

# STEPPER MOTOR CONTROLLER

Microstep Motor Controller for the I2C Bus

Datasheet Rev.: 1.2  
Date: 09.10.2009

## Features

- Trinamic **TMC222** Microstep Stepper Motor Controller
- Controls one stepper motor with four bit microstepping
- Programmable coil current up to **800 mA**
- Motor supply voltage range: **8V ... 29V**
- Fixed frequency PWM current control with automatic selection of fast and slow decay mode
- Full step frequencies up to 1 kHz
- High temperature, open circuit, short, over-current and under-voltage diagnostics
- Internal 16-bit wide position counter
- Configurable speed and acceleration settings
- Built-in ramp generator for autonomous positioning and speed control
- On-the-fly alteration of target position
- Reference switch input available for read out

## Ordering Information

### Art.-No. 01.0018

I2C Stepper Motor Controller

### Contact:

#### Elektronik-Atelier Kallen

Giacomettistrasse 33A

CH-3006 Bern / Switzerland

Web: [www.avrcard.com](http://www.avrcard.com)

Email: [info@avrcard.com](mailto:info@avrcard.com)

Tel: +41-31-832 1441

Fax: +41-31-560 4110

## Specifications

### Motor Interface

Supply Voltage	8...29VDC
Coil Current	Max. 800mA
Step Frequency	Max. 1kHz
Connector	Industry standard 8-way Weidmüller (BL3.5/8)

### MCU Interface

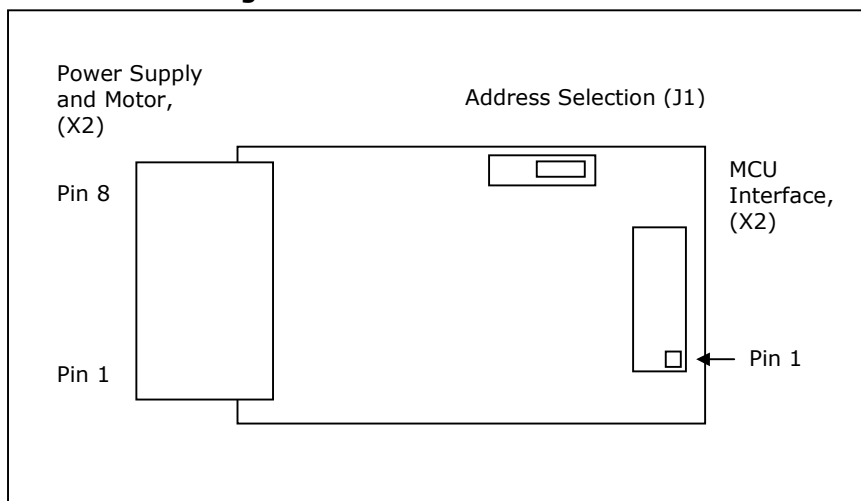
Bus Type	I2C (SCL, SDA)
Bus Speed	Up to 350kHz
Signal levels	TTL
Node address	0x00, 0xC0 (default), 32 field programmable addresses
Connector	Standard 0.1" 10-way header

### Mechanical

Size (L x W)	47 x 32 mm
Weight	25 g

## Connector Specifications

**Figure 1 – Connectors**

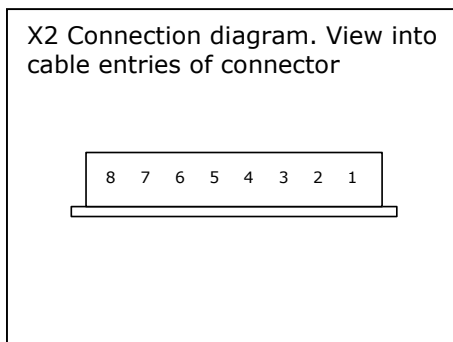


### MCU Interface

X1: MCU Interface	
No.	Function
1	
2	
3	
4	
5	
6	GND
7	
8	
9	SCL
10	SDA

### Power Supply and Motor Interface

X2: Power Supply and Motor		
No.	Function	Alt. Function
1	SWI	Reference Switch Input <sup>1</sup>
2	GND	Power GND
3	A1	Motor Coil 1
4	A2	Motor Coil 1
5	B1	Motor Coil 2
6	B2	Motor Coil 2
7	VBAT	8..29VDC Power Supply
8	GND	Power GND



VBAT min = 6.5V for I2C interface operation

VBAT min = 8.5V for motor operation

<sup>1</sup> To activate the switch, connect this pin to GND or to VBAT. Leave this pin unconnected to deactivate the switch.

