Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

TMCM-1613 FIRMWARE MANUAL

TMCM-1613 Firmware Version 1.00 • 2016-MAR-28 | Document Revision 1.00 • 2016-MAR-28

The TMCM-1613 firmware performs hall sensor-based block commutation using single-shunt current measurement on the TMCM-1613 module. The firmware supports TMCL(TM) commands for standalone operation as well as remote control through an UART interface. Additionally, it implements an analog mode of operation with voltage-controlled speed, which can be selected at startup time. The pre-installed boot loader provides support for firmware updates through the UART interface.

Figure 1:TMCM-1613 Product Photo

Features

- Block commutation for BLDC motors
- Motor current up to 30A Peak
- Supply voltage 6...24V DC
- Configured with TMCL™ software
- TMCL-IDE updates via UART
- Cascaded motor regulation modes

Order Code

Order code	Description	Size
TMCM-1613	1-Axis BLDC controller/driver 500W/24V with PWM and Step/Dir control, hall sensor based	70 x 76 x 25 mm³

Table 1: TMCM-1613 Order Codes

© 2015 TRINAMIC Motion Control GmbH & Co. KG, Hamburg, Germany — technical change reserved. Download newest version at: www.trinamic.co

Read entire documentation; especially the Supplemental Directives in Chapter 6 (page 53)

TABLE OF CONTENTS

© 2015 TRINAMIC Motion Control GmbH & Co. KG, Hamburg, Germany — Terms of delivery and rights to technical change reserved. Download newest version at: www.trinamic.com .

Read entire documentation; especially the "Supplemental Directives" (page 53).

© 2015 TRINAMIC Motion Control GmbH & Co. KG, Hamburg, Germany — Terms of delivery and rights to technical change reserved. Download newest version at: www.trinamic.com .

Read entire documentation; especially the "Supplemental Directives" (page 53).

FIRMWARE MANUAL

1. Basic TMCL Formats and Commands

TMCL is a motion control-oriented command set that provides pre-configures, easily adabtable totation (ROR, ROL) and positioning commands (MVP). TMCL is designed to quickly connect motors to a TMCM module. The TMCM-1613 firmware quickly connects to a three-phase motor.

ℹ TMCL is available for TRINAMIC board level solutions and also for PANdrive.

TMCL-1613 has extensive Command Set

In order to configure the module to your design specification an extensive command set for all necessary motor control parameters is made available to you. Configuration options are explained in detail in this manual.

TMCM-1613 firmware runs on a microprocessor and consists of two parts:

- **Boot loader:**
  The boot loader is installed by TRINAMIC during production. It remains untouched throughout its entire product lifetime.
- **Firmware**:
  The firmware can be updated by the user. New versions can be downloaded free of charge from the product's web page [TMCM-1613].

Functional Scope of TMCM-1613

In this manual the focus is entirely on how to use the TMCM-1613 firmware for the TMCM-1613 module in order to control a 3-phase motor according to your design specification. The TMCM-1613 firmware supports standard TMCL with an additional specified range of parameters and values.

Firmware sample code and TMCL sample scripts are available on the product's web page.

The TMCM module is based on Freescale KE ARM Cortex-M0+ microcontroller and the high performance pre-driver TMCM-1613.

NOTE:

→ *The firmware is related to the standard TMCL firmware [TMCL] with regard to protocol and commands.*

→ *The TMCL firmware is available for USB and field buses like RS232, RS485 and CAN but for configuration of the TMCM-1613 only an UART interface is made available.*

→ *TMCL can be used as script language only. For more information, please refer to the TMCL manual at www.trinamic.com .*

© 2015 TRINAMIC Motion Control GmbH & Co. KG, Hamburg, Germany — Terms of delivery and rights to technical change reserved. Download newest version at: www.trinamic.com .

Read entire documentation; especially the "Supplemental Directives" (page 53).

1.1. Request Format

When commands are sent from a host to a module, the request format has to be used.

Process Description of Request Format

Every request command consists of:

- A one-byte command field,
- A one-byte type field
- A one-byte motor/bank field
- A four-byte value field.

AREAS OF SPECIAL CONCERN

When a command is sent via UART interface, it must be enclosed by an address byte at the beginning and by a checksum byte at te end.

