This is the Revision B verion of the Light4 RoboBrick. The status of this project is finished.

Light4 Robobrick (Revision B)

Table of Contents

This document is also available in PDF format.

- 1. Introduction
- 2. Programming
- 3. Hardware
    - ♦ 3.1 Circuit Schematic
    - ♦ 3.2 Printed Circuit Board
    - ♦ 3.3 Construction Instructions
- 4. Software
- 5. Issues

1. Introduction

The Light4 RoboBrick can connect to up to 4 Photo Sensors (combined light emitter with photodetector.) The inputs are done using analog to digital converters rather than just binary inputs. There are 4 potentiometers to control the current throught the light emitters and 4 pententionmeters to control the gain of the returned signal.

2. Programming

The Light4 RoboBrick is continuously reading the analog inputs from its four A/D pins. The controlling program can just read the results of the digital conversion, or it can have the result down converted into a single binary bit. Each pin has has a threshold high and threshold low register that is used for the down conversion. Whenever the digital conversion exceeds the high threshold register, the down coversion results in a 1. Whenever the digital conversion is lower than the low threshold register, the down conversion results in a 0. A hysterisis effect can be introduced by having some spread between the high and low threshold values.

After the down coversions to binary bits, the result is 4−bits of binary data. A complement mask can be used to selectively invert individual bits in the 4−bit data.

The Light4 RoboBrick supports RoboBrick Interrupt Protocol for those lines that are being used as inputs. 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.

In addition to the common shared commands and the shared interrupt commands, the Light4 RoboBrick supports following commands:

Command	**Send/ Receive**	**Byte Value**								**Discussion**
		7	**6**	**5**	**4**	**3**	**2**	**1**	**0**	
Read Pin	Send	0	0	0	0	0	0	*b*	*b*	Read pin *bb* and respond with 8−bit value *vvvvvvvv*
	Receive	*v*	*v*	*v*	*v*	*v*	*v*	*v*	*v*	
Read Binary Values	Send	0	0	0	0	0	1	0	0	Return the binary values *abcd* (after XOR'ing with complement mask)
	Receive	0	0	0	0	*a*	*b*	*c*	*d*	
Read Raw Binary	Send	0	0	0	0	0	1	0	1	Return the raw binary values *abcd* (no XOR with complement mask)
	Receive	0	0	0	0	*a*	*b*	*c*	*d*	
Reset	Send	0	0	0	0	0	1	1	0	Reset everything to zero
Read Complement Mask	Send	0	0	0	0	1	0	0	0	Return the complement mask *cccc*
	Receive	0	0	0	0	*c*	*c*	*c*	*c*	
Read High Mask	Send	0	0	0	0	1	0	0	1	Return the high mask *hhhh*
	Receive	0	0	0	0	*h*	*h*	*h*	*h*	
Read Low Mask	Send	0	0	0	0	1	0	1	0	Return the high mask *llll*
	Receive	0	0	0	0	*l*	*l*	*l*	*l*	
Read Raising Mask	Send	0	0	0	0	1	0	1	1	Return the raising mask *rrrr*
	Receive	0	0	0	0	*r*	*r*	*r*	*r*	
Read Falling Mask	Send	0	0	0	0	1	1	0	0	Return the falling mask *ffff*
	Receive	0	0	0	0	*f*	*f*	*f*	*f*	
Read High Threshold	Send	0	0	0	1	0	0	*b*	*b*	Return high threshold for pin *bb* of *hhhhhhhh*
	Receive	*h*	*h*	*h*	*h*	*h*	*h*	*h*	*h*	
Read Low Threshold	Send	0	0	0	1	0	1	*b*	*b*	Return low threshold for pin *bb* of *llllllll*
	Receive	*l*	*l*	*l*	*l*	*l*	*l*	*l*	*l*	
Set High Threshold	Send	0	0	0	1	1	0	*b*	*b*	Set high threshold for pin *bb* to *hhhhhhhh*
	Send	*h*	*h*	*h*	*h*	*h*	*h*	*h*	*h*	
Set Low Threshold	Send	0	0	0	1	1	1	*b*	*b*	Set low threshold for pin *bb* to *llllllll*
	Send	*l*	*l*	*l*	*l*	*l*	*l*	*l*	*l*	
Set Complement Mask	Send	0	0	1	0	*c*	*c*	*c*	*c*	Set complement mask to *cccc*
Set High Mask	Send	0	1	0	0	*h*	*h*	*h*	*h*	Set high mask to *hhhh*
Set Low Mask	Send	0	1	0	1	*l*	*l*	*l*	*l*	Set low mask to *llll*
Set Raising Mask	Send	0	1	1	0	*r*	*r*	*r*	*r*	Set raising mask to *rrrr*
Set Falling Mask	Send	0	1	1	1	*f*	*f*	*f*	*f*	Set falling mask to *ffff*
Read Interrupt Bits	Send	1	1	1	0	1	1	1	1	

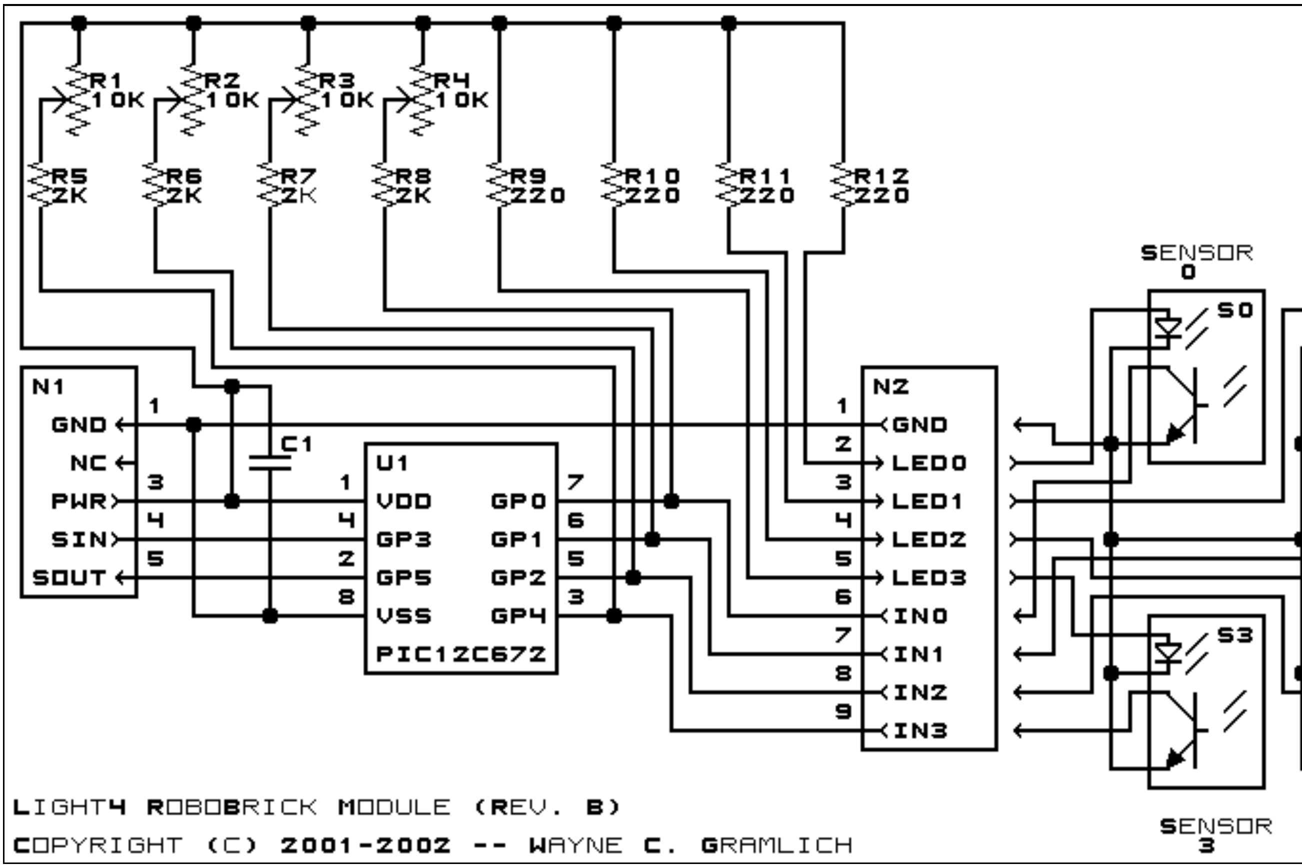
	Receive	0	0	0	0	0	0	*e*	*p*	Return the interrupt pending bit *p* and the interrupt enable bit *e*.
Set Interrupt Commands	Send	1	1	1	1	0	*c*	*c*	*c*	Set Interrupt Command *ccc*.
Shared Commands	Send	1	1	1	1	1	*c*	*c*	*c*	Execute common shared command *ccc*

