b>	keine	
<b>Antworten:</b>	OK, ERROR	
<b>Beispiel:</b>		
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
keine	<b>Voraussetzung:</b> Es muss eine Datei zum schreiben oder lesen geöffnet sein.  <b>Mögliche Rückgabewerte von at+readerror:</b> - FILE NOT OPEN - I AM HAPPY - Filesystem Fehler (Nummer 0x40 bis 0x4d)	

[<Zurück zur Übersicht>](#)

<b>AT+DELETE</b>		
<b>Funktion:</b>	Löscht eine Datei oder ein Verzeichnis	
<b>Beschreibung:</b>	Die Datei „dateiname“ oder das Verzeichnis „directory“ wird gelöscht. Ein zu löschendes Verzeichnis muss leer sein.	
<b>Parameter:</b>	<i>dateiname, \directory\dateiname</i>	
<b>Antworten:</b>	OK, ERROR	
<b>Beispiel:</b>	at+delete test.txt Löscht die Datei text.txt	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
<i>dateiname, \directory\dateiname</i>	<b>Voraussetzung:</b> Es darf keine Datei geöffnet sein.  <b>Mögliche Rückgabewerte von at+readerror:</b> - FILE IS OPEN - I AM HAPPY - Filesystem Fehler (Nummer 0x40 bis 0x4d)	

[<Zurück zur Übersicht>](#)

## BEFEHLE ZUM ABFRAGEN VON INFORMATIONEN

AT+FREESPACE		
<b>Funktion:</b>	Anzahl der freien Bytes	
<b>Beschreibung:</b>	Ermittelt den freien Speicherplatz auf dem Datenträger. Als Antwort wird die Zahl der freien Bytes zurück gegeben.	
<b>Parameter:</b>	<i>keine</i>	
<b>Antworten:</b>	OK, ERROR	
<b>Beispiel:</b>	at+freespace : Zeigt den freien Speicher an	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
<i>keine</i>	<b>Voraussetzung:</b> Es darf keine Datei geöffnet sein. <b>Mögliche Rückgabewerte von at+readerror:</b> - FILE IS OPEN - I AM HAPPY - Filesystem Fehler (Nummer 0x40 bis 0x4d)	

[<Zurück zur Übersicht>](#)

AT+DISKSPACE		
<b>Funktion:</b>	Anzahl der Bytes auf dem Datenträger	
<b>Beschreibung:</b>	Ermittelt die Größe des Datenträgers. Ist keine Speicherkarte eingelegt, wird ,error' zurück gegeben	
<b>Parameter:</b>	<i>keine</i>	
<b>Antworten:</b>	OK, ERROR	
<b>Beispiel:</b>	at+diskspace : Zeigt den Speicher der Karte an	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
<i>keine</i>	<b>Voraussetzung:</b> Es darf keine Datei geöffnet sein. <b>Mögliche Rückgabewerte von at+readerror:</b> - FILE IS OPEN - I AM HAPPY - Filesystem Fehler (Nummer 0x40 bis 0x4d)	

[<Zurück zur Übersicht>](#)

AT+FILESIZE		
<b>Funktion:</b>	Dateigröße in Bytes	
<b>Beschreibung:</b>	Ermittelt die Dateigröße. Die Datei muss vorher mit dem at+open Befehl geöffnet worden sein.	
<b>Parameter:</b>	<i>Keine</i>	
<b>Antworten:</b>	OK, ERROR	
<b>Beispiel:</b>	at+filesize Zeigt die gröÙe der gerade geöffneten Datei an	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
<i>keine</i>	<b>Voraussetzung:</b> Es muss eine Datei geöffnet sein. <b>Mögliche Rückgabewerte von at+readerror:</b> - FILE NOT OPEN - I AM HAPPY	

[<Zurück zur Übersicht>](#)

AT+FILEDATE		
<b>Funktion:</b>	Datum und Zeit der Datei im Format: jjjj/mm/tt hh:mm:ss	
<b>Beschreibung:</b>	Ermittelt das Datum und die Zeit der Datei. Die Datei muss vorher mit dem at+open Befehl geöffnet worden sein. Die Angaben folgen im Abstand mit einer Leerzeichen („Space“).	
<b>Parameter:</b>	<i>Keine</i>	
<b>Antworten:</b>	OK, ERROR	
<b>Beispiel:</b>	at+filedate Zeigt das Datum der gerade geöffneten Datei an	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
<i>keine</i>	<b>Voraussetzung:</b> Es muss eine Datei geöffnet sein. <b>Mögliche Rückgabewerte von at+readerror:</b> - FILE NOT OPEN - I AM HAPPY	

[<Zurück zur Übersicht>](#)

