
CU-WEBBOT

Aufklärungsroboter für den Innenbereich mit Websteuerung.
Dokumentation zur Software



Autor: Christian Ulrich

Datum: 12.01.2008

Version: 1.00

Inhalt

Historie	3
Einleitung	4
Software	5
PC-Software	6
Webserver	6
CGI-Websteuerung	6
Steuerfunktionen der Oberfläche	6
Robotersoftware	7
Kernfunktionen	7
StandBy-Timer	7
Alive-Timer	7
Lifetime-Signal	7
Steuer-Protokoll	8
Verbindungseinstellungen	8
I2C Verbindungseinstellungen	8
Funktionsbeschreibung	9
Befehle der Dokumentation interpretieren	9
Konfigurations-Funktionen	10
128:PrintSettings(NR)	10
Steuer-Funktionen	11
42:PrintControllerStats(NR)	11
Stell-Funktionen	12
Task-Funktionen	12
Webbot-Funktionen	12
Funktionsübersicht/Kurzreferenz	13
Steuer-Funktionen	13
Stell-Funktionen	13
Konfigurations-Funktionen	14
Webbot-Funktionen	15
Task-Funktionen	15
Mikrokontroller-Kommunikation	16
Datentypen	16
Daten Senden	17
Daten empfangen	17
Sonstiges	18
ASCII-Tabelle	18
UART- Kurzreferenz	19

Historie

Erstellt: am 12.01.2008 von Christian Ulrich

Einleitung

Die Dokumentation zur Software, enthält Detailinformationen zu den Implementierungen des CU-WEBBOTS.

Software

Die Software des CU-WEBBOTs teilt sich in zwei Softwareteile.

Zum einen in eine *Robotersteuerung* die am Roboter selbst installiert ist, zum anderen eine *PC-Anbindung* auf einem PC installiert die Steuerung oder auch Anbindung zum Internet ermöglicht.

Beide Softwareteile arbeitet mit einer RS232-Kommunikation zusammen und bilden ein Softwaresystem.

PC-Software

Die Software der PC-Anbindung basiert auf einem Standard Windows-Betriebssystem

Den Kern der eigentlichen Websteuerung bildet ein Webserver, der eine Steuerungsoberfläche zum Internet hin aufbaut, und im Hintergrund den Roboter steuert.

Webserver

Als Webserver, wurde ein Apache-Webserver mit Erweiterungen für Perl installiert.

CGI-Websteuerung

Den Kern der auf dem Webserver installierten Webanwendung, bildet die Websteuerung CU-WWWGUI. Diese Websteuerung wurde als Modul des Websystems bei <http://www.ulrichc.de> entwickelt und als Projekt CU-WWW-GUI projektiert und im Detail beschrieben.

Steuerfunktionen der Oberfläche

Im Zusammenhang mit CU-WEBBOT wurde die Websteuerung CU-WWWGUI mit folgenden Steuerfunktionen realisiert.

Funktionen zur Motorsteuerung (Roboterbewegungen)

(klein) kurve ~20°

(klein) drehung ~60°

(klein) strecke 5 cm

(mittel) kurve ~40°

(mittel) drehung ~90°

(mittel) strecke 10 cm

(groß) kurve ~60°

(groß) drehung ~120°

(groß) strecke 25 cm

Funktionen der Kamera

Kamera Ein/Aus Schalten

Kamera nach oben bewegen

Kamera in Mittelposition

Kamera nach unten bewegen

Sonstige Funktionen

Licht Ein/Aus Schalten

Weitere Funktionen der Steuerung, wie Konfiguration und Trimmung wurden als Direktbefehle über die Weboberfläche von CU-CPORT vorgesehen.

Robotersoftware

Die Robotersoftware basiert auf einem Mikrokontroller-Programm, das auf dem Roboter installiert, alle Steuerungsaufgaben koordiniert und ausführt.

Kernfunktionen

Die Kernfunktionen beschreiben grundlegende Funktionen und Eigenschaften bzw. auch Features der Robotersoftware des CU-WEBBOTS.

StandBy-Timer

Die Robotersteuerung verfügt über einen so genannten StandBy-Timer, der den Roboter zeitweise in den Stromsparmmodus schalten kann. Je nach Konfiguration, werden so Teile der Peripherie des Roboters bzw. die größten Stromverbraucher wie Motorendstufen, Lampen und Kameras abgeschaltet.

Die Steuerung schaltet sich automatisch nach zeitweise ausbleiben von Steuerbefehlen in diesen StandBy-Betrieb. Das Erwecken aus diesem Betriebsmodus geschieht ebenfalls automatisch nach Empfang neuer Steuerbefehle.

