libtmcl-0.2

Generated by Doxygen 1.6.1

Mon Apr 12 14:14:09 2010

Contents

Chapter 1

Trinamic Motion Control Language library

1.1 NOTE

```
This documentation is created with the doxygen source code documentation generator.
It may be regenerated by calling "make doxygen-doc" in the main source tree if
'doxgen' (http://www.stack.nl/~dimitri/doxygen/) is installed on the system.
```

1.2 Introduction

The Trinamic Motion Control Language is a set of commands for the programming of Trinamic motor controller. For the direct control of a motor-controller board these commands have to be translated into a command number and bundled with the command arguments, the motor address and a checksum. The command set and the details of the programming process are documented in the "TMCL Reference Manual" from Trinamic, which can be downloaded from http://www.trinamic.com/.

The aim of this library is to hide the low-level conversion and addressing issues from the user for easier programming. A typical program using libtmcl can be as simple as the following example (error checking omitted!).

```
#include <tmcl/tmcl.h>

int main(void) {

     TMCLInterface *SerialIface; // Stores the interface of the motor-controller
      board
     TMCLMotor *Motor;     // Stores information about the motor to be controlle
     d

     // Init interface structure
     tmcl_init_interface(&SerialIface, TMCL_RSXXX, NULL, NULL, NULL, NULL);

     // Open the interface
     tmcl_open_interface(SerialIface, "/dev/ttyS0");

     // Init motor structure
     tmcl_init_motor(&Motor, SerialIface, TMCM301, 1, 0, TMCL_RSXXX);

     // Rotate motor left
```

```
    tmcl_rol(TestMotor, 100);

    // Cleanup
    tmcl_deinit_motor(&TestMotor);
    tmcl_close_interface(TestIface);
    tmcl_deinit_interface(&TestIface);

}
```

1.3 Installation

There are no special prerequisites for 'libtmcl' installation. Normally it should be enough to call:

- ./configure
- make

and than with 'root' privileges:

- make install

For details refer to the delivered 'INSTALL' file.

1.4 "Let the games begin!"

1.4.1 First steps

The first things you need to know are:

- The model of your trinamic controller (see TMCLModel for supported models)
- The module address and bank of you connected motor(s) (e.g. for the first motor of module "1": address=1, bank=0)
- The interface type. Currently only RS232/RS485 serial interfaces are supported. Custom communication functions (open, close, read, write) may be given to tmcl_init_interface() as pointers. See example01.c in the examples directory and have a look at rsXXX.c how to do this.

With these information first initialize and open you interface struct, e.g. for a serial RS232 connection at /dev/ttyS0:

```
...
tmcl_init_interface(&SerialIface, TMCL_RSXXX, NULL, NULL, NULL, NULL);
tmcl_open_interface(SerialIface, "/dev/ttyS0");
...
```

Remember to check the return codes for errors! 'libtmcl' functions should return values >=0 for success and <0 for failure.

After that init your motor. In this case the motor is the first motor at a TMCM-301 module with address "1":

```
...
tmcl_init_motor(&Motor, SerialIface, TMCM301, 1, 0, TMCL_RSXXX);
...
```

Again: Remember to check for errors!

Some commonly used functions are defined in convenience.h, which is included from tmcl.h by default, e.g.

- Activate limit/reference switches: tmcl_activate_limit_switch(TMCLMotor*, int limit_switch);
- Doing a refsearch: tmcl_refsearch_start(TMCLMotor*) (Remember: Reference switches have to be active for that!)
- Move to position X: tmcl_move_to_pos_abs(TMCLMotor*, int position)
- etc.

1.5 "Advanced" usage

1.5.1 Send commands

There are more commands available than what are defined in convenience.h (see TMCL Reference for details). These functions can be accessed directly by the command number defined in the TMCL reference or, for greater readability, by a command define from tmcldefs.h

For example: If you want to submit the "Move to Position (relative)" command "by hand" you can can use the tmcl_send_command(...) function as follows (Again: No error checking is done here, but you should do it in real code!):

```
...
TMCLCommand command;

command.command = TMCL_MVP;     // TMCL_MVP is defined in tmcldefs.h as command
     number "4"
command.type    = TMCL_MVP_REL; // Relative movement. Type is not necessary for
     all commands (see TMCL reference)
command.value   = 100;          // Move 100 steps relative to current position

tmcl_send_command(Motor, command, NULL);   // Submit the command. We do not exp
    ect a reply, so the last argument is NULL.
...
```

1.5.2 Axis parameter

Axis parameters control the way the motor is moving, e.g. speed, numer of limit switches, etc. The parameters available can be seen in tmcldefs.h or the TMCL reference.

For reading and writing of axis parameters the functions tmcl_get_axis_parameter(...) and tmcl_set_axis_parameter(...) exists.

Examples:

```
...
int speed;

// Get the current speed of the motor
```

```
speed = tmcl_get_axis_parameter(motor, TMCL_AP_CURR_SPEED);

// Set reference search speed. This is set as a fraction of the full positionin
    g speed,
// e.g. 2 means: half the positioning speed, 4: quarter of the full positioning
     speed, etc.
// See TMCL reference for details
tmcl_set_axis_parameter(motor, TMCL_AP_RFS_SPEED, 2);

...
```

Chapter 2

Todo List

Group AxisParam These are not complete.

Global tmcl_deinit Document this.

Global tmcl_init Document this.

Global tmcl_store_axis_parameter : Currently broken and thus not commented

Chapter 3

Module Index

3.1 Modules

Here is a list of all modules:

Chapter 4

Data Structure Index

4.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 5

File Index

5.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 6

Module Documentation

6.1 Misc defines

Defines

- #define TMCL_VERSION 3.27
- #define TMCL_DGRAM_SIZE_CAN 7
- #define TMCL_DGRAM_SIZE_IIC 8
- #define TMCL_DGRAM_SIZE_RSXXX 9
- #define TMCL_MAX_DGRAM_SIZE TMCL_DGRAM_SIZE_RSXXX

6.1.1 Define Documentation

6.1.1.1 #define TMCL_DGRAM_SIZE_CAN 7

Datagram sizes for different busses. Datagram size for CAN bus (in bytes)

Definition at line 39 of file tmcldefs.h.

6.1.1.2 #define TMCL_DGRAM_SIZE_IIC 8

Datagram size for RS232/RS485 (in bytes)

Definition at line 40 of file tmcldefs.h.

6.1.1.3 #define TMCL_DGRAM_SIZE_RSXXX 9

Datagram size for IIC interface (in bytes)

Definition at line 41 of file tmcldefs.h.

6.1.1.4 #define TMCL_MAX_DGRAM_SIZE TMCL_DGRAM_SIZE_RSXXX

Maximum Datagram size

Definition at line 42 of file tmcldefs.h.

6.1.1.5 #define TMCL_VERSION 3.27

Version of TMCL standard

Definition at line 35 of file tmcldefs.h.

6.2 Status Codes.

Defines

- #define TMCL_STATUS_SUCCESS 100
- #define TMCL_STATUS_LOADED_EEPROM 101
- #define TMCL_STATUS_WRONG_CHECKSUM 1
- #define TMCL_STATUS_INVALID_COMMAND 2
- #define TMCL_STATUS_WRONG_TYPE 3
- #define TMCL_STATUS_INVALID_VALUE 4
- #define TMCL_STATUS_EEPROM_LOCKED 5
- #define TMCL_STATUS_COMMAND_NA 6

6.2.1 Detailed Description

These are the status codes returned by the module.

6.2.2 Define Documentation

6.2.2.1 #define TMCL_STATUS_COMMAND_NA 6

Command not available

Definition at line 61 of file tmcldefs.h.

6.2.2.2 #define TMCL_STATUS_EEPROM_LOCKED 5

Configuration EEPROM locked

Definition at line 60 of file tmcldefs.h.

6.2.2.3 #define TMCL_STATUS_INVALID_COMMAND 2

Invalid command

Definition at line 57 of file tmcldefs.h.

6.2.2.4 #define TMCL_STATUS_INVALID_VALUE 4

Invalid value

Definition at line 59 of file tmcldefs.h.

6.2.2.5 #define TMCL_STATUS_LOADED_EEPROM 101

Command loaded into TCML program EEPROM

Definition at line 55 of file tmcldefs.h.

6.2.2.6 #define TMCL_STATUS_SUCCESS 100

Successfully executed, no error

Definition at line 54 of file tmcldefs.h.

6.2.2.7 #define TMCL_STATUS_WRONG_CHECKSUM 1

Wrong checksum

Definition at line 56 of file tmcldefs.h.