<b>AT+DIR</b>		
<b>Funktion:</b>	Zeigt den Inhalt des Wurzelverzeichnis oder eines Unterverzeichnisses an.	
<b>Beschreibung:</b>	Zeigt den Inhalt des Wurzelverzeichnis oder eines Unterverzeichnisses an. Es wird eine Liste mit dem Format Dateiname Größe Datum zurückgegeben. Die Liste wird mit einem ‚OK‘ oder ‚ERROR‘ <cr><lf> abgeschlossen.	
<b>Parameter:</b>	\ , \directory	
<b>Antworten:</b>	Dateiliste	
<b>Beispiel:</b>	at+dir \	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
\ \directory	Dateiname, Größe, Datum <cr><lf> Dateiname, Größe, Datum <cr><lf> ok,error<cr><lf>	
	<b>Voraussetzung:</b> Es darf keine Datei geöffnet sein. <b>Mögliche Rückgabewerte von at+readerror:</b> - FILE IS OPEN	

[<Zurück zur Übersicht>](#)

<b>AT+CARDPRESENT</b>		
<b>Funktion:</b>	Es wird überprüft, ob eine Speicherkarte eingesteckt ist (OK) oder nicht (ERROR).	
<b>Beschreibung:</b>	Es wird überprüft, ob eine Speicherkarte eingesteckt ist (OK) oder nicht (ERROR).	
<b>Parameter:</b>	<i>keine</i>	
<b>Antworten:</b>	OK, ERROR	
<b>Beispiel:</b>	at+cardpresent	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
<i>keine</i>		
	<b>Voraussetzung:</b> Funktioniert unabhängig davon, ob eine Datei geöffnet ist oder nicht. <b>Mögliche Rückgabewerte von at+readerror:</b> - MEMORY CARD NOT PRESENT - I AM HAPPY	

[<Zurück zur Übersicht>](#)

<b>AT+MEDIAINFO</b>		
<b>Funktion:</b>	Mit dem Befehl AT+MEDIAINFO werden Informationen über die eingelegte CF-Karte abgefragt	
<b>Beschreibung:</b>	Mit dem Befehl AT+MEDIAINFO werden Informationen über die eingelegte CF-Karte zur Hardware-Schnittstelle gesendet. Diese können z.B. benutzt werden, wenn aus unerklärlichen Gründen der Zugriff auf die Karte scheitern oder fehlerhaft sein sollte. Benutzer des Moduls können diesen Befehl aufrufen und die Ausgabe an AVISARO zur Fehlerdiagnose schicken.	
<b>Parameter:</b>	<i>Keine</i>	
<b>Antworten:</b>	Informationen / OK, ERROR	
<b>Beispiel:</b>	at+mediainfo	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
<i>keine</i>	Beispiel für eine 128MB CF-Karte: OFFSET_PARTITION_1: 32 OFFSET_PARTITION_2: 0 OFFSET_PARTITION_3: 0 OFFSET_PARTITION_4: 0 PARTITION_1_STATUS: 128 PARTITION_2_STATUS: 0 PARTITION_3_STATUS: 0 PARTITION_4_STATUS: 0 ROOT_DIRECTORY_START: 1095639119 BYTES_PER_SECTOR: 512 SECTORS_PER_CLUSTER: 4 RESERVED_SECTORS: 1 NUMBER_OF_FATS: 2 ROOT_ENTRIES: 512 TOTAL_SECTORS: 0 MEDIA_DESCRIPTOR: 248 SECTORS_PER_FAT: 247 SECTORS_PER_TRACK: 63 HEADS: 255 HIDDEN_SECTORS: 32 TOTAL_SECTORS_32MB: 253408 FIRST_PARTITION: 32 FILESYSTEM: FAT16 HAS_PARTITION_TABLE: YES	

[<Zurück zur Übersicht>](#)

## WEITERE BEFEHLE

AT+SETDATE		
<b>Funktion:</b>	Setzt das Datum im Avisaro Modul.	
<b>Beschreibung:</b>	Setzt das Datum im Avisaro Modul. Dieser Befehl ist immer nach dem Einschalten notwendig, wenn das Avisaro Modul über keine batteriegepufferte Uhr verfügt. So lange das Modul eingeschaltet ist, wird Zeit und Datum weitergezählt. Das Datum wird verwendet, wenn Dateien erzeugt werden. Das Datum wird im Format: jjjj mm tt übertragen (Leerzeichen dazwischen)	
<b>Parameter:</b>	<i>datum</i>	
<b>Antworten:</b>	OK, ERROR	
<b>Beispiel:</b>	at+setdate 2007 01 18 Setzt das Datum auf den 18 Januar 2007	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
<i>datum</i>	<p><b>Voraussetzung:</b></p> <p>Eingaben müssen numerisch sein d.h. dürfen keine Buchstaben haben. Tag und Monat werden auf gültigen Bereich geprüft.</p> <p><b>Mögliche Rückgabewerte von at+readerror:</b></p> <ul style="list-style-type: none"> <li>- INVALID PARAMETER</li> <li>- PARAMETER OUT OF RANGE</li> <li>- I AM HAPPY</li> </ul>	