3. Hardware

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

3.1 Circuit Schematic

The schematic for the Light4 RoboBrick is shown below:



The parts list kept in a separate file −− light4.ptl.

3.2 Printed Circuit Board

The printed circuit board files are listed below:

light4_back.png
    The solder side layer.
light4_front.png
    The component side layer.

light4_artwork.png
The artwork layer.
light4.gbl
The RS−272X "Gerber" back (solder side) layer.
light4.gtl
The RS−272X "Gerber" top (component side) layer.
light4.gal
The RS−272X "Gerber" artwork layer.
light4.drl
The "Excellon" NC drill file.
light4.tol
The "Excellon" tool rack file.

3.3 Construction Instructions

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

4. Software

The Light4 software is available as one of:

light4.ucl
The µCL source file.
light4.asm
The resulting human readable PIC assembly file.
light4.lst
The resulting human readable PIC listing file.
light4.hex
The resulting Intel® Hex file that can be fed into a PIC programmer.

The Light4 test software is available as one of:

light4_test.ucl
The µCL source file.
light4_test.asm
The resulting human readable PIC assembly file.
light4_test.lst
The resulting human readable PIC listing file.
light4_test.hex
The resulting Intel® Hex file that can be fed into a PIC programmer.

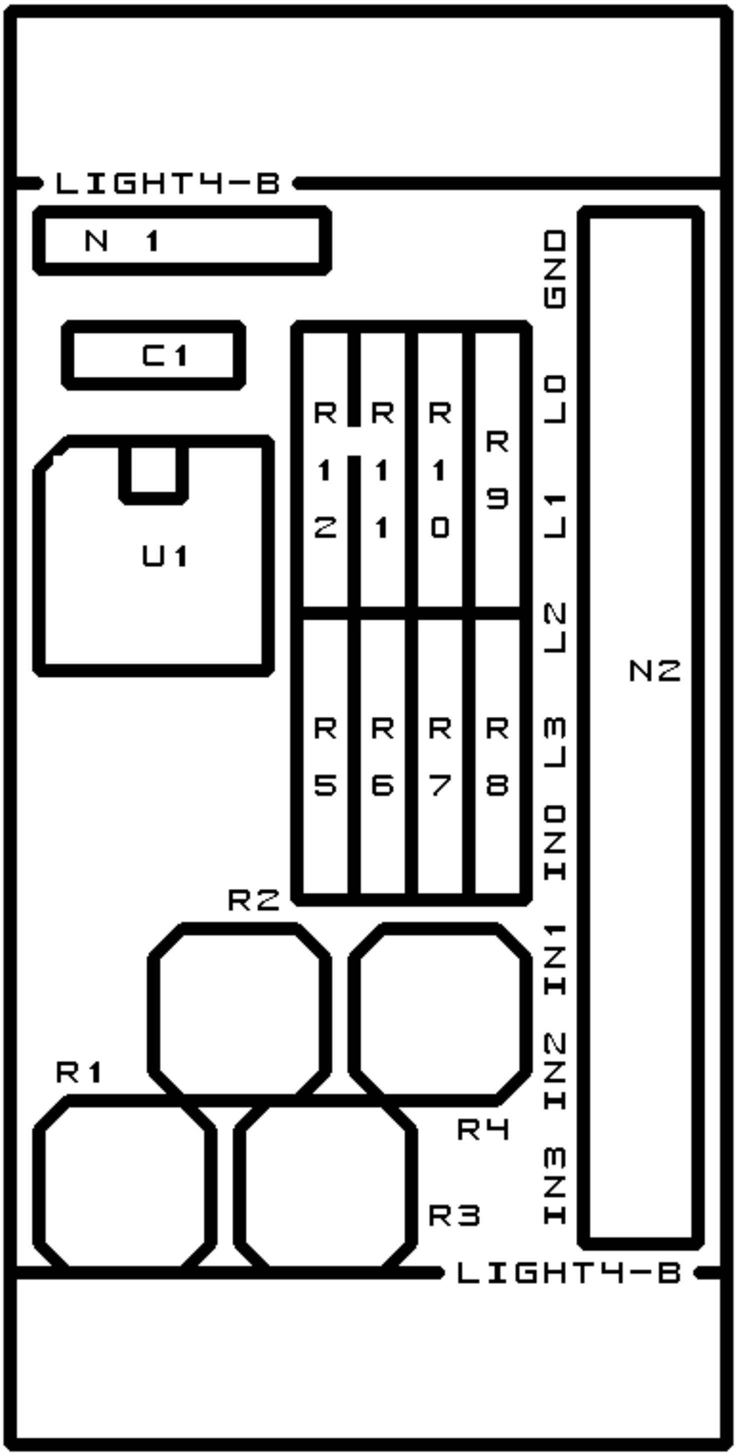
5. Issues

Any fabrication issues will be listed here.