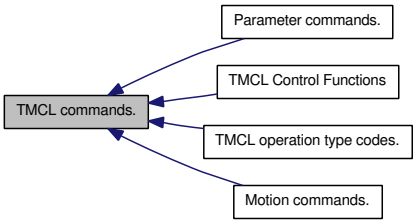
6.2.2.8 #define TMCL_STATUS_WRONG_TYPE 3

Wrong type

Definition at line 58 of file tmcldefs.h.

6.3 TMCL commands.

Collaboration diagram for TMCL commands.:



Modules

- Motion commands.
- Parameter commands.
- TMCL Control Functions
- TMCL operation type codes.

Defines

- #define TMCL_SIO 14
- #define TMCL_GIO 15
- #define TMCL_CALC 19
- #define TMCL_COMP 20
- #define TMCL_JC 21
- #define TMCL_JA 22
- #define TMCL_CSUB 23
- #define TMCL_RSUB 24
- #define TMCL_WAIT 27
- #define TMCL_STOP 28
- #define TMCL_SAC 29
- #define TMCL_SCO 30
- #define TMCL_GCO 31
- #define TMCL_CCO 32
- #define TMCL_CALCX 33
- #define TMCL_AAP 34
- #define TMCL_AGP 35
- #define TMCL_CLE 36
- #define TMCL_UF0 64
- #define TMCL_UF1 65
- #define TMCL_UF2 66
- #define TMCL_UF3 67
- #define TMCL_UF4 68
- #define TMCL_UF5 69
- #define TMCL_UF6 70
- #define TMCL_UF7 71
- #define TMCL_MVP_ABS 0
- #define TMCL_MVP_REL 1

- #define TMCL_MVP_COORD 2
- #define TMCL_RFS_START 0
- #define TMCL_RFS_STOP 1
- #define TMCL_RFS_STATUS 2

6.3.1 Define Documentation

6.3.1.1 #define TMCL_AAP 34

Accumulator to Axis Parameter

Definition at line 114 of file tmcldefs.h.

6.3.1.2 #define TMCL_AGP 35

Accumulator to Global Parameter

Definition at line 115 of file tmcldefs.h.

6.3.1.3 #define TMCL_CALC 19

Calculate

Definition at line 101 of file tmcldefs.h.

6.3.1.4 #define TMCL_CALCX 33

Calculate using the X register

Definition at line 113 of file tmcldefs.h.

6.3.1.5 #define TMCL_CCO 32

Capture Coordinate

Definition at line 112 of file tmcldefs.h.

6.3.1.6 #define TMCL_CLE 36

Clear Error Flag

Definition at line 116 of file tmcldefs.h.

6.3.1.7 #define TMCL_COMP 20

Compare

Definition at line 102 of file tmcldefs.h.

6.3.1.8 #define TMCL_CSUB 23

Call Subroutine

Definition at line 105 of file tmcldefs.h.

6.3.1.9 #define TMCL_GCO 31

Get Coordinate

Definition at line 111 of file tmcldefs.h.

6.3.1.10 #define TMCL_GIO 15

Get Input/Output

Definition at line 100 of file tmcldefs.h.

6.3.1.11 #define TMCL_JA 22

Jump Always

Definition at line 104 of file tmcldefs.h.

6.3.1.12 #define TMCL_JC 21

Jump Conditional

Definition at line 103 of file tmcldefs.h.

6.3.1.13 #define TMCL_MVP_ABS 0

Moving to absolute position

Definition at line 157 of file tmcldefs.h.

6.3.1.14 #define TMCL_MVP_COORD 2

Moving to coordinate

Definition at line 159 of file tmcldefs.h.

6.3.1.15 #define TMCL_MVP_REL 1

Moving to relative position

Definition at line 158 of file tmcldefs.h.

6.3.1.16 #define TMCL_RFS_START 0

Starting reference search

Definition at line 160 of file tmcldefs.h.

6.3.1.17 #define TMCL_RFS_STATUS 2

Checking status of reference search

Definition at line 162 of file tmcldefs.h.

6.3.1.18 #define TMCL_RFS_STOP 1

Stopping reference search

Definition at line 161 of file tmcldefs.h.

6.3.1.19 #define TMCL_RSUB 24

Return from Subroutine

Definition at line 106 of file tmcldefs.h.

6.3.1.20 #define TMCL_SAC 29

SPI Bus Access

Definition at line 109 of file tmcldefs.h.

6.3.1.21 #define TMCL_SCO 30

Set Coordinate

Definition at line 110 of file tmcldefs.h.

6.3.1.22 #define TMCL_SIO 14

Set Output

Definition at line 99 of file tmcldefs.h.

6.3.1.23 #define TMCL_STOP 28

Stop TMCL program execution

Definition at line 108 of file tmcldefs.h.

6.3.1.24 #define TMCL_UF0 64

User definable command 0

Definition at line 117 of file tmcldefs.h.

6.3.1.25 #define TMCL_UF1 65

User definable command 1

Definition at line 118 of file tmcldefs.h.

6.3.1.26 #define TMCL_UF2 66

User definable command 2

Definition at line 119 of file tmcldefs.h.

6.3.1.27 #define TMCL_UF3 67

User definable command 3

Definition at line 120 of file tmcldefs.h.

6.3.1.28 #define TMCL_UF4 68

User definable command 4

Definition at line 121 of file tmcldefs.h.

6.3.1.29 #define TMCL_UF5 69

User definable command 5

Definition at line 122 of file tmcldefs.h.

6.3.1.30 #define TMCL_UF6 70

User definable command 6

Definition at line 123 of file tmcldefs.h.

6.3.1.31 #define TMCL_UF7 71

User definable command 7

Definition at line 124 of file tmcldefs.h.

6.3.1.32 #define TMCL_WAIT 27

Wait for an event to occurr

Definition at line 107 of file tmcldefs.h.

6.4 Motion commands.

Collaboration diagram for Motion commands.:

Defines

- #define TMCL_ROR 1
- #define TMCL_ROL 2
- #define TMCL_MST 3
- #define TMCL_MVP 4
- #define TMCL_RFS 13

6.4.1 Detailed Description

Commands for controlling the motion of the module.

6.4.2 Define Documentation

6.4.2.1 #define TMCL_MST 3

Motor stop

Definition at line 79 of file tmcldefs.h.

6.4.2.2 #define TMCL_MVP 4

Move to position

Definition at line 80 of file tmcldefs.h.

6.4.2.3 #define TMCL_RFS 13

Reference search

Definition at line 81 of file tmcldefs.h.

6.4.2.4 #define TMCL_ROL 2

Rotate left

Definition at line 78 of file tmcldefs.h.

6.4.2.5 #define TMCL_ROR 1

Rotate right

Definition at line 77 of file tmcldefs.h.

6.5 Parameter commands.

Collaboration diagram for Parameter commands.:

Defines

- #define TMCL_SAP 5
- #define TMCL_GAP 6
- #define TMCL_STAP 7
- #define TMCL_RSAP 8
- #define TMCL_SGP 9
- #define TMCL_GGP 10
- #define TMCL_STGP 11
- #define TMCL_RSGP 12

6.5.1 Detailed Description

Commands for setting module parameters.

6.5.2 Define Documentation

6.5.2.1 #define TMCL_GAP 6

Get Axis Parameter

Definition at line 91 of file tmcldefs.h.

6.5.2.2 #define TMCL_GGP 10

Get Global Parameter

Definition at line 95 of file tmcldefs.h.

6.5.2.3 #define TMCL_RSAP 8

Restore Axis Parameter

Definition at line 93 of file tmcldefs.h.

6.5.2.4 #define TMCL_RSGP 12

Restore Global Parameter

Definition at line 97 of file tmcldefs.h.

6.5.2.5 #define TMCL_SAP 5

Set Axis Parameter

Definition at line 90 of file tmcldefs.h.

6.5.2.6 #define TMCL_SGP 9

Set Global Parameter

Definition at line 94 of file tmcldefs.h.

6.5.2.7 #define TMCL_STAP 7

Store Axis Parameter

Definition at line 92 of file tmcldefs.h.

6.5.2.8 #define TMCL_STGP 11

Store Global Parameter

Definition at line 96 of file tmcldefs.h.

6.6 TMCL Control Functions

Collaboration diagram for TMCL Control Functions:

Defines

- #define TMCL_CTL_STOP 128
- #define TMCL_CTL_RUN 129
- #define TMCL_CTL_STEP 130
- #define TMCL_CTL_RST 131
- #define TMCL_CTL_DLM_START 132
- #define TMCL_CTL_DLM_QUIT 133
- #define TMCL_CTL_READMEM 134
- #define TMCL_CTL_STATUS 135
- #define TMCL_CTL_FW_VER 136
- #define TMCL_CTL_FACTORY 137
- #define TMCL_CTL_ASCII 139

6.6.1 Detailed Description

Commands for controlling the TMCL module.

