CU-LINEAR-DRIVE

Kolben Linearantrieb
Softwarebeschreibung

Autor: Christian Ulrich

Datum: 20.01.2009

Version: 1.00

Inhalt

Historie

Freigegeben: am - von -
Version 1.0

Erstellt: am 20.01.2009 von Christian Ulrich

Einleitung

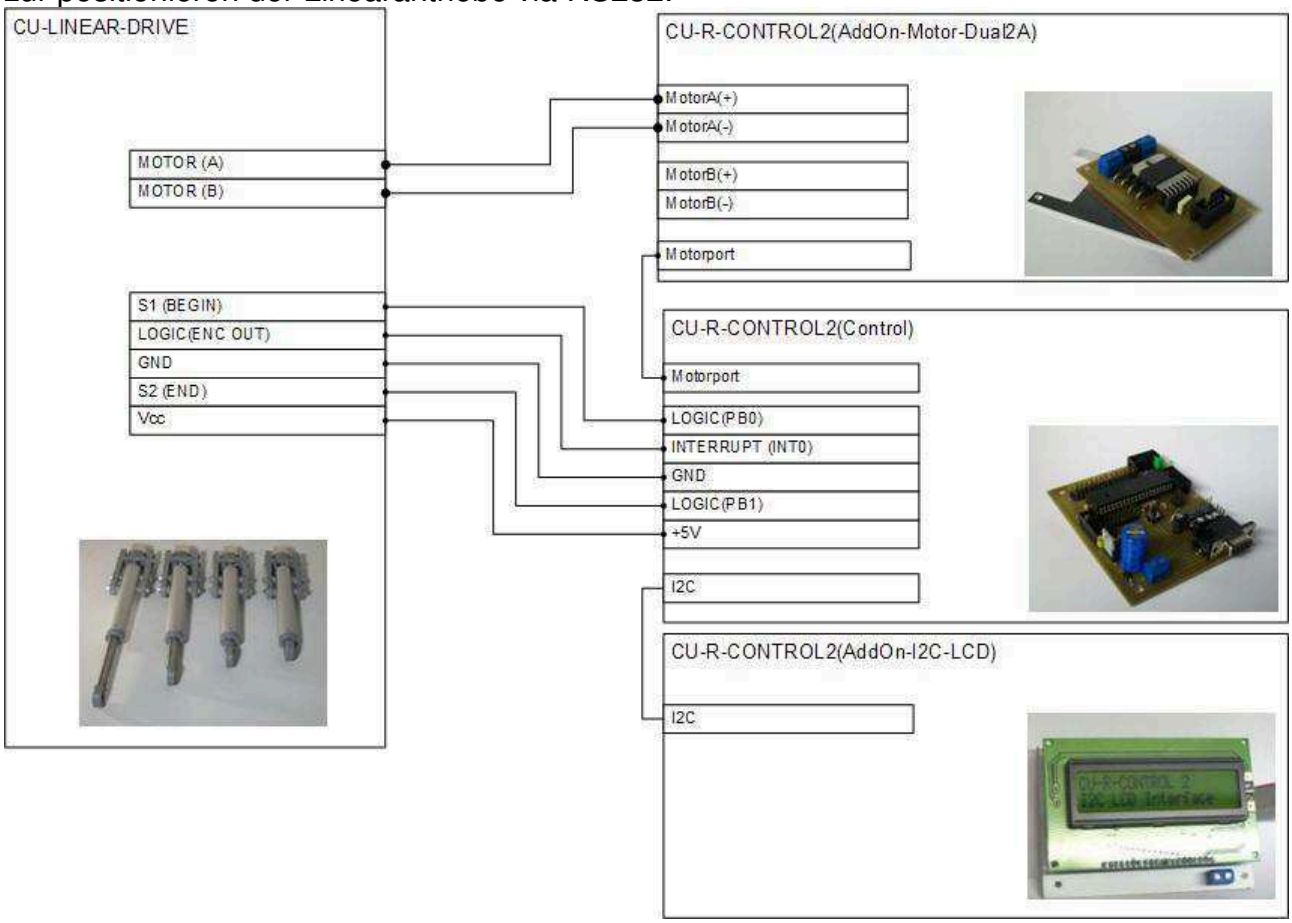
Diese Dokumentation zum Linearantrieb, enthält Detailinformationen zur Software.

