This is the Revision A verion of the InOut10 RoboBrick. The status of this project is finished.

InOut10 Robobrick (Revision C)

Table of Contents

This document is also available in PDF format.

1. Introduction

The InOut10 RoboBrick provides the ability to input and output 10 bits of data. The direction of each bit can be changed under program control.

2. Programming

The basic operation is to send a query to the In8 RoboBrick to read the 4 bits of data. The programmer can download a complement mask to cause any of the bits to be complemented prior to reading.

The In8 RoboBrick supports RoboBrick Interrupt Protocol. The interrupt pending bit is set whenever the the formula:

L&(~I) | H&I | R&(~P)&I | F&P&(~I)]

is non−zero, where:

- I is the current input bits XOR'ed with the complement mask (C)
- P is the previous value of I
- L is the low mask
- H is the high mask
- R is the raising mask
- F is the falling mask

and

- ~ is bit−wise complement
- | is bit−wise OR
- & is bit−wise AND

Once the interrupt pending bit is set, it must be explicitly cleared by the user.

The In8 RoboBrick supports both the standard shared commands and the shared interrupt commands in addition to the following commands:

Command	Send/ Receive	Byte Value								Discussion
		7	6	5	4	3	2	1	0	
Read Inputs Low	Send	0	0	0	0	0	0	0	0	Return low order 5−bits of input *iiiii* (after XOR'ing with complement mask)
	Receive	0	0	0	*i*	*i*	*i*	*i*	*i*	
Read Inputs High	Send	0	0	0	0	0	0	0	1	Return high order 5−bits of input *IIIII* (after XOR'ing with complement mask)
	Receive	0	0	0	*I*	*I*	*I*	*I*	*I*	
Read Complement Mask Low	Send	0	0	0	0	0	0	1	0	Return low order 5−bits of complement mask *ccccc*
	Receive	0	0	0	*c*	*c*	*c*	*c*	*c*	
Read Complement Mask High	Send	0	0	0	0	0	0	1	1	Return high order 5 bits of complement mask *CCCCC*
	Receive	0	0	0	*C*	*C*	*C*	*C*	*C*	
Read Direction Mask Low	Send	0	0	0	0	0	1	0	0	Return low order 5−bits of direction mask *ddddd*
	Receive	0	0	0	*d*	*d*	*d*	*d*	*d*	
Read Direction Mask High	Send	0	0	0	0	0	1	0	1	Return high order 5 bits of direction mask *DDDDD*
	Receive	0	0	0	*D*	*D*	*D*	*D*	*D*	
Read Raw Low	Send	0	0	0	0	0	1	1	0	Return low order 5−bits of raw input data *rrrrr* (without XOR'ing with complement mask)
	Receive	0	0	0	*r*	*r*	*r*	*r*	*r*	
Read Raw High	Send	0	0	0	0	0	1	1	1	Return high order 5−bits of raw input data *RRRRR* (without XOR'ing with complement mask)
	Receive	0	0	0	*R*	*R*	*R*	*R*	*R*	
Read Low Mask Low	Send	0	0	0	0	1	0	0	0	Return low order 5−bits of low mask *lllll*
	Receive	0	0	0	*l*	*l*	*l*	*l*	*l*	
Read Low Mask High	Send	0	0	0	0	1	0	0	1	Return high order 5−bits of low mask *LLLLL*
	Receive	0	0	0	*L*	*L*	*L*	*L*	*L*	
Read High Mask Low	Send	0	0	0	0	1	0	1	0	Return low order 5−bits of the high mask *hhhhh*
	Receive	0	0	0	*h*	*h*	*h*	*h*	*h*	
Read High Mask High	Send	0	0	0	0	1	0	1	1	Return high order 5 bits of the high mask *HHHHH*
	Receive	0	0	0	*H*	*H*	*H*	*H*	*H*	
Read Raising Mask Low	Send	0	0	0	0	1	1	0	0	Return low order 5−bits of the raising mask *rrrrr*
	Receive	0	0	0	*r*	*r*	*r*	*r*	*r*	
Read Raising Mask High	Send	0	0	0	0	1	1	0	1	Return high order 5 bits of the raising mask *RRRRR*
	Receive	0	0	0	*R*	*R*	*R*	*R*	*R*	
Read Falling Mask Low	Send	0	0	0	0	1	1	1	0	Return low order 5−bits of the falling mask *fffff*
	Receive	0	0	0	*f*	*f*	*f*	*f*	*f*	
Read Falling Mask High	Send	0	0	0	0	1	1	1	1	Return high order 5−bits of the falling mask *FFFFF*
	Receive	0	0	0	*F*	*F*	*F*	*F*	*F*	
Read Outputs Low	Send	0	0	0	1	0	0	0	0	

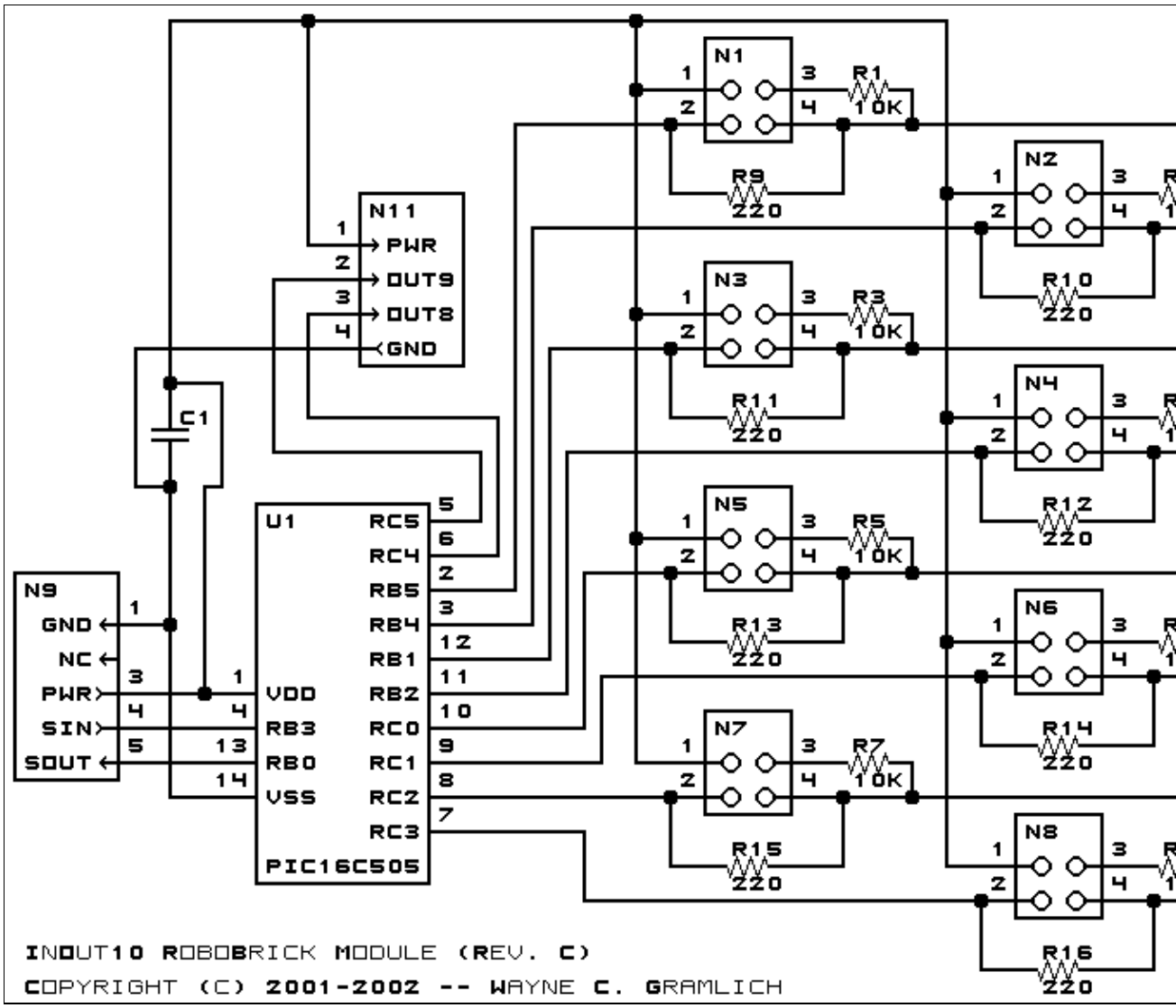
	Receive	0	0	0	*o*	*o*	*o*	*o*	*o*	Return low order 5−bits of the outputs *ooooo*
Read Outputs High	Send	0	0	0	1	0	0	0	1	Return high order 5−bits of the outputs *OOOOO*
	Receive	0	0	0	*O*	*O*	*O*	*O*	*O*	
Set Complement Mask Low	Send	0	0	0	1	0	0	1	0	Set low order 5−bits of complement mask to *ccccc*
	Send	0	0	0	*c*	*c*	*c*	*c*	*c*	
Set Complement Mask High	Send	0	0	0	1	0	0	1	1	Set high order 5 bits of complement mask to *CCCCC*
	Send	0	0	0	*C*	*C*	*C*	*C*	*C*	
Set Direction Mask Low	Send	0	0	0	1	0	1	0	0	Set low order 5−bits of direction mask to *ddddd* 1=input; 0=output
	Send	0	0	0	*d*	*d*	*d*	*d*	*d*	
Set Direction Mask High	Send	0	0	0	1	0	1	0	1	Set high order 5 bits of direction mask of *DDDDD* 1=input; 0=output
	Send	0	0	0	*D*	*D*	*D*	*D*	*D*	
Reset Outputs	Send	0	0	0	1	0	1	1	0	Set all 10 bits of outputs to 0
Reset Everything	Send	0	0	0	1	0	1	1	1	Reset all registers to 0 and set direction bits to 1 (input)
Set Low Mask Low	Send	0	0	0	1	1	0	0	0	Set low order 5−bits of low mask to *lllll*
	Send	0	0	0	*l*	*l*	*l*	*l*	*l*	
Set Low Mask High	Send	0	0	0	1	1	0	0	1	Set high order 5−bits of low mask to *LLLLL*
	Send	0	0	0	*L*	*L*	*L*	*L*	*L*	
Set High Mask Low	Send	0	0	0	1	1	0	1	0	Set low order 5−bits of the high mask to *hhhhh*
	Send	0	0	0	*h*	*h*	*h*	*h*	*h*	
Set High Mask High	Send	0	0	0	1	1	0	1	1	Set high order 5 bits of the high mask to *HHHHH*
	Send	0	0	0	*H*	*H*	*H*	*H*	*H*	
Set Raising Mask Low	Send	0	0	0	1	1	1	0	0	Set low order 5−bits of the raising mask to *rrrrr*
	Send	0	0	0	*r*	*r*	*r*	*r*	*r*	
Set Raising Mask High	Send	0	0	0	1	1	1	0	1	Set high order 5 bits of the raising mask to *RRRRR*
	Send	0	0	0	*R*	*R*	*R*	*R*	*R*	
Set Falling Mask Low	Send	0	0	0	1	1	1	1	0	Set low order 5−bits of the falling mask to *fffff*
	Send	0	0	0	*f*	*f*	*f*	*f*	*f*	
Set Falling Mask High	Send	0	0	0	1	1	1	1	1	Set high order 5−bits of the falling mask to *FFFFF*
	Send	0	0	0	*F*	*F*	*F*	*F*	*F*	
Set Outputs Low	Send	0	0	1	o	o	o	o	o	Set low order 5−bits to *ooooo*
Set Outputs High	Send	0	1	0	O	O	O	O	O	Set high order 5−bits to *OOOOO*
Set Output Bit	Send	0	1	1	*v*	*b*	*b*	*b*	*b*	Set output bit *bbbb* to *v*
Read Interrupt Bits	Send	1	1	1	0	1	1	1	1	Return the interrupt pending bit *p* and the interrupt enable bit *e*.
	Receive	0	0	0	0	0	0	*e*	*p*	
Set Interrupt Commands	Send	1	1	1	1	0	*c*	*c*	*c*	Set Interrupt Command *ccc*.
Shared Commands	Send	1	1	1	1	1	*c*	*c*	*c*	Execute Shared Command *ccc*.

