

# ASTAS-DLL

## Bedienungsanleitung

A.S.T. Angewandte System-Technik GmbH  
Marschnerstraße 26 01307 Dresden  
Telefon (03 51) 44 55 30 Telefax (03 51) 44 55 555  
[www.ast.de](http://www.ast.de) astmr@ast.de

**Inhaltsverzeichnis**

<b>INHALTSVERZEICHNIS .....</b>	<b>1</b>
<b>1    ALLGEMEINES.....</b>	<b>2</b>
<b>2    DATENSTRUKTUREN .....</b>	<b>3</b>
<b>3    FUNKTIONEN .....</b>	<b>5</b>
<b>4    RÜCKGABEWERTE DER FUNKTIONEN.....</b>	<b>11</b>
<b>5    STATUS-BYTE .....</b>	<b>12</b>
<b>6    MASSEINHEITEN .....</b>	<b>13</b>
<b>7    GERÄTETYPEN.....</b>	<b>13</b>
<b>8    SONSTIGE DEFINES .....</b>	<b>13</b>

### 1 ALLGEMEINES

Es können max. 16 Geräte (AE703 / BD341/2 / BA661/2 / FFB201 / FFB204) an der USB-Schnittstelle durch die ASTAS-DLL verwaltet bzw. angesprochen werden.

Die Geräte werden in einem Strukturarray (siehe Kap. 2, *t\_DeviceInfo*) verwaltet. Der Listenindex *list\_idx* geht von 0...15. Die einzelnen Geräte werden über diesen Listenindex (siehe Kap.3, *list\_idx*) angesprochen.

Aktuelle Messerte werden über eine der beiden Strukturen (siehe Kap. 2, *t\_DeviceData* und *t\_DeviceDataFFB*) abgefragt. Darüber hinaus gibt es Funktionen zur Steuerung der Geräte.

Die Definitionen der Datenstrukturen, Funktionsprototypen und der einzelnen Konstanten ist in folgenden Dateien zu finden.

astas\_dll.h                   -> allg. C-Header-Datei

astas\_DLLImport.h   -> spez. Header-Datei für Microsoft Visual C++  
(DLL-Import Header-Datei)

#### **Hinweis 1:**

Für den Einsatz der ASTAS-DLL ist es von Vorteil die Software ASTAS zu installieren, um Einstellungen an den Geräten vornehmen zu können.

<http://www.ast.de/files/downloads/mess-und-regeltechnik/controller/ASTAS.zip>

Die jeweils aktuelle Version der ASTAS-DLL ist ebenfalls im Downloadbereich zu finden.

[http://www.ast.de/files/downloads/mess-und-regeltechnik/controller/ASTAS\\_DLL.zip](http://www.ast.de/files/downloads/mess-und-regeltechnik/controller/ASTAS_DLL.zip)

#### **Hinweis 2:**

***Ab ASTAS-DLL V0.13.x haben sich Reihenfolge und Variablentypen jeweils in den Strukturen der Geräteinformationen und Messwerte geändert! (siehe Kapitel 2 - Datenstrukturen)***

## 2 DATENSTRUKTUREN

### Struktur der Gerätedaten

```
typedef struct
{
    DWORD usb_hid_idx;           // USB-Geraete-Index -> interne Verwendung!
    int open;                    // Zustand -> 1=Open / 0=Close
    char vid_pid[256];           // VID/PID-String
    char dev_info[256];          // GeräteInfo-String
    char sn_info[256];           // S/N-String
    int hw_info;                 // HW-Info (5xxx)
    int fw_var;                  // HW-Variante
    int fw_vers;                 // FW-Version
    int chn;                     // Kanal
} t_DeviceInfo;
```

### Struktur der Messwerte

```
typedef struct
{
    float brutto;                // Brutto / Messwert
    float netto;                 // Netto
    float tara;                  // Tara
    float min;                   // Min (bezogen auf aktuelle An-Zeit)
    float max;                   // Max (bezogen auf aktuelle An-Zeit)
    int me;                      // Maßeinheit
    int dec;                     // Anzahl Nachkommastellen
    int status;                  // Status-Byte
} t_DeviceData;
```