Mechanik

Die mechanischen Konstruktionszeichnungen nebst Schaltungs- und Softwarevorlagen zu CU-LINEAR-DRIVE sind bei http://www.ulrichc.de/ veröffentlicht.
Zu finden bei den Projektaufzeichnungen zu CU-LINEAR-DRIVE.

Elektronik

Die Steuerungselektronik des Linearantriebs ist im Rahmen der Testanwendung am Antrieb und dessen Sensoren Angeschlossen. Basierend auf der Mikrokontroller-Steuerung CU-R-CONTROL² dient diese Steuerung zur Auswertung der Sensoren sowie zur positionieren der Linearantriebe via RS232.



Die Hardware der Steuerungen, wurden aus Platinen des CU-R-CONTROL² (Projekt bei http://www.ulrichc.de/) konfektioniert.
Verwendete Platinen:

- 1x Control (=Hauptsteuerung ATMega16)
- 1x AddOn Dualmotor-2A (= Motorsteuerung)
- 1x AddOn I2C-LCD (= LCD-Display)

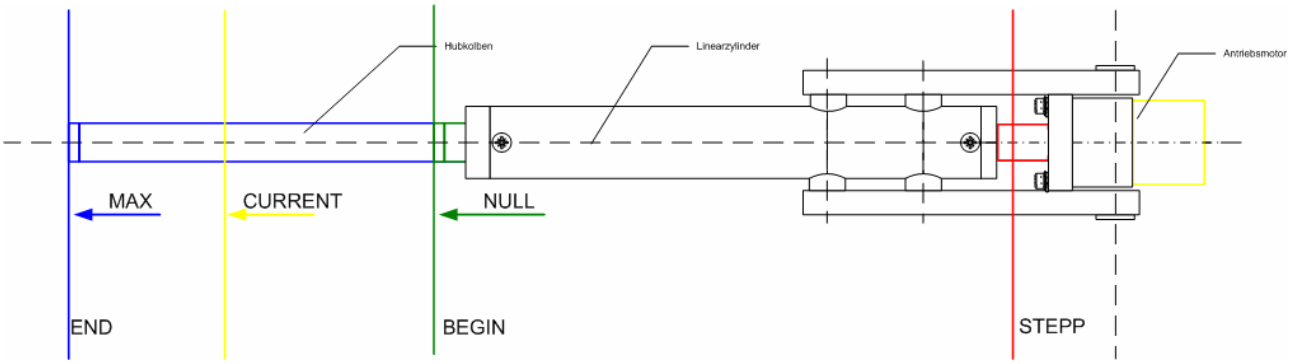
Diese Ausstattung ermöglicht eine elektronische Steuerung von zwei Linearantrieben. Softwaretechnisch ist der Betrieb von zwei Antrieben gleichzeitig weitgehend Vorbereitet. Um der Testanwendung etwas an Komplexität zu nehmen, wurde nur ein Antrieb vollständig implementiert.

Software

Die Softwarebeschreibung des CU-LINEAR-DRIVE beschränkt sich auf die Mikrokontroller-Schaltung zur direkten Ansteuerung des Antriebs. Die innerhalb des Projekts beschriebene Anwendung, wurde aus der Steuerung CU-R-CONTROL² konfektioniert, und als Testanwendung eingesetzt.

Softwaredefinition

Die Softwaredefinition beschreibt Eigenschaften am Linearantrieb, die von der Software ausgewertet oder ggf. gesteuert werden können.



Die Definitionen beziehen sich direkt auf die Elektronik bzw. auch auf die Sensoren und den Antriebsmotor selbst.

Im Detail

END
Endschalter bzw. Endpunkt (MAX) des Kolbens

BEGIN
Endschalter bzw. Beginpunkt (NULL) des Kolbens

STEPP
Encoder bzw. Schritt zur Errechnung der aktuellen (CURRENT) Kolbenposition.

Antriebsmotor
Der Antriebsmotor wird zur direkten Positionierung des Linearantriebs verwendet. Weg, Richtung und Geschwindigkeit werden dabei über Motorsteuerung gesteuert.

Linearzylinder und Hubkolben
Diese technischen Vorrichtungen haben softwaretechnisch betrachtet keine Relevanz. Alle Auswertungen über Weg und Position des Kolbens werden elektronisch erfasst.

