ROT2PROG Protocol Documentation

Table of contents

- ROTn_CMD_CALIBRATION
- ROTn_CMD_CFG_GET
- ROTn_CMD_CFG_SET
- ROTn_CMD_CLEAN
- ROTn_CMD_GET_ANGLES
- ROTn_CMD_GET_ANGLES_100
- ROTn_CMD_GET_MEM
- ROTn_CMD_GET_OUTS
- ROTn_CMD_GET_SOFT_HARD
- ROTn_CMD_MOTORS
- ROTn_CMD_POWER
- ROTn_CMD_RESTART_DEVICE
- ROTn_CMD_SET_ANGLES
- ROTn_CMD_SET_ANGLESX
- ROTn_CMD_SET_ANGLES_100
- ROTn_CMD_SET_MEM_FINISH
- ROTn_CMD_SET_MEM_INIT
- ROTn_CMD_SET_MEM_PACKET
- ROTn_CMD_SET_OUTS
- ROTn_CMD_SET_SOFT_HARD
- ROTn_CMD_STOP

ROTn_CMD_CALIBRATION

Description

Set rotor position(s) without moving.

```
angleToSend = IntToString(360 * divisor + (desiredAngle * divisor))
```

Command value:

0xf9

Data structure

Data structure name

prtROTxRequest

Data structure details

Request Magic Number

```
rotxMagicStart 0x57
```

Request Magic Number End

```
rotxMagicEnd 0x20
```

Bytes												
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**	**0x0c**
magic: prtROTxMagicNumber	payload: prtROTxRequestPayloadRot2setMotor										command: prtROTxCommand	magicEnd: prtROTxMagicNumber
uint8_t	angle1: char[4]				angle1Divisor: uint8_t	angle2: char[4]				angle2Divisor: uint8_t	uint8_t	uint8_t
Example												
0x57	0x33	0x36	0x31	0x30	0x0a	0x33	0x35	0x39	0x30	0x0a	0xf9	0x20
'W'	'3'	'6'	'1'	'0'	\xa	'3'	'5'	'9'	'0'	\xa	xf9	' '
Set Motor 1 to 1 degree and Motor 2 to -1 degree												

Response data structure

Data structure name

prtROTxResponseAngle

Description

Get rotor position.

```
angle = StrToInt(receivedAngle) * divisor - 360 * divisor
```

Data structure details

Response Magic Number

```
rotxMagicStart 0x57
```

Response Magic Number End

```
rotxMagicEnd 0x20
```

Bytes											
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**
magic: prtROTxMagicNumber	angle1: array				angle1Divisor: c_ubyte	angle2: array				angle2Divisor: c_ubyte	magicEnd: prtROTxMagicNumber
(prtROTxMagicNumber/uint8_t)	char[4]				uint8_t	char[4]				uint8_t	(prtROTxMagicNumber/uint8_t)
Example											
0x57	0x33	0x38	0x32	0x33	0x0a	0x33	0x36	0x30	0x35	0x0a	0x20
'W'	'3'	'8'	'2'	'3'	\xa	'3'	'6'	'0'	'5'	\xa	' '
Motor 1 angle: 22.3 degree, Motor 2 angle: 0.5 degree											

ROTn_CMD_CFG_GET

Description

Get settings value. isSketchValue determines, if response provides value for current running settings or for prepared settings to be applied in bulk.
Passing fieldId = 0 in response returns maximum fiedlId in fieldValue.f_word

Command value:

0xef

Data structure

Data structure name

prtROTxRequest

Data structure details

Request Magic Number

```
rotxMagicStart 0x57
```

Request Magic Number End

```
rotxMagicEnd 0x20
```

Bytes												
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**	**0x0c**
magic: prtROTxMagicNumber	payload: prtROTxRequestPayloadRot2cmdGetValue										command: prtROTxCommand	magicEnd: prtROTxMagicNumber
uint8_t	fieldId: uint16_t		isSketchValue: uint8_t	0	0	0	0	0	0	0	uint8_t	uint8_t
Example												
0x57	0x02	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0xef	0x20
'W'	\x2	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\xef	' '
Get value for fieldId = 2 (motor_0_min_angle) for running settings (isSketchValue=0)												