Alive-Timer

Die eigentliche Funktionalität des Alive-Timer wurde bereits als StandBy-Timer behandelt und physikalisch nicht gesondert implementiert. Der Alive-Timer beschreibt lediglich eine erweiterte Schutzfunktion des StandBy-Timers. Der Schutz sieht vor, dass die Robotersteuerung bei defekten automatisch in einen sicheren Modus bzw. in den StandBy wechselt.

Wenn beispielsweise der Steuerrechner ausfällt, Kabel gezogen werden oder Funkübertragungen nicht gelingen, oder allgemein Fehler in der Datenübertragung vorliegen, soll die Steuerung nicht Steuerlos weiterarbeiten.

Um sicher zu stellen, dass keine Fehlfunktion bei der Übertragung von Steuerdaten vorliegt, werden von der Steuerung in regelmäßigen Abständen Signale oder auch Steuerbefehle erwartet. Nach Ausbleiben der Signale, schaltet die Steuerung in den beschriebenen StandBy.

Lifetime-Signal

Die Steuerung des Roboters sendet in regelmäßigen Abständen ein so genanntes Lifetime-Signal. Der zeitliche Intervall des Signals, wurde auf unter eine Sekunde voreingestellt. Das Signal kann von der Hauptsteuerung ähnl. wie der Alive-Timer genutzt werden, um die Funktionalität der Robotersteuerung sicher zu stellen.

Dieses Signal sendet die intern festgelegte ID der Steuerung. Das Signal wird auch zur Identifikation der Kommunikationsschnittstelle auf der Gegenseite verwendet.

Steuer-Protokoll

Das Steuer-Protokoll beschreibt das Protokoll zwischen Roboter und dem Steuerungs-PC über RS232 Schnittstelle.

Die Kommunikation basiert auf einer einfachen Protokollkommunikation nach DIN 66258 und wird je nach Hardware über Funk oder auch Kabelverbindung hergestellt.

Verbindungseinstellungen

Die Einstellungen zur RS232-Verbindung wurden auch Softwaretechnisch berücksichtigt. PC-Steuerung sowie Robotersteuerung wurden gleichermaßen eingestellt. Bedingt durch die Gegebenheiten der Hardware bzw. der Steuerelektronik des Roboters, wurde folgende Parameter eingestellt.

Übertragungsrate: 19200 Baud (Bit/Sekunde)

Datenbits: 8

Parität: keine

Stopbit: 1

Die Befehle werden wie üblich als Byte als HEX oder auch DEC übermittelt. Alle Befehle werden in dieser Beschreibung, der Einfachheit halber, als DECimal-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: 124 (bzw. &H7C)

(Die Adresse kann mit den Einstellungen unter Kapitel „Konfigurations-Funktionen“ geändert werden.)

Funktionsbeschreibung

Die Beschreibung der Funktionen, umschreibt die wichtigsten Funktionen der Steuerung im Detail.

Vorwiegend 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 Controllers*

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! (Ein)

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.

42:PrintControllerStats(NR)

Sendet aktuelle Werte zur Steuerung und Software.

Parameter

Byte(Nr)	Typ	Wertebereich	Beschreibung
1	Byte = 42	Festwert	
2	Byte	0-N	Definiert die Art der Steuerungseinstellungen 0=Alle Werte 1-N 1=Controller-Werte 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-Werte

Byte(Nr)	Typ	Wertebereich	Beschreibung
3	Byte	1-254	Aktuellen Modus der Steuerung. 0= CTRL_MODE_STANDBY = 0 1= CTRL_MODE_DEFAULT 2= CTRL_MODE_PWRSTAT 3= CTRL_MODE_REBOOT 4= CTRL_MODE_AUTO_STANDBY 5= CTRL_MODE_SHOOTDOWN 99 = CTRL_MODE_UNDEFINED

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 Encoder und eine vollständige Konfiguration voraus.

Webbot-Funktionen

Die Webbot-Funktionen beschreiben einfache Sonder-Steuerfunktion zur Ansteuerung via Websteuerung.

In diesen Befehlen wurden so genannte Befehlsfolgen (mehrere Befehle), für ein einfaches Ansteuern, für jeweils einen erforderlichen Befehlsaufruf implementiert.

Funktionsübersicht/Kurzreferenz

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

Steuer-Funktionen

COMMAND	Name	Beschreibung
41	PrintControllerInfo()	Sendet Informationen zur Steuerung und Software.
42	PrintControllerStats(NR)	Sendet aktuelle Statis zur Steuerung und Software.
43		
44	Reboot()	Neustart, NotStop, NotHalt
45	StartMotors()	Starten der Motoren (gemäß Konfiguration)
46	StopMotors()	Stopp der Motoren
47	BrakeMotors()	Bremst die Motoren
48	ShootDown()	Bereitet abschalten der Steuerung vor.
49		
50	WakeUp()	Schaltet alle Nebensteuerungen Ein
51	StandBy()	Schaltet alle Nebensteuerungen Aus
52	SetCamPowerOn()	Schaltet Kamera ein
53	SetCamPowerOff()	Schaltet Kamera aus
54	IncreaseCamPos()	Bewegt Kamera nach oben (Gemäss vorkonfigurierte Schrittweite)
55	DecreaseCamPos()	Bewegt Kamera nach unten (Gemäss vorkonfigurierte Schrittweite)
56	SetCamPosMiddel()	Setzt Kamera wieder auf den vordefinierten Mittelpunkt zurück
57		
58		
59	SetLightPowerOn()	Schaltet das Licht ein
60	SetLightPowerOff()	Schaltet das Licht aus