Features

Die Software zum Linearantrieb wurde für keine spezielle Anwendung erstellt. Jedoch wurden in Hinblick auf eine zweckgemäße Verwendung verschiedene Schlüsselfunktionen softwaretechnisch Vorbereitet.

Absolute Position

Im Betrieb mit Drehgebern, kann der Antrieb absolut positioniert werden. Einfache Steuerbefehle ermöglichen das Anfahren und positionieren den Kolben in einer Auflösung von einem Millimeter. Auch genauere Positionierungen können einfach umgesetzt werden. Abhängig von der Drehgeberauflösung sind positionieren im 0,01 mm Bereich keine Unwägbarkeit für diese Antriebsform.

Drehgeberauswertung

Die Mechanisch und Elektronisch eingeplanten Drehgeber werden mit der Software ausgewertet. Aber auch ohne Drehgeber arbeitet die Software mit Funktionen zur Positionierung.

Sicherheitsfunktionen

Speziell bei der Kalibrierung werden auch die Endschalter getestet. Sodass sich evtl. Ausfällte nicht umgehend auf die Mechanik der Linearantriebe auswirken.

Positionsspeicher

Die Software ist vorbereitet für Positionsspeicher. Mit dem Positionsspeicher können bis zu fünf Positionen direkt oder mittels Konfiguration gespeichert werden. Die gespeicherten Positionen können zur Absoluten Positionierung der Kolben verwendet werden.

Kommunikation

Die Software ist für serielle RS232 Kommunikation sowie I2C-Bus Kommunikation ausgelegt.

Ansteuerung

Die Ansteuerung beschreibt Wege zur Kommunikation mit der Elektronik von CU-LINEAR-DRIVE. Damit verbunden, wird auch die Software der Steuerung in Form von Befehlsbeschreibungen beschrieben.

Steuer-Protokoll

Das Steuer-Protokoll beschreibt das Protokoll zum ansteuern des Kamerastativs mittels RS232-Schnittstelle.

Die RS232-Kommunikation basiert auf einer einfachen Protokollkommunikation nach DIN 66258 und wird via. Kabelverbindung hergestellt. Je nach Wahl der Hardware kann diese Ansteuerung auch über eine drahtlose Funkübertragung arbeiten.

RS232 Verbindungseinstellungen

Folgende Einstellungen zur RS232-Verbindung wurden Softwaretechnisch eingestellt. PC-Steuerung sowie Stativ-Steuerung müssen gleichermaßen eingestellt sein. Bedingt durch die Gegebenheiten der Hardware bzw. der Steuerelektronik, wurden folgende Parameter eingestellt.

Übertragungsrate: 19200 Baud (Bit/Sekunde)
Datenbits: 8
Parität: keine
Stopbit: 1

Die Befehle werden wie üblich als Byte als **HEX***adecimal* oder auch **DEC***imal* übermittel. Alle Befehle werden in dieser Beschreibung, der einfacheren Darstellung wegen, als **DEC***imal*-Werte behandelt.

I2C Verbindungseinstellungen

Alternativ zur seriellen Schnittstelle, kann zur Ansteuerung auch der I2C-Bus verwendet werden. Die Befehle werden analog zur seriellen Schnittstelle behandelt.
I2C-Adresse: 128 (bzw. &H80)
(Die Adresse kann mit den Einstellungen unter Kapitel „Konfigurations-Funktionen“ geändert werden.)

Funktionsbeschreibung

Die Beschreibung der Funktionen, umschreibt Funktionen der Steuerung im Detail. Lediglich parametrisierbare bzw. auch Befehle mit gesteuerter Ausgabe sind in diesem Kapitel beschrieben. Weitere Befehle der Steuerung sind ergänzend im Kapitel „Kurzreferenz“ beschrieben.
Ausgenommen besonderer Protokoll- und Debug-Funktionen, können alle Funktionen, wahlweise über Rs232 oder auch I2C verwendet werden. Folglich ist die Beschreibung der Funktionen in beiden Fällen gleich.

Hinweis: Grundlegendes zur Interpretation und Ansteuerung der Steuerung befindet sich im Kapitel „Mikrokontroller-Kommunikation“.