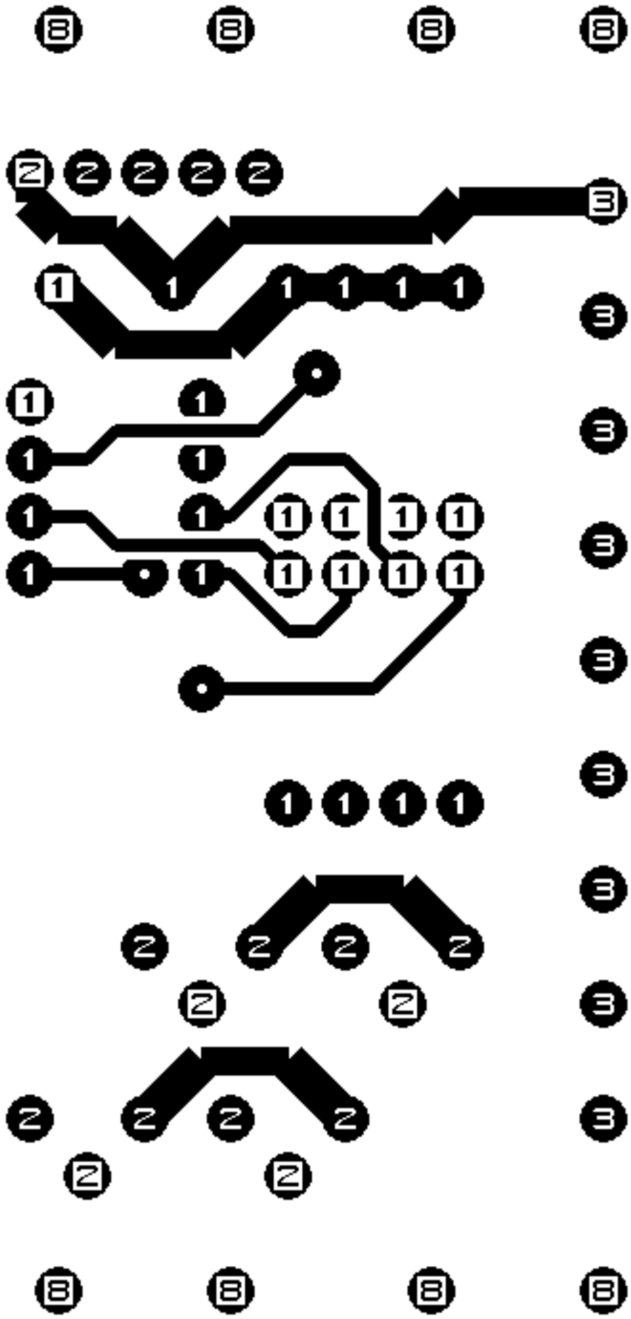
A. Appendix A: Parts List

```
# Parts list for Light4 RoboBrick (Rev. B)
#
C1: Capacitor10pF − 10 pF Ceramic Capacitor [Jameco: 15333]
N1: Header1x5.RBSlave − 1x5 Male Header [5/40 Jameco: 160881]
N2: TerminalStrip9.Light4 − 9 Junction Terminal Strip [3 Jameco: 189667]
R1−4: ResistorTrimPot10K.Light4 − 10K Ohm 1/2 Watt Potentiometer [Jameco: 96719]
R5−8: Resistor2K − 2K Ohm 1/4 Watt Resistor [Jameco: 30277]
R9−12: Resistor220 − 220 Ohm 1/4 Watt Resistor [Jameco: 30470]
U1: PIC12C672.Light4 − Microchip PIC12C672 [Digikey: PIC12C672−04/P−ND]
# The reflective photosensors are not mounted on the PCB:
U2−5: QRB1134 − IR Reflective Photosensor [Digikey: QRB1134−ND]
# Each RoboBrick should have a connector cable:
N3−4: CableHeader1x5 − 1 x 5 Female Shell [Jameco: 163686]
N5−12: CablePinFemale − Female Pin [Jameco: 100765]
N13−14: CablePinMale − Male Pin [Jameco: 145357]
```

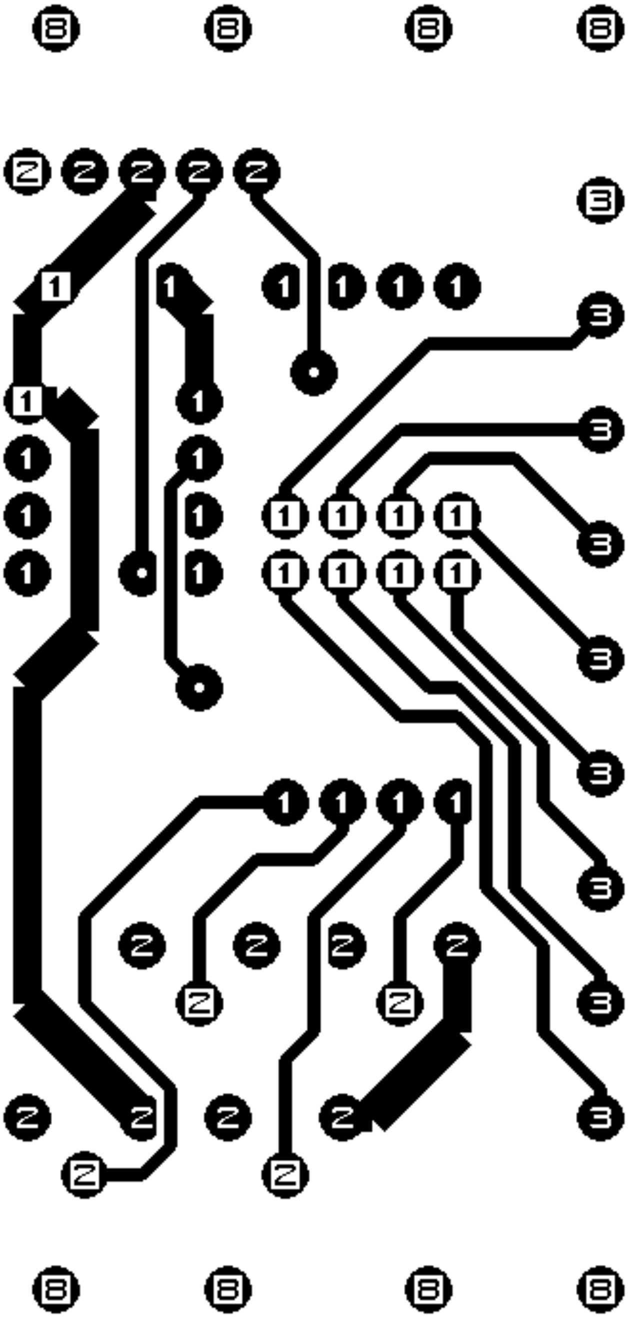
B. Appendix B: Artwork Layer



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 Light4 (Rev. B) RoboBrick are listed below:

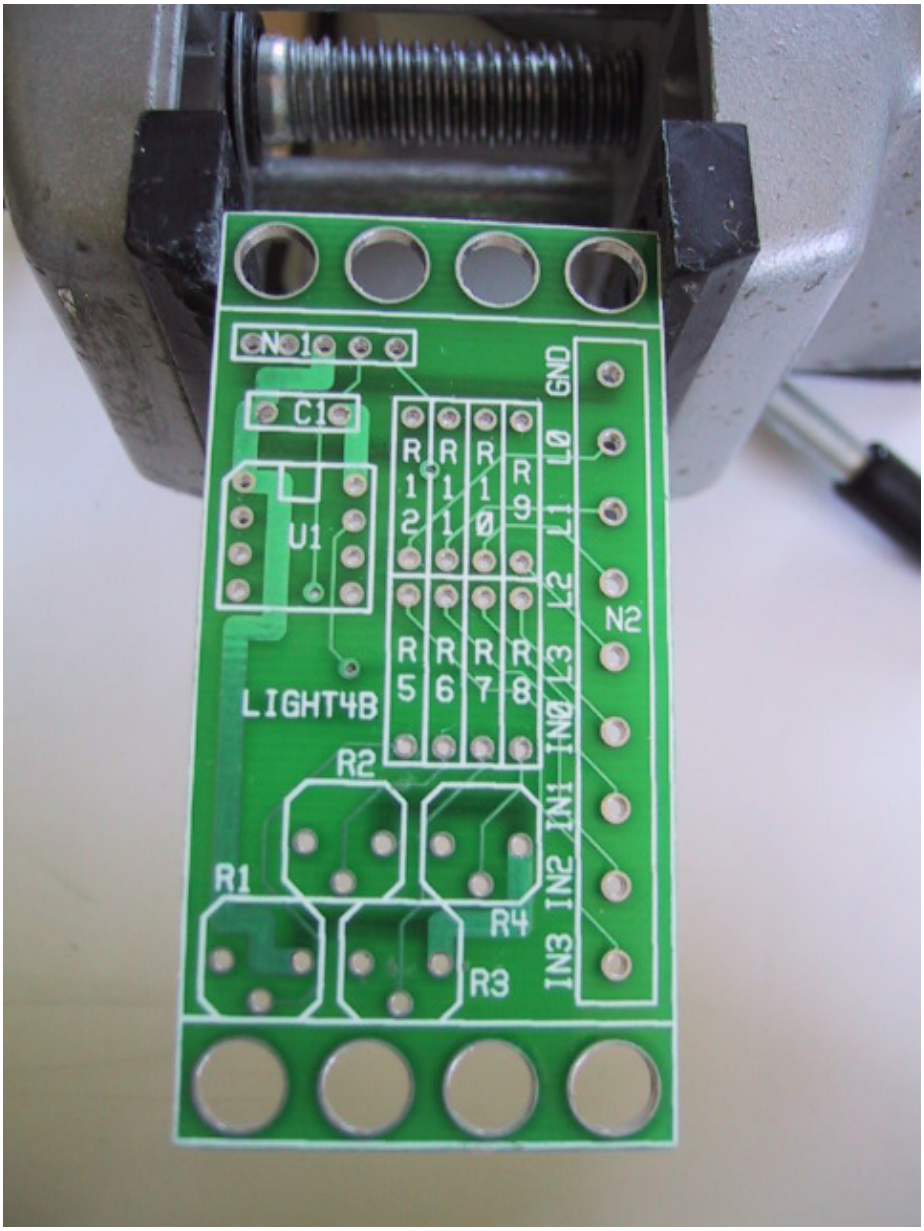
1. Orient the board vertically. By convention the upper edge is north, the lower edge is south, the left edge is west and the right edge is east. Orient the board so that N1 is in the north west corner. [step1.jpg]
2. Take a 1×5 male header and using some diagonal cutters, snip off pin 2 marked with an `X' in the the diagram below:

1	X	3	4	5

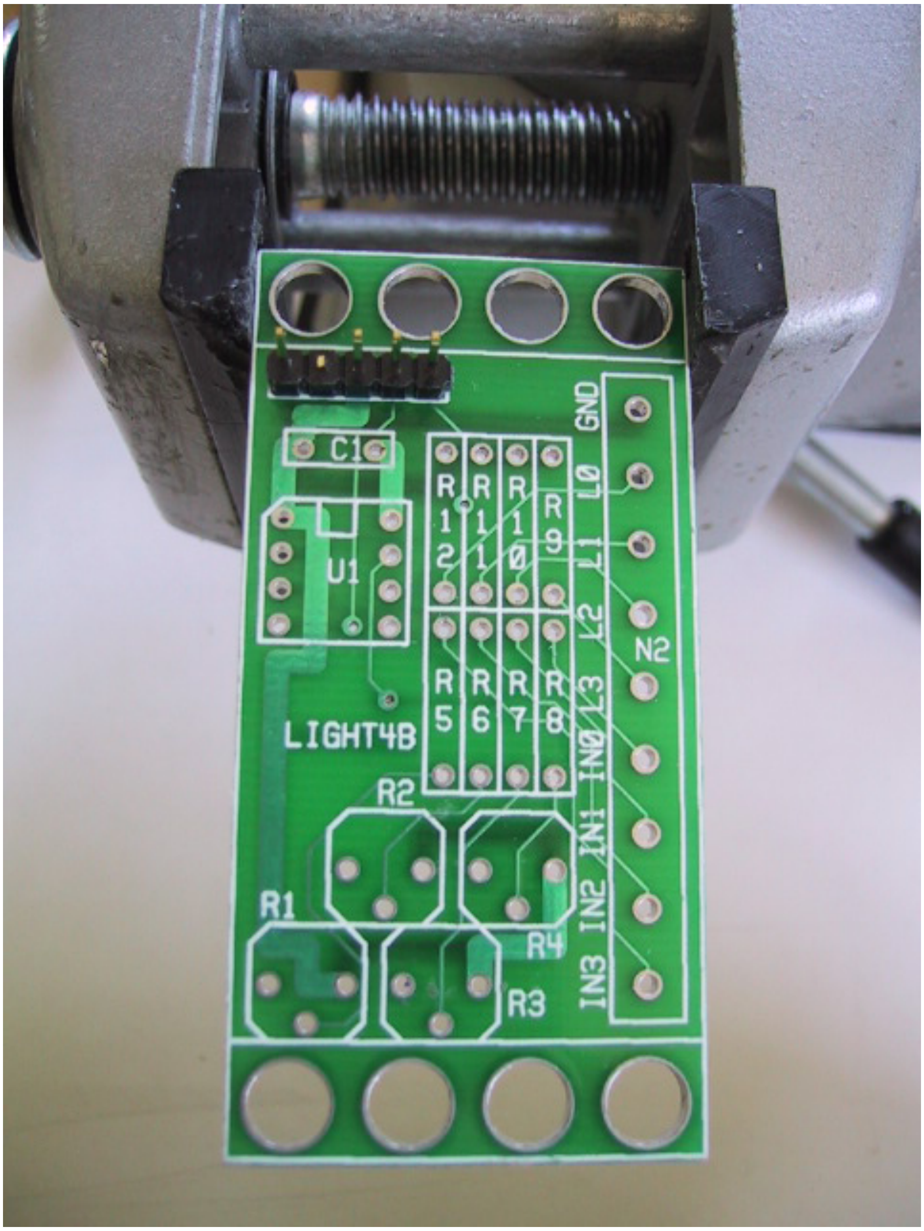
   Insert the header into N1 with pin 1 to the west. Solder one pin and verify that the header is flat on the board. If not, re−heat the solder pin and reposition the header until it is flat and vertical on the board. Solder the remaining 9 pins. [step2.jpg]
3. Find a 10pF capacitor and insert it into C1. Spread the leads a little so it will not fall out. Turn the board over and solder one lead. Turn the board over again and verify that the capacitor is still positioned vertically. If not, re−heat the lead you just soldered an reposition the capacitor. Now solder the remaining lead. Snip off the excess capacitor leads. [step3.jpg]
4. Find a 2K resistor (Red Black Red). If you can not find a 2K resistor, look for a 2.2K resistor (Red Red Red) instead. Bend both leads 90 degrees so that the resistor comfortably fits into R5. Just for consistency, place the red band up. (Resistors are symmetric, so there is no damage done if you install the resistor with the red band down.) 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. [step4.jpg]
5. Repeat the preceding instruction with three more of the same resistors and insert them into R6 through R8. Solder them as in the preceding instruction. [step5.jpg]
6. Find a 220 Ohm resistor (Red Red Brown) and install it into position R9 and solder as before. [step6.jpg]
7. Install the remaining three 220 Ohm resistors in positions R10 through R12. [step7.jpg]
8. Find the 9−terminal terminal strip and insert it into N2. If all you can find is some 3−terminal strips, a 9−terminal strip can be assembled from three 3−terminal strip by sliding them together. Make sure that the wire holes point towards the east. Solder in 1 pin, verify position, and solder the remaining pins. (Note: this picture was taken with an 8−terminal terminal strip; you should use a 9−terminal terminal strip instead.) [step8.jpg]
9. Find a 10K Ohm trim potentiometer. Insert it into R1 so that the leads go in straight. Solder one lead, verify that it is flat on the board, solder in the remaining two pins and snip off any excess leads. [step9.jpg]
10. Perform the previous instruction on the remaining three potentiometers and install them into R2 through R4. [step10.jpg]
11. Find the pre−programmed PIC12C672 and insert it into U1 with pin 1 pointing up. Solder in 1 pin, verify position, and solder in the remaining pins. The picture highlights the notch in U1 with some white marker. [step11.jpg]

The assembly of the Light4 (Rev. B) RoboBrick is complete.