In this case it consists of nine bytes. The binary command format for UART and USB is structured as follows:

TMCL Request Format	
Bytes	**Description**
1	Module address
1	Command number
1	Type number
1	Motor or Bank number
4	Value (MSB first!)
1	Checksum

Table 2:TMCL Request Format

Checksum Calculation

The checksum is calculated by adding up all bytes (including the module address byte) using 8-bit addition.

Here is a C-example for the calculation:

```
unsigned char i, Checksum;
unsigned char Command[9];

Checksum = Command[0];
for(i=1; i<8; i++) { Checksum+=Command[i]; }

Command[8]=Checksum;
                        // insert checksum as last byte of the command
                        // Now, send the command back to the module
```

Calculation Examples 1: TMCL Request Format

© 2015 TRINAMIC Motion Control GmbH & Co. KG, Hamburg, Germany — Terms of delivery and rights to technical change reserved. Download newest version at: www.trinamic.com .

Read entire documentation; especially the "Supplemental Directives" (page 53).

1.2. Reply Format

Whenever a command is sent to a module, the module sends a reply.

TMCL Reply Format Structure

The reply format for UART and USB is structured as follows:

TMCL Reply Format	
Bytes	**Description**
1	Reply address
1	Module address
1	Status (e.g. 100 means no error)
1	Command number
4	Value (MSB first!)
1	Checksum

Table 3: TMCL Reply Format

TMCL Reply Status Code

The checksum is calculated similar to the checksum of the request format. The status code can have one of the following values:

TMCL Reply Status Codes	
Code	**Description**
100	Successfully executed, no error
101	Command loaded into TMCL program EEPROM
1	Wrong checksum
2	Invalid command
3	Wrong type
4	Invalid value
5	Configuration EEPROM locked
6	Command not available

Table 4: TMCL Reply Status Codes

Motion Commands

These commands control the motion of the motor. They are the most important commands and can be used in direct mode or in standalone mode.

TMCL Motion Commands		
Mnemonic	**Command Number**	**Description**
ROR	1	Rotate right
ROL	2	Rotate left
MST	3	Motor stop
MVP	4	Move to position

Table 5: TMCL Motion Commands

© 2015 TRINAMIC Motion Control GmbH & Co. KG, Hamburg, Germany — Terms of delivery and rights to technical change reserved. Download newest version at: www.trinamic.com .

Read entire documentation; especially the "Supplemental Directives" (page 53).

1.3. Parameter Commands

These commands are used to set, read, and store axis parameters or global parameters. Axis parameters can be set independently for the axis, whereas global parameters control the behavior of the module itself. These commands can also be used in direct mode and in standalone mode.

TMCL Parameter Commands		
Mnemonic	**Command Number**	**Description**
SAP	5	Set axis parameter
GAP	6	Get axis parameter
STAP	7	Store axis parameter into EEPROM
RSAP	8	Restore axis parameter from EEPROM
SGP	9	Set global parameter
GGP	10	Get global parameter
STGP	11	Store global parameter into EEPROM
RSGP	12	Restore global parameter from EEPROM

Table 6: TMCL Parameter Commands

1.4. I/O Port Commands

Direct Mode and Standalone Mode

These commands control the external I/O ports and can be used in direct mode and in standalone mode.

TMCL I/O Port Commands		
Mnemonic	**Command Number**	**Meaning**
SIO	14	Set output
GIO	15	Get input

Table 7: TMCL I/O Port Commands

© 2015 TRINAMIC Motion Control GmbH & Co. KG, Hamburg, Germany — Terms of delivery and rights to technical change reserved. Download newest version at: www.trinamic.com .

Read entire documentation; especially the "Supplemental Directives" (page 53).

• FIRMWARE MANUAL | TABLE OF CONTENTS •

2. Detailed TMCL Commands Description

The module specific commands are explained in more detail on the following pages. They are listed according to their command number.

2.1. ROR (Rotate Right)

The motor is instructed to rotate with a specified velocity in right direction (increasing the position counter).

Process Description:

Internal function:

- First, velocity mode is selected.
- Then, the velocity value is transferred to axis parameter #2 (*target velocity*).