Befehle der Dokumentation interpretieren

Die Funktionen werden nach dem jeweiligen ersten Byte identifiziert und mit den darauf folgenden Bytes Parametrisiert.

Beispiel:
Befehlsbeschreibung: Zeige Kontrollereinstellungen.
Befehl in Kurzreferenz der Dokumentation: 128:PrintSettings(NR)
Gesendeter Befehl (RS232 oder I2C): 128 1
Erstes Byte = Name der Funktion (Bsp. 128 = PrintSettings)
Zweites Byte = Erster Parameter (Bsp. 1 Nummer der Einstellung.

Beispielhafte Rückgabe gemäß Funktionsbeschreibung
1 Byte *Nummer des Befehls*
2 Byte *Detaillierung des Befehls (Erster Parameter)*
3 Byte *Kontroller Id*
4 Byte *Sekunden Wartezeit bis zum umschalten in den StandBy-Modus*
5 Byte *Die I2C Adresse des Kontrollers*
Wahlweise empfangen über RS232: Zeichenkette „128,1,222,255,128“
Wahlweise empfangen über I2C: *Byte1*=128, *Byte2*=1, *Byte3*=222, *Byte4*=255, *Byte5*=128

Konfigurations-Funktionen

Die Konfigurationsfunktionen, umschreiben alle Voreinstellungen zur Steuerung.

Die Voreinstellungen, sollten nur nach einem Reset der Steuerung bzw. frisch bei in Betriebnahme getroffen werden. Nach der Konfiguration, sollte wieder ein Reset erfolgen. Anstatt des Hardware-Resets (Taster auf Steuerung), kann auch die Direktfunktion der Software für General-Reset ausgeführt werden.

Hinweis: Alle Konfigurationseinstellungen, bleiben auch nach einem Reset oder nach Abschalten der Steuerung erhalten.

Getätigte Einstellungen werden erst nach einem Reload der Einstellungen oder auch (Software-)Reset wirksam!

128:PrintSettings(NR)

Sendet die aktuellen Steuerungseinstellungen zurück.

Parameter

Byte(Nr)	Typ	Wertebereich	Beschreibung
1	Byte = 128	Festwert	
2	Byte	0-N	Definiert die Art der Steuerungseinstellungen 0=Alle Einstellungen 1=Controller-Einstellungen 2=Motor-Einstellungen
3			
4			
5			

Rückgabe(n)

Byte(Nr)	Typ	Wertebereich	Beschreibung
1	Byte = 128	Festwert	(Identisch mit Befehlsnummer)
2	Byte	0-N	Art der Steuerungseinstellungen (Siehe Parameter)

Byte2=0 Alle Einstellung

Alle Einstellungen unterhalb in Reihenfolge 1-N.

Byte2=1 Controller-Einstellung

Byte(Nr)	Typ	Wertebereich	Beschreibung
3	Byte	1-254	ID der Steuerung
4	Byte	1-254 (255=aus)	Maximale Wartezeit bis zum StandBy-Modus.
5	Byte	2-254	I2C-Adresse

Steuer-Funktionen

Die Steuerfunktionen umschreiben die eigentlichen Funktionen zur Robotersteuerung. Diese elementaren Befehle führen umgehend zu einer Reaktion der Steuerung.

Stell-Funktionen

Die Stellfunktionen, umschreiben alle Einstellungen während dem Betrieb der Steuerung.

Die Einstellungen, wirken sich je nach Betriebszustand der Steuerung, ähnlich wie die Steuerfunktionen (oberhalb), direkt auf die Steuerung aus. Können aber auch zur Voreinstellung genutzt werden.

Hinweis:
Die Einstellungen, bleiben nach einem Reset oder nach Abschalten der Steuerung *nicht* erhalten.

Task-Funktionen

Die Task-Funktionen beschreiben übergeordnete Steuer-Funktion. Es handelt sich hierbei um komplexe Befehle, die in Form von Aufgaben formuliert werden.

Die Funktionen, setzen eine vollständige Konfiguration voraus.

Funktionsübersicht/Kurzreferenz

Die Funktionsübersicht zu den Funktionen der Steuerung beschreibt alle Befehle in Kürze.