Note:

Not to be used in stand-alone mode

6.6.2 Define Documentation

6.6.2.1 #define TMCL_CTL_ASCII 139

Enter ASCII mode

Definition at line 147 of file tmcldefs.h.

6.6.2.2 #define TMCL_CTL_DLM_QUIT 133

Stop download mode

Definition at line 141 of file tmcldefs.h.

6.6.2.3 #define TMCL_CTL_DLM_START 132

Start download mode

Definition at line 140 of file tmcldefs.h.

6.6.2.4 #define TMCL_CTL_FACTORY 137

Restore factory settings

Definition at line 145 of file tmcldefs.h.

6.6.2.5 #define TMCL_CTL_FW_VER 136

Get firmware version

Definition at line 144 of file tmcldefs.h.

6.6.2.6 #define TMCL_CTL_READMEM 134

Read TMCL memory

Definition at line 142 of file tmcldefs.h.

6.6.2.7 #define TMCL_CTL_RST 131

Reset application

Definition at line 139 of file tmcldefs.h.

6.6.2.8 #define TMCL_CTL_RUN 129

Run application

Definition at line 137 of file tmcldefs.h.

6.6.2.9 #define TMCL_CTL_STATUS 135

Get application status

Definition at line 143 of file tmcldefs.h.

6.6.2.10 #define TMCL_CTL_STEP 130

Only exectute next command of application

Definition at line 138 of file tmcldefs.h.

6.6.2.11 #define TMCL_CTL_STOP 128

Stop application

Definition at line 136 of file tmcldefs.h.

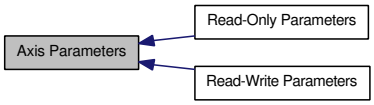
6.7 TMCL operation type codes.

Collaboration diagram for TMCL operation type codes.:

Operation type codes for TMCL commands.

6.8 Axis Parameters

Collaboration diagram for Axis Parameters:



Modules

- Read-Write Parameters
- Read-Only Parameters

6.8.1 Detailed Description

Axis parameters to be used with TMCL_SAP, TMCL_GAP, TMCL_AAP, TMCL_STAP and TMCL_-RSAP for TMCM-3xx/11x/109/61x modules.

Todo

These are not complete.

6.9 Read-Write Parameters

Collaboration diagram for Read-Write Parameters:

Defines

- #define TMCL_AP_TARGET_POS 0
- #define TMCL_AP_CURR_POS 1
- #define TMCL_AP_TARGET_SPEED 2
- #define TMCL_AP_MAX_POS_SPEED 4
- #define TMCL_AP_MAX_ACCEL 5
- #define TMCL_AP_ABS_CURRENT 6
- #define TMCL_AP_STBY_CURRENT 7
- #define TMCL_AP_DISABLE_LIMIT_R 12
- #define TMCL_AP_DISABLE_LIMIT_L 13
- #define TMCL_AP_SR_PRESC 14
- #define TMCL_AP_MICROSTEPS 140
- #define TMCL_AP_MAX_CURR_REST 143
- #define TMCL_AP_MAX_CURR_LOW_ACCEL 144
- #define TMCL_AP_MAX_CURR_HIGH_ACCEL 145
- #define TMCL_AP_RFS_MODE 193
- #define TMCL_AP_RFS_SPEED 194
- #define TMCL_AP_RFS_SW_SPEED 195

6.9.1 Detailed Description

Parameters that can be read and written

6.9.2 Define Documentation

6.9.2.1 #define TMCL_AP_ABS_CURRENT 6

Maximum absolute current

Definition at line 187 of file tmcldefs.h.

6.9.2.2 #define TMCL_AP_CURR_POS 1

Current position

Definition at line 183 of file tmcldefs.h.

6.9.2.3 #define TMCL_AP_DISABLE_LIMIT_L 13

Disable the left limit switch

Definition at line 190 of file tmcldefs.h.

6.9.2.4 #define TMCL_AP_DISABLE_LIMIT_R 12

Disable the right limit switch

Definition at line 189 of file tmcldefs.h.

6.9.2.5 #define TMCL_AP_MAX_ACCEL 5

Maximum acceleration

Definition at line 186 of file tmcldefs.h.

6.9.2.6 #define TMCL_AP_MAX_CURR_HIGH_ACCEL 145

Maximal current at high acceleration (Normally use TMCL_AP_ABS_CURRENT and TMCL_AP_-STBY_CURRENT)

Definition at line 198 of file tmcldefs.h.

6.9.2.7 #define TMCL_AP_MAX_CURR_LOW_ACCEL 144

Maximal current at low acceleration (Normally use TMCL_AP_ABS_CURRENT and TMCL_AP_-STBY_CURRENT)

Definition at line 197 of file tmcldefs.h.

6.9.2.8 #define TMCL_AP_MAX_CURR_REST 143

Maximal current at rest (Normally use TMCL_AP_ABS_CURRENT and TMCL_AP_STBY_CURRENT)

Definition at line 196 of file tmcldefs.h.

6.9.2.9 #define TMCL_AP_MAX_POS_SPEED 4

Maximum positioning speed

Definition at line 185 of file tmcldefs.h.

6.9.2.10 #define TMCL_AP_MICROSTEPS 140

Extended Parameters Microstep mode (

See also:

TMCLMicrosteps)

Definition at line 195 of file tmcldefs.h.

6.9.2.11 #define TMCL_AP_RFS_MODE 193

Reference search mode

Definition at line 199 of file tmcldefs.h.

6.9.2.12 #define TMCL_AP_RFS_SPEED 194

Reference search speed mode

Definition at line 200 of file tmcldefs.h.

6.9.2.13 #define TMCL_AP_RFS_SW_SPEED 195

Reference search speed at switch position

Definition at line 201 of file tmcldefs.h.

6.9.2.14 #define TMCL_AP_SR_PRESC 14

Note:

Currently not used

Definition at line 191 of file tmcldefs.h.

6.9.2.15 #define TMCL_AP_STBY_CURRENT 7

Maximum standby current

Definition at line 188 of file tmcldefs.h.

6.9.2.16 #define TMCL_AP_TARGET_POS 0

Basic parameters Target (next) postition

Definition at line 182 of file tmcldefs.h.

6.9.2.17 #define TMCL_AP_TARGET_SPEED 2

Desired speed in velocity mode

Definition at line 184 of file tmcldefs.h.

6.10 Read-Only Parameters

Collaboration diagram for Read-Only Parameters:

Defines

- #define TMCL_AP_CURR_SPEED 3
- #define TMCL_AP_POS_REACHED 8
- #define TMCL_AP_LIMIT_R 9
- #define TMCL_AP_LIMIT_L 10

6.10.1 Detailed Description

196 and 197 reserved

Parameters that can only be read

6.10.2 Define Documentation

6.10.2.1 #define TMCL_AP_CURR_SPEED 3

Basic parameters Current speed

Definition at line 212 of file tmcldefs.h.

6.10.2.2 #define TMCL_AP_LIMIT_L 10

Left limit switch status

Definition at line 215 of file tmcldefs.h.

6.10.2.3 #define TMCL_AP_LIMIT_R 9

Right limit switch status

Definition at line 214 of file tmcldefs.h.

6.10.2.4 #define TMCL_AP_POS_REACHED 8

Target position reached

Definition at line 213 of file tmcldefs.h.

Chapter 7

Data Structure Documentation

7.1 TMCLCommandStruct Struct Reference

`#include <src/tmcl/tmcldefs.h>`

Data Fields

- uint8_t command
- uint8_t type
- uint32_t value

7.1.1 Detailed Description

Structure containing a TMCL command and related data.

See also:

TMCL commands.

Definition at line 297 of file tmcldefs.h.

7.1.2 Field Documentation

7.1.2.1 uint8_t TMCLCommandStruct::command

Command

Definition at line 298 of file tmcldefs.h.

7.1.2.2 uint8_t TMCLCommandStruct::type

Type

Definition at line 299 of file tmcldefs.h.

7.1.2.3 uint32_t TMCLCommandStruct::value

Value

Definition at line 300 of file tmcldefs.h.