### Struktur der Nominalwerte

```
typedef struct
{
    float nominal_value;         // Nennkennwert/Nominalwert
    unsigned int teds;           // Nennkennwert von TEDS (=1)
    unsigned int teds_sn;        // TEDS-S/N, wnn TEDS-Aufnehmer (!=0)
} t_NominalData;
```

Die Deklaration der Datenstruktur/des Datenarrays für max. 16, durch die ASTAS-DLL verwaltete, Geräte kann folgendermaßen aussehen.

```
t_DeviceInfo    myDeviceInfo[16];                // list_idx = 0...15
```

Diese muss vom aufrufenden Programm intern angelegt werden. Der Zeiger auf diese Datenstruktur bzw. das Datenarray wird dann an die Search-Funktion übergeben.

### **Hinweis 1:**

usb\_hid\_idx entspricht nicht list\_idx und wird nur für interne Zwecke verwendet!

### **Hinweis 2:**

In der Datenstruktur t\_DeviceData hat die Variable <chn> folgende Bedeutung.

- chn=1: FFB204-A / FFB201 / AE703 / BD341/2 / BA661/2
- chn=2: FFB204-B
- chn=3: FFB204-C
- chn=4: FFB204-D

### **Hinweis 3:**

Die verwendeten Datentypen entsprechen bei einer 32bit-DLL:

DWORD	32bit	INT32	signed int
int	32bit	INT32	signed int
unsigned int	32bit	UINT32	unsigned int
char	8bit	INT8	signed char
unsigned char	8bit	UINT8	unsigned char

### 3 FUNKTIONEN

```
extern "C" DLL_API int AstasDllInit(void);
extern "C" DLL_API int AstasDllReInit(void);
extern "C" DLL_API int AstasDllSearch(t_DeviceInfo *p_dev_info);
extern "C" DLL_API int AstasDllOpen(int list_idx);
extern "C" DLL_API int AstasDllClose(int list_idx);
extern "C" DLL_API int AstasDllReadData(int list_idx,
                                       t_DeviceData *dev_data);
extern "C" DLL_API int AstasDllNull(int list_idx);
extern "C" DLL_API int AstasDllTara(int list_idx);
extern "C" DLL_API int AstasDllGrossNet(int list_idx);
extern "C" DLL_API int AstasDllMax(int list_idx);
extern "C" DLL_API int AstasDllChangeUnit(int list_idx);
extern "C" DLL_API int AstasDllChangeRange(int list_idx);
extern "C" DLL_API int AstasDllAckChangeStatusBits(int list_idx);
extern "C" DLL_API int AstasDllCalibNull(int list_idx);
extern "C" DLL_API int AstasDllCalibEnd(int list_idx, float cal_val);
extern "C" DLL_API int AstasDllCalibSave(int list_idx);
extern "C" DLL_API int AstasDllCalibDelete(int list_idx);
extern "C" DLL_API int AstasDllSetMinMax(int list_idx, int mode);
extern "C" DLL_API int AstasDllReadNominalData(int list_idx,
                                              t_NominalData *nom_data);
extern "C" DLL_API int AstasDllReset(int list_idx);
extern "C" DLL_API int AstasDllDLLVersion(void);
```

**Hinweis 1:** Abhängig von der Firmwareversion des angeschlossenen Gerätes und des gewählten Gerätes können Funktionen nicht implementiert sein.

**Hinweis2:** Ab ASTAS-DLL V0.10.0 sind die Funktionen zum Lesen der Daten  
- ReadData und ReadDataFFB - getrennt für AE703 / BD341/2 / BA661/2 bzw. FFB201 / FFB204 zu verwenden.

**Hinweis3:** Ab ASTAS-DLL V0.11.0 haben alle Funktionen einen vorgestellten Präfix AstasDll... erhalten!

### **int AstasDllInit(void)**

Initialisierung der ASTAS-DLL. Dieser Schritt ist notwendig für eine korrekte Funktion der ASTAS-DLL.

Rückgabe: NO\_ERR / ERR

**Hinweis:** Diese Funktion muss zwingend als erste Funktion zur Initialisierung der DLL aufgerufen werden.

### **int AstasDllReInit(void)**

Re-Initialisierung der ASTAS-DLL.