Steuerfunktionen

COMMAND	Name	Beschreibung
41	PrintControllerInfo()	Sendet Informationen zur Steuerung und Software.
42	PrintControllerStats()	Sendet aktuelle Statis zur Steuerung und Software.
43	CalibrateLinear()	Startet Initialisierung und Grundkalibrierung des Linearantriebs. Die Kalibrierung wird auch automatisch beim ersten Start durchgeführt.
44	Reboot()	Neustart, NotStop, NotHalt
48	ShootIn()	Schaltet die Programmfunktionen aus.
50	WakeUp()	
51	StandBy()	
80	LinearPosToBegin()	Fährt der Linearantrieb in die Beginposition.
81	LinearPosMoveIn(STEPS)	Ändert die aktuelle Position einwärts in definierter weite. Parameter STEPS bestimmt die Anzahl der Schritte.
82	LinearPosToSaved(NR)	Fährt den Linearantrieb in gespeicherte Position (1-5).
83	LinearPosMoveOut(STEPS)	Ändert die aktuelle Position auswärts in definierter weite. Parameter STEPS bestimmt die Anzahl der Schritte.
84	LinearPosToEnd()	Fährt den Linearantrieb in die Endposition.
85	LinearPosToBegin()	Fährt den Linearantrieb in die Beginposition.
86	LinearPosMoveRight(STEPS)	Ändert die aktuelle Position einwärts in definierter weite. Parameter STEPS bestimmt die Anzahl der Schritte.
87	LinearPosToSaved (NR)	Fährt den Linearantrieb in gespeicherte Position (1-5).
88	LinearPosMoveLeft(STEPS)	Ändert die aktuelle Position auswärts in definierter weite. Parameter STEPS bestimmt die Anzahl der Schritte.
89	LinearPosToEnd()	Fährt der Linearantrieb in die Endposition.
90	SaveCurrentAsLinearPos(NR)	Speichert aktuelle Position in NR.
91	SaveCurrentAsLinearPos(NR)	Speichert aktuelle Position in NR.
98	LinearMoveToAbsPos(NR)	Fährt gespeicherte Position NR an.
99	LinearMoveToAbsPos(NR)	Fährt gespeicherte Position NR an.

Ausgegraute Funktionen fungieren als Platzhalter. Die Funktionen können im Zuge einer Programmierung des zweiten Linearantriebs implementiert werden.

Grundfunktionen

COMMAND	Name	Beschreibung
45	StartMorors(MOTOR_BOTH)	Startet beide Motoren des Linearantriebs.
46	StopMorors(MOTOR_BOTH)	Stoppt beide Motoren des Linearantriebs.
47	BrakeMotors (MOTOR)	Bremst Motoren des Linearantriebs
61	SetMotorSpeed(MOTOR,SPEED)	Setzt aktuelle Motorgeschwindigkeit.
62	SetMotorDirection(MOTOR,DIRECTION)	Setzt Drehrichtung der Motoren.