3. Hardware

The hardware consists of a circuit schematic and a printed circuit board.

3.1 Circuit Schematic

The schematic for the InOut10 RoboBrick is shown below:



The parts list kept in a separate file −− inout10.ptl.

3.2 Printed Circuit Board

The printed circuit files are listed below:

inout10_back.png
: The solder side layer.

inout10_front.png

The component side layer.

inout10_artwork.png
The artwork layer.

inout10.gbl
The RS−274X "Gerber" back (solder side) layer.

inout10.gtl
The RS−274X "Gerber" top (component side) layer.

inout10.gal
The RS−274X "Gerber" artwork layer.

inout10.drl
The "Excellon" NC drill file.

inout10.tol
The "Excellon" tool rack file.

3.3 Construction Instructions

The construction Instructions are located in a separate file to be a little more printer friendly.

4. Software

The InOut10 software is available as one of:

inout10.ucl
The µCL source file.

inout10.asm
The resulting human readable PIC assembly file.

inout10.lst
The resulting human readable PIC listing file.

inout10.hex
The resulting Intel® Hex file that can be fed into a PIC12C5xx programmer.

The InOut10 test suite is available as one of:

inout10_test.ucl
The µCL source file.

inout10_test.asm
The resulting human readable PIC assembly file.

inout10_test.lst
The resulting human readable PIC listing file.

inout10_test.hex
The resulting Intel® Hex file that can be fed into a PIC16F84 programmer.

5. Issues

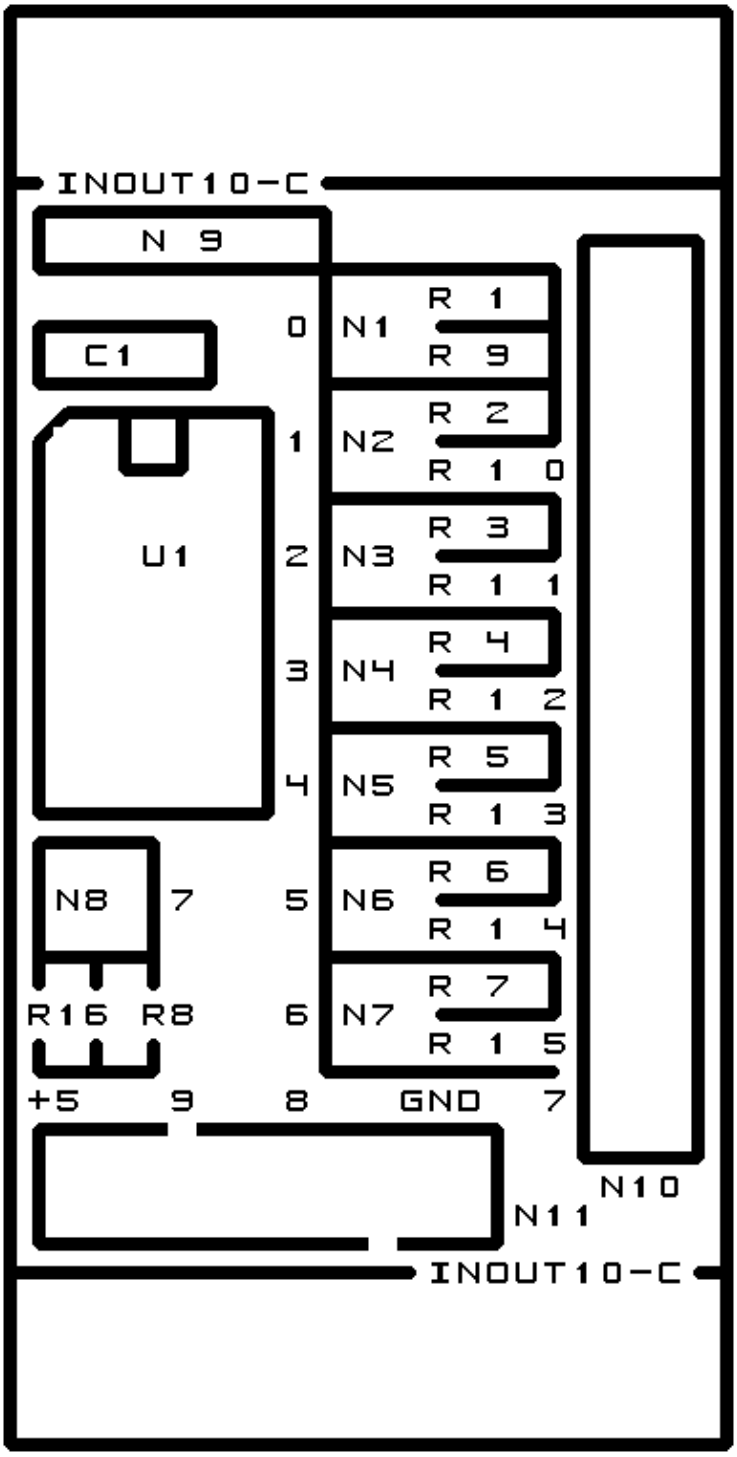
The following fabrication issues came up:

- Think about adding some in−line 220 Ohm resistors for powering LED's.
- Think about adding some 10K Ohm pull up resistors for bump sensors.

