 **Controller**        **Motor** (MS17HA2P4100)

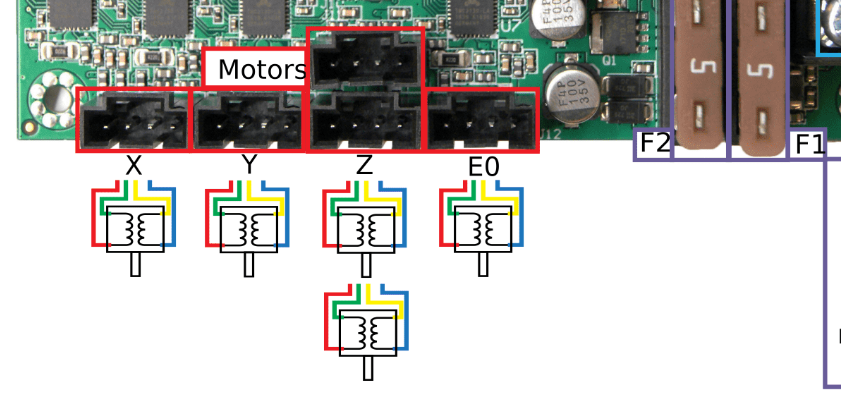
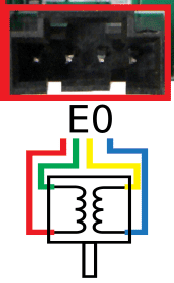
**A+** PIN1 (schwarz) --------- PIN1 (schwarz)

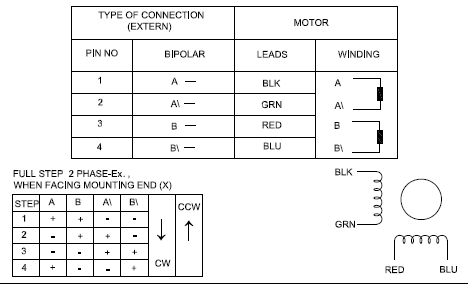
**A-** PIN2 (grün) ------------- PIN2 (grün)

**B+** PIN3 (rot) --------------- PIN3 (rot)

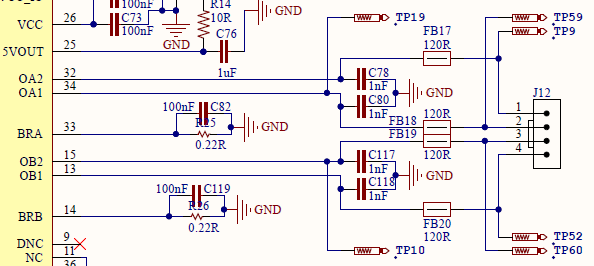
**B-** PIN4 (blau) ------------- PIN4 (blau)

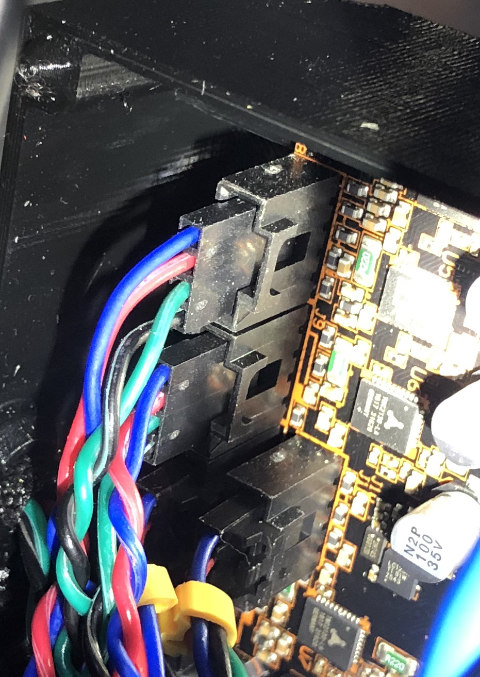
Einsy Rambo 1.1a - Trinamic (OA1-Pin2, OA2-Pin1, OB1-Pin4 & OB2-Pin3)