## Jumper Settings

HW Bit Select			
No.	Des.	LEFT	RIGHT
J1	HW	HW = 0	HW = 1

### Physical Address on I2C Bus

HW = 0: I2C Slave address = 0xC0

HW = 1: I2C Slave address = 0xC2

The TCM222 supports a "general call" address. Therefore the circuit is addressable using either the physical slave address or address "000 0000".

## Application Information

### Hardware Setup

1. Connect the host microcontroller to X1. Make sure the I2C interface of the host works in master mode and has appropriate pull-up resistors on the bus.
2. Set Jumper J1 to left position (HW=0).
3. Connect VBAT on X2
4. Verify that Communication on the I2C bus

### I2C Bus Addressing

A total of 30 devices can be operated in parallel on the same I2C bus. They are discriminated by means of the physical address. The default address is 0xC0. This address can be field programmed in the OTP, using the reserved address '0'.

All new devices are shipped with the OTP physical address bits set to zero. In order to program the address, it is therefore necessary to connect only one device at a time to the bus for programming. Please refer to the TMC222 datasheet for a description of the programming sequence.

Note: All OTP parameters are programmable only once from '0' to '1'.

### Motor Functional Test

Following sequence of I2C commands can be used for testing the module.

#### Step 1: Get full status

```
I2COut (TMC222, GetFullStatus1);
```

#### Step 2: Set motor parameters

```
ar[0]:= $FF;
ar[1]:= $FF;
ar[2]:= $55; // Irun, Ihold
ar[3]:= $81; // Vmax, Vmin
ar[4]:= $10; // Shaft = 1
ar[5]:= $00; // SecPos
ar[6]:= $0C; // StepMode
I2COut (TMC222, SetMotorParam, ar);
```

#### Step 3: Get full status

```
I2COut (TMC222, GetFullStatus1);
for i:= 0 to 7 do
  I2COut (TMC222, i);
  I2CInp (TMC222, v);
  Write (SerOut, ByteToHex (v) + ' ');
endfor;
```

Now, Ihold is flowing into the motor.

**Step 4: Drive motor**

```
    ar1[0]:= $FF;
    ar1[1]:= $FF;
    ar1[2]:= $FF; // Pos byte 1
    ar1[3]:= $7F; // Pos byte 2
    ar1[4]:= $00; //
    ar1[5]:= $00; //
    ar1[6]:= $00; //
    I2COut (TMC222, SetPosition, ar1);
```

The motor drives to the set position and is then hold.

**Step 5: Reset position**

```
    I2COut (TMC222, ResetPosition);
```

**Step 6: Drive Motor again, etc.**

## Constant definitions used

```
TMC222:           Byte = %1100000;
GetFullStatus1:  Byte = $81;
GetFullStatus2:  Byte = $FC;
SetMotorParam:   Byte = $89;
SetPosition:     Byte = $8B;
GetOtpParam:     Byte = $82;
ResetDefault:    Byte = $87;
ResetPosition:   Byte = $86;
```

## Notice to Users

The intended use of the Stepper modules is described in this document. Other than the described uses are not permitted or only after consultation with the manufacturer.

Stepper modules are not authorized for use as critical components in life-support devices or systems.

Life-support devices or systems are devices or systems intended for surgical implantation into the body or to sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling and user's manual, can be reasonably expected to result in significant injury.

No complex software or hardware system is perfect. Bugs are always present in a system of any size. In order to prevent danger to life or property, it is the responsibility of the system designer to incorporate redundant protective mechanisms appropriate to the risk involved.

All Stepper modules are 100 percent functionally tested. Additional testing may include visual quality control inspections. Specifications are based on characterization of tested sample units rather than testing over temperature and voltage of each unit. Stepper modules may qualify components to operate within a range of parameters that is different from the manufacturer's recommended range.