Konfigurationsfunktionen

CMD	Funktion (Parameter)	Format	Beschreibung
100	SetControllerId(ID)	1-255	Legt die ID des Controllers fest
101	SetStandByOrAliveTimer(SEC)	1-254 (255=aus)	Setzt die Zeit bis zum automatischen StandBy.
102	MotorDefaultSpeed1(SPEED)	1-255	Legt die Minimalgeschwindigkeit des Motors für den ersten Linearantrieb fest.
103	MotorDefaultSpeed2(SPEED)	1-255	Legt die Minimalgeschwindigkeit des Motors für den zweiten Linearantrieb fest.
105	MotorSpeedMin(SPEED)	1-255	Legt die Minimalgeschwindigkeit der Motoren fest.
106	MotorSpeedMax(SPEED)	1-255	Legt die Maximalgeschwindigkeit der Motoren fest.
109	SetMotorBreakPower(POWER)	1-254 (255=aus)	Bremskraft der automatischen Bremse.
110	SaveLinearPos (VAL1,VAL2)	1-255	Speichert eine Position an Speicherplatz 0. VAL1 und VAL2 werden im Sinne von Byte1 und Byte2 für -Word- Datentypen angewendet. (Die Funktion wird intern bei der Kalibrierung für die Mittelposition genutzt.)
111	SaveLinearPos (VAL1,VAL2)	1-255	Speichert eine Position an Speicherplatz 1. VAL1 und VAL2 werden im Sinne von Byte1 und Byte2 für -Word- Datentypen angewendet.
112	SaveLinearPos (VAL1,VAL2)	1-255	Speichert eine Position an Speicherplatz 2. VAL1 und VAL2 werden im Sinne von Byte1 und Byte2 für -Word- Datentypen angewendet.
113	SaveLinearPos (VAL1,VAL2)	1-255	Speichert eine Position an Speicherplatz 3. VAL1 und VAL2 werden im Sinne von Byte1 und Byte2 für -Word- Datentypen angewendet.
114	SaveLinearPos (VAL1,VAL2)	1-255	Speichert eine Position an Speicherplatz 4. VAL1 und VAL2 werden im Sinne von Byte1 und Byte2 für -Word- Datentypen angewendet.
115	SaveLinearPos (VAL1,VAL2)	1-255	Speichert eine Position an Speicherplatz 5. VAL1 und VAL2 werden im Sinne von Byte1 und Byte2 für -Word- Datentypen angewendet.
117	SaveCurrentPos (VAL1,VAL2)	1-255	Manipuliert die aktuelle Position des Linearantriebs. VAL1 und VAL2 werden im Sinne von Byte1 und Byte2 für -Word- Datentypen angewendet.
118	SaveMaxPos (VAL1,VAL2)	1-255	Manipuliert die maximale Position. Also die Hublänge des Linearantriebs. Dieser Parameter wird beim Kalibrieren automatisch gesetzt. VAL1 und VAL2 werden im Sinne von Byte1 und Byte2 für -Word- Datentypen angewendet.
119	SetLinearStepWidth (WIDTH)	1-255	Setzt die Stepp-Weite in zehntel Millimeter.

			Entspricht der Hubweite pro Encoder-Schritt. Formel: Gewindesteigung / Encoder-Schritte pro Umdrehung => In zehntel Millimeter.
125	SetI2CAdress(ADRESS)	0-254 (255=aus)	Setzt die I2C-Adresse der Steuerung
126	ReLoadSettings()		Ladet (aktuell) geänderte Einstellungen
127	ResetSettings()		Stellt die Werkseinstellungen wieder her.
128	PrintSettings(NR)	1-n (0=alle)	Gibt alle aktuellen Einstellungen nach NR aus.

Ausgegraute Funktionen fungieren als Platzhalter. Die Funktionen können im Zuge einer Programmierung des zweiten Linearantriebs implementiert werden.

Mikrokontroller-Kommunikation

Das Ansteuern der Steuerung, geschieht ob über RS232 oder auch I2C nach einem Kommunikationsprotokoll.
Unterhalb wurden die grundlegendsten und zugleich wichtigsten Punkte zur Kommunikation via. Datenprotokoll beschrieben.

Vorweg, die Datenkommunikation verbirgt keine Besonderheit und folgt den allgemeinen Grundregeln zur Datenkommunikation für Mikrokontroller. Fortgeschrittene Nutzer, können folglich getrost manches überfliegen.

Datentypen

Die Datenkommunikation vom und zum Kontroller geschieht in Byte(s).
Dieser Datentyp ist Standard für die RS232- und I2C-Kommunikation der Steuerung.

Grundlegend hat ein Byte insgesamt acht Datenbits (0 oder 1) und einen Wertebereich von 0 bis 255.
Je nach Anwendungsfall wird dieser Datentyp verschieden verwendet. Unterhalb wurden die jeweiligen Anwendungsfälle beschrieben.

Byte als Zahlenwert

Das Grundlegende Byte wird als Zahlenwert verwendet. Dann als „Byte“ bezeichnet, werden damit Werte von 0 bis 255 versendet oder auch empfangen.

Byte als ASCII / Zeichenketten interpretieren

Zeichenketten (wie „Hallo Welt!) können in einzelnen Bytes in Standard ASCII interpretiert werden.
Folglich besteht eine Zeichenkette aus mehreren Bytes, die seriell (nacheinander) zur Zeichenkette zusammengesetzt werden können. Siehe auch Kapitel „ASCII-Tabelle“ im Anschluss an diesem Dokument.

Byte als Word / 2 Byte Datentypen interpretieren

Datentypen die aus zwei Bytes bestehen, werden aus Low- und High-Byte interpretiert. Das HighByte entspricht dabei einem Multiplikator. Das LowByte enthält die Restsumme der Multiplikation.