Stell-Funktionen

COMMAND	Funktion (Parameter)	Beschreibung
61	SetMotorSpeed(MOTOR, SPEED)	Konfiguriert die Motorgeschwindigkeit oder bzw. auch die Bremskraft (EMK)
62	SetMotorDirection(MOTOR1, MOTOR2)	Konfiguriert die Motordrehrichtung oder auch Bremse

Konfigurations-Funktionen

CMD	Funktion (Parameter)	Format	Beschreibung
100	SetControllerId(ID)	1-255	Legt die ID des Controllers fest.
101	SetAliveOrStandByTimer(SEC)	1-254 (255=aus)	Setzt die Zeit bis zum automatischen StandBy.
105	SetMotorMinSpeed(SPEED)	1-255	Legt die Minimalgeschwindigkeit des Motors fest.
106	SetMotorMaxSpeed(SPEED)	1-255	Legt die Maximalgeschwindigkeit des Motors fest.
109	SetMotorBreakPower(POWER)	1-254 (255=aus)	Bremskraft/Haltestrom der automatischen Bremse.
110	SetCamRange(RANGE)	1-255	Legt den möglichen Schwenkradius der Kamera fest.
111	SetCamPosStep(STEPPS)	1-50	Legt die Auflösung der Teilschritte beim Auf- und Abfahren der Kamera fest.
112	SetCamPosMiddel(POS)	1-RANGE	Legt den Mittelpunkt der Kamera fest.
115	SetWheelDiameter(RADIUS)	1 - 254 mm	Konfiguriert den Radius des Raddurchmessers in Millimeter
116	SetWheelMiddelSpacing(SPACE)	1 - 254 mm	Konfiguriert den mittleren Radabstand (Abstand / 2) in Millimeter
117	SetWheelEncoderSteps(STEPS)	1-254 (255=aus)	Konfiguriert die Teilschritte des Drehgebers
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.

Webbot-Funktionen

CMD	Funktion (Parameter)	Format	Beschreibung
220	RobotMoveForward(TIMEOUT,SPEED)	TIMEOUT= 0-255 sec SPEED=0-255	
221	RobotMoveBackward(TIMEOUT,SPEED)	TIMEOUT= 0-255 sec SPEED=0-255	
222	RobotCurveLeft(TIMEOUT,SPEED)	TIMEOUT= 0-255 sec SPEED=0-255	
223	RobotMoveRight(TIMEOUT,SPEED)	TIMEOUT= 0-255 sec SPEED=0-255	
224	RobotSpinLeft(TIMEOUT,SPEED)	TIMEOUT= 0-255 sec SPEED=0-255	
225	RobotSpinRight(TIMEOUT,SPEED)	TIMEOUT= 0-255 sec SPEED=0-255	
226	RobotStop()		Stopp bzw. auch Nothalt

Task-Funktionen

CMD	Funktion (Parameter)	Format	Beschreibung
139	SetTaskTimeOut(SECONDS)	0-255 sec	Setzt die maximale Laufzeit eines Tasks in Sekunden.
140	DriveTaskStart(SPEED,OPT)	Speed=0-255 OPT=0-255	Startet den gesetzten Task.
141	DriveTaskAbort()	1-254 (255=aus)	Abbruch des aktuell gesetzten Tasks. Bremse.
142	PrintDriveTaskInfo()	1-254 (255=aus)	Informationen zum aktuellen Task.

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 *DECimal-Werte* gelistet.