Related commands: ROL, MST, SAP, GAP

Mnemonic: ROR 0, <velocity>

ROR Request in Direct Mode			
COMMAND	**TYPE**	**MOT/BANK**	**VALUE <velocity>**
1	don't care	0	-200000... +200000

Table 8: ROR Request in Direct Mode

ROR Reply in Direct Mode		
STATUS	**COMMAND**	**VALUE**
100 – OK	1	don't care

Table 9: ROR Reply in Direct Mode

ROR Example: Rotate right with velocity = 350: Mnemonic: ROR 0, 350								
Byte Index	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Function	Target Address	Instruction Number	Type	Motor / Bank	Operand Byte3	Operand Byte2	Operand Byte1	Operand Byte0
Value (hex)	$01	$01	$00	$00	$00	$00	$01	$5e

Table 10: ROR Example: Rotate right with Velocity=350

© 2015 TRINAMIC Motion Control GmbH & Co. KG, Hamburg, Germany — Terms of delivery and rights to technical change reserved. Download newest version at: www.trinamic.com .

Read entire documentation; especially the "Supplemental Directives" (page 53).

2.2. ROL (Rotate Left)

The motor is instructed to rotate with a specified velocity (opposite direction compared to ROR, decreasing the position counter).

Process Description:

Internal function:

- First, velocity mode is selected.
- Then, the velocity value is transferred to axis parameter #2 (*target velocity*).

Related commands: ROR, MST, SAP, GAP
Mnemonic: ROL 0, <velocity>

ROL Request in Direct Mode			
COMMAND	**TYPE**	**MOT/BANK**	**VALUE <velocity>**
2	don't care	0	-200000... +200000

Table 11: ROL Request in Direct Mode

ROL Reply in Direct Mode		
STATUS	**COMMAND**	**VALUE**
100 – OK	2	don't care

Table 12: ROL Reply in Direct Mode

ROL Example: Request: Rotate left with Velocity = 1200: Mnemonic: ROL 0, 1200								
Byte Index	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Function	Target Address	Instruction Number	Type	Motor / Bank	Operand Byte3	Operand Byte2	Operand Byte1	Operand Byte0
Value (hex)	$01	$02	$00	$00	$00	$00	$04	$b0

Table 13: ROL Example: Rotate left with Velocity=1200

© 2015 TRINAMIC Motion Control GmbH & Co. KG, Hamburg, Germany — Terms of delivery and rights to technical change reserved. Download newest version at: www.trinamic.com .

Read entire documentation; especially the "Supplemental Directives" (page 53).

2.3. MST (Motor Stop)

The motor is instructed to stop.

Process Description:

Internal function: The axis parameter *target velocity* is set to zero.

Related commands: ROL, ROR, SAP, GAP

Mnemonic: MST 0

ℹ An example for MST is provided in Table 16.

MST Request in Direct Mode			
COMMAND	**TYPE**	**MOT/BANK**	**VALUE**
3	don't care	0	don't care

Table 14: MST (Motor Stop) Request in Direct Mode

MST Reply in Direct Mode		
STATUS	**COMMAND**	**VALUE**
100 – OK	3	don't care

Table 15: MST (Motor Stop) Reply in Direct Mode

MST Example: Stop Motor at Mnemonic: MST 0								
Byte Index	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Function	Target-address	Instruction Number	Type	Motor/ Bank	Operand Byte3	Operand Byte2	Operand Byte1	Operand Byte0
Value (hex)	$01	$03	$00	$00	$00	$00	$00	$00

Table 16: MST Example: Stop Motor at Mnemonic: MST 0

© 2015 TRINAMIC Motion Control GmbH & Co. KG, Hamburg, Germany — Terms of delivery and rights to technical change reserved. Download newest version at: www.trinamic.com .

2.4. MVP (Move to Position)

The motor is instructed to move to a specified relative or absolute position.

Two available Operation Types

The motor is instructed to move to a specified relative or absolute position. It uses the predefined acceleration/deceleration ramp and the positing speed. This setting can be changed by the user. The command is non-blocking (like all commands). A reply is sent immediately after command interpretation. Further commands can follow – even if the motor has not yet reached its target position. The maximum velocity and acceleration are defined by axis parameters #4 and #11.

Two operation types are available:

- **ABS:**
  Moving to an absolute position in the range from:
  -2147483648... +2147483647.
- **REL:**
  Starting a relative movement by means of an offset to the actual position.

Internal function:
A new position value is transferred to the axis parameter #0 *target position*.