Der Multiplikationswert des ersten Bytes (HighByte) ist 254.
Das zweite Byte (LowByte) wird addiert.

Beispiel:
<Byte1> <Byte2>
<001> <002>
Formel:
<1*254> <+2>
Ergebnis:
254 + 2 = 256

Als Besonderheit, wir 255 als NULL interpretiert.

Beispiel:

<Byte1>	<Byte2>	
<255>	<012>	= 12
<001>	<255>	= 254

Daten Senden

Daten bzw. auch Befehle, werden Byteweise versandt. Dem Protokoll entsprechend, werden jeweils Datenblöcke von 1 bis n Bytes versendet. Zwischen dem Versenden mehrerer Befehlsdatenblöcke, müssen 10 Millisekunden verstreichen. Andernfalls werden die Daten ggf. nicht als Einzelbefehl interpretiert.

Daten empfangen

Nach jedem senden von Daten, werden wiederum Daten empfangen. Je nach Einstellungen an der Steuerung, können auch fortwährend Daten von der Steuerung empfangen werden.

Der Datenempfang unterscheidet sich je nach Kommunikationsweg.
Über RS232, können alle Daten als ASCII-Zeichenketten empfangen werden.
Als Komma getrennte Zeichenkette, bestückt mit den jeweiligen Daten, können die empfangenen Daten zwischen den Kommas ausgelesen werden. Die einzelnen Bytes, werden dabei ebenfalls als ASCII-Zeichenkette übertragen.

Über I2C, können die Daten Byteweise empfangen werden. Die Daten folgen in derselben Reihenfolge wie in der Dokumentation beschrieben.

Sonstiges

ASCII-Tabelle

Die ASCII Tabelle enthält alle Zeichen entsprechend dem Bytewert bzw. auch Hex.
Dem entsprechend, kann Beispielsweise die ASCII-Zeichenkette „Hallo Welt“ in Bytes wie folgt verstanden werden.
72 97 108 108 111 32 87 101 108 116
Gemäß dem ASCII-Standard werden die Bytewerte unterhalb als DEC*imal-Werte* gelistet.

Ctrl	Dec	Hex	Char	Code
^@	0	00		NUL
^A	1	01	☺	SOH
^B	2	02	☻	SIX
^C	3	03	♥	EIX
^D	4	04	♦	EOI
^E	5	05	♣	ENQ
^F	6	06	♠	ACK
^G	7	07	•	BEL
^H	8	08	◘	BS
^I	9	09	○	HI
^J	10	0A	◙	LF
^K	11	0B	♂	VI
^L	12	0C	♀	FF
^M	13	0D	♪	CR
^N	14	0E	♫	SO
^O	15	0F	☼	SI
^P	16	10	►	SLE
^Q	17	11	◄	CS1
^R	18	12	↕	DC2
^S	19	13	‼	DC3
^I	20	14	¶	DC4
^U	21	15	§	NAK
^V	22	16	▬	SYN
^W	23	17	↨	EIB
^X	24	18	↑	CAN
^Y	25	19	↓	EM
^Z	26	1A	→	SIB
^[	27	1B	←	ESC
^\	28	1C	∟	FS
^]	29	1D	↔	GS
^^	30	1E	▲	RS
^_	31	1F	▼	US

Dec	Hex	Char
32	20	sp
33	21	!
34	22	"
35	23	#
36	24	$
37	25	%
38	26	&
39	27	'
40	28	(
41	29	)
42	2A	*
43	2B	+
44	2C	,
45	2D	-
46	2E	.
47	2F	/
48	30	0
49	31	1
50	32	2
51	33	3
52	34	4
53	35	5
54	36	6
55	37	7
56	38	8
57	39	9
58	3A	:
59	3B	;
60	3C	<
61	3D	=
62	3E	>
63	3F	?