Rückgabe: NO\_ERR / ERR

### **int AstasDllSearch(t\_DeviceInfo \*p\_dev\_info)**

Mit dieser Funktion werden alle angeschlossenen Geräte AE703 / BD341/2 / BA661/2 / FFB201 / FFB204 (max. 16) gesucht und die Informationen in der DeviceInfo-Struktur abgelegt. Diese Struktur muss vom aufrufenden Program als Zeiger/Pointer übergeben werden.

Rückgabe: Anzahl der gefundenen Geräte

**Hinweis:** Ab ASTAS-DLL V0.9.9 wird zusätzlich an den GeräteInfo-String (s.o.) der Gerätenamen aus „Setup -> Gerät-> Gerätebezeichnung“ (siehe ASTAS) nach einem Trennzeichen (Semikolon / „;“ bzw. 0x3B mit Leerzeichen / „ „ bzw. 0x20) angehängt.

Geräte-Info (ab ASTAS-DLL V0.9.9):

AE703: "AST\_AE703"  
BD341/2: "AST\_BD342"  
BA661/2: "AST\_BA662"  
FFB201: "AST\_FFB201"  
FFB204: "AST\_FFB204"

Beispiel: „AST\_AE703; AE703\_1 „

B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	T0	T1	S0	S1	S2	S3	S4	S5	S6	S7
	A	S	T	_	A	E	7	0	3	;		A	E	7	0	3	_	1	

B0..9: 10 Zeichen/Bytes interner Gerätebezeichner  
(linksbündig mit Leerzeichen aufgefüllt)  
T0: Trennzeichen(1) (Semikolon/0x3B)  
T1: Trennzeichen(2) (Leerzeichen/0x20)  
S0..7: 8 Zeichen/Bytes des Setup-Gerätebezeichners  
(rechtsbündig mit Leerzeichen aufgefüllt)

**`int AstasDllOpen(int list_idx)`**

Öffnen des gewählten Gerätes.

Rückgabe: NO\_ERR / *Fehlercode*

Geräte: AE703 / BD341/2 / BA661/2 / FFB201 / FFB204

**`int AstasDllClose(int list_idx)`**

Schliessen des gewählten Gerätes.

Rückgabe: NO\_ERR / *Fehlercode*

Geräte: AE703 / BD341/2 / BA661/2 / FFB201 / FFB204

**`int AstasDllReadData(int list_idx,  
t_DeviceData *p_dev_data)`**

Lesen der aktuellen Messserte(s) des gewählten Gerätes. Es muss die Struktur t\_DeviceData als Zeiger/Pointer übergeben werden.

Rückgabe: NO\_ERR / *Fehlercode*

Geräte: AE703 / BD341/2 / BA661/2 / FFB201 / FFB204

### **Hinweis:**

Als Rückgabe erfolgt der aktuelle Messwert/Anzeigewert des mobilen Gerätes.

**`int AstasDllNull(int list_idx)`**

Nullung des gewählten Gerätes.

Rückgabe: NO\_ERR / *Fehlercode*

Geräte: AE703 / BD341/2 / BA661/2

### **Hinweis:**

Eine nicht erfolgte Nullung wird nicht als fehlerhaft zurückgegeben.

Eine Nullung kann über die Rückgabe des Status-Bytes bei Lesen des aktuellen Messwertes kontrolliert werden.



***int AstasDllTara(int list\_idx)***

Tarierung des gewählten Gerätes.

Rückgabe: NO\_ERR / *Fehlercode*

Geräte: FFB201 / FFB204

***int AstasDllGrossNet(int list\_idx)***

Umschaltung zwischen Anzeige Brutto/Netto des gewählten Gerätes.

Rückgabe: NO\_ERR / *Fehlercode*

Geräte: AE703 / BD341/2 / FFB201 / FFB204

***int AstasDllMax(int list\_idx)***

Umschaltung zwischen Anzeige Brutto/Netto bzw. Max des gewählten Gerätes.

Rückgabe: NO\_ERR / *Fehlercode*

Geräte: FFB201 / FFB204

***int AstasDllChangeUnit(int list\_idx)***

Umschaltung der Maßeinheit des gewählten Gerätes.