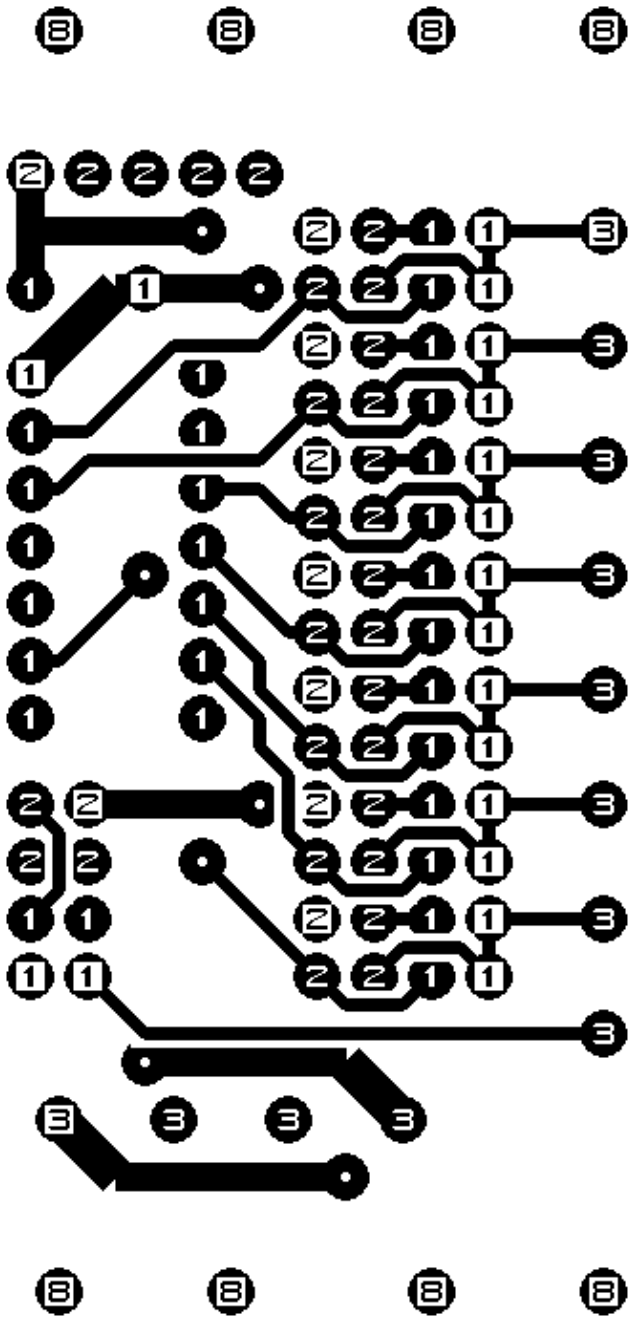
A. Appendix A: Parts List

```
# Parts list for InOut10 RoboBrick (Rev. C)
#
C1: Capacitor10pF − 10 pF Ceramic Capacitor [Jameco: 15333]
# N1−7 can be replaced by 2x14 Male Header [Jameco: 203852]
# N8 still needs to be 2x2 Male Header
N1−8: Header2x2.InOut10 − 2x2 Male Header [4/80 Jameco: 117196]
N9: Header1x5.RBSlave − 1x5 Male Header [5/40 Jameco: 160881]
N10: TerminalStrip8.InOut10 − 8 Junction Terminal Strip [4 Jameco: 189675]
N11: TerminalStrip4.InOut10 − 4 Junction Terminal Strip [2 Jameco: 189675]
N12−27: ShortingBlock − Shorting Block .1" [Jameco: 22023]
R1−8: Resistor10K.Vertical − 10K Ohm 1/4 resistor [Jameco: 29911]
R9−16: Resistor220.Vertical ResistorVertical [Jameco: 30470]
U1: PIC16C505.InOut10 − Microchip PIC16C505 [Digikey: PIC16C505−04/P−ND]
# Each RoboBrick should have a connector cable:
N28−29: CableHeader1x5 − 1 x 5 Female Shell [Jameco: 163686]
N30−37: CablePinFemale − Female Pin [Jameco: 100765]
N38−39: CablePinMale − Male Pin [Jameco: 145357]
```

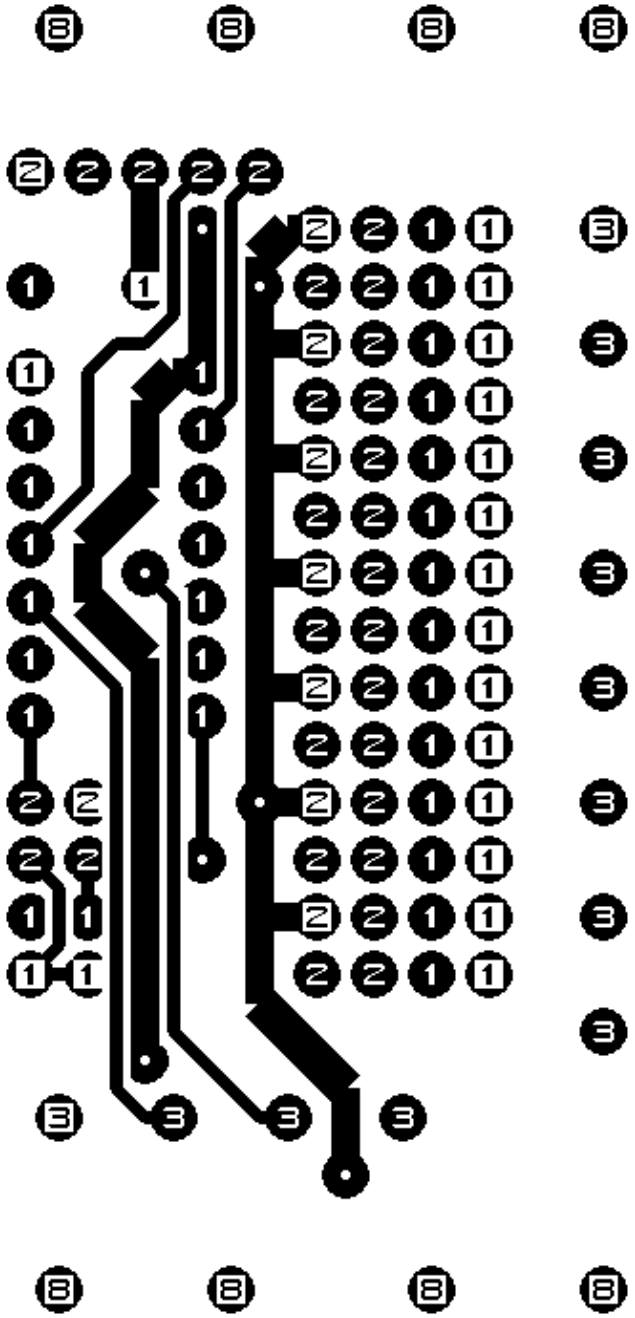
B. Appendix B: Artwork Layer



C. Appendix C: Back (Solder Side) Layer



D. Appendix D: Front (Component Side) Layer



E. Appendix E: Construction Instructions

The instruction steps for building the InOut10 (Rev. C) RoboBrick are listed below:

1. Orient the board vertically. By convention the upper edge is north, the lower edge is south, the left edge is west, and the right edge is east. Orient the board so that N9 is in the north west corner. [step1.jpg]
2. Take a 1×5 male header and using some diagonal cutters, snip off pin 2 using the diagram below:

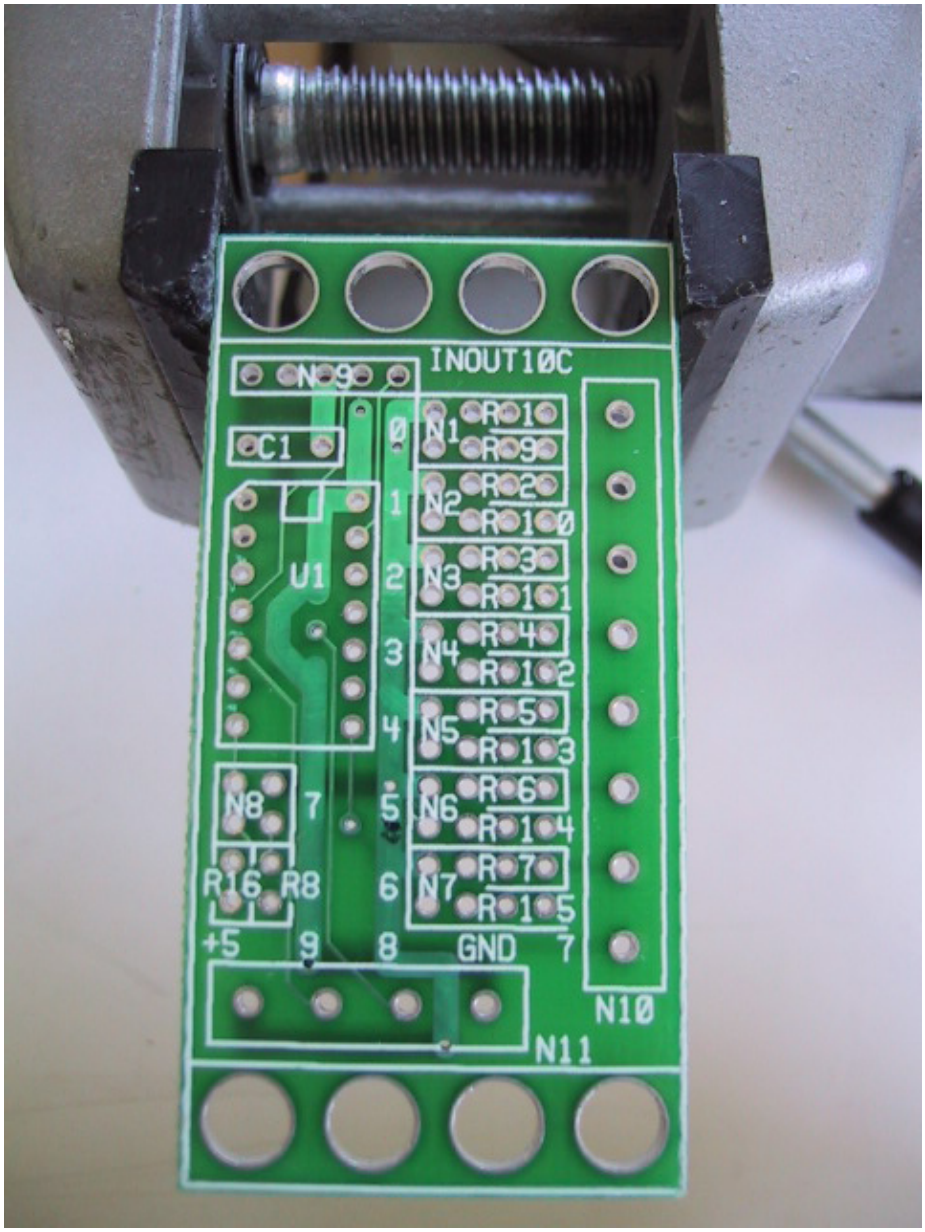
1	X	3	4	5

   Pin 2 is the pin marked with an `X'. Install the 1×5 male header into N9 with pin 1 to the west. First, solder only one pin and verify that the connector is pointing straight up. If not, use the soldering iron to melt solder around the pin you just solder and reposition the header so that it is pointing straight up. When you are happy with the position, solder in the remaining 4 pins. [step2.jpg]
3. Take a 2×14 male header, orient it so that it is vertical, and insert it into N1 through N7. As in the previous instruction, solder 1 pin, verify that it is completely vertical, and solder the remaining 13 pins in. [step3.jpg]
4. Take a 2×2 male header, insert it into N8, solder one pin, verify position, and solder in the remaining 3 pins. [step4.jpg]
5. Find a 10K resistor (Brown Black Orange). Take the lead that is closest to the brown band and bend it around 180 degrees. Insert the resistor into R1 with the resistor to the east and the bend lead to the west. Turn the board over, and spread the leads a little to keep it from coming out. Solder one lead. Now turn it over to verify that the position is acceptable. If not, re−heat the lead you just soldered and get it positioned to your liking. Solder in the remaining lead. Snip off the excess leads sticking out the back. [step5.jpg]
6. Repeat the preceding instruction with 6 more 10K resistors and insert them into R2 through R7. [step6.jpg]
7. Install the last 10K resistor in R8. The resistor base should go into the south hole and the the bent lead should go into the north hole. [step7.jpg]
8. Next, find a 220 Ohm resistor (Red, Red, Brown). Bend the lead closes to the red band over 180 degrees. Install the resistor into R9 with the resistor to the west and the bent lead to the east (i.e. the opposite of R1.) Spread the leads, solder 1 lead, verify position, solder the remaining lead, and snip of the excess leads. [step8.jpg]
9. Find 6 more 220 Ohm resistors (Red, Red, Brown) and install them into R9 through R15 using the same process as the previous step. [step9.jpg]
10. Find the last 220 Ohm resistor (Red, Red, Brown) and install it into R16 using the same process as the previous two steps. [step10.jpg]
11. Find the 4−terminal terminal strip and insert it into N11. If all you can find is some 2−terminal strips, a 4−terminal strip can be assembled from two 2−terminal strip by sliding them together. Make sure that the wire holes point towards the south. Solder in 1 pin, verify position, and solder the remaining pins. [step11.jpg]
12. Find the 8−terminal terminal strip and insert it into N10. If all you can find is some 2−terminal strips, an 8−terminal strip can be assembled from four 2−terminal strip by sliding them together. Make sure that the wire holes point towards the east. Solder in 1 pin, verify position, and solder in the remaining pin. [step12.jpg]
13. Find the 10pF capacitor and install it at C1. Snip off the excess leads. [step13.jpg]
14. Find the pre−programmed PIC16C509 and insert it into U1 with pin 1 pointing up. Solder in 1 pin, verify position, and solder in the remaining pins. The picture marks the notch in the chip with a little white marker. [step14.jpg]

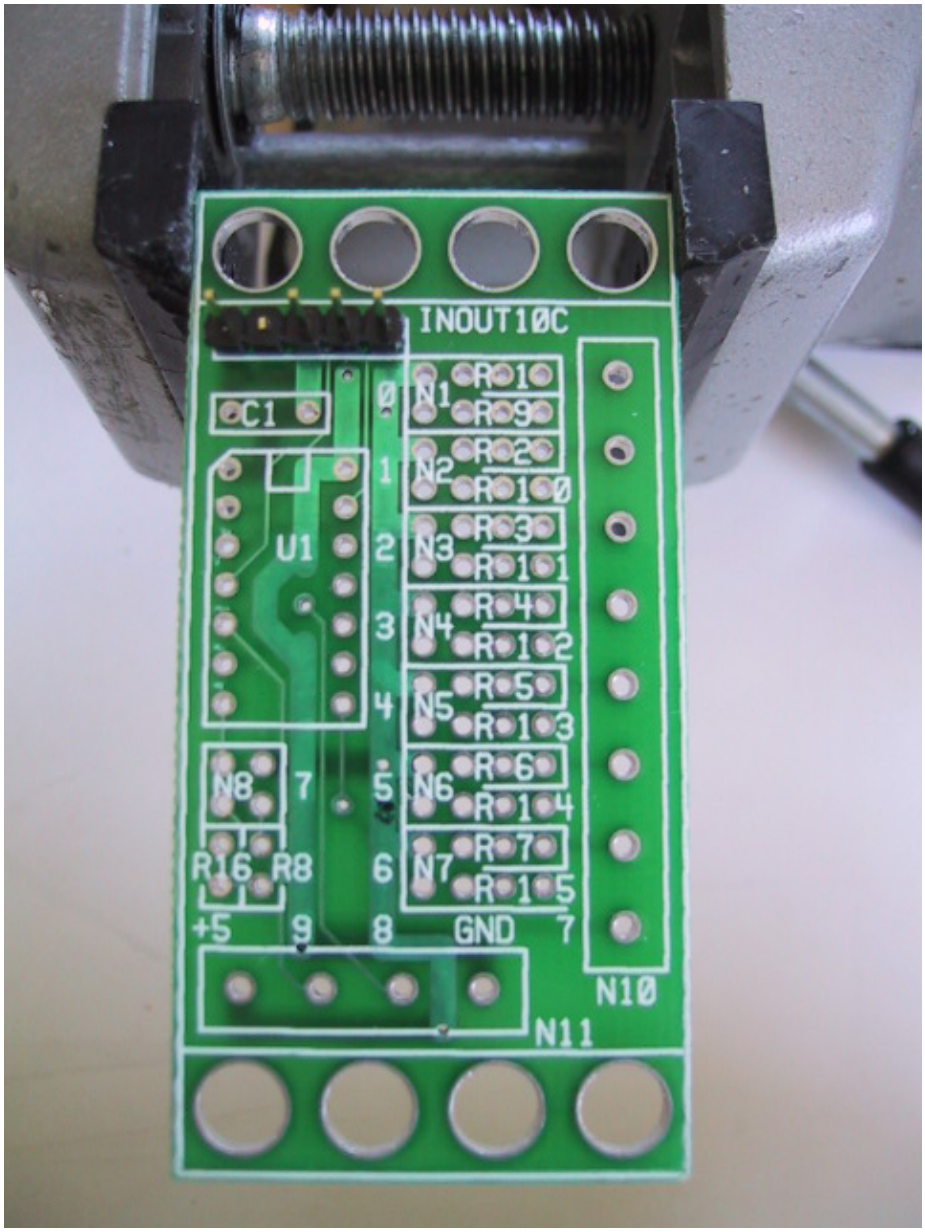
15. Install the shorting blocks on N1 through N16. [{Missing Picture}]

The assembly of the InOut10 (Rev. C) RoboBrick is complete.