Related commands: SAP, GAP, and MST
Mnemonic: MVP <ABS|REL>, 0, <position|offset value>

MVP Process Description

A new position value is transferred to the axis parameter #0 target position.

Related commands: SAP, GAP, and MST

Mnemonic: MVP <ABS|REL>, 0, <position | offset value>

MVP (ABS / REL) Request in Direct Mode			
COMMAND	**TYPE**	**MOT/BANK**	**VALUE**
4	0 ABS – absolute	0	<position> -2147483648... +2147483647
	1 REL – relative	0	<offset> -2147483648... +2147483647

Table 17: MVP ABS/ REL Request in Direct Mode

MVP Reply in Direct Mode		
STATUS	**COMMAND**	**VALUE**
100 – OK	4	don't care

Table 18: MVP ABS / REL Reply in Direct Mode

•→ Please turn page for ABS and REL examples.

© 2015 TRINAMIC Motion Control GmbH & Co. KG, Hamburg, Germany — Terms of delivery and rights to technical change reserved. Download newest version at: www.trinamic.com .

Read entire documentation; especially the "Supplemental Directives" (page 53).

MVP ABS Example: Move Motor to Absolute Position 9000: Mnemonic: MVP ABS, 0, 9000								
Byte Index	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Function	Target Address	Instruction Number	Type	Motor / Bank	Operand Byte3	Operand Byte2	Operand Byte1	Operand Byte0
Value (hex)	$01	$04	$00	$00	$00	$00	$23	$28

Table 19: MVP ABS Example: Move Motor to Absolute Position 9000

MVP REL Example: Move Motor 1000 steps to Relative Position *(move relative -1000)*: Mnemonic: MVP REL, 0, -1000								
Byte Index	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Function	Target Address	Instruction Number	Type	Motor / Bank	Operand Byte3	Operand Byte2	Operand Byte1	Operand Byte0
Value (hex)	$00	$04	$01	$00	$ff	$ff	$fc	$18

Table 20: (MVP REL Example: Move Motor 1000 Steps to Relative Position

© 2015 TRINAMIC Motion Control GmbH & Co. KG, Hamburg, Germany — Terms of delivery and rights to technical change reserved. Download newest version at: www.trinamic.com .

2.5. SAP (Set Axis Parameter)

Most of the motion control parameters of the module can be specified by using the SAP command.

SRAM Settings Process Description

The settings are stored in SRAM and therefore are volatile. Thus, information is lost after power-off.

Related commands: GAP, STAP, and RSAP
Mnemonic: SAP <parameter number>, 0, <value>

NOTE:

→ *You must use command STAP in order to store your specified setting permanently.*

→ *An example for setting the axis parameter is provided in Table 21 (page 13).*

SAP Request in Direct Mode			
COMMAND	**TYPE**	**MOT/BANK**	**VALUE**
5	<parameter number>	0	<value>

Table 21: SAP (Set Axis Parameter) Request in Direct Mode

SAP Reply in Direct Mode		
STATUS	**COMMAND**	**VALUE**
100 – OK	5	don't care

Table 22: SAP (Set Axis Parameter) Reply in Direct Mode

ℹ A list of all parameters that can be used for the SAP command is shown in section 3.

SAP Example: Absolute Maximum Current 2000mA: Mnemonic: SAP6, 0, 2000								
Byte Index	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Function	Host-address	Target-Address	Status	Instruction	OperandByte3	Operand Byte2	Operand Byte1	Operand Byte0
Value (hex)	$01	$05	$06	$00	$00	$00	$07	$D0

Table 23: SAP Example: Absolute Max. Current 2000MA:

© 2015 TRINAMIC Motion Control GmbH & Co. KG, Hamburg, Germany — Terms of delivery and rights to technical change reserved. Download newest version at: www.trinamic.com .

2.6. GAP (Get Axis Parameter)

SRAM Settings Process Description

Most parameters of the TMC6130-EVAL can be adjusted individually. They can be read out using the GAP command.

Related commands: SAP, STAP, and RSAP
Mnemonic: GAP <parameter number>, 0

NOTE:

→ *A GAP request example is provided in Table 26, and for a GAP reply example in Table 25 (page 14).*

GAP Request in Direct Mode			
COMMAND	**TYPE**	**MOT/BANK**	**VALUE**
6	<parameter number>	0	don't care

Table 24: GAP (Get Axis Parameter) Request in Direct Mode

GAP Reply in Direct Mode		
STATUS	**COMMAND**	**VALUE**
100 – OK	6	don't care

Table 25: GAP (Get Axis Parameter) Reply in Direct Mode

ℹ A list of all parameters which can be used for the GAP command is shown in section 3.