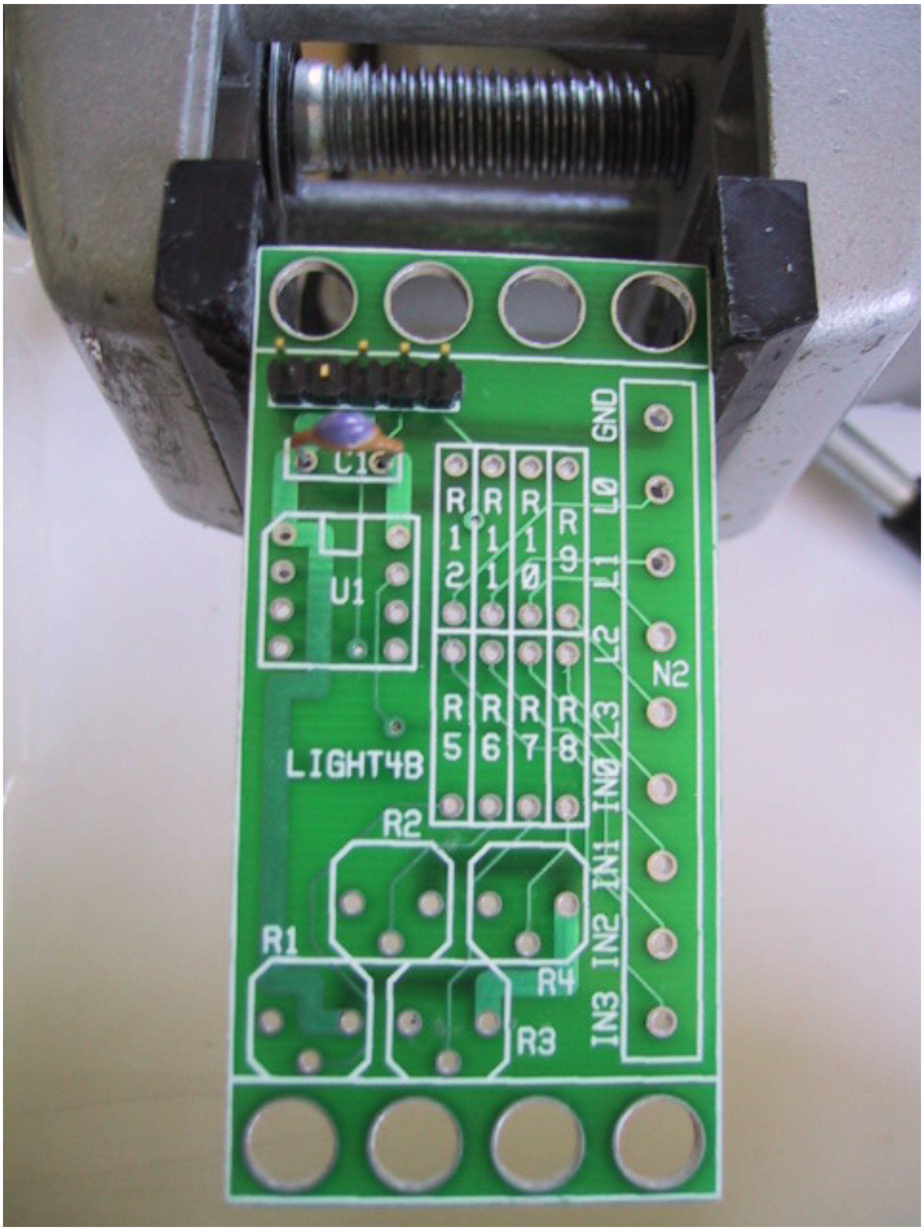
Step 1


N1
C1
U1
R12
R11
R10
R9
R5
R6
R7
R8
LIGHT4B
R2
R1
R4
R3
GND
LØ
L1
L2
L3
INØ
IN1
IN2
IN3
N2

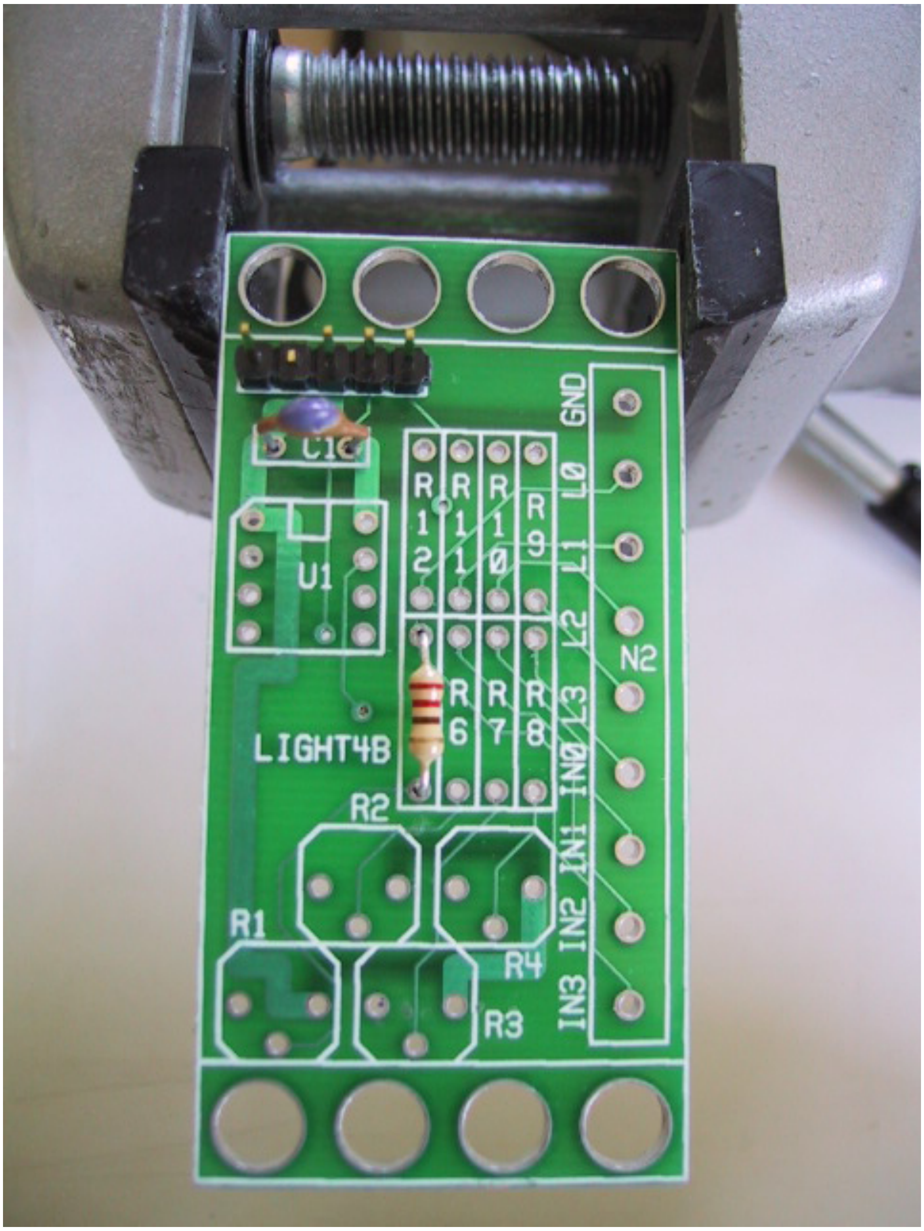
Step 2


C1
U1
R12
R11
R10
R9
R5
R6
R7
R8
LIGHT4B
R2