Rückgabe: NO\_ERR / *Fehlercode*

Geräte: AE703 / BD341/2

***int AstasDllChangeRange(int list\_idx)***

Umschaltung des Messbereichs des gewählten Gerätes.

Rückgabe: NO\_ERR / *Fehlercode*

Geräte: AE703 / BD341/2

***int AstasDllAckChangeStatusBits(int list\_idx)***

Bestätigung der Statusbits ChangeUnit/ChangeRange. Gerät setzt darauf hin beide Statusbits zurück.

Rückgabe: NO\_ERR / *Fehlercode*

Geräte: AE703 / BD341/2

**`int AstasDllCalibNull(int list_idx)`**

Lastkalibrierung Nullpunkt (fest 0.0 = 0%).

Rückgabe: NO\_ERR / *Fehlercode*

Geräte: AE703 / BD341/2 / BA661/2

**`int AstasDllCalibEnd(int list_idx, float cal_val)`**

Lastkalibrierung Endpunkt (normierter variabler Wert vom Nennkennwert (100%)).

cal\_val: 1.0 = 100%

0.5 = 50%

0.0 = 0%

Rückgabe: NO\_ERR / *Fehlercode*

Geräte: AE703 / BD341/2 / BA661/2

**`int AstasDllCalibSave(int list_idx)`**

Lastkalibrierung abschließen und im Gerät speichern.

Rückgabe: NO\_ERR / *Fehlercode*

Geräte: AE703 / BD341/2 / BA661/2

**`int AstasDllCalibDelete(int list_idx)`**

Lastkalibrierung im Gerät löschen. Notwendig vor jedem Kalibriervorgang!

Rückgabe: NO\_ERR / *Fehlercode*

Geräte: AE703 / BD341/2 / BA661/2

**`int AstasDllSetMinMax(int list_idx, int mode)`**

Setzen der min/Max-Werte auf aktuelle Meßwerte.

mode: 1 – MODE\_SETMIN

2 – MODE\_SETMAX

Rückgabe: NO\_ERR / *Fehlercode*

Geräte: AE703 / BD341/2

```
int AstasDllReadNominalData(int list_idx,  
                             t_NominalData *p_nom_data)
```

Lesen des aktuellen Nennkennwertes aus TEDS oder Messbereich. Es muss die Struktur t\_NominalData als Zeiger/Pointer übergeben werden.

Rückgabe: NO\_ERR / Fehlercode

Geräte: AE703 / BD341/2

Wenn ein TEDS-Aufnehmer vorhanden ist wird der Nennkennwert aus dem TEDS gelesen und in der Struktur p\_nom\_data.teds = 1 und die Seriennummer des TEDS-Aufnehmers in p\_norm\_data.teds\_sn != 0 zurück gegeben..

Wenn kein TEDS-Aufnehmer am Gerät angeschlossen ist wird der Nennkennwert aus dem aktuellen Messbereich gelesen und in der Struktur p\_nom\_data teds=0 zurück gegeben.

```
int AstasDllReset(int list_idx)
```

Reset des gewählten Gerätes. Danach müssen die Geräte neu gesucht und geöffnet werden.

Rückgabe: NO\_ERR

Geräte: AE703 / BD341/2 / BA661/2 / FFB201 / FFB204

```
int AstasDllDLLVersion(void)
```

Abfrage der ASTAS-DLL-Version. Rückgabe ist ein int-Wert, der folgende Informationen enthält.

Byte3: Hauptversion

Byte2: Unterversion

Byte1: Revisionsversion

Byte0: Reserve (int. Zähler)

Beispiel:

Hex: 0x000E01yy – V0.14.1.yy

### 4 RÜCKGABEWERTE DER FUNKTIONEN

NO_ERR	0	kein Fehler
ERR	1	allg. Fehler (z.B. Gerät nicht ansprechbar)
ERR_DEVICE_NOT_OPEN	10	Gerät nicht geöffnet
ERR_FUNC_NOT_IMPL	20	Funktion nicht implementiert (abh. von der FW-Version des Gerätes)
ERR_DEVICE_READ	30	Lesefehler von Gerät (z.B. Gerät von USB getrennt)
ERR_FFB_RF	40	Funk-Fehler (FFB20x)
ERR_ONE_DEVICE_OPEN	50	Fehler bei Öffnen Gerät (wenn ein Gerät geöffnet ist)
ERR_DEVICE_READ_FALSE	60	Lesefehler (FFB20x <-> andere Geräte)
ERR_DEVICE_CALIB_MVV	100	bei Maßeinheit mV/V (UNIT_mV_V) keine Kalibrierung möglich
ERR_DEVICE_CALIB_READ_VAL	101	erst Messwert lesen, dann werden Kalibrier-Funktionen ausgeführt
ERR_DLL_REINIT	250	Fehler bei ReInit der ASTAS-DLL
ERR_DLL_MEM_INIT	251	Speicherfehler beim init. der ASTAS-DLL
ERR_DLL_CURR_INIT	252	DLL ist schon initialisiert
ERR_DEVICE_FALSE_IDX	253	falscher Wert für <i>list_idx</i>
ERR_DEVICE_NOT_EXIST	254	gewähltes Gerät nicht vorhanden
ERR_DLL_NOT_INIT	255	ASTAS-DLL wurde nicht initialisiert

### 5 STATUS-BYTE

Bei Abfrage der Messwerte wird das Status-Byte zurückgegeben.  
Die einzelnen Bitpositionen sind folgend erklärt.

#### AE703 / BD341/2 / BA661/2

-Reserviert-	0x01	0b00000001	// Reserviert
STATUS_NULL	0x02	0b00000010	// Status -> Null-Anzeige // (Netto-Wert)
STATUS_TARA	0x04	0b00000100	// Status -> Tara/Null
STATUS_OVER	0x08	0b00001000	// Status -> Ueberlast
STATUS_UNDER	0x10	0b00010000	// Status -> Unterlast
STATUS_CHG_UNIT	0x20	0b00100000	// Status -> Änd. Maßeinheit
STATUS_CHG_RANGE	0x40	0b01000000	// Status -> Änd. Messbereich
STATUS_ERR	0x80	0b10000000	// Status -> ADC-Error

#### FFB201 / FFB204

-Reserviert-	0x01	0b00000001	// Reserviert
STATUS_NULL	0x02	0b00000010	// Status -> Null-Anzeige // (Netto-Wert)
STATUS_TARA	0x04	0b00000100	// Status -> Tara/Null
STATUS_OVER	0x08	0b00001000	// Status -> Ueberlast
STATUS_UNDER	0x10	0b00010000	// Status -> Unterlast
STATUS_MAX	0x20	0b00100000	// Status -> Max.-Wert
STATUS_RF_IO	0x40	0b01000000	// Status -> RF i.O.
-Reserviert-	0x80	0b10000000	// Reserviert

### 6 MASSEINHEITEN

Die Bedeutung des Maßeinheiten-Byte ist im Folgenden erklärt

UNIT_USER	0	// User-Einheit	(Definierbar im Gerät bzw. per ASTAS-Software)
UNIT_N	1	// N	
UNIT_kN	2	// kN	
UNIT_g	3	// g	
UNIT_kg	4	// kg	
UNIT_t	5	// t	
UNIT_lbf	6	// lbf/Pfund	
UNIT_oz	7	// oz/Unze	
UNIT_mV_V	8	// mV/V	
UNIT_M300	9	// M300	
UNIT_M600	10	// M600	
UNIT_to	11	// to	(amerikanische Tonne)
// Drehmomenteneinheiten (nur bei AE703_PM, es fallen dafür die			
// Maßeinheiten 9=M300, 10=M600 weg)			
UNIT_Nm	9	// Nm	
UNIT_Ncm	10	// Ncm	
UNIT_ozIn	12	// ozIn	
UNIT_Inlb	13	// Inlb	
UNIT_ftlb	14	// ftlb	
UNIT_klbf	15	// klbf	

### 7 GERÄTETYPEN

AE703	5100
AE703_PM	5110
BD341/2	5140
FFB201	5200
FFB204	5240
BA661	5310
BA662	5330

### 8 SONSTIGE DEFINES

MODE_SETMIN	1	// Setzen des Min-Wertes auf aktuellen Meßwert
MODE_SETMAX	2	// Setzen des Max-Wertes auf aktuellen Meßwert

Benötigt für Funktion SetMinMax().