Response data structure

Data structure name

prtROTxResponseGetValue

Description

Data structure details

Response Magic Number

```
rotxMagicStartResponseCfgGetValue 0x59
```

Bytes																				
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**	**0x0c**	**0x0d**	**0x0e**	**0x0f**	**0x10**	**0x11**	**0x12**	**0x13**	**0x14**
magic: prtROTxMagicNumber	fieldId: c_ushort		fieldName: array																	
(prtROTxMagicNumber/uint8_t)	uint16_t		char[32]																	
Example																				
0x59	0x02	0x00	0x6d	0x6f	0x74	0x6f	0x72	0x5f	0x30	0x5f	0x6d	0x69	0x6e	0x5f	0x61	0x6e	0x67	0x6c	0x65	0x00
'Y'	\x2	\x0	'm'	'o'	't'	'o'	'r'	'_'	'0'	'_'	'm'	'i'	'n'	'_'	'a'	'n'	'g'	'l'	'e'	\x0

0x15	**0x16**	**0x17**	**0x18**	**0x19**	**0x1a**	**0x1b**	**0x1c**	**0x1d**	**0x1e**	**0x1f**	**0x20**	**0x21**	**0x22**	**0x23**	**0x24**	**0x25**	**0x26**	**0x27**	**0x28**	**0x29**	**0x2a**
														isSketchValue: c_ubyte	isReadonly: c_ubyte	fieldDataType: rotCfgDataType	fieldDataTypeSimple: rotCfgDataTypeSimple	fieldValue: cfgFieldValue			
														uint8_t	uint8_t	(rotCfgDataType/uint8_t)	(rotCfgDataTypeSimple/uint8_t)	None			
0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x04	0x03	0x00	0x00	0xb4	0x43
\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x4	\x3	\x0	\x0	\xb4	'C'

Response for fieldId = 2.
For meaning of fieldDataType and fieldDataTypeSimple see ProtocolRoTxCfg.h

ROTn_CMD_CFG_SET

Description

Set settings value. isSketchValue determines, if response provides value for current running settings or for prepared settings to be applied in bulk.
applySketchValues = 1 will write all stored with isSketchValue = 1 settings to device settings.

Command value:

0xf0

Data structure

Data structure name

prtROTxRequest

Data structure details

Request Magic Number

```
rotxMagicStart 0x57
```

Request Magic Number End

```
rotxMagicEnd 0x20
```

Bytes												
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**	**0x0c**
magic: prtROTxMagicNumber	payload: prtROTxRequestPayloadRot2cmdSetValue										command: prtROTxCommand	magicEnd: prtROTxMagicNumber
uint8_t	fieldId: uint16_t		applySketchValues: uint8_t	isSketchValue: uint8_t	fieldValue: None				0	0	uint8_t	uint8_t
Example												
0x57	0x02	0x00	0x00	0x00	0x00	0x00	0xb4	0x43	0x00	0x00	0xf0	0x20
'W'	\x2	\x0	\x0	\x0	\x0	\x0	\xb4	'C'	\x0	\x0	\xf0	' '
Set value of 360 (float) for field fieldId = 2 (motor_0_min_angle) for running settings (isSketchValue=0)												

Response data structure

Data structure name

prtROTxResponseGetValue

Description

Data structure details

Response Magic Number

```
rotxMagicStartResponseCfgGetValue 0x59
```

Bytes																																										
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**	**0x0c**	**0x0d**	**0x0e**	**0x0f**	**0x10**	**0x11**	**0x12**	**0x13**	**0x14**	**0x15**	**0x16**	**0x17**	**0x18**	**0x19**	**0x1a**	**0x1b**	**0x1c**	**0x1d**	**0x1e**	**0x1f**	**0x20**	**0x21**	**0x22**	**0x23**	**0x24**	**0x25**	**0x26**	**0x27**	**0x28**	**0x29**	**0x2a**
magic: prtROTxMagicNumber	fieldId: c_ushort		fieldName: array																																isSketchValue: c_ubyte	isReadonly: c_ubyte	fieldDataType: rotCfgDataType	fieldDataTypeSimple: rotCfgDataTypeSimple	fieldValue: cfgFieldValue			
(prtROTxMagicNumber/uint8_t)	uint16_t		char[32]																																uint8_t	uint8_t	(rotCfgDataType/uint8_t)	(rotCfgDataTypeSimple/uint8_t)	None			
Example																																										
0x59	0x02	0x00	0x6d	0x6f	0x74	0x6f	0x72	0x5f	0x30	0x5f	0x6d	0x69	0x6e	0x5f	0x61	0x6e	0x67	0x6c	0x65	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x04	0x03	0x00	0x00	0xb4	0x43
'Y'	\x2	\x0	'm'	'o'	't'	'o'	'r'	'_'	'0'	'_'	'm'	'i'	'n'	'_'	'a'	'n'	'g'	'l'	'e'	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x4	\x3	\x0	\x0	\xb4	'C'
Response for fieldId = 2. For meaning of fieldDataType and fieldDataTypeSimple see ProtocolRoTxCfg.h																																										

ROTn_CMD_CLEAN

Description

Set both motors position to 0, without moving.