GAP Request Example: Get the actual position of motor 0: Mnemonic: GAP 1, 0								
Byte Index	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Function	Target Address	Instruction Number	Type	Motor / Bank	Operand Byte3	Operand Byte2	Operand Byte1	Operand Byte0
Value (hex)	$01	$06	$01	$00	$00	$00	$00	$00

Table 26: GAP Request Example: Get actual Position of Motor 0

GAP Reply Example								
Byte Index	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Function	Host-address	Target-Address	Status	Instruction	Operand Byte3	Operand Byte2	Operand Byte1	Operand Byte0
Value (hex)	$00	$01	$64	$06	$00	$00	$02	$c7

Table 27: GAP Reply Example

© 2015 TRINAMIC Motion Control GmbH & Co. KG, Hamburg, Germany — Terms of delivery and rights to technical change reserved. Download newest version at: www.trinamic.com .

Read entire documentation; especially the "Supplemental Directives" (page 53).

2.7. STAP (Store Axis Parameter)

STAP Settings stored in SRAM

The STAP command stores an axis parameter previously set with a Set Axis Parameter command (SAP) permanently.

ℹ Most parameters are automatically restored after power up.

Internal function:
An axis parameter stored in SRAM will be transferred to EEPROM and loaded from EEPORM after next power up.

Related commands: SAP, RSAP, and GAP
Mnemonic: STAP <parameter number>, 0

STAP Request in Direct Mode			
COMMAND	**TYPE**	**MOT/BANK**	**VALUE**
7	<parameter number>	0	don't care[1]

Table 28: STAP (Store Axis Parameter) Request in Direct Mode

[1] *The value operand of this function has no effect. Instead, the currently used value (e.g. selected by SAP) is saved.*

STAP Reply in Direct Mode		
STATUS	**COMMAND**	**VALUE**
100 – OK	7	don't care

Table 29: STAP (Store Axis Parameter) Reply in Direct Mode

ℹ A list of all parameters which can be used for the STAP command is shown in section 3.

STAP Example: Store Maximum Speed: STAP 4,0								
Byte Index	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Function	Target Address	Instruction Number	Type	Motor / Bank	Operand Byte3	Operand Byte2	Operand Byte1	Operand Byte0
Value (hex)	$01	$07	$04	$00	$00	$00	$00	$00

Table 30: STAP Example: Store Maximum Speed: STAP 4,0

ℹ The STAP command has no effect when the configuration EEPROM is locked. In this case, the error code 5 (configuration EEPROM locked) is returned.

© 2015 TRINAMIC Motion Control GmbH & Co. KG, Hamburg, Germany — Terms of delivery and rights to technical change reserved. Download newest version at: www.trinamic.com .

Read entire documentation; especially the "Supplemental Directives" (page 53).

2.8. RSAP (Restore Axis Parameter)

For all configuration related axis parameters non-volatile memory locations are provided.

Resetting a single Parameter

By default, most parameters are automatically restored after power up. A single parameter that has been changed before can be reset by this instruction also.

Internal function:
The specified parameter is copied from the configuration EEPROM memory to its RAM location.

Related commands: SAP, STAP, and GAP
Mnemonic: RSAP <parameter number>, 0

An example for RSAP is provided below:

RSAP Request in Direct Mode			
COMMAND	**TYPE**	**MOT/BANK**	**VALUE**
8	<parameter number>	0	don't care

Table 31: RSAP Request in Direct Mode

RSAP Reply in Direct Mode		
STATUS	**COMMAND**	**VALUE**
100 – OK	8	don't care

Table 32: RSAP Reply in Direct Mode

ℹ A list of all parameters which can be used for the RSAP command is shown in section 3.