The documentation for this struct was generated from the following file:

- src/tmcl/tmcldefs.h

7.2 TMCLDeviceStruct Struct Reference

`#include <src/tmcl/tmcldefs.h>`

Data Fields

- uint8_t address
- uint8_t bank
- TMCLBusType bus
- TMCLModel model
- int num_refswitches
- TMCLParameters parameter

7.2.1 Detailed Description

Information of the TMCL module.

See also:

TMCLBusType

Definition at line 268 of file tmcldefs.h.

7.2.2 Field Documentation

7.2.2.1 uint8_t TMCLDeviceStruct::address

Address of device

Definition at line 269 of file tmcldefs.h.

7.2.2.2 uint8_t TMCLDeviceStruct::bank

Bank/channel of device

Definition at line 270 of file tmcldefs.h.

7.2.2.3 TMCLBusType TMCLDeviceStruct::bus

Bus or interface of device

Definition at line 271 of file tmcldefs.h.

7.2.2.4 TMCLModel TMCLDeviceStruct::model

TMCL Device Model

Definition at line 272 of file tmcldefs.h.

7.2.2.5 int TMCLDeviceStruct::num_refswitches

Number of reference switches on axis

Definition at line 273 of file tmcldefs.h.

7.2.2.6 TMCLParameters TMCLDeviceStruct::parameter

Array containing device parameters

Definition at line 274 of file tmcldefs.h.

The documentation for this struct was generated from the following file:

- src/tmcl/tmcldefs.h

7.3 TMCLInterfaceStruct Struct Reference

`#include <src/tmcl/interface.h>`

Data Fields

- union {
  int **fd**
  } handle

- char * **ifacename**
- TMCLBusType **bus**
- tmcl_open_funcPtr **_open**
- void * tmcl_open_void
- tmcl_close_funcPtr **_close**
- void * tmcl_close_void
- tmcl_write_funcPtr **_write**
- void * tmcl_write_void
- tmcl_read_funcPtr **_read**
- void * tmcl_read_void
- unsigned int timeout_sec
- unsigned int timeout_msec
- unsigned int timewait_sec
- unsigned int timewait_msec

7.3.1 Detailed Description

Struct to store information about the controller interface

Definition at line 36 of file interface.h.

7.3.2 Field Documentation

7.3.2.1 union { ... } TMCLInterfaceStruct::handle

handle to access the interface

7.3.2.2 unsigned int TMCLInterfaceStruct::timeout_msec

Timeout for reading from device (milliseconds) (Default 0)

Definition at line 56 of file interface.h.

7.3.2.3 unsigned int TMCLInterfaceStruct::timeout_sec

Timeouts Timeout for reading from device (seconds) (Default 2)

Definition at line 55 of file interface.h.

7.3.2.4 unsigned int TMCLInterfaceStruct::timewait_msec

Time to wait for reply of board (milliseconds) (Default 10)

Definition at line 60 of file interface.h.

7.3.2.5 unsigned int TMCLInterfaceStruct::timewait_sec

Due to the processing time of the board it may be necessary to wait some microseconds until the reply is ready. Time to wait for reply of board (seconds) (Default 0)

Definition at line 59 of file interface.h.

7.3.2.6 void* TMCLInterfaceStruct::tmcl_close_void

Void pointer store a custom data close function

Definition at line 48 of file interface.h.

7.3.2.7 void* TMCLInterfaceStruct::tmcl_open_void

Void pointer store a custom open function

Definition at line 46 of file interface.h.

7.3.2.8 void* TMCLInterfaceStruct::tmcl_read_void

Void pointer store a custom data read function

Definition at line 52 of file interface.h.

7.3.2.9 void* TMCLInterfaceStruct::tmcl_write_void

Void pointer store a custom data write function

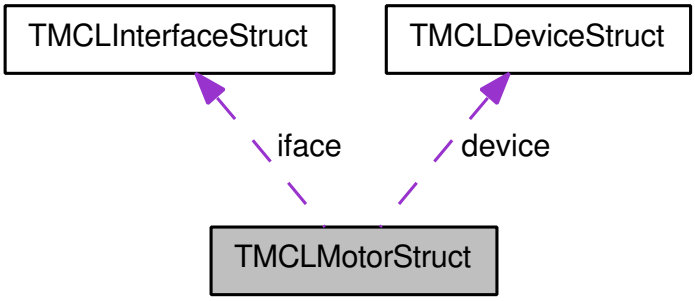
Definition at line 50 of file interface.h.

The documentation for this struct was generated from the following file:

- src/tmcl/interface.h

7.4 TMCLMotorStruct Struct Reference

`#include <src/tmcl/motor.h>`Collaboration diagram for TMCLMotorStruct:



Data Fields

- TMCLDevice **device**
- TMCLInterface * **iface**

7.4.1 Detailed Description

Motor handler

Stores information about the motor and the interface of the controller board

Definition at line 32 of file motor.h.

The documentation for this struct was generated from the following file:

- src/tmcl/motor.h

7.5 TMCLReplyStruct Struct Reference

`#include <src/tmcl/tmcldefs.h>`

Data Fields

- uint8_t reply_address
- uint8_t module_address
- uint8_t status
- uint8_t command
- uint32_t value
- uint8_t checksum

7.5.1 Detailed Description

Structure for holding the reply of a module.

See also:

Status Codes., TMCL commands.

Definition at line 283 of file tmcldefs.h.

7.5.2 Field Documentation

7.5.2.1 uint8_t TMCLReplyStruct::checksum

Checksum

Definition at line 289 of file tmcldefs.h.

7.5.2.2 uint8_t TMCLReplyStruct::command

Command

Definition at line 287 of file tmcldefs.h.

7.5.2.3 uint8_t TMCLReplyStruct::module_address

Module address

Definition at line 285 of file tmcldefs.h.

7.5.2.4 uint8_t TMCLReplyStruct::reply_address

Reply address

Definition at line 284 of file tmcldefs.h.

7.5.2.5 uint8_t TMCLReplyStruct::status

Status Code

Definition at line 286 of file tmcldefs.h.

7.5.2.6 uint32_t TMCLReplyStruct::value

Value

Definition at line 288 of file tmcldefs.h.

The documentation for this struct was generated from the following file:

- src/tmcl/tmcldefs.h

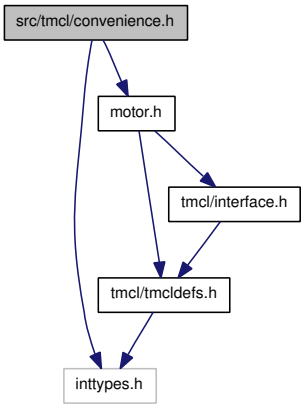
Chapter 8

File Documentation

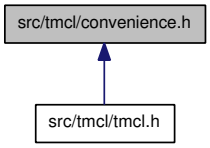
8.1 src/tmcl/convenience.h File Reference

```
#include <inttypes.h>
#include "motor.h"
```

Include dependency graph for convenience.h:



This graph shows which files directly or indirectly include this file:



Functions

- int tmcl_move_to_pos_abs (TMCLMotor *motor, int position)
- int tmcl_move_to_pos_rel (TMCLMotor *motor, int position)
- int tmcl_move_to_coord (TMCLMotor *motor, int coordinate)

- int tmcl_stop (TMCLMotor *motor)
- int tmcl_refsearch_start (TMCLMotor *motor)
- int tmcl_refsearch_stop (TMCLMotor *motor)
- int tmcl_refsearch_status (TMCLMotor *motor)
- int32_t tmcl_get_position (TMCLMotor *motor)
- int tmcl_ror (TMCLMotor *motor, int velocity)
- int tmcl_rol (TMCLMotor *motor, int velocity)
- int tmcl_set_max_current (TMCLMotor *motor, unsigned int percent)
- int tmcl_get_max_current (TMCLMotor *motor)
- int tmcl_set_max_standby_current (TMCLMotor *motor, unsigned int percent)
- int tmcl_get_max_standby_current (TMCLMotor *motor)
- int tmcl_set_microsteps (TMCLMotor *motor, int microsteps)
- int tmcl_get_microsteps (TMCLMotor *motor)
- int tmcl_activate_limit_switch (TMCLMotor *motor, int limit_switch)
- int tmcl_deactivate_limit_switch (TMCLMotor *motor, int limit_switch)
- int tmcl_get_limit_switch (TMCLMotor *motor, int limit_switch)
- int tmcl_set_no_ref_switch (TMCLMotor *motor, int number)
- int tmcl_get_current_speed (TMCLMotor *motor)
- int tmcl_set_refsearch_speed (TMCLMotor *motor, int fraction)
- int tmcl_get_refsearch_speed (TMCLMotor *motor)
- int tmcl_set_pos_speed (TMCLMotor *motor, int speed)
- int tmcl_get_pos_speed (TMCLMotor *motor)
- int tmcl_get_limit_status (TMCLMotor *motor, int limit_switch)

8.1.1 Detailed Description

Convenience function for regularly used actions

Definition in file convenience.h.