Command value:

0xf8

Data structure

Data structure name

prtROTxRequest

Data structure details

Request Magic Number

```
rotxMagicStart 0x57
```

Request Magic Number End

```
rotxMagicEnd 0x20
```

Bytes												
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**	**0x0c**
magic: prtROTxMagicNumber	payload: prtROTxRequestPayloadIgnored										command: prtROTxCommand	magicEnd: prtROTxMagicNumber
uint8_t	ignored: uint8_t[10]										uint8_t	uint8_t
Example												
0x57	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0xf8	0x20
'W'	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\xf8	' '

Move Motor 1 to 5.54 degree and Motor 2 to 10.05 degree

Response data structure

Data structure name

prtROTxResponseAngle

Description

Get rotor position.

```
angle = StrToInt(receivedAngle) * divisor - 360 * divisor
```

Data structure details

Response Magic Number

```
rotxMagicStart 0x57
```

Response Magic Number End

```
rotxMagicEnd 0x20
```

Bytes											
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**
magic: prtROTxMagicNumber	angle1: array				angle1Divisor: c_ubyte	angle2: array				angle2Divisor: c_ubyte	magicEnd: prtROTxMagicNumber
(prtROTxMagicNumber/uint8_t)	char[4]				uint8_t	char[4]				uint8_t	(prtROTxMagicNumber/uint8_t)
Example											
0x57	0x33	0x38	0x32	0x33	0x0a	0x33	0x36	0x30	0x35	0x0a	0x20
'W'	'3'	'8'	'2'	'3'	\xa	'3'	'6'	'0'	'5'	\xa	' '

Motor 1 angle: 22.3 degree, Motor 2 angle: 0.5 degree

ROTn_CMD_GET_ANGLES

Description

Get current motors positions

Command value:

0x1f

Data structure

Data structure name

prtROTxRequest

Data structure details

Request Magic Number

```
rotxMagicStart 0x57
```

Request Magic Number End

```
rotxMagicEnd 0x20
```

Bytes												
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**	**0x0c**
magic: prtROTxMagicNumber	payload: prtROTxRequestPayloadIgnored										command: prtROTxCommand	magicEnd: prtROTxMagicNumber
uint8_t	ignored: uint8_t[10]										uint8_t	uint8_t
Example												
0x57	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x1f	0x20
'W'	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x1f	' '

Response data structure

Data structure name

prtROTxResponseAngle

Description

Get rotor position.

```
angle = StrToInt(receivedAngle) * divisor - 360 * divisor
```

Data structure details

Response Magic Number

```
rotxMagicStart 0x57
```

Response Magic Number End

```
rotxMagicEnd 0x20
```

Bytes											
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**
magic: prtROTxMagicNumber	angle1: array				angle1Divisor: c_ubyte	angle2: array				angle2Divisor: c_ubyte	magicEnd: prtROTxMagicNumber
(prtROTxMagicNumber/uint8_t)	char[4]				uint8_t	char[4]				uint8_t	(prtROTxMagicNumber/uint8_t)
Example											
0x57	0x33	0x38	0x32	0x33	0x0a	0x33	0x36	0x30	0x35	0x0a	0x20
'W'	'3'	'8'	'2'	'3'	\xa	'3'	'6'	'0'	'5'	\xa	' '
Motor 1 angle: 22.3 degree, Motor 2 angle: 0.5 degree											

ROTn_CMD_GET_ANGLES_100

Description

Get current motors positions. 0.01 resolution

Command value:

0x6f

Data structure

Data structure name

prtROTxRequest

Data structure details

Request Magic Number

```
rotxMagicStartResponseAngle100 0x58
```

Request Magic Number End

```
rotxMagicEnd 0x20
```

Bytes												
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**	**0x0c**
magic: prtROTxMagicNumber	payload: prtROTxRequestPayloadIgnored										command: prtROTxCommand	magicEnd: prtROTxMagicNumber
uint8_t	ignored: uint8_t[10]										uint8_t	uint8_t
Example												
0x57	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x6f	0x20
'W'	'x0	'x0	'x0	'x0	'x0	'x0	'x0	'x0	'x0	'x0	'o'	' '

