Specification for Communication Protocol

Contents

1 Packet Structur **2**
- 1.1 Command Structur . . . 2
- 1.2 Response Structur . . . 2

2 Command List **2**
- 2.1 Generic Commands . . . 2
- 2.2 Specific Commands . . . 3
- 2.3 InitMove (MOTOR, DIR) . . . 3
- 2.4 MoveTo (MOTOR, DIR, ABS_POS, SPEED, ACC, DEC) . . . 3
- 2.5 WaitMoved (MOTOR, TIMEOUT) . . . 4
- 2.6 IsReady (MOTOR) . . . 5
- 2.7 Move (MOTOR, DIR, SPEED, ACC, DEC) . . . 5
- 2.8 StopMove (MOTOR, IS_HARDSTOP) . . . 6
- 2.9 GetAbsPos (MOTOR) . . . 6
- 2.10 SetPin (PIN_NR,IS_HIGH) . . . 6
- 2.11 GetPin (PIN_NR) . . . 6
- 2.12 ConfigPin (PIN_NR, IS_OUTPUT) . . . 7
- 2.13 DcMove (DIR, TIME, GO_HiZ) . . . 7
- 2.14 SaveHome (MOTOR) . . . 7
- 2.15 GoHome (MOTOR) . . . 8
- 2.16 SaveWayPoint (MOTOR) . . . 8
- 2.17 MoveToWayPoint (MOTOR, WAY_POINT, SPEED, ACC, DEC) . . . 8

3 Errorcodes **9**
- 3.1 Full Buffer . . . 9
- 3.2 Invalide Command . . . 9
- 3.3 Invalide Address . . . 9
- 3.4 Motor not ready . . . 9
- 3.5 Motor Error . . . 10
- 3.6 Full Way Point Buffer . . . 10
- 3.7 Invalid Way Point . . . 10

Definitions

FALSE := 0x00
TRUE := 0x01..0xFF

1 Packet Structur

1.1 Command Structur

Payload (0..9 Bytes)	Padding (9 Bytes - Length)	*Checksum* *(1 Byte)*

Payload: Befehlsfolge
Padding: Füllbytes
Checksum: *Prüfsumme CRC8(Payload) (optional)*

1.2 Response Structur

Ack (1 Byte)	Payload (0..3 Bytes)	Padding (3 Bytes - Length)	*Checksum* *(1 Byte)*

Ack: Acknowledge (TRUE wenn Befehl gültig)
Payload: Antwort
Padding: Füllbytes
Checksum: *Prüfsumme CRC8(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	Speed	Acc	Dec			
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						
DcMove	0x0E	Dir	Time (HB)	Time (LB)	goHiz				

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							
Byte 3	SPEED							
Byte 4	ACC							
Byte 5	DEC							

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, setzt die absolute Position auf null und ist bereit. Die angegebene Geschwindigkeit kann folgendermassen umgerechnet werden:

$$[step/s] = \frac{SPEED \cdot 2^{-16}}{250ns}$$

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 gewählt MOTOR fährt zu eine bestimmten absoluten Position (ABS_POS). Dabei kann dem Motor eine Richtung (DIR) sowie eine Geschwindigkeit (SPEED) vorgegeben werden. Des weiteren kann eine Beschleunigungsrampe (ACC) und eine Verzögerungsrampe vorgegeben werden (DEC).
Falls Standardwerte für Geschwindigkeit, Beschleunigung und Verzögerung verwendet werden sollen, können die jeweiligen Parameter auf 0x00 gesetzt werden.
Umrechnung der Geschwindigkeit:

$$[step/s] = \frac{SPEED \cdot 2^{-16}}{250ns}$$

Umrechnung der Beschleunigung:

$$\left[step/s^2\right] = \frac{ACC \cdot 2^{-36}}{(250ns)^2}$$
$$\left[step/s^2\right] = \frac{DEC \cdot 2^{-36}}{(250ns)^2}$$

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 die Richtung DIR. Es kann eine Beschleunigungsrampe (ACC) und eine Verzögerungsrampe vorgegeben werden (DEC).
Falls Standardwerte für Geschwindigkeit, Beschleunigung und Verzögerung verwendet werden sollen, können die jeweiligen Parameter auf 0x00 gesetzt werden.
Umrechnung der Geschwindigkeit:

$$[step/s] = \frac{SPEED \cdot 2^{-16}}{250ns}$$

Umrechnung der Beschleunigung:

$$\left[step/s^2\right] = \frac{ACC \cdot 2^{-36}}{(250ns)^2}$$

$$\left[step/s^2\right] = \frac{DEC \cdot 2^{-36}}{(250ns)^2}$$

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 Befehlt setzt einen IO-Pin (PIN_NR) des μC 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 μC 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 2	IS_OUTPUT							

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

2.13 DcMove (DIR, TIME, GO_HiZ)

Command code: **0x0E**

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0								
Byte 1	DIR							
Byte 2	TIME (HB)							
Byte 3	TIME (LB)							
Byte 4	GO_HiZ							

Description
Der DC-Motor fährt die angegebene Zeit (TIME in ms) in die Richtung DIR. Wenn GO_HiZ auf TRUE ist, wird der Motor nach dem Fahren so geschaltet, dass dieser ein Haltemoment hat.

2.14 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.

2.15 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

2.16 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).

2.17 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), sowie eine Beschleunigungsrampe (ACC) und Verzögerungsrampe (DEC) vorgegeben werden.
Falls Standardwerte für Geschwindigkeit, Beschleunigung und Verzögerung verwendet werden sollen, können die jeweiligen Parameter auf 0x00 gesetzt werden.
Umrechnung der Geschwindigkeit:

$$[step/s] = \frac{SPEED \cdot 2^{-16}}{250ns}$$

Umrechnung der Beschleunigung:

$$\left[step/s^2\right] = \frac{ACC \cdot 2^{-36}}{(250ns)^2}$$
$$\left[step/s^2\right] = \frac{DEC \cdot 2^{-36}}{(250ns)^2}$$

3 Errorcodes

Errorcodes werden mit der Response Structure gesendet. Das Acknowledge Byte ist auf den Wert 0x00 und das erste Byte beinhaltet den Error-Code.

Ack	Byte 0	Byte 1 - Byte 2	*Checksum*
0x00	Error code	Padding (0x00)	

3.1 Full Buffer

Error code: **0xE0**

Description
Der Prozessor hat noch nicht alle Befehle abgearbeitet. Der Befehls-Buffer ist voll und es können keine Weiteren Befehle gespeichert werden.

3.2 Invalide Command

Error code: **0xE1**

Description
Es wurde ein ungültiger Befehl geschickt, welche nicht verarbeitet werden kann.

3.3 Invalide Address

Error code: **0xE2**

Description
Die Motorennummer ist ungültig.

3.4 Motor not ready

Error code: **0xE3**

Description
Der Motor ist noch nicht bereit (Befindet sich noch in Bewegung).

3.5 Motor Error

Error code: **0xE4**

Description
Der Motor hat einen Error.

3.6 Full Way Point Buffer

Error code: **0xE5**

Description
Die maximale Anzahl Way Points für diesen Motor sind erreicht.

3.7 Invalid Way Point

Error code: **0xE6**

Description
Der angegebene Way Point existiert nicht.