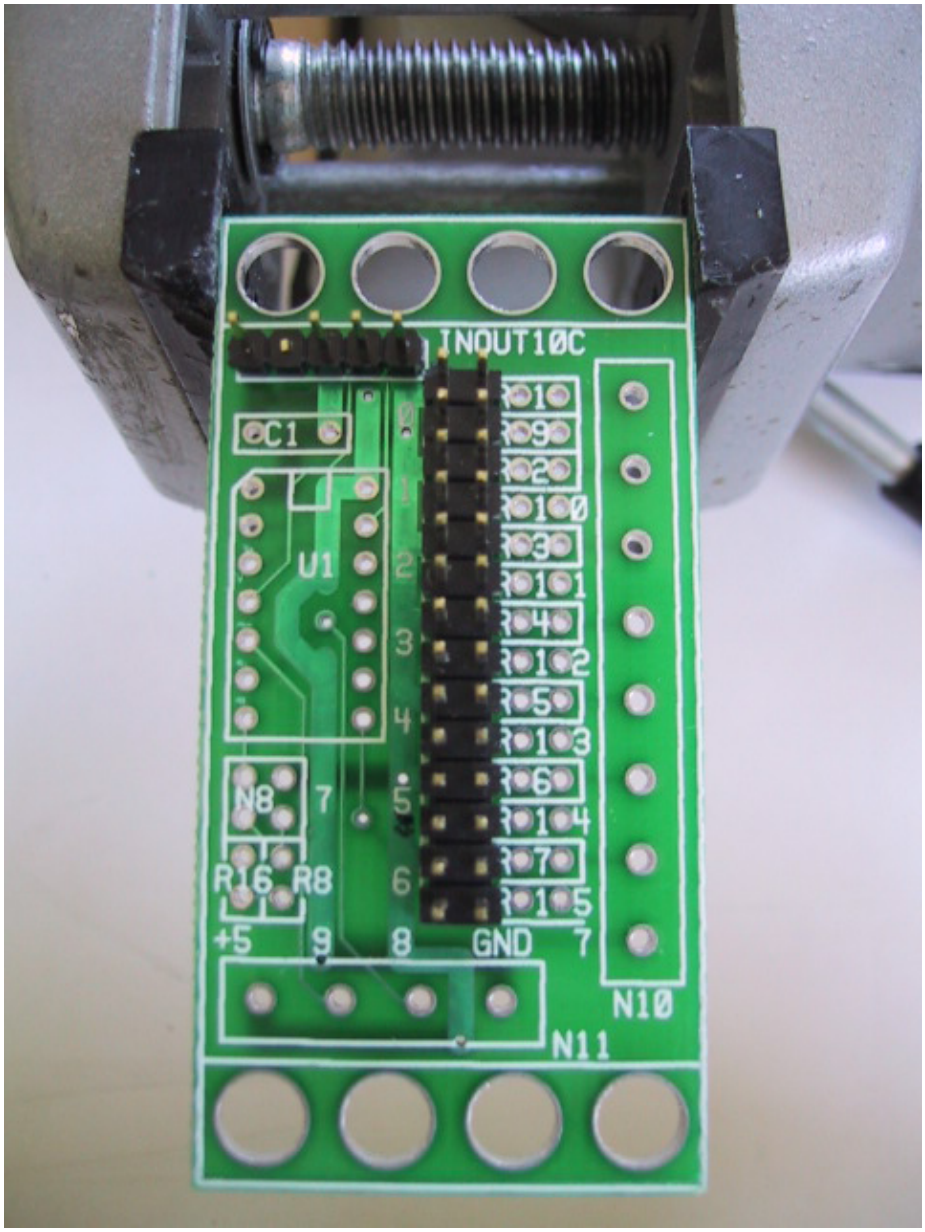
Step 1


INOUT10C
N9
C1
U1
N1
R1
R9
N2
R2
R10
N3
R3
R11
N4
R4
R12
N5
R5
R13
N6
R6
R14
N7
R7
R15
N8
R16
R8
+5
GND
N10
N11

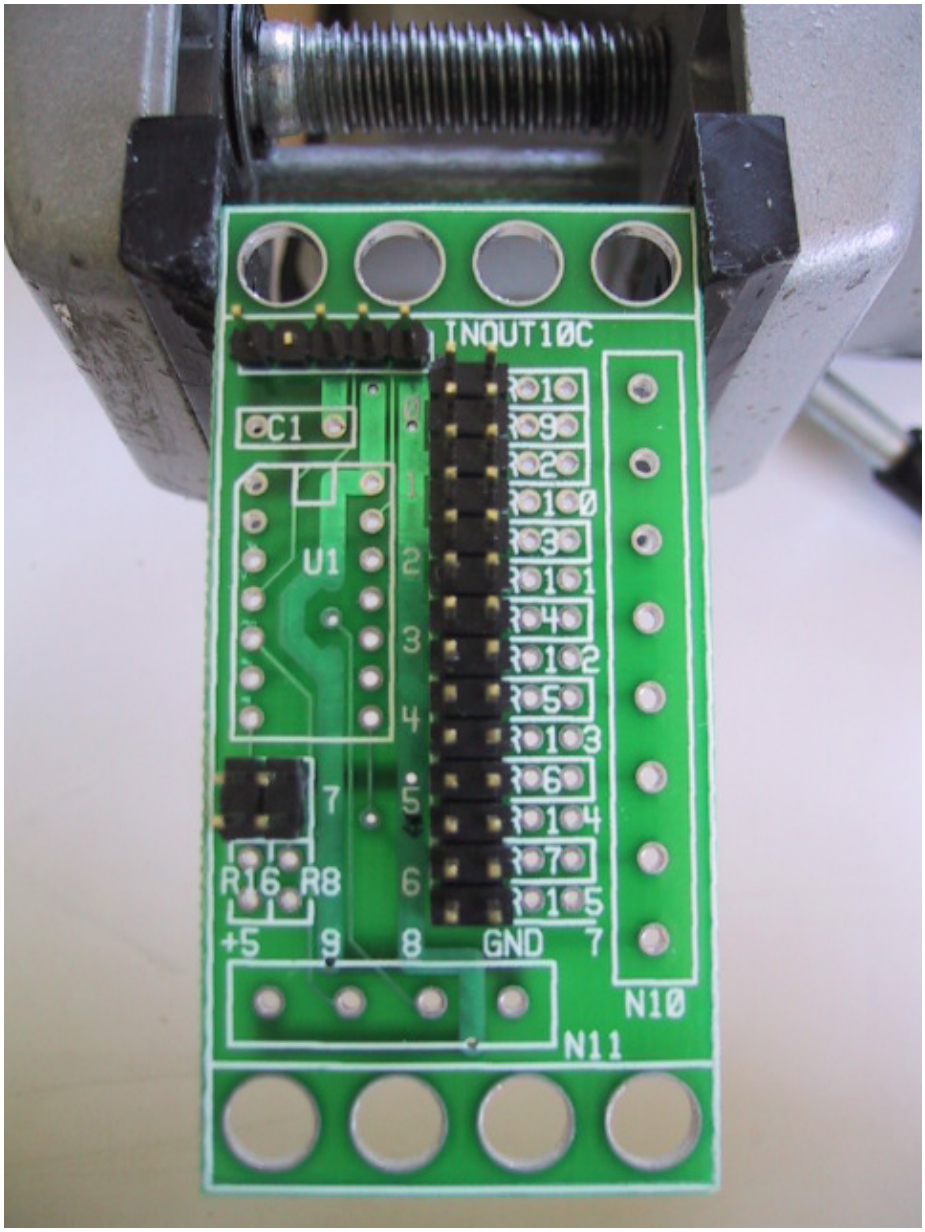