R1
R4
R3
GND
LØ
L1
L2
L3
INØ
IN1
IN2
IN3
N2

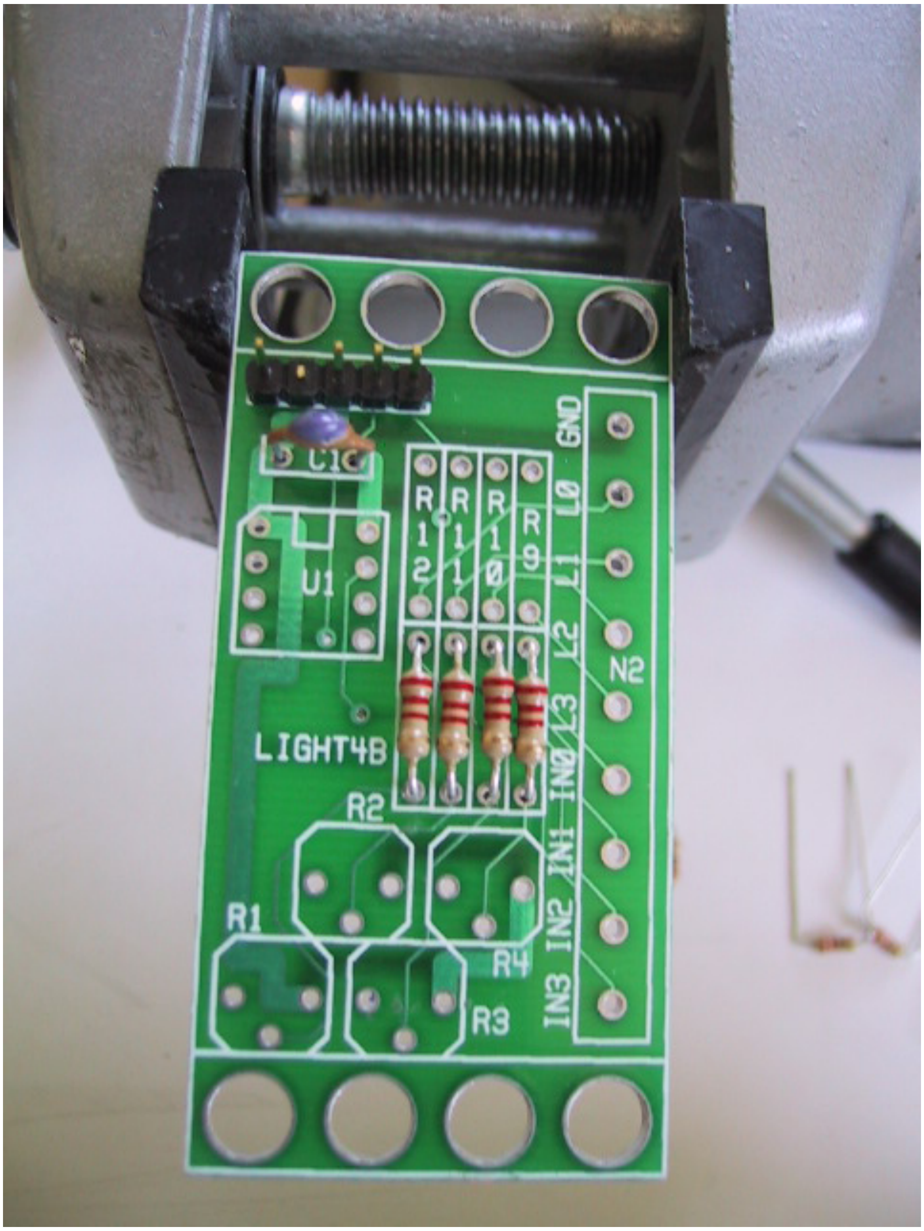
Step 3


GND
L0
L1
L2
L3
IN0
IN1
IN2
IN3
N2
C1
U1
R12
R11
R10
R9
R5
R6
R7
R8
LIGHT4B
R2
R1
R4
R3

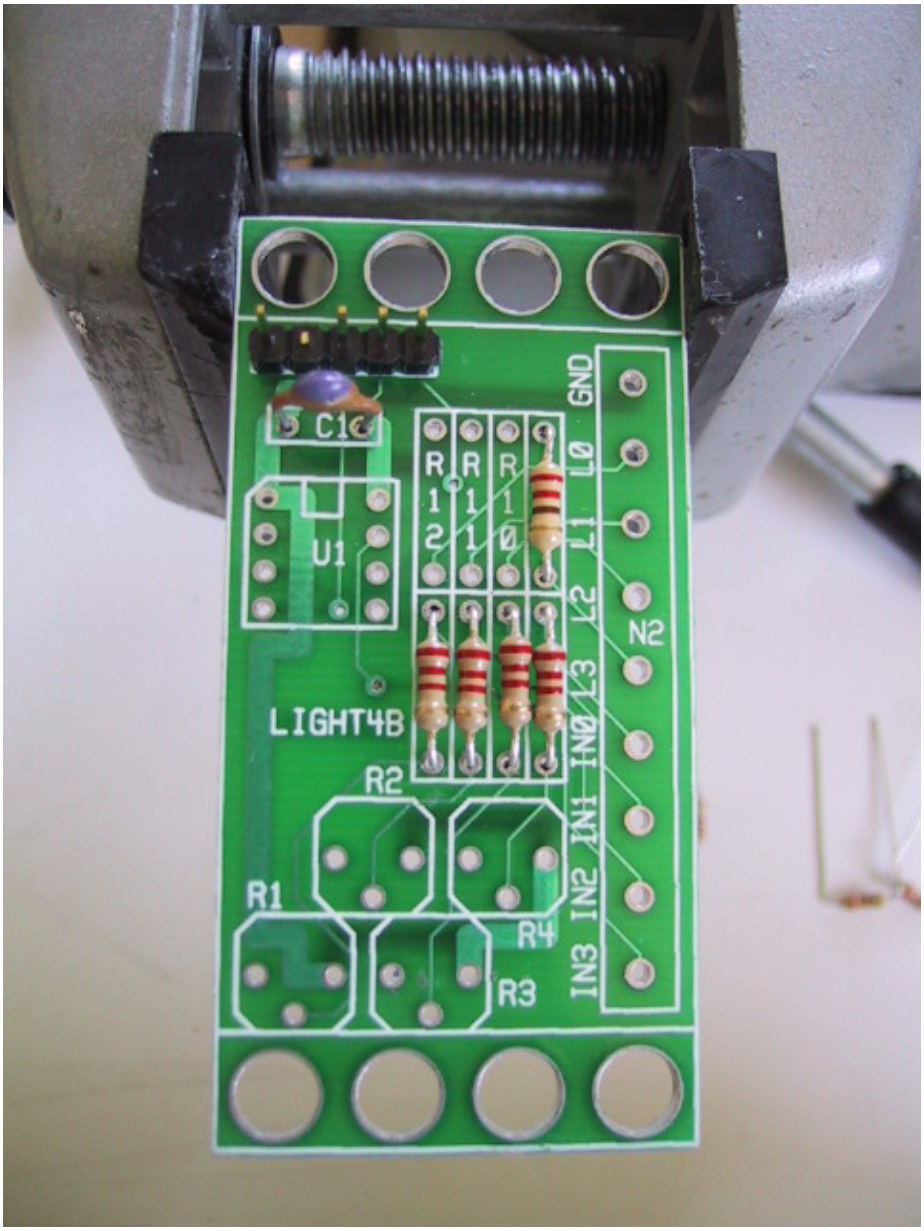
Step 4