Response data structure

Data structure name

prtROTxResponseAngle100

Description

Get rotor position.

```
angle = StrToInt(receivedAngle) * 100 - 360 * 100
```

Data structure details

Response Magic Number

```
rotxMagicStartResponseAngle100 0x58
```

Response Magic Number End

```
rotxMagicEnd 0x20
```

Bytes											
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**
magic: prtROTxMagicNumber	angle1: array					angle2: array					magicEnd: prtROTxMagicNumber
(prtROTxMagicNumber/uint8_t)	char[5]					char[5]					(prtROTxMagicNumber/uint8_t)
Example											
0x58	0x33	0x38	0x32	0x33	0x33	0x33	0x36	0x30	0x35	0x32	0x20
'X'	'3'	'8'	'2'	'3'	'3'	'3'	'6'	'0'	'5'	'2'	' '

Motor 1 angle: 22.33 degree, Motor 2 angle: 0.52 degree

ROTn_CMD_GET_MEM

Description

Get configuration memory data

Command value:

0x4f

Data structure

Data structure name

prtROTxRequest

Data structure details

Request Magic Number

```
rotxMagicStart 0x57
```

Request Magic Number End

```
rotxMagicEnd 0x20
```

Bytes												
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**	**0x0c**
magic: prtROTxMagicNumber	payload: prtROTxRequestPayloadRot2getMem										command: prtROTxCommand	magicEnd: prtROTxMagicNumber
uint8_t	pageId: uint8_t	0	0	0	0	0	0	0	0	0	uint8_t	uint8_t
Example												
0x57	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x4f	0x20
'W'	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	'O'	' '

Response data structure

ROTn_CMD_GET_OUTS

Description

Get SW01 outputs state

Command value:

0x3f

Data structure

Data structure name

prtROTxRequest

Data structure details

Request Magic Number

```
rotxMagicStart 0x57
```

Request Magic Number End

```
rotxMagicEnd 0x20
```

Bytes												
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**	**0x0c**
magic: prtROTxMagicNumber	payload: prtROTxRequestPayloadIgnored										command: prtROTxCommand	magicEnd: prtROTxMagicNumber
uint8_t	ignored: uint8_t[10]										uint8_t	uint8_t
Example												
0x57	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x3f	0x20
'W'	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x3f	' '

Response data structure

Data structure name

prtROTxResponseGetOuts

Description

SW01 outputs

Data structure details

Response Magic Number

```
rotxMagicStartResponseGetOuts 0x3f
```

Bytes	
0x00	**0x01**
magic: prtROTxMagicNumber	pins: c_ubyte
(prtROTxMagicNumber/uint8_t)	uint8_t
Example	
0x3f	0x23
\x3f	\x23
Outputs state: 100011	

ROTn_CMD_GET_SOFT_HARD

Description

Get START and STOP settings (IMMEDIATELY/SOFTLY)

Command value:

0xa1

Data structure

Data structure name

prtROTxRequest

Data structure details

Request Magic Number

```
rotxMagicStart 0x57
```

Request Magic Number End

```
rotxMagicEnd 0x20
```

Bytes												
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**	**0x0c**
magic: prtROTxMagicNumber	payload: prtROTxRequestPayloadIgnored										command: prtROTxCommand	magicEnd: prtROTxMagicNumber
uint8_t	ignored: uint8_t[10]										uint8_t	uint8_t
Example												
0x57	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0xa1	0x20
'W'	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\xa1	' '

Response data structure

Data structure name

prtROTxResponseGetSoftHard

Description

Start and Stop settings

Data structure details

Response Magic Number

```
rotxMagicStart 0x57
```

Response Magic Number End

```
rotxMagicEnd 0x20
```

Bytes											
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**
magic: prtROTxMagicNumber	unused: array				manualCtlStart: seMotorSoftStart	unused2: array				manualCtlStop: seMotorSoftStart	magicEnd: prtROTxMagicNumber
(prtROTxMagicNumber/uint8_t)	uint8_t[4]				(seMotorSoftStart/uint8_t)	uint8_t[4]				(seMotorSoftStart/uint8_t)	(prtROTxMagicNumber/uint8_t)
Example											
0x57	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x20
'W'	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	' '

START and STOP set to SOFTLY

seMotorSoftStart values:

- sstHard = 0
- sstSoft = 1

ROTn_CMD_MOTORS

Description

Command motors move (left/right etc.)

