

Specification for Communication Protocol

Contents

1	Packet Structur	1
1.1	Command Structur	1
1.2	Response Structur	2
2	Command List	2
2.1	Generic Commands	2
2.2	Specific Commands	2
2.3	InitMove (MOTOR, DIR)	3
2.4	MoveTo (MOTOR, DIR, ABS_POS, SPEED, ACC, DEC)	3
2.5	WaitMoved (MOTOR, TIMEOUT)	3
2.6	IsReady (MOTOR)	4
2.7	Move (MOTOR, DIR, SPEED, ACC, DEC)	4
2.8	StopMove (MOTOR, IS_HARDSTOP)	5
2.9	GetAbsPos (MOTOR)	5
2.10	SetPin (PIN_NR, IS_HIGH)	5
2.11	GetPin (PIN_NR)	5
2.12	ConfigPin (PIN_NR, IS_OUTPUT)	6
2.13	SaveHome (MOTOR)	6
2.14	GoHome (MOTOR)	6
2.15	SaveWayPoint (MOTOR)	7
2.16	MoveToWayPoint (MOTOR, WAY_POINT, SPEED, ACC, DEC)	7
3	Errorcodes	7

Definitions

FALSE := 0x00

TRUE := 0x01..0xFF

1 Packet Structur

1.1 Command Structur

Payload (0..10 Bytes)	Padding (10 Bytes - Length)	Checksum (1 Byte)
--------------------------	--------------------------------	----------------------

Payload: Befehlsfolge

Padding: Füllbytes

Checksum: Prüfsumme $CRC8(Length + Payload)$ (optional)

1.2 Response Structur

Ack (1 Byte)	Payload (0..10 Bytes)	Padding (10 Bytes - Length)	Checksum (1 Byte)
-----------------	--------------------------	--------------------------------	----------------------

Ack: Acknowledge (TRUE wenn Befehl gültig)

Payload: Antwort

Padding: Füllbytes

Checksum: Prüfsumme $CRC8(Length + Payload)$ (optional)

2 Command List

2.1 Generic Commands

Command mnemonic	Command Bytes (Payload)								
	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
InitMove	0x00	Motor	Dir						
MoveTo	0x01	Motor	Dir	AbsPos (HB)	AbsPos	AbsPos (LB)	Speed	Acc	Dec
WaitMoved	0x02	Motor	Timeout (HB)	Timeout (LB)					
IsReady	0x03	Motor							
Move	0x04	Motor	Dir	Speed	Acc	Dec			
StopMove	0x05	Motor	IsHard Stop						
GetAbsPos	0x06	Motor							
SetPin	0x07	PinNr	IsHigh						
GetPin	0x08	PinNr							
ConfigPin	0x09	PinNr	IsOutput						

2.2 Specific Commands

Command mnemonic	Command Bytes (Payload)								
	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
SaveHome	0x0A	Motor							
GoHome	0x0B	Motor							
Save WayPoint	0x0C	Motor							
MoveTo WayPoint	0x0D	Motor	WayPoint	Speed	Acc	Dec			

2.3 InitMove (MOTOR, DIR)

Command code: **0x00**

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0								
Byte 1	MOTOR							
Byte 2	DIR							

Description

Der gewählte MOTOR fährt zur Initialisierung in der angegebenen Richtung (DIR) bis an eine Endposition (zB. mechanischer Anschlag). Sobald diese Position erreicht ist, bleibt der Motor stehen und ist bereit.

2.4 MoveTo (MOTOR, DIR, ABS_POS, SPEED, ACC, DEC)

Command code: **0x01**

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0								
Byte 1	MOTOR							
Byte 2	DIR							
Byte 3	ABS_POS (High Byte)							
Byte 4	ABS_POS (Middle Byte)							
Byte 5	ABS_POS (Low Byte)							
Byte 6	SPEED							
Byte 7	ACC							
Byte 8	DEC							

Description

Der gewählte MOTOR fährt zu einer bestimmten absoluten Position (ABS_POS). Dabei kann dem Motor eine Richtung (DIR) sowie eine Geschwindigkeit (SPEED in $\frac{Steps}{s}$) vorgegeben werden. Des Weiteren kann eine Beschleunigungsrampe (ACC in $\frac{Steps}{s^2}$) und eine Verzögerungsrampe vorgegeben werden (DEC in $\frac{Steps}{s^2}$).

Falls Standardwerte für Geschwindigkeit, Beschleunigung und Verzögerung verwendet werden sollen, können die jeweiligen Parameter auf 0x00 gesetzt werden.

2.5 WaitMoved (MOTOR, TIMEOUT)

Command code: **0x02**

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0								
Byte 1	MOTOR							
Byte 2	TIMEOUT (High Byte)							
Byte 3	TIMEOUT (Low Byte)							

Description

Bei diesem Befehl wird die Antwort erst zurück gesendet, wenn der spezifizierte MOTOR still steht. Zusätzlich muss ein TIMEOUT (in *ms*) mitgegeben werden, nach welchem die Antwort auf jeden Fall gesendet wird. Falls das Timeout abgelaufen ist, wird ein Errorcode mitgesendet.