RSAP Example: Restore Maxium Motor Current 0: Mnemonic: RSAP 6,0								
Byte Index	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Function	Target Address	Instruction Number	Type	Motor/ Bank	Operand Byte3	Operand Byte2	Operand Byte1	Operand Byte0
Value (hex)	$01	$08	$06	$00	$00	$00	$00	$00

Table 33: RSAP Example: Restore Maximum Motor Current 0

© 2015 TRINAMIC Motion Control GmbH & Co. KG, Hamburg, Germany — Terms of delivery and rights to technical change reserved. Download newest version at: www.trinamic.com .

2.9. SGP (Set Global Parameter)

Global parameters are related to the host interface, peripherals or other application specific variables.

Organization of Parameters in Banks

The different groups of these parameters are organized in banks to allow a larger total number for future products.

Currently, bank 0 is used for global parameters and bank 2 is intended for user variables.

Related commands: GGP, STGP, RSGP
Mnemonic: SGP <parameter number>, <bank number>, <value>

An example for SGP is provided below:

SGP Request in Direct Mode			
COMMAND	**TYPE**	**MOT/BANK**	**VALUE**
9	<parameter number>	<bank number>	<value>

Table 34: SGP (Set Global Parameter) Request in Direct Mode

SGP Reply in Direct Mode	
STATUS	**VALUE**
100 – OK	don't care

Table 35: SGP (Set Global Parameter) Reply in Direct Mode

ℹ A list of all parameters which can be used for the SGP command is shown in section 4.

SGP Example: Set Variable 0 at Bank 2 to 100: Mnemonic: SGP, 0, 2, 100								
Byte Index	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Function	Target Address	Instruction Number	Type	Motor/ Bank	Operand Byte3	Operand Byte2	Operand Byte1	Operand Byte0
Value (hex)	$01	$09	$00	$02	$00	$00	$00	$64

Table 36: SBP Example: Mnemonic: SGP, 0, 2,100

© 2015 TRINAMIC Motion Control GmbH & Co. KG, Hamburg, Germany — Terms of delivery and rights to technical change reserved. Download newest version at: www.trinamic.com .

Read entire documentation; especially the "Supplemental Directives" (page 53).

2.10. GGP (Get Global Parameter)

Read out all global Parameters

All global parameters can be read with this function.

Related commands: SGP, STGP, RSGP
Mnemonic: GGP <parameter number>, <bank number>

GGP Request in Direct Mode			
COMMAND	**TYPE**	**MOT/BANK**	**VALUE**
10	<parameter number>	<bank number>	don't care

Table 37: GGP (Get Global Parameter) Request in Direct Mode

GGP Reply in Direct Mode	
STATUS	**VALUE**
100 – OK	<value>

Table 38: GGP (Get Global Parameter) Reply in Direct Mode

ℹ A list of all parameters which can be used for the GGP command is shown in section 4.

GGP Example: Get Variable 0 from Bank 2: Mnemonic: GGP, 0, 2								
Byte Index	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Function	Target Address	Instruction Number	Type	Motor/ Bank	Operand Byte3	Operand Byte2	Operand Byte1	Operand Byte0
Value (hex)	$01	$0a	$00	$02	$00	$00	$00	$00

Table 39:GGP Example: Get Variable 0 from Bank 2

© 2015 TRINAMIC Motion Control GmbH & Co. KG, Hamburg, Germany — Terms of delivery and rights to technical change reserved. Download newest version at: www.trinamic.com .

Read entire documentation; especially the "Supplemental Directives" (page 53).

2.11. STGP (Store Global Parameter)

STGP Configuration

Some global parameters are located in RAM memory.
Consequently, modifications are lost at power-down.

The instruction copies a value from its RAM location to the configuration EEPROM and enables permanent storing. Most parameters are automatically restored after power up.

Related commands: SGP, GGP, RSGP
Mnemonic: STGP <parameter number>, <bank number>

STGP Request in Direct Mode			
COMMAND	**TYPE**	**MOT/BANK**	**VALUE**
11	<parameter number>	<bank number>	don't care

Table 40: STGP Request (Store Global Parameter) in Direct Mode

GGP Reply in Direct Mode	
STATUS	**VALUE**
100 – OK	<value>

Table 41: STGP Reply (Store Global Parameter) in Direct Mode

ℹ A list of all parameters which can be used for the STGP command is shown in section 4.