Command value:

0x14

Data structure

Data structure name

prtROTxRequest

Data structure details

Request Magic Number

```
rotxMagicStart 0x57
```

Request Magic Number End

```
rotxMagicEnd 0x20
```

Bytes												
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**	**0x0c**
magic: prtROTxMagicNumber	payload: prtROTxRequestPayloadRot2cmdMotors										command: prtROTxCommand	magicEnd: prtROTxMagicNumber
uint8_t	command: (prtROTxMoveMotorsCmd/uint8_t)	0	0	0	0	0	0	0	0	0	uint8_t	uint8_t
Example												
0x57	0x05	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x14	0x20
'W'	\x5	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x14	' '

Command Motor 1 to move Left, and Motor 2 to move Up
prtROTxRequestPayloadRot2cmdMotors values:

- mmCmdStop = 0x00
- mmCmdLeft = 0x01
- mmCmdRight = 0x02
- mmCmdUp = 0x04
- mmCmdDown = 0x08
- mmCmdLeftUp = 0x05
- mmCmdRightUp = 0x06
- mmCmdLeftDown = 0x09
- mmCmdRightDown = 0x0A

ROTn_CMD_POWER

Description

Set motors power (0-100%). (Applied immediately, without stoping current move)

Command value:

0xf7

Data structure

Data structure name

prtROTxRequest

Data structure details

Request Magic Number

```
rotxMagicStart 0x57
```

Request Magic Number End

```
rotxMagicEnd 0x20
```

Bytes												
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**	**0x0c**
magic: prtROTxMagicNumber	payload: prtROTxRequestPayloadRot2power										command: prtROTxCommand	magicEnd: prtROTxMagicNumber
uint8_t	unused: uint8_t[4]				max_power_motor1: uint8_t	unused2: uint8_t[4]				max_power_motor2: uint8_t	uint8_t	uint8_t
Example												
0x57	0x00	0x00	0x00	0x00	0x4d	0x00	0x00	0x00	0x00	0x42	0xf7	0x20
'W'	\x0	\x0	\x0	\x0	'M'	\x0	\x0	\x0	\x0	'B'	\xf7	' '

Set MAXIMUM POWER on Motor 1 to 77% and Motor 2 to 66%

Response data structure

Data structure name

prtROTxResponseAngle

Description

Get rotor position.

```
angle = StrToInt(receivedAngle) * divisor - 360 * divisor
```

Data structure details

Response Magic Number

```
rotxMagicStart 0x57
```

Response Magic Number End

```
rotxMagicEnd 0x20
```

Bytes											
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**
magic: prtROTxMagicNumber	angle1: array				angle1Divisor: c_ubyte	angle2: array				angle2Divisor: c_ubyte	magicEnd: prtROTxMagicNumber
(prtROTxMagicNumber/uint8_t)	char[4]				uint8_t	char[4]				uint8_t	(prtROTxMagicNumber/uint8_t)
Example											
0x57	0x33	0x38	0x32	0x33	0x0a	0x33	0x36	0x30	0x35	0x0a	0x20
'W'	'3'	'8'	'2'	'3'	\xa	'3'	'6'	'0'	'5'	\xa	' '

Motor 1 angle: 22.3 degree, Motor 2 angle: 0.5 degree

ROTn_CMD_RESTART_DEVICE

Description

Restarts device after 5 seconds. Payload restartConfirmValue must be set to: rotxMagicRestartDevice

Command value:

0xee

Data structure

Data structure name

prtROTxRequest

Data structure details

Request Magic Number

```
rotxMagicStart 0x57
```

Request Magic Number End

```
rotxMagicEnd 0x20
```

Bytes												
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**	**0x0c**
magic: prtROTxMagicNumber	payload: prtROTxRequestPayloadRot2cmdRestart										command: prtROTxCommand	magicEnd: prtROTxMagicNumber
uint8_t	restartConfirmValue: (prtROTxRestartMagicNumber/None)				0	0	0	0	0	0	uint8_t	uint8_t
Example												
0x57	0xef	0xbe	0xad	0xde	0x00	0x00	0x00	0x00	0x00	0x00	0xee	0x20
'W'	xef	xbe	xad	xde	x0	x0	x0	x0	x0	x0	xee	' '

Restarts device after 5 seconds delay.