Ctrl	Dec	Hex	Char	Code	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char			
^@	0	00		NUL	32	20	sp	64	40	@	96	60	`	128	80	Ç	160	A0	á	192	C0	Ā	224	E0	α
^A	1	01	☐	SCH	33	21	!	65	41	A	97	61	a	129	81	ü	161	A1	í	193	C1	Ā	225	E1	β
^B	2	02	☐	SIX	34	22	"	66	42	B	98	62	b	130	82	é	162	A2	ó	194	C2	Ā	226	E2	Γ
^C	3	03	♥	EIX	35	23	#	67	43	C	99	63	c	131	83	â	163	A3	ú	195	C3	Ā	227	E3	Π
^D	4	04	♣	EOI	36	24	\$	68	44	D	100	64	d	132	84	à	164	A4	û	196	C4	Ā	228	E4	Σ
^E	5	05	♠	ENQ	37	25	%	69	45	E	101	65	e	133	85	ä	165	A5	ü	197	C5	Ā	229	E5	ϰ
^F	6	06	♣	ACK	38	26	&	70	46	F	102	66	f	134	86	ã	166	A6	ë	198	C6	Ā	230	E6	ρ
^G	7	07	•	BEL	39	27	'	71	47	G	103	67	g	135	87	ä	167	A7	ë	199	C7	Ā	231	E7	Υ
^H	8	08	BS		40	28	(72	48	H	104	68	h	136	88	å	168	A8	ë	200	C8	Ā	232	E8	ϰ
^I	9	09	o	HI	41	29)	73	49	I	105	69	i	137	89	æ	169	A9	ë	201	C9	Ā	233	E9	ϰ
^J	10	0A	☐	LF	42	2A	*	74	4A	J	106	6A	j	138	8A	è	170	AA	ë	202	CA	Ā	234	EA	Ω
^K	11	0B	♀	VI	43	2B	+	75	4B	K	107	6B	k	139	8B	é	171	AB	ë	203	CB	Ā	235	EB	ø
^L	12	0C	♀	FF	44	2C	,	76	4C	L	108	6C	l	140	8C	ê	172	AC	ë	204	CC	Ā	236	EC	ε
^M	13	0D	♯	CR	45	2D	-	77	4D	M	109	6D	m	141	8D	ë	173	AD	ë	205	CD	Ā	237	ED	ϰ
^N	14	0E	♯	SO	46	2E	.	78	4E	N	110	6E	n	142	8E	î	174	AE	ë	206	CE	Ā	238	EE	€
^O	15	0F	♯	SI	47	2F	/	79	4F	O	111	6F	o	143	8F	ï	175	AF	ë	207	CF	Ā	239	EF	∩
^P	16	10	♯	SLE	48	30	0	80	50	P	112	70	p	144	90	ê	176	B0	ë	208	D0	Ā	240	FO	∩
^Q	17	11	♯	CS1	49	31	1	81	51	Q	113	71	q	145	91	ë	177	B1	ë	209	D1	Ā	241	F1	∩
^R	18	12	♯	DC2	50	32	2	82	52	R	114	72	r	146	92	ê	178	B2	ë	210	D2	Ā	242	F2	∩
^S	19	13	!!	DC3	51	33	3	83	53	S	115	73	s	147	93	ë	179	B3	ë	211	D3	Ā	243	F3	∩
^T	20	14	!!	DC4	52	34	4	84	54	T	116	74	t	148	94	ë	180	B4	ë	212	D4	Ā	244	F4	∩
^U	21	15	☐	NAK	53	35	5	85	55	U	117	75	u	149	95	ò	181	B5	ë	213	D5	Ā	245	F5	∩
^V	22	16	☐	SYN	54	36	6	86	56	V	118	76	v	150	96	ù	182	B6	ë	214	D6	Ā	246	F6	∩
^W	23	17	☐	EIB	55	37	7	87	57	W	119	77	w	151	97	ú	183	B7	ë	215	D7	Ā	247	F7	∩
^X	24	18	☐	CAN	56	38	8	88	58	X	120	78	x	152	98	û	184	B8	ë	216	D8	Ā	248	F8	∩
^Y	25	19	☐	EM	57	39	9	89	59	Y	121	79	y	153	99	ü	185	B9	ë	217	D9	Ā	249	F9	∩
^Z	26	1A	→	SIB	58	3A	:	90	5A	Z	122	7A	z	154	9A	Û	186	BA	ë	218	DA	Ā	250	FA	∩
^[27	1B	+	ESC	59	3B	;	91	5B	[123	7B	{	155	9B	ü	187	BB	ë	219	DB	Ā	251	FB	∩
^\	28	1C	L	FS	60	3C	<	92	5C	\	124	7C	}	156	9C	ÿ	188	BC	ë	220	DC	Ā	252	FC	∩
^]	29	1D	+	G\$	61	3D	=	93	5D]	125	7D	~	157	9D	ÿ	189	BD	ë	221	DD	Ā	253	FD	∩
^^	30	1E	▲	RS	62	3E	>	94	5E	^	126	7E	ˆ	158	9E	ÿ	190	BE	ë	222	DE	Ā	254	FE	∩
^_	31	1F	▼	US	63	3F	?	95	5F	_	127	7F	ˆ	159	9F	ÿ	191	BF	ë	223	DF	Ā	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-WEBBOT](#) von UlrichC.DE. Weitere Dokumente sowie Konstruktionsunterlagen und Bilder zum Projekt sind auf der Internetpräsenz <http://www.ulrichc.de/> zum Download bereitgestellt.