STGP Example: Restore Variable 0 to Bank 2 to EEPROM Configuration: Mnemonic: STGP, 0, 2								
Byte Index	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Function	Target Address	Instruction Number	Type	Motor/ Bank	Operand Byte3	Operand Byte2	Operand Byte1	Operand Byte0
Value (hex)	$01	$0b	$00	$02	$00	$00	$00	$00

Table 42: STGP Example: Restore Variable 0 to Bank 2 to EEPROM Configuration

© 2015 TRINAMIC Motion Control GmbH & Co. KG, Hamburg, Germany — Terms of delivery and rights to technical change reserved. Download newest version at: www.trinamic.com .

Read entire documentation; especially the "Supplemental Directives" (page 53).

2.12. RSGP (Restore Global Parameter)

RSGP Configuration

This instruction copies a value from the EEPROM configuration to its RAM location. Thereby, the permanently stored value of a RAM-located parameter is recovered. Most parameters are automatically restored after power-up.

Related commands: SGP, GGP, STGP
Mnemonic: RSGP <parameter number>, <bank number>

RSGP Request in Direct Mode			
COMMAND	**TYPE**	**MOT/BANK**	**VALUE**
12	<parameter number>	<bank number>	don't care

Table 43: RSGP Request (Store Global Parameter) in Direct Mode

RSGP Reply in Direct Mode	
STATUS	**VALUE**
100 – OK	don't care

Table 44: RSGP Reply (Store Global Parameter) in Direct Mode

ℹ A list of all parameters which can be used for the RSGP command is shown in section 4.

RSGP Example: Restore Variable 0 to Bank 2 to EEPROM Configuration: Mnemonic: RSGP, 0, 2								
Byte Index	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Function	Target Address	Instruction Number	Type	Motor/ Bank	Operand Byte3	Operand Byte2	Operand Byte1	Operand Byte0
Value (hex)	$01	$0c	$00	$02	$00	$00	$00	$00

Table 45: RSGP Example: Restore Variable 0 to Bank 2 to EEPROM Configuration

© 2015 TRINAMIC Motion Control GmbH & Co. KG, Hamburg, Germany — Terms of delivery and rights to technical change reserved. Download newest version at: www.trinamic.com .

2.13. SIO (Set Output) and GIO (Get Input / Output)

Command Settings

The TMC6130-EVAL provides two commands for dealing with inputs and outputs:

SIO:
Sets the status of the general digital output either to low (0) or to high (1).

GIO:
Reads out the status of the two available general purpose inputs of the module.

NOTE:

→ *The command reads out a digital or analogue input port.*
→ *Digital lines read 0 and 1. ADC channel that delivers 12 bit (value of 0... 4095).*

Correlation between I/Os and Banks		
Inputs/ Outputs	**Bank**	**Description**
Digital inputs	Bank 0	Digital inputs are accessed in bank 0.
Analogue inputs	Bank 1	Analog inputs are accessed in bank 1.
Digital outputs	Bank 2	The states of the OUT lines (that have been set by SIO commands) can be read back using bank 2.

Table 46: Correlation between I/Os (SIO and GIO) and Banks

© 2015 TRINAMIC Motion Control GmbH & Co. KG, Hamburg, Germany — Terms of delivery and rights to technical change reserved. Download newest version at: www.trinamic.com .

2.14. SIO (Set Output)

Setup of General Output Status

Bank 2 is used for setting the status of the general digital output either to low (0) or to high (1).

Internal function:
The passed value is transferred to the specified output line.

Related commands: GIO, WAIT
Mnemonic: SIO <port number>, <bank number>, <value>

SIO Request in Direct Mode			
INSTRUCTION NO.	**TYPE**	**MOT/BANK**	**VALUE**
14	<port number>	<bank number> 2	<value> 0/1

Table 47: SIO Request in Direct Mode

SIO Reply in Direct Mode	
STATUS	**VALUE**
100 – OK	don't care

Table 48: SIO Reply in Direct Mode

SIO Example								
Byte Index	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Function	Target Address	Instruction Number	Type	Motor/ Bank	Operand Byte3	Operand Byte2	Operand Byte1	Operand Byte0
Value (hex)	$01	$0e	$07	$02	$00	$00	$00	$01

Table 49: SIO Example

© 2015 TRINAMIC Motion Control GmbH & Co. KG, Hamburg, Germany — Terms of delivery and rights to technical change reserved. Download newest version at: www.trinamic.com .

Read entire documentation; especially the "Supplemental Directives" (page 53).