Dec	Hex	Char
64	40	@
65	41	A
66	42	B
67	43	C
68	44	D
69	45	E
70	46	F
71	47	G
72	48	H
73	49	I
74	4A	J
75	4B	K
76	4C	L
77	4D	M
78	4E	N
79	4F	O
80	50	P
81	51	Q
82	52	R
83	53	S
84	54	T
85	55	U
86	56	V
87	57	W
88	58	X
89	59	Y
90	5A	Z
91	5B	[
92	5C	\
93	5D	]
94	5E	^
95	5F	_

Dec	Hex	Char
96	60	`
97	61	a
98	62	b
99	63	c
100	64	d
101	65	e
102	66	f
103	67	g
104	68	h
105	69	i
106	6A	j
107	6B	k
108	6C	l
109	6D	m
110	6E	n
111	6F	o
112	70	p
113	71	q
114	72	r
115	73	s
116	74	t
117	75	u
118	76	v
119	77	w
120	78	x
121	79	y
122	7A	z
123	7B	{
124	7C	|
125	7D	}
126	7E	~
127	7F	⌂

Dec	Hex	Char
128	80	Ç
129	81	ü
130	82	é
131	83	â
132	84	ä
133	85	à
134	86	å
135	87	ç
136	88	ê
137	89	ë
138	8A	è
139	8B	ï
140	8C	î
141	8D	ì
142	8E	Ä
143	8F	Å
144	90	É
145	91	æ
146	92	Æ
147	93	ô
148	94	ö
149	95	ò
150	96	û
151	97	ù
152	98	ÿ
153	99	Ö
154	9A	Ü
155	9B	¢
156	9C	£
157	9D	¥
158	9E	₧
159	9F	ƒ

Dec	Hex	Char
160	A0	á
161	A1	í
162	A2	ó
163	A3	ú
164	A4	ñ
165	A5	Ñ
166	A6	ª
167	A7	º
168	A8	¿
169	A9	⌐
170	AA	¬
171	AB	½
172	AC	¼
173	AD	¡
174	AE	«
175	AF	»
176	B0	▒
177	B1	▒
178	B2	▒
179	B3	│
180	B4	┤
181	B5	╡
182	B6	╢
183	B7	╖
184	B8	╕
185	B9	╣
186	BA	║
187	BB	╗
188	BC	╝
189	BD	╜
190	BE	╛
191	BF	┐

Dec	Hex	Char
192	C0	└
193	C1	┴
194	C2	┬
195	C3	├
196	C4	─
197	C5	┼
198	C6	╞
199	C7	╟
200	C8	╚
201	C9	╔
202	CA	╩
203	CB	╦
204	CC	╠
205	CD	═
206	CE	╬
207	CF	╧
208	D0	╨
209	D1	╤
210	D2	╥
211	D3	╙
212	D4	╘
213	D5	╒
214	D6	╓
215	D7	╫
216	D8	╪
217	D9	┘
218	DA	┌
219	DB	█
220	DC	▄
221	DD	▌
222	DE	▐
223	DF	▀

Dec	Hex	Char
224	E0	α
225	E1	ß
226	E2	Γ
227	E3	π
228	E4	Σ
229	E5	σ
230	E6	µ
231	E7	τ
232	E8	Φ
233	E9	Θ
234	EA	Ω
235	EB	δ
236	EC	∞
237	ED	ø
238	EE	ε
239	EF	∩
240	F0	≡
241	F1	±
242	F2	≥
243	F3	≤
244	F4	⌠
245	F5	⌡
246	F6	÷
247	F7	≈
248	F8	°
249	F9	∙
250	FA	·
251	FB	√
252	FC	n
253	FD	2
254	FE	■
255	FF	

UART- Kurzreferenz

Tabelle der gängigsten Steuerzeichen für UART-Kommunikation.

DEC	COMMAND		DEC	COMMAND
0	<NUL>		16	<DLE>
1	<SOH>		17	<DC1>
2	<STX>		18	<DC2>
3	<ETX>		19	<DC3>
4	<EOT>		20	<DC4>
5	<ENQ>		21	<NAK>
6	<ACK>		23	<ETB>
8	<BS>		27	<ESC>
9	<HT>		28	<FS>
10	<LF>		29	<GS>
11	<VT>		30	<RS>
12	<FF>			
13	<CR>			
14	<SO>			
15	<SI>			

Dieses Dokument gehört zum Projekt CU-LINEAR-DRIVE von UlrichC.DE. Weitere Dokumente sowie Konstruktionsunterlagen und Bilder zum Projekt sind auf der Internetpräsenz

http://www.ulrichc.de/

zum Download bereitgestellt.