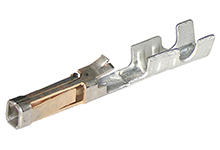


4 3 **2** **1**





## Einsy Rambo 1.1a Controller: MOLEX 5057-9404 Crimpgehäuse - SL - 1x4-polig



1 2 3 4

## MOLEX Crimpkontakt 70058-0006 (Gold)

## oder 70058-0004 (Zinn)

## Schrittmotor Nema 17 25mm: MOLEX 87369-0600 Crimpgehäuse 2mm, 1x6-polig

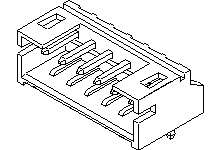
## (End-of-Life-Produkt : Gilt als veraltet und wurde vom Hersteller eingestellt)

## 87369-0600Molex 505487-8000

1 3 4 6

## MOLEX Crimpkontakt 50212-8000 (Zinn)

## oder 50212-8100 (Zinn)

**

# passend zu Molex 89401-0610

The following table provides the length of the stock motor cable wires on the MK3. This is measured from the motor body to the start of the connector housing. If you plan on twisting the cable like those on the Prusa-supplied motors, make sure you do that before measuring.

| **Axis** | **Length (mm)** |
| --- | --- |
| E | 720 |
| X | 365 |
| Y | 510 |
| Z (short) | 255 |
| Z (long) | 555 |

**Änderungen für X/Y 0.9er Stepper (OMC 17HM15-0904S) FW 3.8.1**

**Configuration\_prusa.h**

//#define DEFAULT\_AXIS\_STEPS\_PER\_UNIT {100,100,3200/8,140}

#define DEFAULT\_AXIS\_STEPS\_PER\_UNIT {**201.5,202**,**400**,**423.232324**} // XYZE axes calibrated (XY-Messlänge 100mm, Z-Messlänge 20mm, E-Messlänge 300mm);**E changed from 32 to** **16** **microsteps**

**(define TMC2130\_USTEPS\_E16)**

//#define TMC2130\_USTEPS\_XY 16 // microstep resolution for XY axes  
#define TMC2130\_USTEPS\_XY **16 (unverändert nachdem 1/8 getestet wurde)**

//#define Y\_MAX\_POS 212.5 // selftest error, therefore reduced

#define Y\_MAX\_POS **210**

//#define TMC2130\_CURRENTS\_H {16, 20, 35, 30} // default holding currents for all axes

//#define TMC2130\_CURRENTS\_R {16, 20, 35, 30} // default running currents for all axes

#define TMC2130\_CURRENTS\_H {**18**, 22, 35, 30} // default holding currents for all axes

#define TMC2130\_CURRENTS\_R {16, 20, 35, 30} // default running currents for all axes

//#define HOMING\_FEEDRATE {3000, 3000, 800, 0} // set the homing speeds (mm/min) // 3000 is also valid for stallGuard homing. Valid range: 2200 - 3000

#define HOMING\_FEEDRATE {**2200**, **2200**, 800, 0} // set the homing speeds (mm/min) // 3000 is also valid for stallGuard homing. Valid range: 2200 – 3000

//#define TMC2130\_PWM\_GRAD\_X 2 // PWMCONF  
#define TMC2130\_PWM\_GRAD\_X **4** // higher limits current overshoot and avoids squeak (stealthmode), but must be low enough to control accel/decel

//#define TMC2130\_PWM\_GRAD\_Y 2 // PWMCONF  
#define TMC2130\_PWM\_GRAD\_Y **4** // higher limits current overshoot and avoids squeak (stealthmode), but must be low enough to control accel/decel

//#define TMC2130\_SG\_THRS\_X 3 // stallguard sensitivity for X axis

//#define TMC2130\_SG\_THRS\_Y 3 // stallguard sensitivity for Y axis

#define TMC2130\_SG\_THRS\_X **2** // stallguard sensitivity for X axis

#define TMC2130\_SG\_THRS\_Y **2** // stallguard sensitivity for Y axis

**tmc2130.ccp**

//uint8\_t tmc2130\_current\_r\_home[4] = {8, 10, 20, 18};

uint8\_t tmc2130\_current\_r\_home[4] = {**12**, **12**, 20, 18};

*if (dir == 2)*

*{*

*dir = tmc2130\_get\_inv(axis)?0:1;*

*int steps = (int)step - (int)(mscnt >> shift);*

//if (steps < 0)

**if (steps > static\_cast<int>(cnt / 2))**

*{*

*dir ^= 1;*

//steps = -steps;

**steps = cnt - steps; // This can create a negative step value**

}

//if (steps > static\_cast<int>(cnt / 2))

**if (steps < 0)**

{

dir ^= 1;

steps = cnt - steps;

}

cnt = steps;

}