8.1.2 Function Documentation

8.1.2.1 int tmcl_activate_limit_switch (TMCLMotor * *motor*, int *limit_switch*)

Activate limit switch

Parameters:

← ***limit_switch*** ID of limit switch to activate

Returns:

- 0 on success
- -1 on failure

Definition at line 410 of file convenience.c.

8.1.2.2 int tmcl_deactivate_limit_switch (TMCLMotor * *motor*, int *limit_switch*)

Deactivate limit switch

Parameters:

← ***limit_switch*** ID of limit switch to deactivate

Returns:

- 0 on success
- -1 on failure

Definition at line 415 of file convenience.c.

8.1.2.3 int tmcl_get_current_speed (TMCLMotor * *motor*)

Get current speed of motor

Returns:

- >=0: current speed of motor
- -1 on failure

Definition at line 482 of file convenience.c.

8.1.2.4 int tmcl_get_limit_status (TMCLMotor * *motor*, int *limit_switch*)

Get status of limit switch

Parameters:

← ***limit_switch*** Limit switch to check 0: left switch, 1: right switch

Returns:

- 0: when limit switch is open
- 1: when limit switch is closed
- -1: on failure

Definition at line 504 of file convenience.c.

8.1.2.5 int tmcl_get_limit_switch (TMCLMotor * *motor*, int *limit_switch*)

Check if limit switch is active

Parameters:

← ***limit_switch*** Switch to check

Returns:

- 1 when `limit_switch` is active
- 0 when not active
- -1 on error

Definition at line 384 of file convenience.c.

8.1.2.6 int tmcl_get_max_current (TMCLMotor * *motor*)

Get maximum current

Returns:

- current in percent of available full current
- -1 on error

Definition at line 248 of file convenience.c.

8.1.2.7 int tmcl_get_max_standby_current (TMCLMotor * *motor*)

Get maximum standby current

Returns:

- current in percent of available full current
- -1 on error

Definition at line 272 of file convenience.c.

8.1.2.8 int tmcl_get_microsteps (TMCLMotor * *motor*)

Get used microsteps

Note:

not for TMCM100 model

Returns:

- microsteps: 0 (full step mode), 1 (half step mode), 2, 4, 8, 16, 32, 64 microsteps
- -1 on failure

Definition at line 320 of file convenience.c.

8.1.2.9 int tmcl_get_pos_speed (TMCLMotor * *motor*)

Get positioning speed

Definition at line 498 of file convenience.c.

8.1.2.10 int32_t tmcl_get_position (TMCLMotor * *motor*)

Get current position of motor

Definition at line 166 of file convenience.c.

8.1.2.11 int tmcl_get_refsearch_speed (TMCLMotor * *motor*)

Get reference search speed

Returns:

- >=0: reference search speed in fraction of positioning speed
- -1: failure

Definition at line 477 of file convenience.c.

8.1.2.12 int tmcl_move_to_coord (TMCLMotor * *motor*, int *coordinate*)

Move to previously stored coordinate See TMCL reference for details

Definition at line 76 of file convenience.c.

8.1.2.13 int tmcl_move_to_pos_abs (TMCLMotor * *motor*, int *position*)

Move motor to absolute position

Definition at line 26 of file convenience.c.

8.1.2.14 int tmcl_move_to_pos_rel (TMCLMotor * *motor*, int *position*)

Move motor relative to current position

Definition at line 63 of file convenience.c.

8.1.2.15 int tmcl_refsearch_start (TMCLMotor * *motor*)

Start reference search

Definition at line 101 of file convenience.c.

8.1.2.16 int tmcl_refsearch_status (TMCLMotor * *motor*)

Get status of reference search

Returns:

- 1 when reference search is running
- 0 when no reference search is running
- -1 on error

Definition at line 137 of file convenience.c.

8.1.2.17 int tmcl_refsearch_stop (TMCLMotor * *motor*)

Stop reference search

Definition at line 118 of file convenience.c.

8.1.2.18 int tmcl_rol (TMCLMotor * *motor*, int *velocity*)

Rotate motor left

Definition at line 51 of file convenience.c.

8.1.2.19 int tmcl_ror (TMCLMotor * *motor*, int *velocity*)

Rotate motor right

Definition at line 39 of file convenience.c.

8.1.2.20 int tmcl_set_max_current (TMCLMotor * *motor*, unsigned int *percent*)

Set maximum current

Parameters:

← ***current*** in percent of full current

Definition at line 231 of file convenience.c.

8.1.2.21 int tmcl_set_max_standby_current (TMCLMotor * *motor*, unsigned int *percent*)

Set maximum standby current

Parameters:

← ***maximum*** standby current in percent

Definition at line 257 of file convenience.c.

8.1.2.22 int tmcl_set_microsteps (TMCLMotor * *motor*, int *microsteps*)

Set microsteps for movement

Note:

not for TMCM100 model

Parameters:

← ***microsteps*** 0 (full step mode), 1 (half step mode), 2, 4, 8, 16, 32, 64 microsteps

Definition at line 281 of file convenience.c.

8.1.2.23 int tmcl_set_no_ref_switch (TMCLMotor * *motor*, int *number*)

Set number of reference switches

Parameters:

← ***number*** Number of reference switches (1-3)

Returns:

- 0 on success
- -1 on failure

Definition at line 420 of file convenience.c.

8.1.2.24 int tmcl_set_pos_speed (TMCLMotor * *motor*, int *speed*)

Set positioning speed

Parameters:

← ***speed*** Positioning speed 0-2047

Returns:

- 0 on success
- -1 on failure

Definition at line 488 of file convenience.c.

8.1.2.25 int tmcl_set_refsearch_speed (TMCLMotor * *motor*, int *fraction*)

Set reference search speed

Parameters:

← ***fraction*** Set reference search speed to 1/fraction of positioning speed

Definition at line 453 of file convenience.c.

8.1.2.26 int tmcl_stop (TMCLMotor * *motor*)

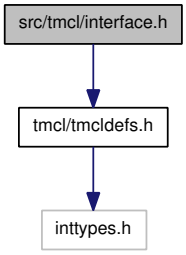
Stop motor

Definition at line 89 of file convenience.c.

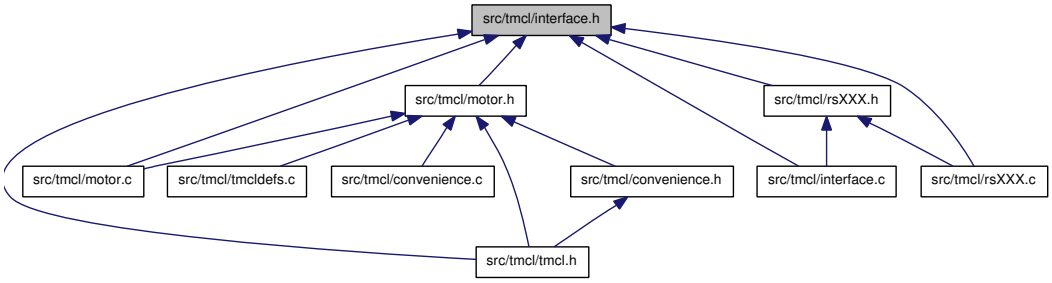
8.2 src/tmcl/interface.h File Reference

`#include <tmcl/tmcldefs.h>`

Include dependency graph for interface.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct TMCLInterfaceStruct

Typedefs

- typedef int(∗ tmcl_open_funcPtr)(struct TMCLInterfaceStruct ∗iface, const char ∗ifacename, void ∗)
- typedef int(∗ **tmcl_close_funcPtr**)(struct TMCLInterfaceStruct ∗iface, void ∗)
- typedef int(∗ **tmcl_write_funcPtr**)(struct TMCLInterfaceStruct ∗iface, const void ∗buffer, int length, void ∗)
- typedef int(∗ **tmcl_read_funcPtr**)(struct TMCLInterfaceStruct ∗iface, char ∗buffer, void ∗)
- typedef struct TMCLInterfaceStruct TMCLInterface

Functions

- int tmcl_init_interface (TMCLInterface ∗∗iface, TMCLBusType bus, tmcl_open_funcPtr open, tmcl_close_funcPtr close, tmcl_read_funcPtr read, tmcl_write_funcPtr write)
- void tmcl_set_open_data (TMCLInterface ∗iface, void ∗func_pointer)
- void tmcl_set_close_data (TMCLInterface ∗iface, void ∗func_pointer)
- void tmcl_set_read_data (TMCLInterface ∗iface, void ∗func_pointer)

- void tmcl_set_write_data (TMCLInterface *iface, void *func_pointer)
- void tmcl_deinit_interface (TMCLInterface **iface)
- int tmcl_open_interface (TMCLInterface *iface, const char *filename)
- int tmcl_close_interface (TMCLInterface *iface)
- void tmcl_interface_set_timeout (TMCLInterface *iface, unsigned int sec, unsigned int msec)
- void tmcl_interface_set_timewait (TMCLInterface *iface, unsigned int sec, unsigned int msec)

8.2.1 Detailed Description

Functions, structures, etc. to access the interface of the controller board

Definition in file interface.h.