Step 2


INOUT10C
C1
U1
N1
N2
N3
N4
N5
N6
N7
N8
R1
R9
R2
R10
R3
R11
R4
R12
R5
R13
R6
R14
R7
R15
R16
R8
+5
GND
N10
N11

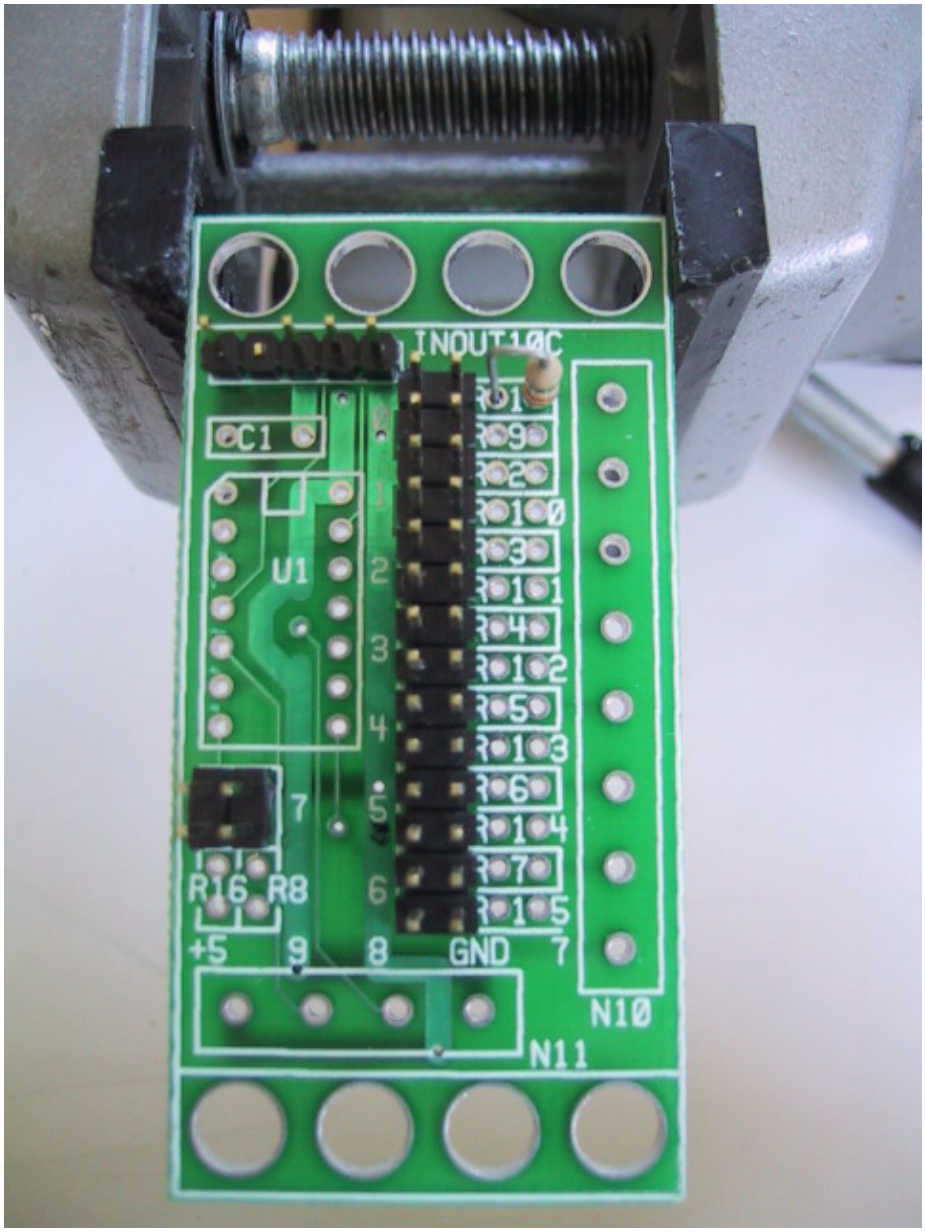
Step 3



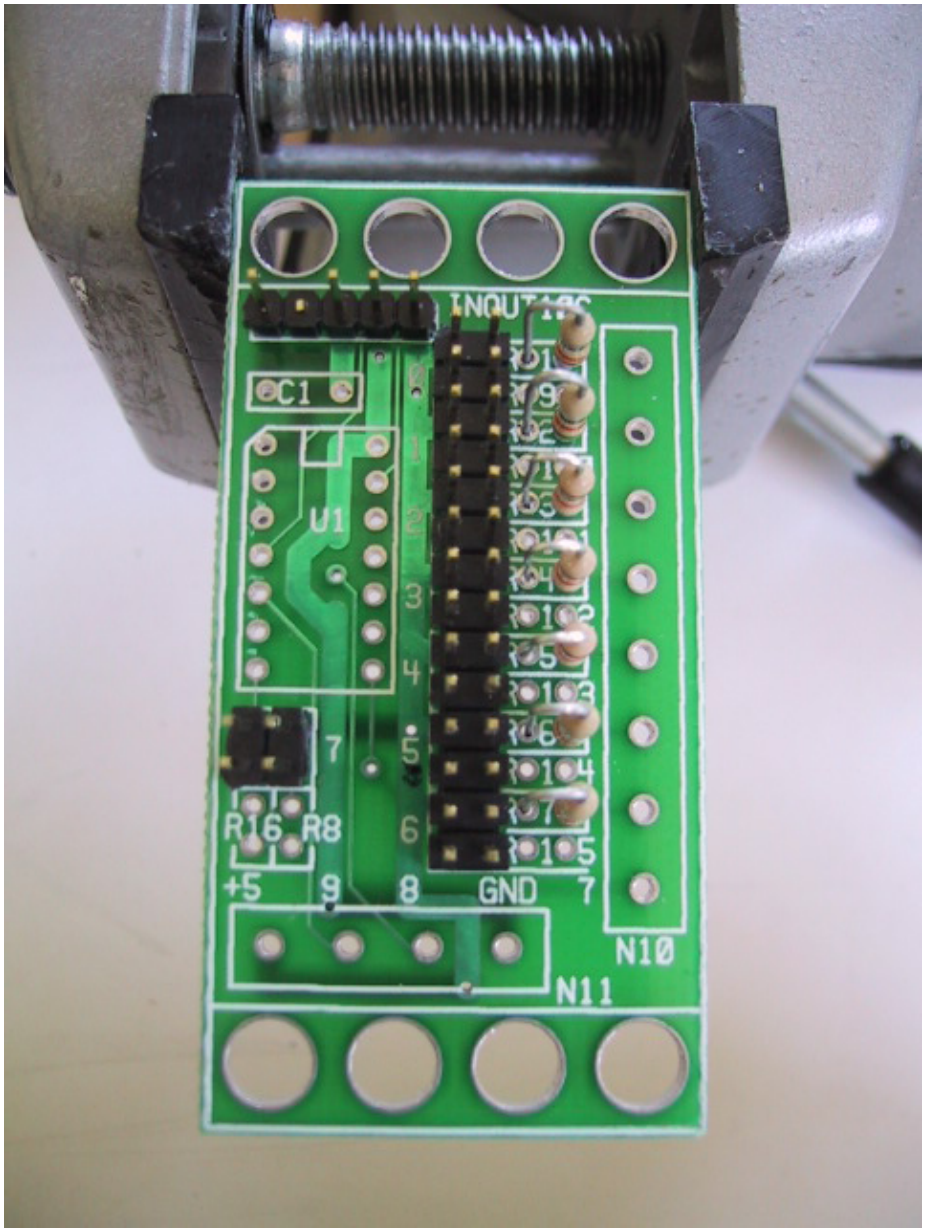