GND
L0
L1
L2
L3
IN0
IN1
IN2
IN3
N2
C1
U1
R12
R11
R10
R9
R6
R7
R8
LIGHT4B
R2
R1
R4
R3

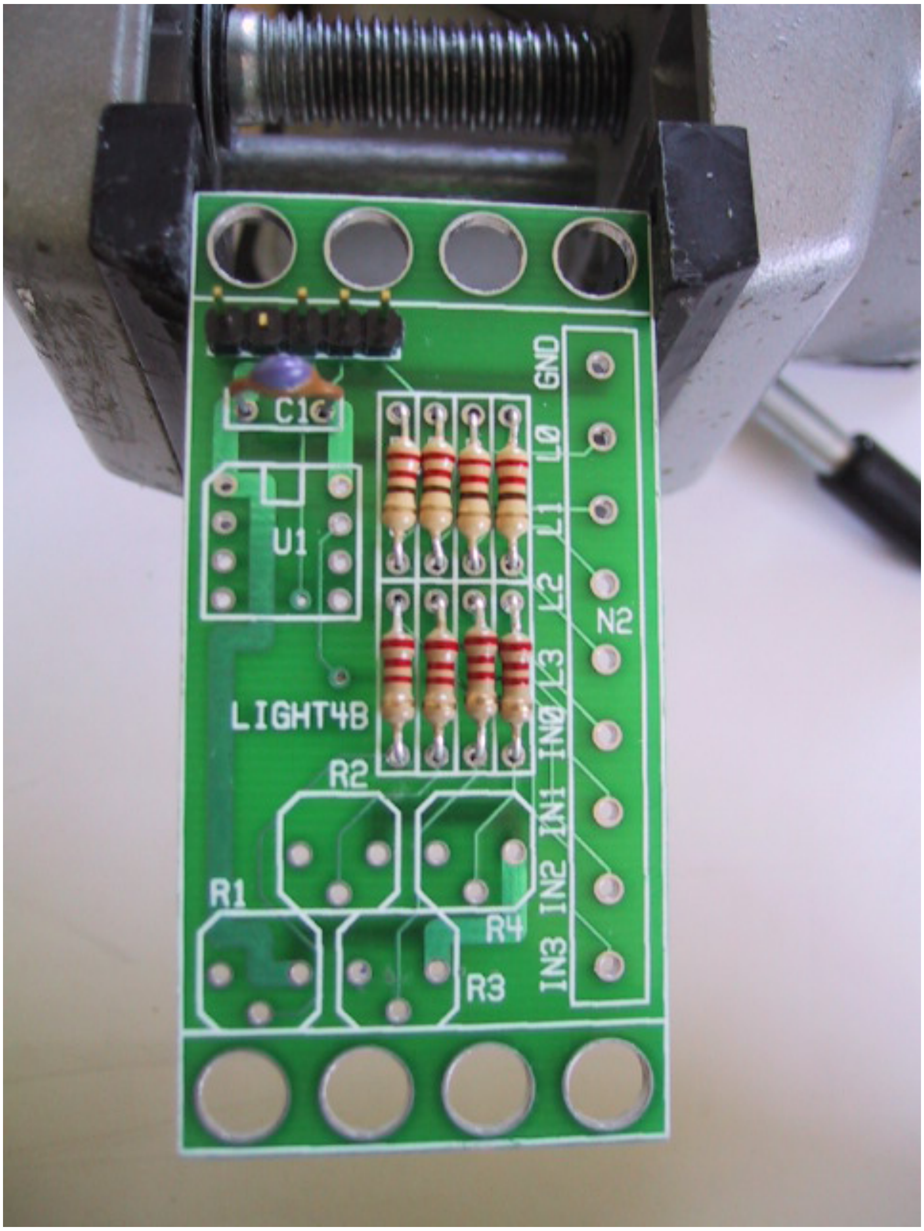
Step 5


GND
L0
L1
L2
L3
IN0
IN1
IN2
IN3
N2
C1
U1
R12
R11
R10
R9
LIGHT4B
R2
R1
R4
R3

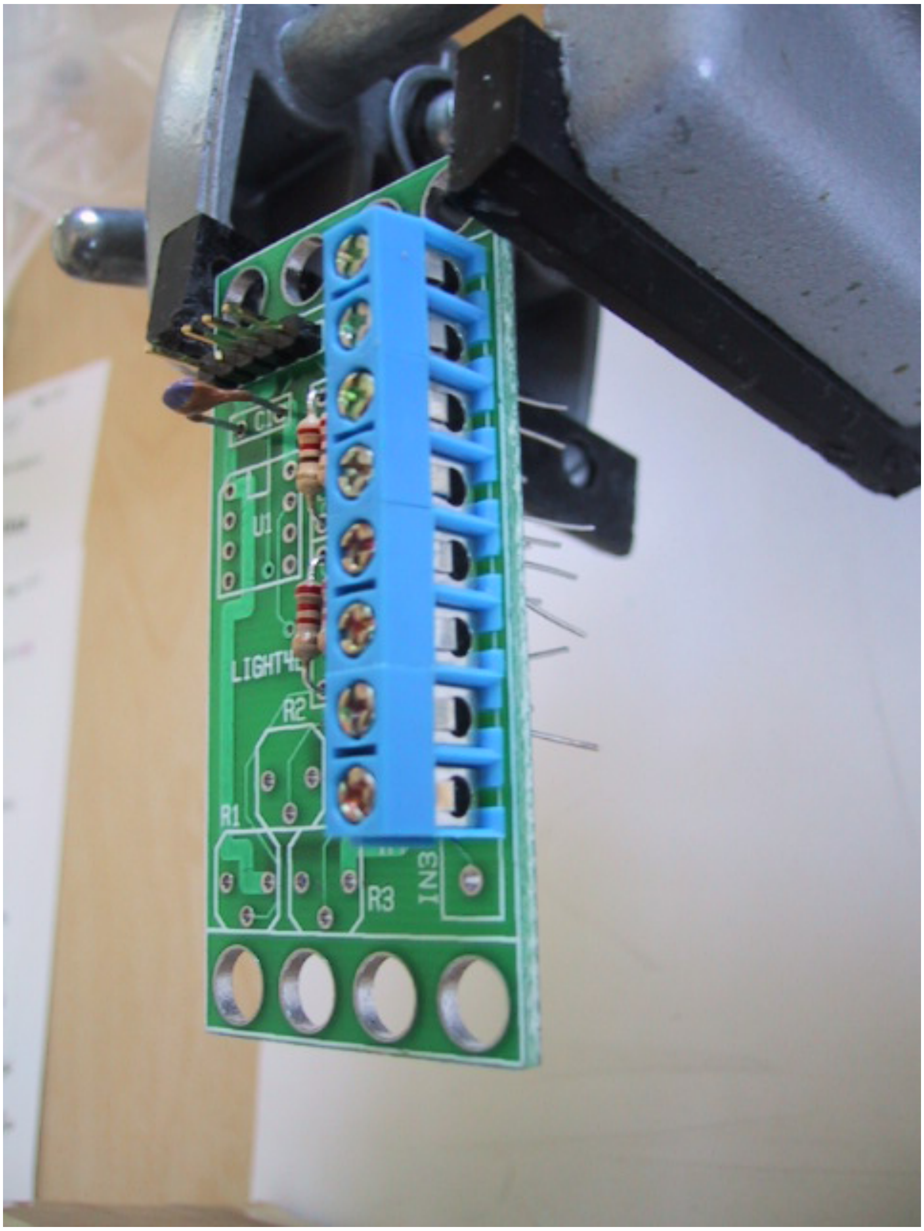