8.2.2 Typedef Documentation

8.2.2.1 typedef int(* tmcl_open_funcPtr)(struct TMCLInterfaceStruct *iface, const char *ifacename, void *)

Function pointers for open/close and read/write interface communication functions

Definition at line 30 of file interface.h.

8.2.2.2 typedef struct TMCLInterfaceStruct TMCLInterface

Struct to store information about the controller interface

8.2.3 Function Documentation

8.2.3.1 int tmcl_close_interface (TMCLInterface * *iface*)

Close interface *

Returns:

- 0 on success
- -1 on failure

Definition at line 130 of file interface.c.

8.2.3.2 void tmcl_deinit_interface (TMCLInterface ** *iface*)

Deinitialize interface

Definition at line 107 of file interface.c.

8.2.3.3 int tmcl_init_interface (TMCLInterface ** *iface*, TMCLBusType *bus*, tmcl_open_funcPtr *open*, tmcl_close_funcPtr *close*, tmcl_read_funcPtr *read*, tmcl_write_funcPtr *write*)

Initialize TMCLInterface struct

Custom open/close/read/write functions may be given here. Use `NULL` to use the builtin functions.

Returns:

- 0 on success
- -1 on failure

Definition at line 39 of file interface.c.

8.2.3.4 void tmcl_interface_set_timeout (TMCLInterface * *iface*, unsigned int *sec*, unsigned int *msec*)

Adjust timout for interface communication

Definition at line 143 of file interface.c.

8.2.3.5 void tmcl_interface_set_timewait (TMCLInterface * *iface*, unsigned int *sec*, unsigned int *msec*)

Adjust how long to wait for reply from motor controller

Definition at line 151 of file interface.c.

8.2.3.6 int tmcl_open_interface (TMCLInterface * *iface*, const char * *filename*)

Open interface *

Parameters:

← ***iface*** TMCLInterface struct
← ***filename*** filename of interface device (for RSXXX)

Returns:

- 0 on success
- -1 on failure

Definition at line 113 of file interface.c.

8.2.3.7 void tmcl_set_close_data (TMCLInterface * *iface*, void * *func_pointer*)

Set custom close function for interface

Definition at line 95 of file interface.c.

8.2.3.8 void tmcl_set_open_data (TMCLInterface * *iface*, void * *func_pointer*)

Set custom open function for interface

Definition at line 91 of file interface.c.

8.2.3.9 void tmcl_set_read_data (TMCLInterface * *iface*, void * *func_pointer*)

Set custom read function for interface

Definition at line 99 of file interface.c.

8.2.3.10 void tmcl_set_write_data (TMCLInterface * *iface*, void * *func_pointer*)

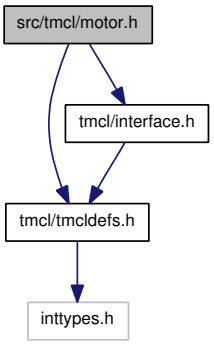
Set custom write function for interface

Definition at line 103 of file interface.c.

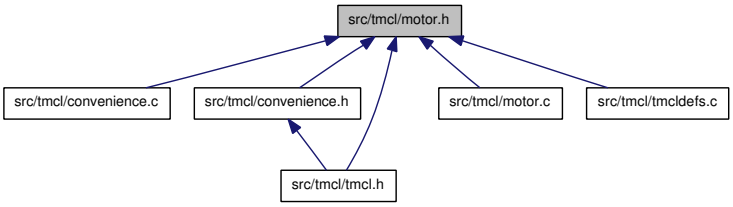
8.3 src/tmcl/motor.h File Reference

```
#include <tmcl/tmcldefs.h>
#include <tmcl/interface.h>
```

Include dependency graph for motor.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct TMCLMotorStruct

Typedefs

- typedef struct TMCLMotorStruct TMCLMotor

Functions

- int tmcl_init_motor (TMCLMotor **mot, TMCLInterface *iface, TMCLModel model, uint8_t address, uint8_t bank, TMCLBusType bus)
- void tmcl_deinit_motor (TMCLMotor **mot)
- int tmcl_send_command (TMCLMotor *mot, TMCLCommand tcom, TMCLReply *reply)
- int tmcl_update_axis_parameter (TMCLMotor *mot, int axis_parameter)
- int tmcl_set_axis_parameter (TMCLMotor *mot, int axis_parameter, int value)
- int tmcl_get_axis_parameter (TMCLMotor *mot, int axis_parameter)
- int tmcl_store_axis_parameter (TMCLMotor *mot, int axis_parameter)

8.3.1 Detailed Description

Motor communication and configuration

Definition in file motor.h.

8.3.2 Typedef Documentation

8.3.2.1 typedef struct TMCLMotorStruct TMCLMotor

Motor handler

Stores information about the motor and the interface of the controller board

8.3.3 Function Documentation

8.3.3.1 void tmcl_deinit_motor (TMCLMotor ** *mot*)

Deinitialize motor handling structure

Definition at line 57 of file motor.c.

8.3.3.2 int tmcl_get_axis_parameter (TMCLMotor * *mot*, int *axis_parameter*)

Get axis parameter from motor struct

This does not read the parameter from the board, but just from the TMCLMotor struct. To update the value in the TMCLMotor struct call tmcl_update_axis_parameter() before.

Parameters:

← ***mot*** Motor struct

← ***axis_parameter*** Parameter to get

See also:

Axis Parameters

Returns:

- 0: on success
- -1: on failure

Definition at line 173 of file motor.c.

8.3.3.3 int tmcl_init_motor (TMCLMotor ** *mot*, TMCLInterface * *iface*, TMCLModel *model*, uint8_t *address*, uint8_t *bank*, TMCLBusType *bus*)

Initialize motor handling structure

Returns:

- 0: on success

- -1: on failure

Definition at line 30 of file motor.c.

8.3.3.4 int tmcl_send_command (TMCLMotor * *mot*, TMCLCommand *tcom*, TMCLReply * *reply*)

Send command to motor

See also:

TMCLCommand

Returns:

- 0: on success
- -1: on failure

Definition at line 69 of file motor.c.

8.3.3.5 int tmcl_set_axis_parameter (TMCLMotor * *mot*, int *axis_parameter*, int *value*)

Set axis parameter in motor controller board

Parameters:

← ***mot*** Motor struct

← ***axis_parameter*** Parameter to set

See also:

Axis Parameters

Parameters:

← ***value*** New value for parameter

Returns:

- 0: on success
- -1: on failure

Definition at line 184 of file motor.c.

8.3.3.6 int tmcl_store_axis_parameter (TMCLMotor * *mot*, int *axis_parameter*)

Read all available axis parameters from the controller board and store them in the TMCLMotor struct

Returns:

- 0: on success
- -1: on failure

Todo

: Currently broken and thus not commented

Copy axis parameter from RAM to non-volatile EEPROM on board

Returns:

- 0: on success
- -1: on failure

Definition at line 213 of file motor.c.

8.3.3.7 int tmcl_update_axis_parameter (TMCLMotor * *mot*, int *axis_parameter*)

Read axis parameter 'axis_parameter' from the motor and saves it in the 'TMCLMotor' struct

Parameters:

← ***mot*** Motor struct

← ***tmcl_parameter*** Parameter to read

Returns:

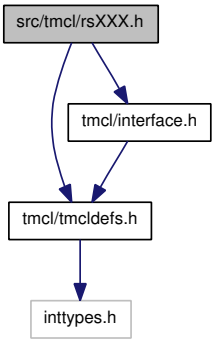
- 0: on success
- -1: on failure

Definition at line 141 of file motor.c.