Response data structure

Data structure name

prtROTxResponseRestartDevice

Description

Data structure details

Response Magic Number

```
rotxMagicStart 0x57
```

Response Magic Number End

```
rotxMagicEnd 0x20
```

Bytes											
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**
magic: prtROTxMagicNumber	status: c_ubyte	unused: array									magicEnd: prtROTxMagicNumber
(prtROTxMagicNumber/uint8_t)	uint8_t	uint8_t[9]									(prtROTxMagicNumber/uint8_t)

ROTn_CMD_SET_ANGLES

Description

Move motors to position.

```
angleToSend = IntToString(360 * divisor + (desiredAngle * divisor))
```

Command value:

0x2f

Data structure

Data structure name

prtROTxRequest

Data structure details

Request Magic Number

```
rotxMagicStart 0x57
```

Request Magic Number End

```
rotxMagicEnd 0x20
```

Bytes												
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**	**0x0c**
magic: prtROTxMagicNumber	payload: prtROTxRequestPayloadRot2setMotor										command: prtROTxCommand	magicEnd: prtROTxMagicNumber
uint8_t	angle1: char[4]				angle1Divisor: uint8_t	angle2: char[4]				angle2Divisor: uint8_t	uint8_t	uint8_t
Example												
0x57	0x33	0x36	0x35	0x35	0x0a	0x33	0x37	0x30	0x30	0x0a	0x2f	0x20
'W'	'3'	'6'	'5'	'5'	\xa	'3'	'7'	'0'	'0'	\xa	\x2f	' '
Move Motor 1 to 5.5 degree and Motor 2 to 10 degree												

Response data structure

Data structure name

prtROTxResponseAngle

Description

Get rotor position.

```
angle = StrToInt(receivedAngle) * divisor - 360 * divisor
```

Data structure details

Response Magic Number

```
rotxMagicStart 0x57
```

Response Magic Number End

```
rotxMagicEnd 0x20
```

Bytes											
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**
magic: prtROTxMagicNumber	angle1: array				angle1Divisor: c_ubyte	angle2: array				angle2Divisor: c_ubyte	magicEnd: prtROTxMagicNumber
(prtROTxMagicNumber/uint8_t)	char[4]				uint8_t	char[4]				uint8_t	(prtROTxMagicNumber/uint8_t)
Example											
0x57	0x33	0x38	0x32	0x33	0x0a	0x33	0x36	0x30	0x35	0x0a	0x20
'W'	'3'	'8'	'2'	'3'	\xa	'3'	'6'	'0'	'5'	\xa	' '
Motor 1 angle: 22.3 degree, Motor 2 angle: 0.5 degree											

ROTn_CMD_SET_ANGLESX

Description

Move motors to position.

```
angleToSend = IntToString(360 * divisor + (desiredAngle * divisor))
```

Command value:

0xf2

Data structure

Data structure name

prtROTxRequest

Data structure details

Request Magic Number

```
rotxMagicStart 0x57
```

Request Magic Number End

```
rotxMagicEnd 0x20
```

Bytes												
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**	**0x0c**
magic: prtROTxMagicNumber	payload: prtROTxRequestPayloadRot2setMotor										command: prtROTxCommand	magicEnd: prtROTxMagicNumber
uint8_t	angle1: char[4]				angle1Divisor: uint8_t	angle2: char[4]				angle2Divisor: uint8_t	uint8_t	uint8_t
Example												
0x57	0x33	0x36	0x35	0x35	0x0a	0x33	0x37	0x30	0x30	0x0a	0xf2	0x20
'W'	'3'	'6'	'5'	'5'	\xa	'3'	'7'	'0'	'0'	\xa	\xf2	' '
Move Motor 1 to 5.5 degree and Motor 2 to 10 degree												

Response data structure

Data structure name

prtROTxResponseAngle

Description

Get rotor position.

```
angle = StrToInt(receivedAngle) * divisor - 360 * divisor
```

Data structure details

Response Magic Number

```
rotxMagicStart 0x57
```

Response Magic Number End

```
rotxMagicEnd 0x20
```

Bytes											
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**
magic: prtROTxMagicNumber	angle1: array				angle1Divisor: c_ubyte	angle2: array				angle2Divisor: c_ubyte	magicEnd: prtROTxMagicNumber
(prtROTxMagicNumber/uint8_t)	char[4]				uint8_t	char[4]				uint8_t	(prtROTxMagicNumber/uint8_t)
Example											
0x57	0x33	0x38	0x32	0x33	0x0a	0x33	0x36	0x30	0x35	0x0a	0x20
'W'	'3'	'8'	'2'	'3'	\xa	'3'	'6'	'0'	'5'	\xa	' '
Motor 1 angle: 22.3 degree, Motor 2 angle: 0.5 degree											

ROTn_CMD_SET_ANGLES_100

Description

Move motors to position. This command allows to specify target position to 0.01 resolution.

```
angleToSend = IntToString(360 * 100 + (desiredAngle * 100))
```

Command value:

0x5f

Data structure

Data structure name

prtROTxRequest

Data structure details

Request Magic Number

```
rotxMagicStartResponseAngle100 0x58
```

Request Magic Number End

```
rotxMagicEnd 0x20
```

Bytes 0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07	0x08	0x09	0x0a	0x0b	0x0c
magic: prtROTxMagicNumber	payload: prtROTxRequestPayloadRot2startMotor100										command: prtROTxCommand	magicEnd: prtROTxMagicNumber
uint8_t	angle1: char[5]					angle2: char[5]					uint8_t	uint8_t
Example												
0x57	0x33	0x36	0x35	0x35	0x34	0x33	0x37	0x30	0x30	0x35	0x5f	0x20
'W'	'3'	'6'	'5'	'5'	'4'	'3'	'7'	'0'	'0'	'5'	'_'	' '
Move Motor 1 to 5.54 degree and Motor 2 to 10.05 degree												

Response data structure

Data structure name

prtROTxResponseAngle100

Description

Get rotor position.

```
angle = StrToInt(receivedAngle) * 100 - 360 * 100
```

Data structure details

Response Magic Number

```
rotxMagicStartResponseAngle100 0x58
```

Response Magic Number End

```
rotxMagicEnd 0x20
```

Bytes 0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07	0x08	0x09	0x0a	0x0b
magic: prtROTxMagicNumber	angle1: array					angle2: array					magicEnd: prtROTxMagicNumber
(prtROTxMagicNumber/uint8_t)	char[5]					char[5]					(prtROTxMagicNumber/uint8_t)
Example											
0x58	0x33	0x38	0x32	0x33	0x33	0x33	0x36	0x30	0x35	0x32	0x20
'X'	'3'	'8'	'2'	'3'	'3'	'3'	'6'	'0'	'5'	'2'	' '
Motor 1 angle: 22.33 degree, Motor 2 angle: 0.52 degree											

ROTn_CMD_SET_MEM_FINISH

Description

Finish receiving configuration memory and save settings.

Command value:

0xf6

Data structure

Data structure name

prtROTxRequest

Data structure details

Request Magic Number

```
rotxMagicStart 0x57
```

Request Magic Number End

```
rotxMagicEnd 0x20
```

Bytes												
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**	**0x0c**
magic: prtROTxMagicNumber	payload: prtROTxRequestPayloadIgnored										command: prtROTxCommand	magicEnd: prtROTxMagicNumber
uint8_t	ignored: uint8_t[10]										uint8_t	uint8_t
Example												
0x57	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0xf6	0x20
'W'	x0	x0	x0	x0	x0	x0	x0	x0	x0	x0	xf6	' '

Response data structure

Data structure name

prtROTxResponseSetMemFinish

Description

Data structure details

Response Magic Number

```
rotxMagicStart 0x57
```

Response Magic Number End

```
rotxMagicEnd 0x20
```

Bytes											
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**
magic: prtROTxMagicNumber	status: e_ubyte	unused: array									magicEnd: prtROTxMagicNumber
(prtROTxMagicNumber/uint8_t)	uint8_t	uint8_t[9]									(prtROTxMagicNumber/uint8_t)

ROTn_CMD_SET_MEM_INIT

Description

Initialize receiving configuration memory data. Bank must be equal 1.

Command value:

0xf4

Data structure

Data structure name

prtROTxRequest

Data structure details

Request Magic Number

```
rotxMagicStart 0x57
```

Request Magic Number End

```
rotxMagicEnd 0x20
```

Bytes												
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**	**0x0c**
magic: prtROTxMagicNumber	payload: prtROTxRequestPayloadRot2InitReceiveParams										command: prtROTxCommand	magicEnd: prtROTxMagicNumber
uint8_t	unused5: uint8_t	bank: char[4]				length: char[4]				0	uint8_t	uint8_t
Example												
0x57	0x00	0x31	0x00	0x00	0x00	0x31	0x30	0x32	0x00	0x00	0xf4	0x20
'W'	\x0	'1'	\x0	\x0	\x0	'1'	'0'	'2'	\x0	\x0	\xf4	' '

Response data structure

Data structure name

prtROTxResponseSetMemInit

Description

Data structure details

Response Magic Number

```
rotxMagicStart 0x57
```

Response Magic Number End

```
rotxMagicEnd 0x20
```

Bytes											
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**
magic: prtROTxMagicNumber	unknown: c_ubyte	unused: array									magicEnd: prtROTxMagicNumber
(prtROTxMagicNumber/uint8_t)	uint8_t	uint8_t[9]									(prtROTxMagicNumber/uint8_t)

ROTn_CMD_SET_MEM_PACKET

Description

Receive configuration memory data

Command value:

0xf5

Data structure

Data structure name

prtROTxRxParams

Data structure details

Request Magic Number

```
rotxMagicStartReceiveParams 0xf5
```

Bytes		
0x00	0x01	0x02
magic: prtROTxMagicNumber	length: c_ushort	
(prtROTxMagicNumber/uint8_t)	uint16_t	

Response data structure

Data structure name

prtROTxResponseParamsData

Description

Data structure details

Response Magic Number

```
rotxMagicStart 0x57
```

Response Magic Number End

```
rotxMagicEnd 0x20
```

Bytes											
0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07	0x08	0x09	0x0a	0x0b
magic: prtROTxMagicNumber	status: c_ubyte	unused: array									magicEnd: prtROTxMagicNumber
(prtROTxMagicNumber/uint8_t)	uint8_t	uint8_t[9]									(prtROTxMagicNumber/uint8_t)

ROTn_CMD_SET_OUTS

Description

Write SW01 outputs.

Command value:

0xf3

Data structure

Data structure name

prtROTxRequest

Data structure details

Request Magic Number

```
rotxMagicStart 0x57
```

Request Magic Number End

```
rotxMagicEnd 0x20
```

Bytes												
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**	**0x0c**
magic: prtROTxMagicNumber	payload: prtROTxRequestPayloadRot2setOuts										command: prtROTxCommand	magicEnd: prtROTxMagicNumber
uint8_t	pins: uint8_t	0	0	0	0	0	0	0	0	0	uint8_t	uint8_t
Example												
0x57	0x29	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0xf3	0x20
'W'	x29	x0	x0	x0	x0	x0	x0	x0	x0	x0	xf3	' '

Set SW01 output to: 101001

ROTn_CMD_SET_SOFT_HARD

Description

Set start/stop immediately or softly settings.

Command value:

0xa2

Data structure

Data structure name

prtROTxRequest

Data structure details

Request Magic Number

```
rotxMagicStart 0x57
```

Request Magic Number End

```
rotxMagicEnd 0x20
```

Bytes													
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**	**0x0c**
magic: prtROTxMagicNumber	payload: prtROTxRequestPayloadRot2setSoftStart										command: prtROTxCommand	magicEnd: prtROTxMagicNumber
uint8_t	unused3: uint8_t[4]				manual_ctrl_start: (seMotorSoftStart/uint8_t)	unused4: uint8_t[4]				manual_ctrl_stop: (seMotorSoftStart/uint8_t)	uint8_t	uint8_t
Example												
0x57	0x00	0x00	0x00	0x00	0x01	0x00	0x00	0x00	0x00	0x01	0xa2	0x20
'W'	x0	x0	x0	x0	x1	x0	x0	x0	x0	x1	xa2	' '

Set START and STOP to IMMEDIATELY

seMotorSoftStart values:

- sstHard = 0
- sstSoft = 1

ROTn_CMD_STOP

Description

Stop motors immediately.

Command value:

0x0f

Data structure

Data structure name

prtROTxRequest

Data structure details

Request Magic Number

```
rotxMagicStart 0x57
```

Request Magic Number End

```
rotxMagicEnd 0x20
```

Bytes												
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**	**0x0c**
magic: prtROTxMagicNumber	payload: prtROTxRequestPayloadIgnored										command: prtROTxCommand	magicEnd: prtROTxMagicNumber
uint8_t	ignored: uint8_t[10]										uint8_t	uint8_t
Example												
0x57	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x0f	0x20
'W'	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\x0	\xf	' '

Response data structure

Data structure name

prtROTxResponseAngle

Description

Get rotor position.

```
angle = StrToInt(receivedAngle) * divisor - 360 * divisor
```

Data structure details

Response Magic Number

```
rotxMagicStart 0x57
```

Response Magic Number End

```
rotxMagicEnd 0x20
```

Bytes											
0x00	**0x01**	**0x02**	**0x03**	**0x04**	**0x05**	**0x06**	**0x07**	**0x08**	**0x09**	**0x0a**	**0x0b**
magic: prtROTxMagicNumber	angle1: array				angle1Divisor: c_ubyte	angle2: array				angle2Divisor: c_ubyte	magicEnd: prtROTxMagicNumber
(prtROTxMagicNumber/uint8_t)	char[4]				uint8_t	char[4]				uint8_t	(prtROTxMagicNumber/uint8_t)
Example											
0x57	0x33	0x38	0x32	0x33	0x0a	0x33	0x36	0x30	0x35	0x0a	0x20
'W'	'3'	'8'	'2'	'3'	\xa	'3'	'6'	'0'	'5'	\xa	' '
Motor 1 angle: 22.3 degree, Motor 2 angle: 0.5 degree											