Step 6


C1
U1
R
1
2
R
1
1
R
1
0
GND
L0
L1
L2
L3
IN0
IN1
IN2
IN3
N2
LIGHT4B
R1
R2
R3
R4

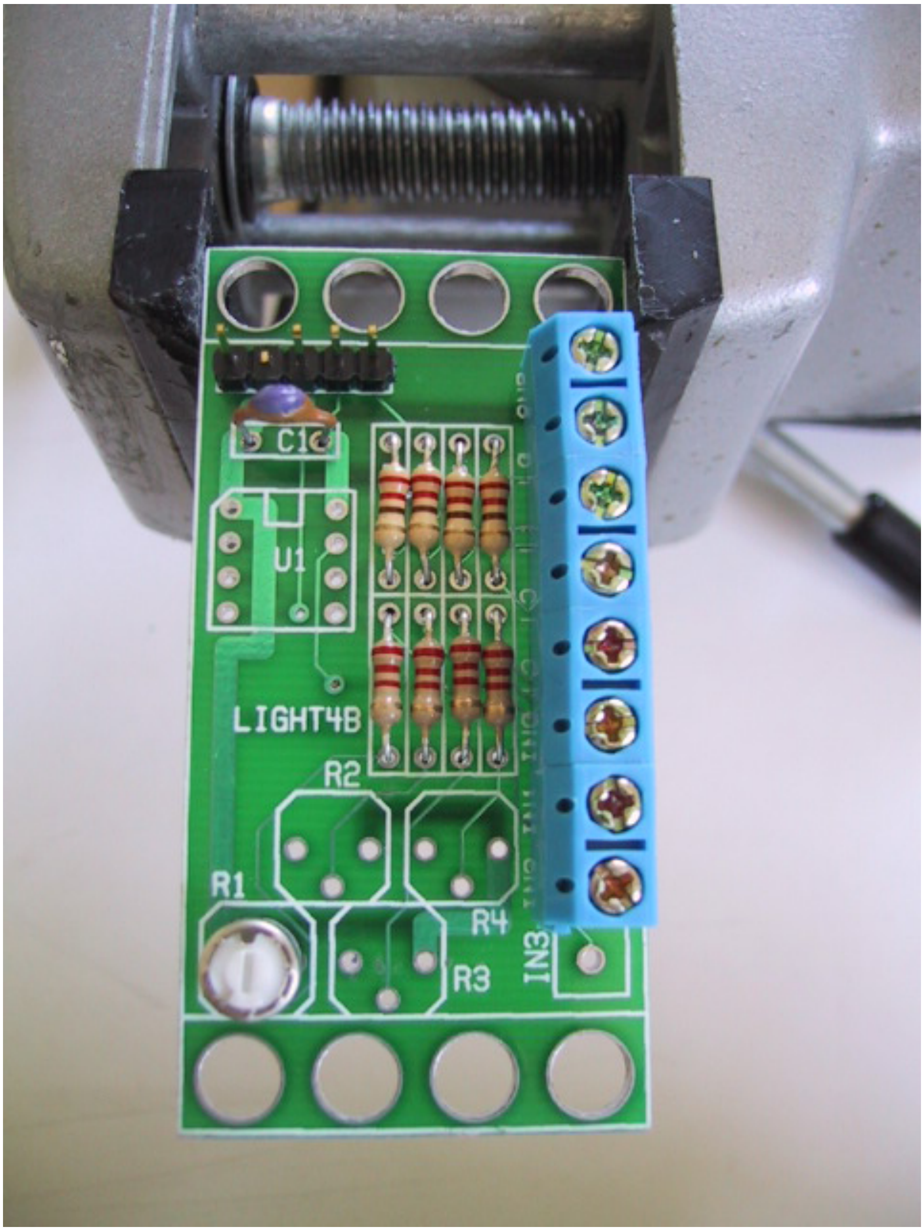
Step 7



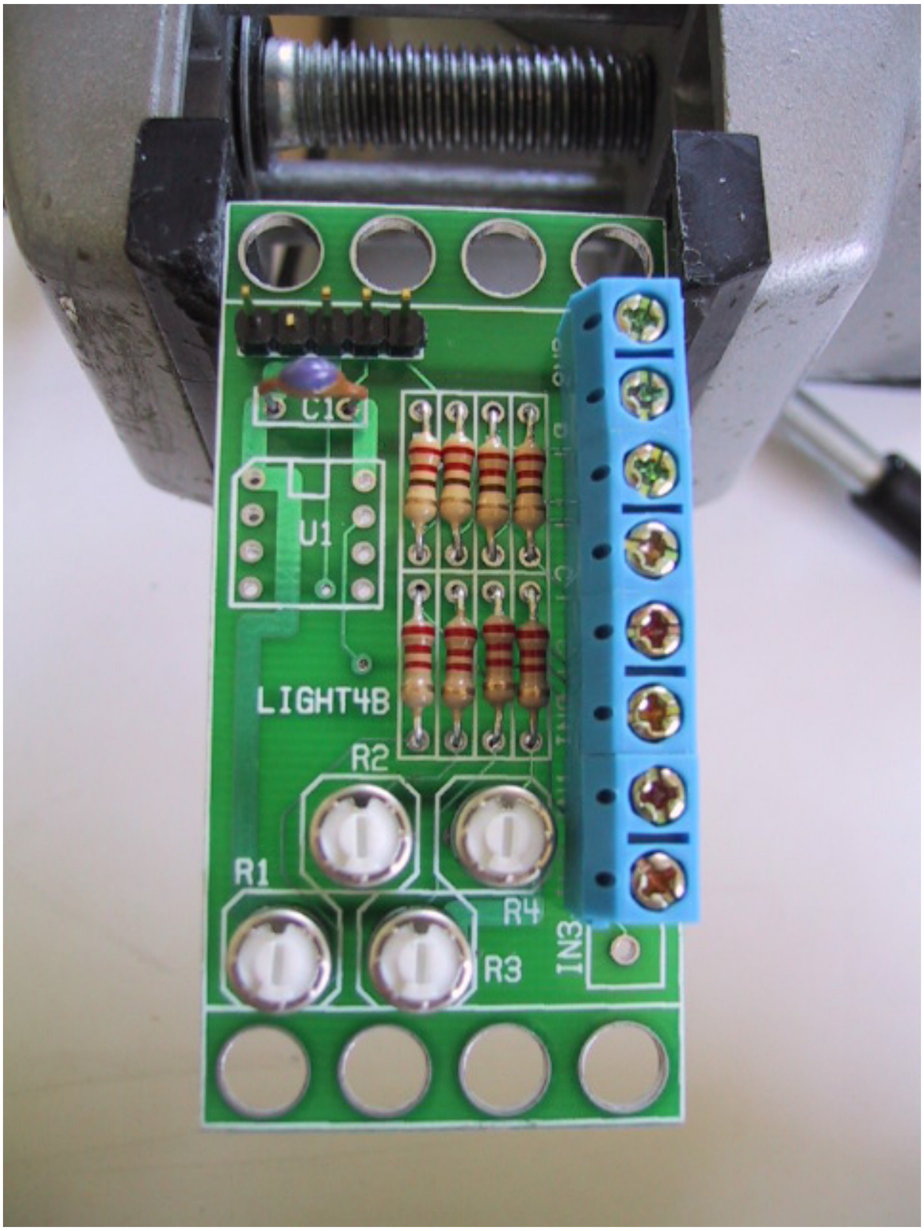
Step 8



Step 9



Step 10



Step 11