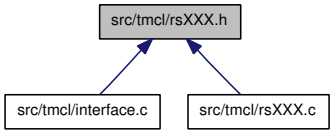
8.4 src/tmcl/rsXXX.h File Reference

```
#include <tmcl/tmcldefs.h>
#include <tmcl/interface.h>
```

Include dependency graph for rsXXX.h:



This graph shows which files directly or indirectly include this file:



Functions

- int tmcl_open_rsXXX (TMCLInterface *iface, const char *filename, void *pointer)
- int tmcl_close_rsXXX (TMCLInterface *iface, void *pointer)
- int tmcl_write_rsXXX (TMCLInterface *iface, const void *buf, int length, void *pointer)
- int tmcl_poll_rsXXX (TMCLInterface *iface, char *buffer, void *pointer)

8.4.1 Detailed Description

Communication function for RS232 and RS485 interfaces

Normally there should not be any need to call these directly.

Definition in file rsXXX.h.

8.4.2 Function Documentation

8.4.2.1 int tmcl_close_rsXXX (TMCLInterface * *iface*, void * *pointer*)

Closes the RSXXX port

- `pointer`: NOT USED!

Returns:

- 0 on success
- -1 on failure

Definition at line 143 of file rsXXX.c.

8.4.2.2 int tmcl_open_rsXXX (TMCLInterface * *iface*, const char * *filename*, void * *pointer*)

Opens the RSXXX port

- `filename`: Device node of RSXXX port
- `pointer`: NOT USED!

Returns:

- File descriptor of RSXXX port on success
- -1 on failure

Definition at line 128 of file rsXXX.c.

8.4.2.3 int tmcl_poll_rsXXX (TMCLInterface * *iface*, char * *buffer*, void * *pointer*)

Waits for data from the RSXXX port.

- `buffer`: Buffer to store received data
- `pointer`: NOT USED! RETURNS
  - >0: length of data read (in bytes)
  - -1 on failure
  - -2 on wrong length of read data

Definition at line 170 of file rsXXX.c.

8.4.2.4 int tmcl_write_rsXXX (TMCLInterface * *iface*, const void * *buf*, int *length*, void * *pointer*)

Writes to RSXXX port

- `buf`: buffer of data to be written
- `length`: length of data buffer
- `pointer`: NOT USED!

Returns:

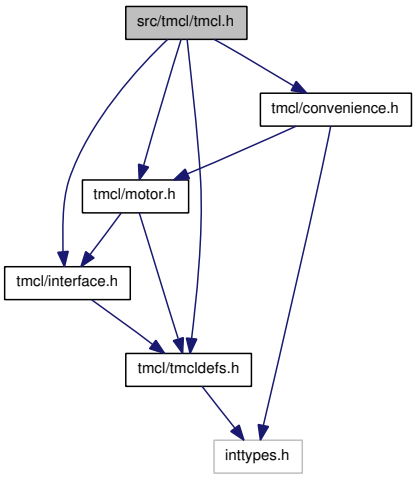
- 0 on success
- -1 on failure

Definition at line 154 of file rsXXX.c.

8.5 src/tmcl/tmcl.h File Reference

```
#include <tmcl/tmcldefs.h>
#include <tmcl/motor.h>
#include <tmcl/convenience.h>
#include <tmcl/interface.h>
```

Include dependency graph for tmcl.h:



8.5.1 Detailed Description

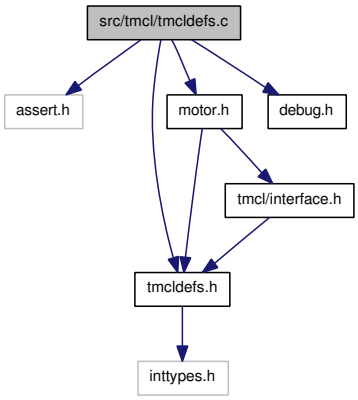
Main libtmcl include

Definition in file tmcl.h.

8.6 src/tmcl/tmcldefs.c File Reference

```
#include <assert.h>
#include "tmcldefs.h"
#include "motor.h"
#include "debug.h"
```

Include dependency graph for tmcldefs.c:



Functions

- void tmcl_init (TMCLDevice *device)
- void tmcl_deinit (TMCLDevice *device)
- uint8_t tmcl_checksum (uint8_t *commands, int length)
- int tmcl_datagram (uint8_t *datagram, TMCLDevice device, uint8_t command, uint8_t type, uint32_t value)
- int tmcl_valid_checksum (TMCLReply reply)
- int tmcl_dgram2reply (TMCLReply *reply, uint8_t *datagram, int length)

8.6.1 Detailed Description

Internal functions

Definition in file tmcldefs.c.

8.6.2 Function Documentation

8.6.2.1 uint8_t tmcl_checksum (uint8_t * *commands*, int *length*)

Parameters:

commands Buffer containing the datagram

length length of datagram

Returns:

Checksum

See also:

TMCL_DGRAM_SIZE_CAN, TMCL_DGRAM_SIZE_RSXXX, TMCL_DGRAM_SIZE_IIC

Definition at line 61 of file tmcldefs.c.

8.6.2.2 int tmcl_datagram (uint8_t * *datagram*, TMCLDevice *device*, uint8_t *command*, uint8_t *type*, uint32_t *value*)

Parameters:

datagram Buffer to store the datagram

device The device for which the datagram is intended

command The command

type Type

value Value

Returns:

Length of datagram

See also:

TMCL Commands

Definition at line 82 of file tmcldefs.c.

8.6.2.3 void tmcl_deinit (TMCLDevice * *device*)

Todo

Document this.

Definition at line 48 of file tmcldefs.c.

8.6.2.4 int tmcl_dgram2reply (TMCLReply * *reply*, uint8_t * *datagram*, int *length*)

Parameters:

reply tmcl_reply structure to store the data

datagram The datagram received from the module

length Lengh of the datagram

Returns:

- 0 on success
- -1 undefined length

Definition at line 172 of file tmcldefs.c.

8.6.2.5 void tmcl_init (TMCLDevice * *device*)

Todo

Document this.

Definition at line 35 of file tmcldefs.c.

8.6.2.6 int tmcl_valid_checksum (TMCLReply *reply*)

Parameters:

reply The reply of the module

Returns:

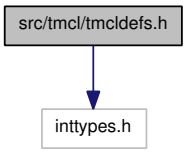
- 1 on good checksum
- 0 on bad checksum

Definition at line 138 of file tmcldefs.c.

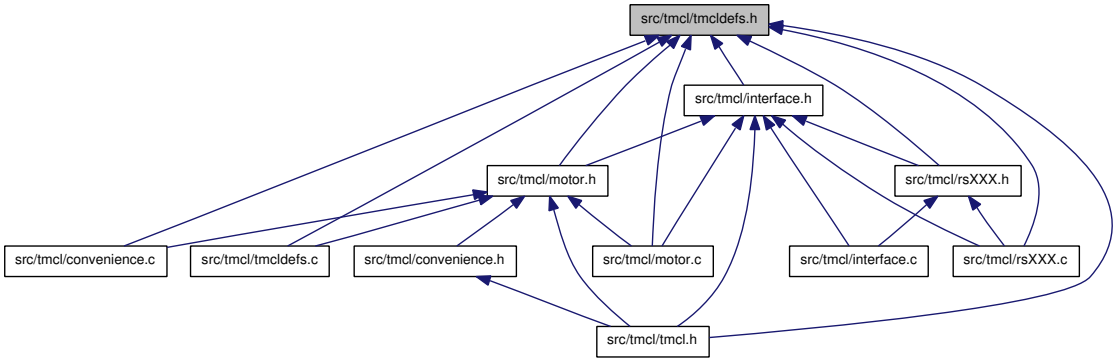
8.7 src/tmcl/tmcldefs.h File Reference

```
#include <inttypes.h>
```

Include dependency graph for tmcldefs.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct TMCLDeviceStruct
- struct TMCLReplyStruct
- struct TMCLCommandStruct

Defines

- #define **__TMCL_TMCLDEFS_H_** 1
- #define TMCL_VERSION 3.27
- #define TMCL_DGRAM_SIZE_CAN 7
- #define TMCL_DGRAM_SIZE_IIC 8
- #define TMCL_DGRAM_SIZE_RSXXX 9
- #define TMCL_MAX_DGRAM_SIZE TMCL_DGRAM_SIZE_RSXXX
- #define **TMCL_MAX_PAR_NO** 211
- #define TMCL_STATUS_SUCCESS 100
- #define TMCL_STATUS_LOADED_EEPROM 101
- #define TMCL_STATUS_WRONG_CHECKSUM 1
- #define TMCL_STATUS_INVALID_COMMAND 2
- #define TMCL_STATUS_WRONG_TYPE 3
- #define TMCL_STATUS_INVALID_VALUE 4
- #define TMCL_STATUS_EEPROM_LOCKED 5