Step 4



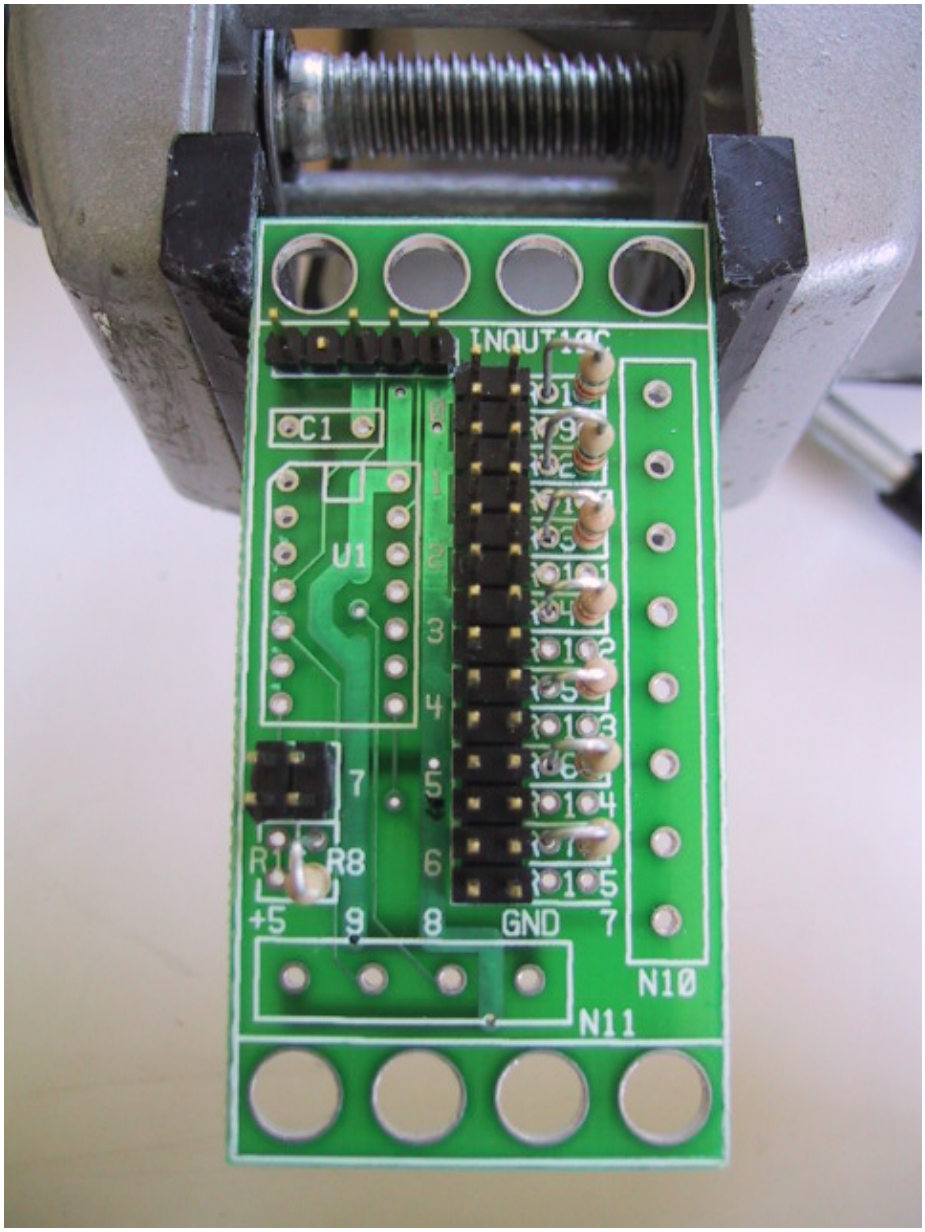
Step 5



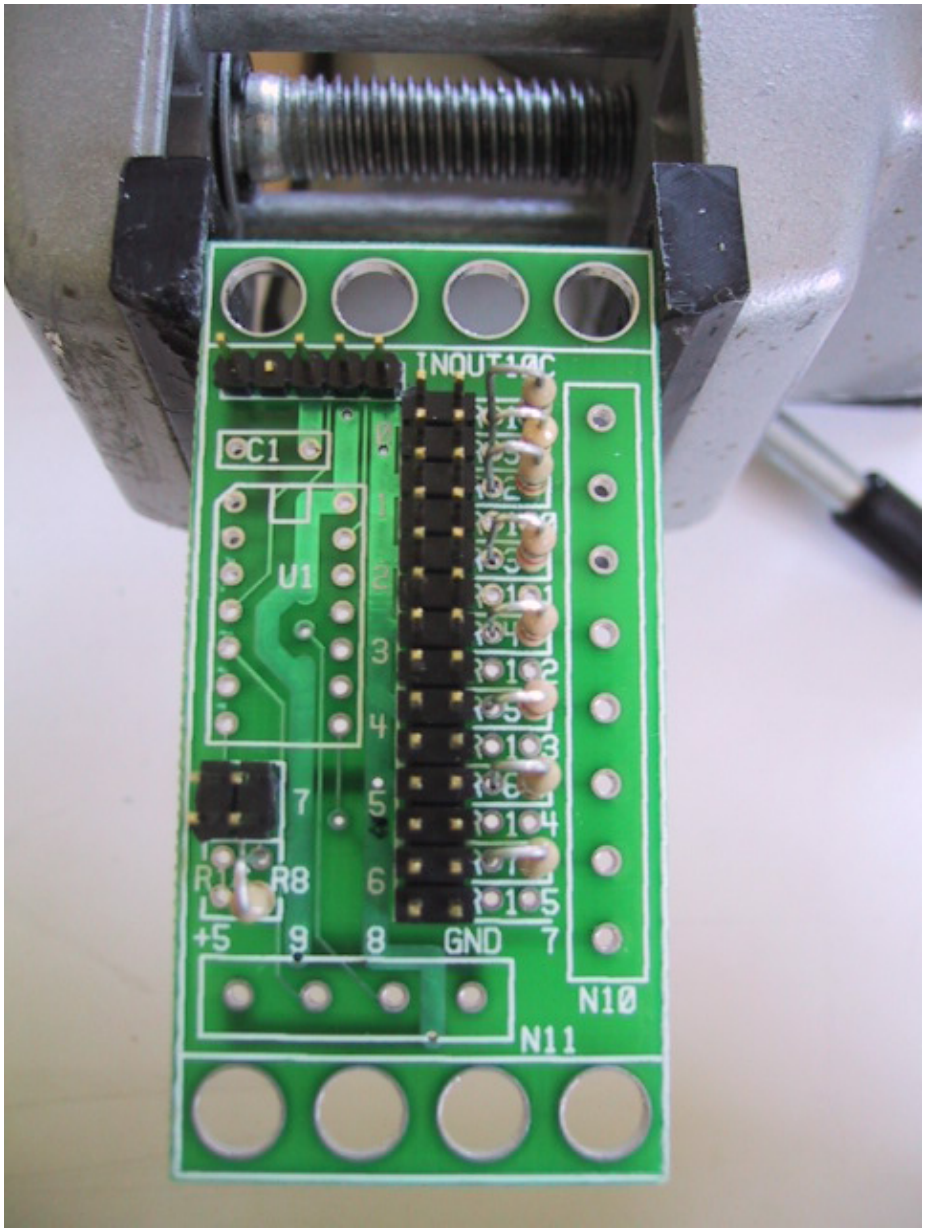
Step 6



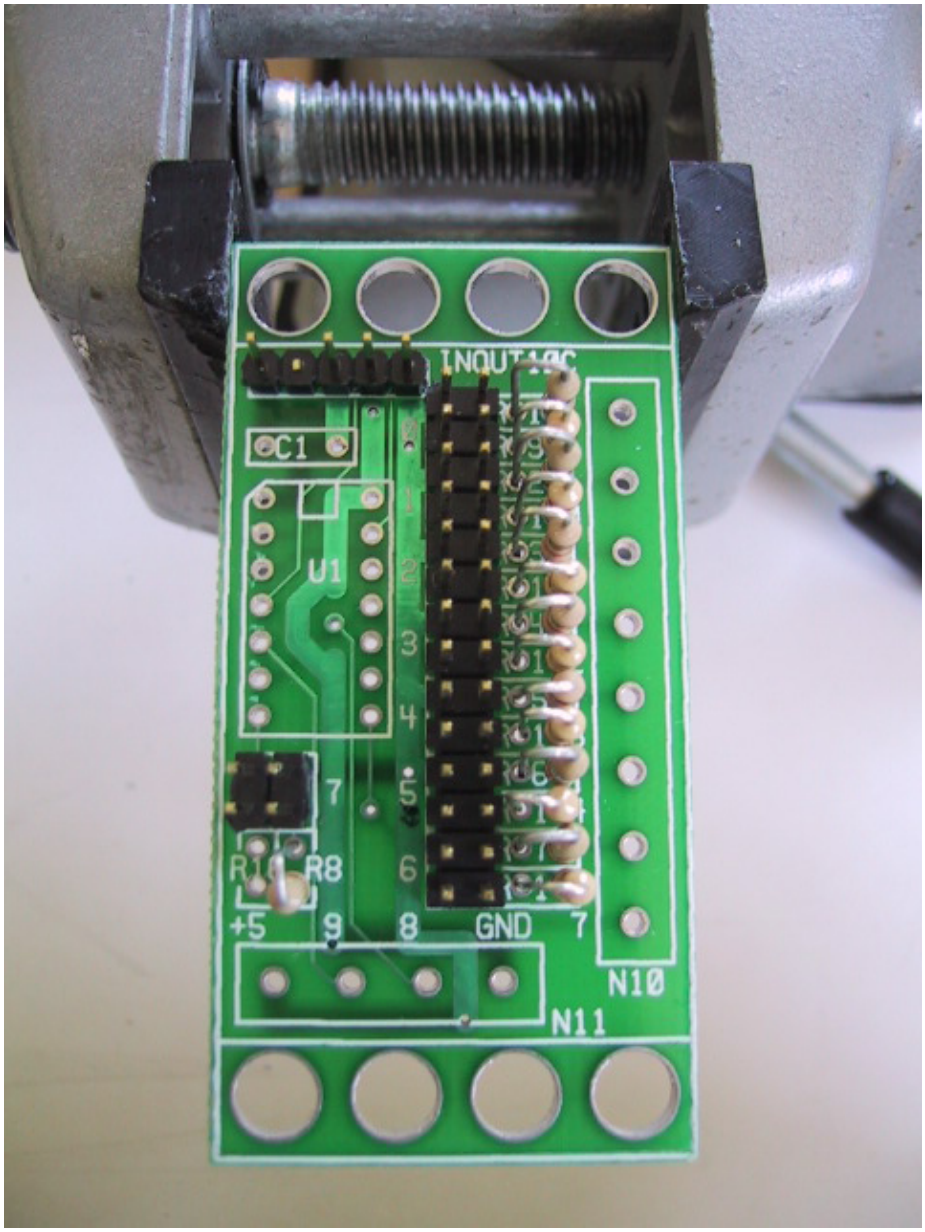
Step 7



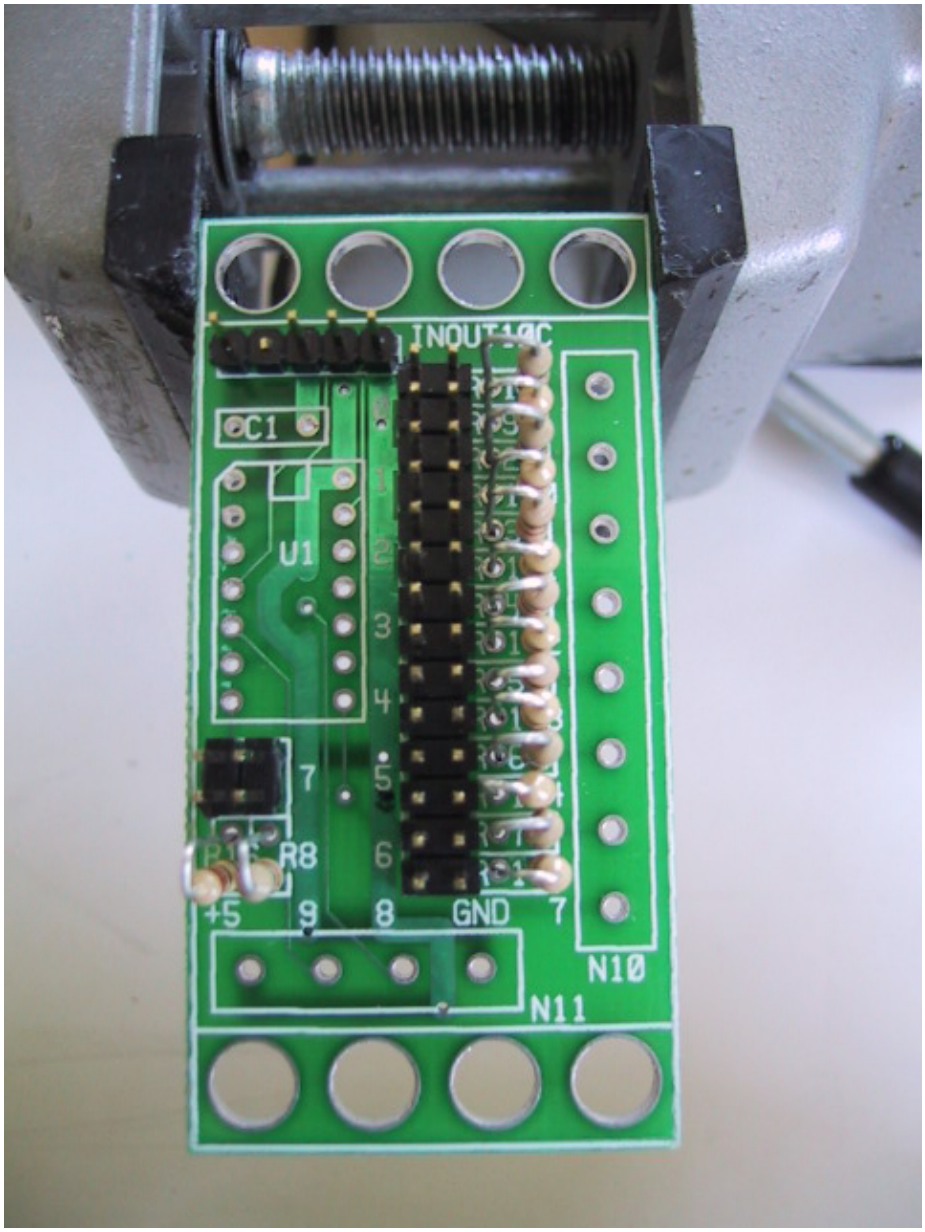