[<Zurück zur Übersicht>](#)

AT+SETTIME		
<b>Funktion:</b>	Setzt die Zeit im Avisaro Modul.	
<b>Beschreibung:</b>	Setzt die Zeit im Avisaro Modul. Dieser Befehl ist immer nach dem Einschalten notwendig, wenn das Avisaro Modul über keine batteriegepufferte Uhr verfügt. Die Zeit wird verwendet, wenn Dateien erzeugt werden. Die Zeit wird im Format: hh mm ss übertragen. (Leerzeichen dazwischen).	
<b>Parameter:</b>	<i>time</i>	
<b>Antworten:</b>	OK, ERROR	
<b>Beispiel:</b>	at+settime 18 10 15 Setzt die Zeit auf 18 Uhr, 10 Minuten und 15 Sekunden	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>

<i>time</i>	<p><b>Voraussetzung:</b></p> <p>Eingaben müssen numerisch sein d.h. dürfen keine Buchstaben haben. Stunde, Minute und Sekunde werden auf gültigen Bereich geprüft.</p> <p><b>Mögliche Rückgabewerte von at+readerror:</b></p> <ul style="list-style-type: none"> <li>- INVALID PARAMETER</li> <li>- PARAMETER OUT OF RANGE</li> <li>- I AM HAPPY</li> </ul>
-------------	---

[<Zurück zur Übersicht>](#)

AT+DUMP		
<b>Funktion:</b>	Listet die Daten eines Sectors aus.	
<b>Beschreibung:</b>	Listet die Daten eines Sectors aus. Diese Funktion dient zur Diagnostik im Fehlerfall. Detaillierte Kenntnisse von Dateisystemen sind erforderlich.	
<b>Parameter:</b>	<i>sector</i>	
<b>Antworten:</b>	Sektor, OK, ERROR	
<b>Beispiel:</b>	at+sector 498  Gibt den Sector 498 aus.	
<b>Parameter</b>	<b>Antworten / Rückgabe</b>	<b>Beschreibung</b>
<i>sector</i>	<p><b>Voraussetzung:</b></p> <p>Funktioniert nur, wenn keine Datei geöffnet ist. Gibt einen Sektor der FlashDisk aus (512 Bytes).</p> <p><b>Mögliche Rückgabewerte von at+readerror:</b></p> <ul style="list-style-type: none"> <li>- FILE IS OPEN</li> <li>- I AM HAPPY</li> </ul>	

[<Zurück zur Übersicht>](#)

---

## PROGRAMMIERBEISPIELE

---

### QUELLCODE CRC BERECHNUNG

Im Paketmodus kann eine CRC Prüfsumme verwendet werden, um Daten bei der Übertragung zu sichern. Die Berechnung des CRCs kann nach folgendem Schema erfolgen:

```
static const unsigned int crc_tab[16] =
{
    0x0000, 0x1081, 0x2102, 0x3183,
    0x4204, 0x5285, 0x6306, 0x7387,
    0x8408, 0x9489, 0xA50A, 0xB58B,
    0xC60C, 0xD68D, 0xE70E, 0xF78F
};

unsigned short crc_update (unsigned short crc, unsigned char c)
{
    crc = (((crc >> 4) & 0x0FFF) ^ crc_tab[((crc ^ c) & 0x000F)]);
    crc = (((crc >> 4) & 0x0FFF) ^ crc_tab[((crc ^ (c>>4)) & 0x000F)]);
    return crc;
}
```

- Eine Variable (16 Bits, der CRC-Wert) wird mit 0xffff initialisiert:  
**unsigned short crc\_wert = 0xffff;**
- Für jedes Byte des Pakets wird die Variable verändert, indem man die obenstehende Funktion aufruft (in einer Schleife):  
**crc\_wert = crc\_update (crc\_wert, aktuelles\_byte);**
- Nachdem das letzte Byte verarbeitet wurde, wird die Variable negiert:  
**crc\_wert = ~crc\_wert;**

## QUELLCODE „WRITE PACKET“

Das folgende Beispiel illustriert, wie eine Anwendung ein Paket generieren könnte, das zum Avisaro Modul übertragen wird.

```
void CommWritePacket (HANDLE hcomm, char *data, short size)
{
    unsigned short crc = 0xffff;

    CommWrite (hcomm, size & 0xff);
    CommWrite (hcomm, size >> 8);
    while (size--)
    {
        CommWrite (hcomm, *data);
        crc = crc_update (crc, *data);
        data++;
    }

    //CommWrite (hcomm, 0); //diese Zeilen verwenden, wenn kein
    //CommWrite (hcomm, 0); //CRC verwendet werden soll
    crc = ~crc;
    CommWrite (hcomm, crc & 0xff); // diese Zeilen weglassen, wenn
    CommWrite (hcomm, crc >> 8); // kein CRC verwendet werden soll
}
```