- #define TMCL_STATUS_COMMAND_NA 6
- #define TMCL_ROR 1
- #define TMCL_ROL 2
- #define TMCL_MST 3
- #define TMCL_MVP 4
- #define TMCL_RFS 13
- #define TMCL_SAP 5
- #define TMCL_GAP 6
- #define TMCL_STAP 7
- #define TMCL_RSAP 8
- #define TMCL_SGP 9
- #define TMCL_GGP 10
- #define TMCL_STGP 11
- #define TMCL_RSGP 12
- #define TMCL_SIO 14
- #define TMCL_GIO 15
- #define TMCL_CALC 19
- #define TMCL_COMP 20
- #define TMCL_JC 21
- #define TMCL_JA 22
- #define TMCL_CSUB 23
- #define TMCL_RSUB 24
- #define TMCL_WAIT 27
- #define TMCL_STOP 28
- #define TMCL_SAC 29
- #define TMCL_SCO 30
- #define TMCL_GCO 31
- #define TMCL_CCO 32
- #define TMCL_CALCX 33
- #define TMCL_AAP 34
- #define TMCL_AGP 35
- #define TMCL_CLE 36
- #define TMCL_UF0 64
- #define TMCL_UF1 65
- #define TMCL_UF2 66
- #define TMCL_UF3 67
- #define TMCL_UF4 68
- #define TMCL_UF5 69
- #define TMCL_UF6 70
- #define TMCL_UF7 71
- #define TMCL_CTL_STOP 128
- #define TMCL_CTL_RUN 129
- #define TMCL_CTL_STEP 130
- #define TMCL_CTL_RST 131
- #define TMCL_CTL_DLM_START 132
- #define TMCL_CTL_DLM_QUIT 133
- #define TMCL_CTL_READMEM 134
- #define TMCL_CTL_STATUS 135
- #define TMCL_CTL_FW_VER 136
- #define TMCL_CTL_FACTORY 137

- #define TMCL_CTL_ASCII 139
- #define TMCL_MVP_ABS 0
- #define TMCL_MVP_REL 1
- #define TMCL_MVP_COORD 2
- #define TMCL_RFS_START 0
- #define TMCL_RFS_STOP 1
- #define TMCL_RFS_STATUS 2
- #define TMCL_AP_TARGET_POS 0
- #define TMCL_AP_CURR_POS 1
- #define TMCL_AP_TARGET_SPEED 2
- #define TMCL_AP_MAX_POS_SPEED 4
- #define TMCL_AP_MAX_ACCEL 5
- #define TMCL_AP_ABS_CURRENT 6
- #define TMCL_AP_STBY_CURRENT 7
- #define TMCL_AP_DISABLE_LIMIT_R 12
- #define TMCL_AP_DISABLE_LIMIT_L 13
- #define TMCL_AP_SR_PRESC 14
- #define TMCL_AP_MICROSTEPS 140
- #define TMCL_AP_MAX_CURR_REST 143
- #define TMCL_AP_MAX_CURR_LOW_ACCEL 144
- #define TMCL_AP_MAX_CURR_HIGH_ACCEL 145
- #define TMCL_AP_RFS_MODE 193
- #define TMCL_AP_RFS_SPEED 194
- #define TMCL_AP_RFS_SW_SPEED 195
- #define TMCL_AP_CURR_SPEED 3
- #define TMCL_AP_POS_REACHED 8
- #define TMCL_AP_LIMIT_R 9
- #define TMCL_AP_LIMIT_L 10

Typedefs

- typedef int32_t TMCLParameter
- typedef TMCLParameter **TMCLParameters** [TMCL_MAX_PAR_NO+1]
- typedef enum tmcl_busses TMCLBusType
- typedef enum TMCLModelEnum TMCLModel
- typedef struct TMCLDeviceStruct TMCLDevice
- typedef struct TMCLReplyStruct TMCLReply
- typedef struct TMCLCommandStruct TMCLCommand

Enumerations

- enum tmcl_busses { TMCL_CAN, TMCL_RSXXX, TMCL_IIC, TMCL_NONE }
- enum TMCLModelEnum {

  TMCM300, TMCM301, TMCM302, TMCM303,

  TMCM310, TMCM11x, TMCM109, TMCM110,

  TMCM100, TMCM610, TMCM611, TMCM612 }

Functions

- void tmcl_init (TMCLDevice *)
- void tmcl_deinit (TMCLDevice *)
- uint8_t tmcl_checksum (uint8_t *, int)
- int tmcl_datagram (uint8_t *, TMCLDevice, uint8_t, uint8_t, uint32_t)
- int tmcl_valid_checksum (TMCLReply)
- int tmcl_dgram2reply (TMCLReply *, uint8_t *, int)

8.7.1 Detailed Description

Definitions for TMCLlib

Definition in file tmcldefs.h.

8.7.2 Typedef Documentation

8.7.2.1 typedef enum tmcl_busses TMCLBusType

Supported busses and interfaces.

8.7.2.2 typedef struct TMCLCommandStruct TMCLCommand

Structure containing a TMCL command and related data.

See also:

TMCL commands.

8.7.2.3 typedef struct TMCLDeviceStruct TMCLDevice

Information of the TMCL module.

See also:

TMCLBusType

8.7.2.4 typedef enum TMCLModelEnum TMCLModel

Supported TMCL Device Models

8.7.2.5 typedef int32_t TMCLParameter

Extended Parameters Storage space for parameter of a device

Definition at line 225 of file tmcldefs.h.

8.7.2.6 typedef struct TMCLReplyStruct TMCLReply

Structure for holding the reply of a module.

See also:

Status Codes., TMCL commands.

8.7.3 Enumeration Type Documentation

8.7.3.1 enum tmcl_busses

Supported busses and interfaces.

Enumerator:

TMCL_CAN CAN bus (currently unsupported)

TMCL_RSXXX RS232/RS485 interface

TMCL_IIC IIC interface (currently unsupported)

TMCL_NONE Marker for uninitialized interface

Definition at line 237 of file tmcldefs.h.

8.7.3.2 enum TMCLModelEnum

Supported TMCL Device Models

Enumerator:

TMCM300 TMCM-300

TMCM301 TMCM-301

TMCM302 TMCM-302

TMCM303 TMCM-303

TMCM310 TMCM-310

TMCM11x TMCM-11x, except TMCL-110

TMCM109 TMCM-109

TMCM110 TMCM-110

TMCM100 TMCM-100

TMCM610 TMCM-610

TMCM611 TMCM-611

TMCM612 TMCM-612

Definition at line 248 of file tmcldefs.h.

8.7.4 Function Documentation

8.7.4.1 uint8_t tmcl_checksum (uint8_t * *commands*, int *length*)

Calculate the checksum for a datagram

Parameters:

commands Buffer containing the datagram

length length of datagram

Returns:

Checksum

See also:

TMCL_DGRAM_SIZE_CAN, TMCL_DGRAM_SIZE_RSXXX, TMCL_DGRAM_SIZE_IIC

Definition at line 61 of file tmcldefs.c.

8.7.4.2 int tmcl_datagram (uint8_t * *datagram*, TMCLDevice *device*, uint8_t *command*, uint8_t *type*, uint32_t *value*)

Build a datagram for sending to the device.

Parameters:

datagram Buffer to store the datagram

device The device for which the datagram is intended

command The command

type Type

value Value

Returns:

Length of datagram

See also:

TMCL Commands

Definition at line 82 of file tmcldefs.c.

8.7.4.3 void tmcl_deinit (TMCLDevice * *device*)

Deinitialize TMCLDevice data structure

Todo

Document this.

Definition at line 48 of file tmcldefs.c.

8.7.4.4 int tmcl_dgram2reply (TMCLReply * *reply*, uint8_t * *datagram*, int *length*)

Convert a datagram received from a module to a tmcl_reply struct.

Parameters:

reply tmcl_reply structure to store the data

datagram The datagram received from the module

length Lengh of the datagram

Returns:

- 0 on success
- -1 undefined length

Definition at line 172 of file tmcldefs.c.

8.7.4.5 void tmcl_init (TMCLDevice * *device*)

Initialize TMCLDevice data structure

Todo

Document this.

Definition at line 35 of file tmcldefs.c.

8.7.4.6 int tmcl_valid_checksum (TMCLReply *reply*)

Check the checksum of a TMCL reply

Parameters:

reply The reply of the module

Returns:

- 1 on good checksum
- 0 on bad checksum

Definition at line 138 of file tmcldefs.c.

Index