Step 8



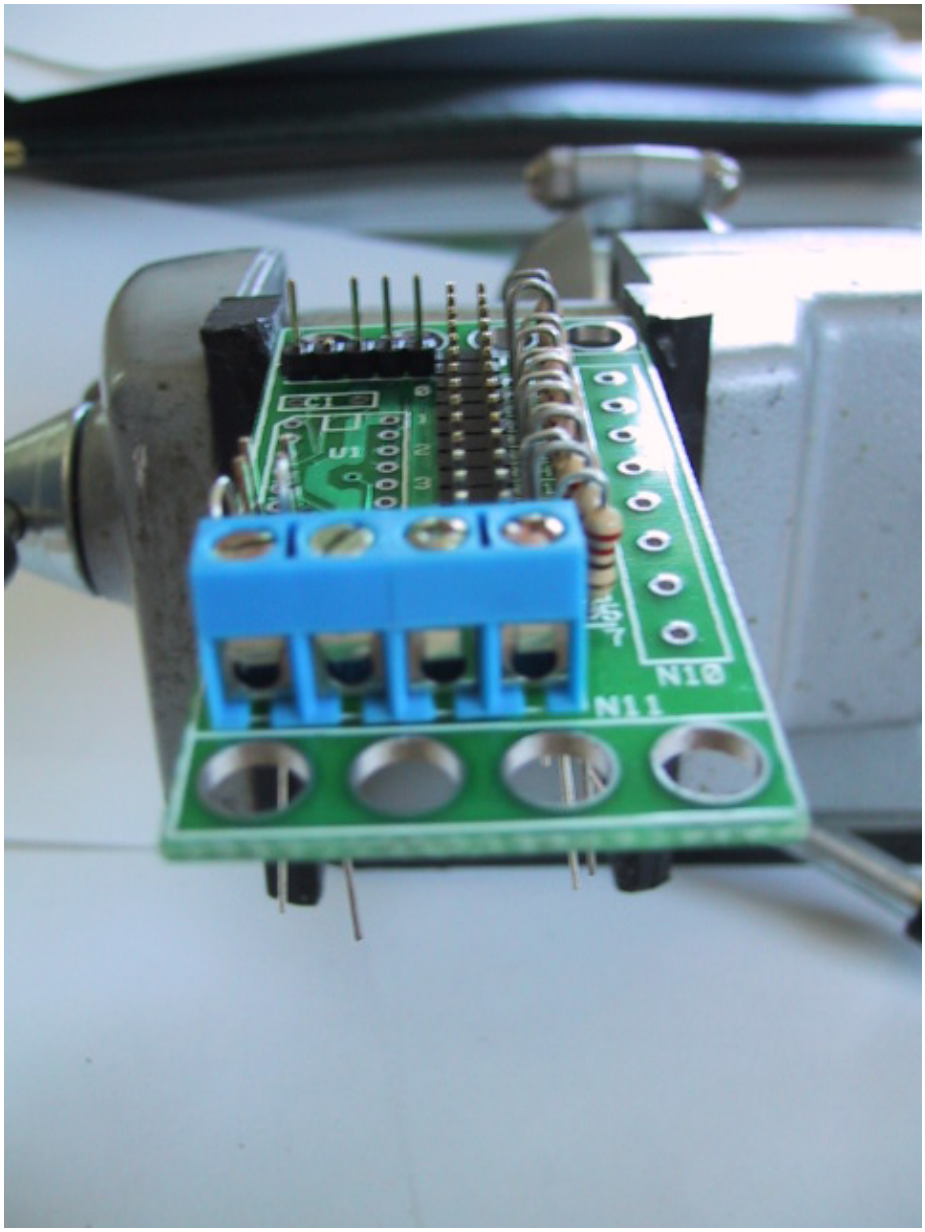
Step 9



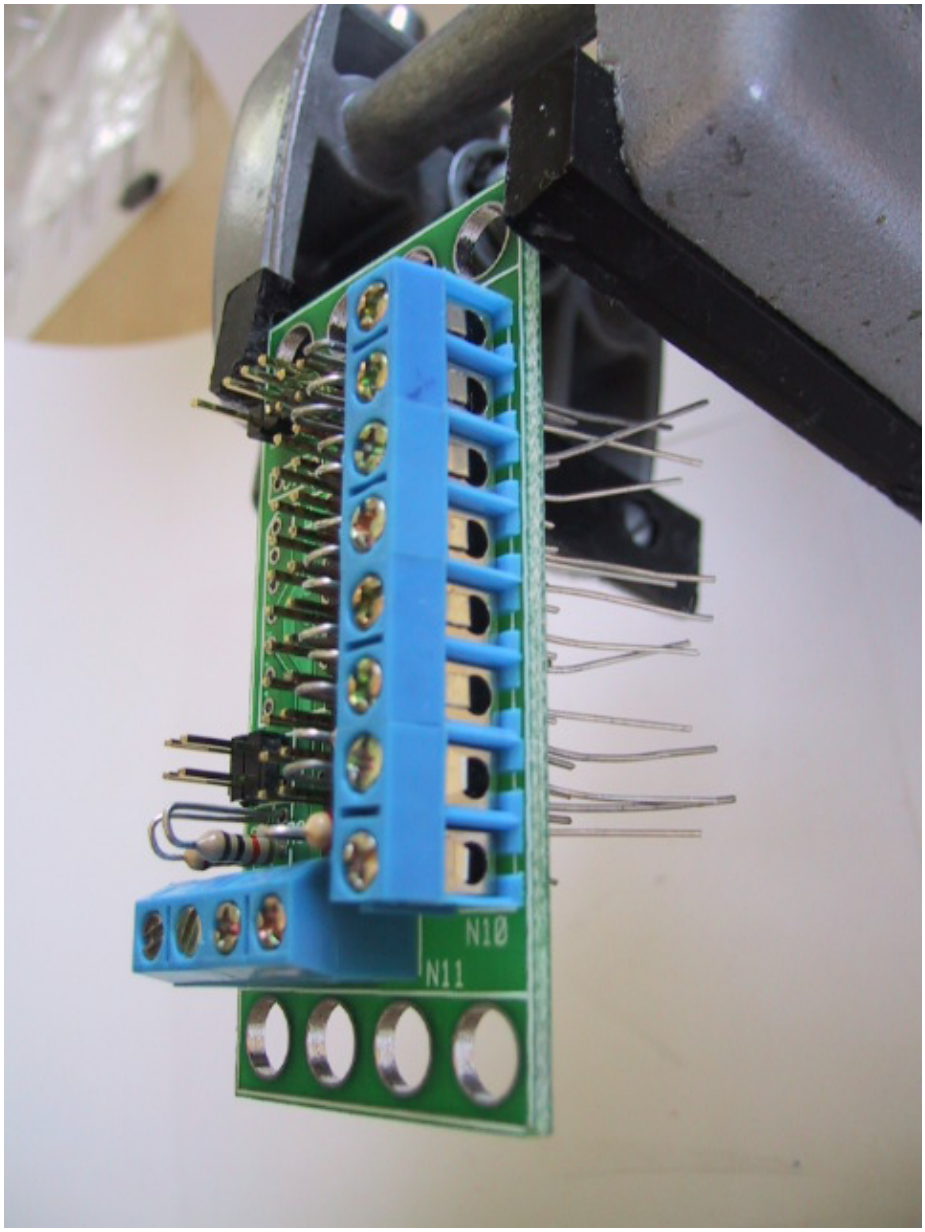
Step 10



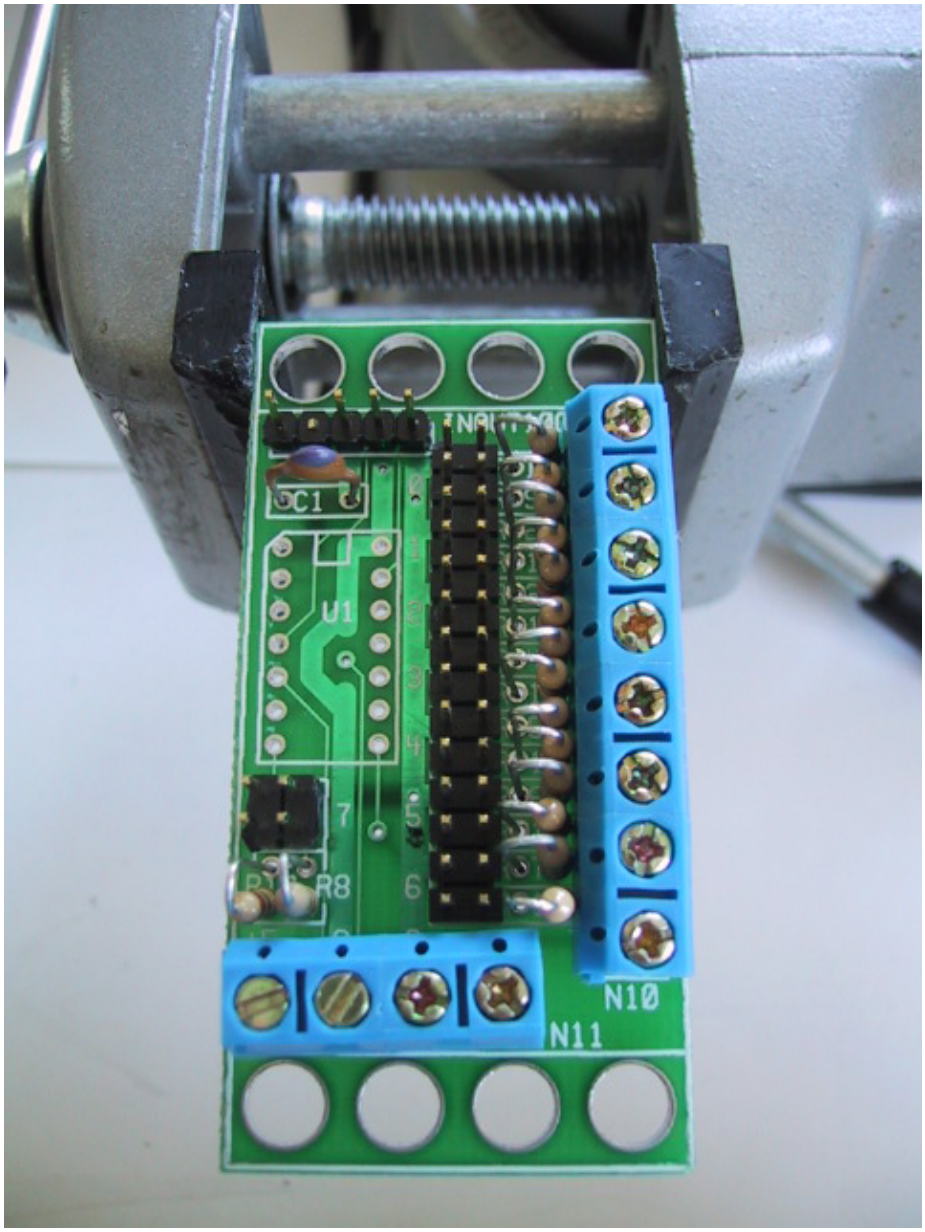
Step 11



Step 12



Step 13



Step 14