2.6 IsReady (MOTOR)

Command code: **0x03**

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0								
Byte 1	MOTOR							

Description

Sendet mit der Antwort, ob der spezifizierte MOTOR bereit ist. Die Antwort ist TRUE, wenn der Motor keinen aktuellen Befehl am ausführen ist, andernfalls ist sie FALSE.

2.7 Move (MOTOR, DIR, SPEED, ACC, DEC)

Command code: **0x04**

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0								
Byte 1	MOTOR							
Byte 2	DIR							
Byte 3	SPEED							
Byte 4	ACC							
Byte 5	DEC							

Description

Der gewählt MOTOR fährt mit der angegebenen Geschwindigkeit (SPEED in $\frac{Steps}{s}$) in die Richtung DIR. Es kann eine Beschleunigungsrampe (ACC in $\frac{Steps}{s^2}$) und eine Verzögerungsrampe vorgegeben werden (DEC in $\frac{Steps}{s^2}$).

Falls Standardwerte für Geschwindigkeit, Beschleunigung und Verzögerung verwendet werden sollen, können die jeweiligen Parameter auf 0x00 gesetzt werden.

2.8 StopMove (MOTOR, IS_HARDSTOP)

Command code: **0x05**

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0								
Byte 1	MOTOR							
Byte 2	IS_HARDSTOP							

Description

Stoppt den MOTOR. Mit dem zusätzlichen Parameter IS_HARDSTOP kann angegeben werden, ob der Motor auf der Stelle stoppt (IS_HARDSTOP = TRUE) oder mit der programmierten Verzögerungsrampe abbremst (IS_HARDSTOP = FALSE).

2.9 GetAbsPos (MOTOR)

Command code: **0x06**

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0								
Byte 1	MOTOR							

Description

Fragt die aktuelle absolute Position des MOTORS ab. Die absolute Position wird in der Antwort mit 3 Bytes beschrieben, wobei das höchste Byte als erstes geschickt wird.

2.10 SetPin (PIN_NR,IS_HIGH)

Command code: **0x07**

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0								
Byte 1	PIN_NR							
Byte 1	IS_HIGH							

Description

Dieser Befehl setzt einen IO-Pin (PIN_NR) des μ C auf den entsprechenden Pegel (IS_HIGH). Der Pin wird auf Vcc gesetzt, wenn der Parameter IS_HIGH = TRUE ist, andernfalls auf 0V. Der IO-Pin muss als Output definiert sein. Wenn der gewählte Pin nicht gesetzt werden kann, enthält die Antwort einen Errorcode.

2.11 GetPin (PIN_NR)

Command code: **0x08**

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0								
Byte 1	PIN_NR							

Description

Mit diesem Befehl kann der Wert eines IO-Pins (PIN_NR) des μ C abgefragt werden. In der Antwort ist der Zustand des Pins direkt enthalten (Low = FALSE, High = 0x01).

Wenn der gewählte Pin nicht gelesen werden kann, enthält die Antwort einen Errorcode.

2.12 ConfigPin (PIN_NR, IS_OUTPUT)

Command code: **0x09**

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0								
Byte 1	PIN_NR							
Byte	IS_OUTPUT							

Description

Der gewählte IO-Pin (PIN_NR) des μ Cs wird anhand des zweiten Parameters (IS_OUTPUT) konfiguriert. Der Pin wird zum Output, wenn IS_OUTPUT = TRUE ist, andernfalls zum Input.

2.13 SaveHome (MOTOR)

Command code: **0x0A**

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0								
Byte 1	MOTOR							

Description

Speichert die aktuelle Position als Home-Position für den gewählten MOTOR.

2.14 GoHome (MOTOR)

Command code: **0x0B**

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0								
Byte 1	MOTOR							

Description

Der entsprechende MOTOR fährt auf die Home-Position

2.15 SaveWayPoint (MOTOR)

Command code: **0x0C**

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0								
Byte 1	MOTOR							

Description

Speichert die aktuelle Position als Wegpunkt für den gewählten MOTOR. In der Antwort ist die Nummer des gespeicherten Wegpunktes enthalten (max. 255).

2.16 MoveToWayPoint (MOTOR, WAY_POINT, SPEED, ACC, DEC)

Command code: **0x0D**

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0								
Byte 1	MOTOR							
Byte 2	WAY_POINT							
Byte 3	SPEED							
Byte 4	ACC							
Byte 5	DEC							

Description

Der MOTOR fährt zum angegebenen Wegpunkt (WAY_POINT). Dabei kann die Geschwindigkeit ((SPEED in $\frac{Steps}{s}$)), sowie eine Beschleunigungsrampe (ACC in $\frac{Steps}{s^2}$) und Verzögerungsrampe (DEC in $\frac{Steps}{s^2}$) vorgegeben werden.

Falls Standardwerte für Geschwindigkeit, Beschleunigung und Verzögerung verwendet werden sollen, können die jeweiligen Parameter auf 0x00 gesetzt werden.

3 Errorcodes