2.15. GIO (Get Input / Output)

Two Options: Direct Mode or Standalone Mode

***GIO* can be used in direct mode or in standalone mode.**
In standalone mode, the requested value is copied to the accumulator (accu) for further processing purposes; such as conditioned jumps.

OPTION 1: IN STANDALONE MODE

The requested value is copied to the acculmular (accu) for further processing purposes; such as conditioned jumps.

OPTION 2: IN DIRECT MODE

In direct mode, the value is output in the value field of the reply without affecting the accumulator. The actual status of a digital output line can also be read.

Internal function: The specified line is read.

Related commands: SIO, WAIT
Mnemonic: GIO <port number>, <bank number>

GIO Request in Direct Mode			
INSTRUCTION NO.	**TYPE**	**MOT/BANK**	**VALUE**
15	<port number>	<bank number>	don't care

Table 50: GIO Request in Direct Mode

GIO Reply in Direct Mode	
STATUS	**VALUE**
100 – OK	<status of the port>

Table 51: GIO Reply in Direct Mode

GIO Request Example								
Byte Index	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Function	Target Address	Instruction Number	Type	Motor/ Bank	Operand Byte3	Operand Byte2	Operand Byte1	Operand Byte0
Value (hex)	$01	$0f	$00	$01	$00	$00	$00	$00

Table 52: GIO Request Example

GIO Reply Example								
Byte Index	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Function	Target Address	Instruction Number	Type	Motor/ Bank	Operand Byte3	Operand Byte2	Operand Byte1	Operand Byte0
Value (hex)	$02	$01	$64	$0f	$00	$00	$01	$2e

Table 53: GIO Reply Example

© 2015 TRINAMIC Motion Control GmbH & Co. KG, Hamburg, Germany — Terms of delivery and rights to technical change reserved. Download newest version at: www.trinamic.com .

Read entire documentation; especially the "Supplemental Directives" (page 53).

Available *SIO* and *GIO* Commands					
I/O	**Digital**	**Analog**	**GIO <port>, <bank>**	**SIO <port>, <bank>, <value>**	**Value Range**
Digital input 0	X	-	GIO 0, 0	-	0/1
Digital input 1	X	-	GIO 1, 0	-	0/1
VIN+ Detect	X	-	GIO 2, 0	-	0/1
ADC Input 0	-	X	GIO 0, 1	-	0... 4095
ADC single shunt	-	X	GIO 1, 1	-	0... 4095
ADC VSupply	-	X	GIO 2, 1	-	0... 4095
ADC Motor Temp	-	X	GIO 3, 1	-	0... 4095
UART1-Tx State	X	-	GIO 0, 2	-	0/1
UART1-Tx	X	-	-	SIO 0, 2	0/1

Table 54: Available SIO and GIO Commands

© 2015 TRINAMIC Motion Control GmbH & Co. KG, Hamburg, Germany — Terms of delivery and rights to technical change reserved. Download newest version at: www.trinamic.com .

2.16. TMCL Control Functions

TMCL Control Command 136

There are several TMCL control functions. The most important one for users is command 136.

- ℹ Other control functions can be used with axis parameters.
- ℹ Two possible reply examples are provided below.

Command 136: Request in Direct Mode				
Command	**Type**	**Parameter**	**Description**	**Access**
136	0 – string 1 – binary	Firmware version	Get the module type and firmware revision as a string or in binary format. (Motor/Bank and Value are ignored.)	read

Table 55: TMCL Command 136: Request in Direct Mode

Command 136: Request in Direct Mode Type set to 0. Reply as a String:[1]	
Byte index	**Contents**
1	Host Address
2... 9	Version string (8 characters, e.g. 1613V101)

Table 56: TMCL Command 136: Request in Direct Mode

[1] *There is no checksum in this reply format!*

TMCL Control Functions: Type Set to 1. Version Number in Binary Format:[1]	
Byte Index in Value Field	**Contents**
1	Version number, low byte.
2	Version number, high byte.
3	Type number, low byte.
4	Type number, high byte.

Table 57: TMCL Control Funtions: Type Set to 1.

[1] *The version number is output in the value field.*

© 2015 TRINAMIC Motion Control GmbH & Co. KG, Hamburg, Germany — Terms of delivery and rights to technical change reserved. Download newest version at: www.trinamic.com .