This is the Revision A verion of the InOut4 RoboBrick. The status of this project is that it has been replaced by the InOut10 RoboBrick.

InOut4 Robobrick (Revision A)

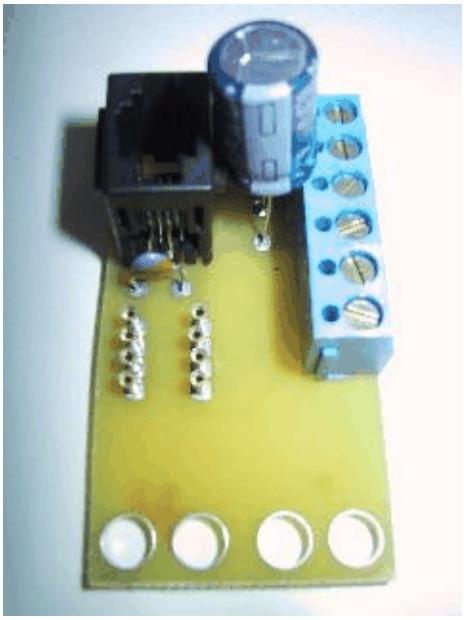
Table of Contents

This document is also available as a PDF document.

1. Introduction

The InOut4 RoboBrick allows you to read up to 8 digital inputs. An interrupt can be generated on the states of selected inptus.

A picture of the InOut4−A RoboBrick is shown below:



2. Programming

The InOut4 RobBrick has 4 I/O lines that can changed between input and output lines to meet the user's needs. The first operation is to specify the line direction. Next, the user does some input and output until it is time to change the line direction. The user may change the line direction as many times as needed.

The InOut4 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.

The InOut4 RoboBrick supports both the standard shared commands and the shared interrupt commands in addition to the following commands:

Read Inputs
: This command requests the that InOut4 RoboBrick return the 4−bits of its inputs XOR'ed with the complement mask.

Read Complement Mask
: This command will return the contents of the complement mask.

Read Low Mask
: This command will return the contents of the low mask.

Read High Mask
: This command will return the contents of the high mask.

Read Raising Mask
: This command will return the contents of the raising mask.

Read Falling Mask
: This command will return the contents of the falling mask.

Read Direction Mask
: This command will return the contents of the direction mask.

Read Outputs
: This command will return the contents of the outputs.

Read Raw
: This command requests the that InOut4 RoboBrick return the 4−bits of its inputs without XOR'ing them with the complement mask.

Set Complement Mask
: This command will set the contents of the complement mask.

Set Low Mask
: This command will set the contents of the Low mask.

Set High Mask
: This command will set the contents of the high mask.

Set Raising Mask
This command will set the contents of the raising mask.
Set Falling Mask
This command will set the contents of the falling mask.
Set Direction Mask
This command will set the contents of the direction mask.
Set Outputs
This command will set the outputs.
Bit Clear
This command will clear a specified bit in the outputs.
Bit Set
This command will set a specified bit in the outputs.
Bit Toggle
This command will clear a specified bit in the outputs.
Bit Read
This command will read a specified bit in the outputs.

These commands are summarized in the table below:

Command	**Send/ Receive**	**Byte Value**								**Discussion**
		7	**6**	**5**	**4**	**3**	**2**	**1**	**0**	
Read Inputs	Send	0	0	0	0	0	0	0	0	Return input values *abcd* (after XOR'ing with complement mask)
	Receive	0	0	0	0	*a*	*b*	*c*	*d*	
Read Complement Mask	Send	0	0	0	0	0	0	0	1	Return complement mask *cccc*
	Receive	0	0	0	0	*c*	*c*	*c*	*c*	
Read Low Mask	Send	0	0	0	0	0	0	1	0	Return low mask *llll*
	Receive	0	0	0	0	*l*	*l*	*l*	*l*	
Read High Mask	Send	0	0	0	0	0	0	1	1	Return high mask *hhhh*
	Receive	0	0	0	0	*h*	*h*	*h*	*h*	
Read Raising Mask	Send	0	0	0	0	0	1	0	0	Return raising mask *rrrr*
	Receive	0	0	0	0	*r*	*r*	*r*	*r*	
Read Falling Mask	Send	0	0	0	0	0	1	0	1	Return falling mask *ffff*
	Receive	0	0	0	0	*f*	*f*	*f*	*f*	
Read Direction Mask	Send	0	0	0	0	0	1	1	0	Return direction mask *dddd*
	Receive	0	0	0	0	*d*	*d*	*d*	*d*	
Read Outputs	Send	0	0	0	0	0	1	1	1	Return outputs *oooo*
	Receive	0	0	0	0	*o*	*o*	*o*	*o*	
Read Raw	Send	0	0	0	0	1	0	0	0	Return raw data *abcd* (without XOR'ing with complement mask)
	Receive	0	0	0	0	*a*	*b*	*c*	*d*	
Set Complement Mask	Send	0	0	0	0	1	0	0	1	Set complement mask to *cccc*
	Send	0	0	0	0	*c*	*c*	*c*	*c*	
Set High Mask	Send	0	0	0	0	1	0	1	0	Set high mask to *hhhh*
	Send	0	0	0	0	*h*	*h*	*h*	*h*	
Set Low Mask	Send	0	0	0	0	1	0	1	1	Set low mask to *llll*
	Send	0	0	0	0	*l*	*l*	*l*	*l*	

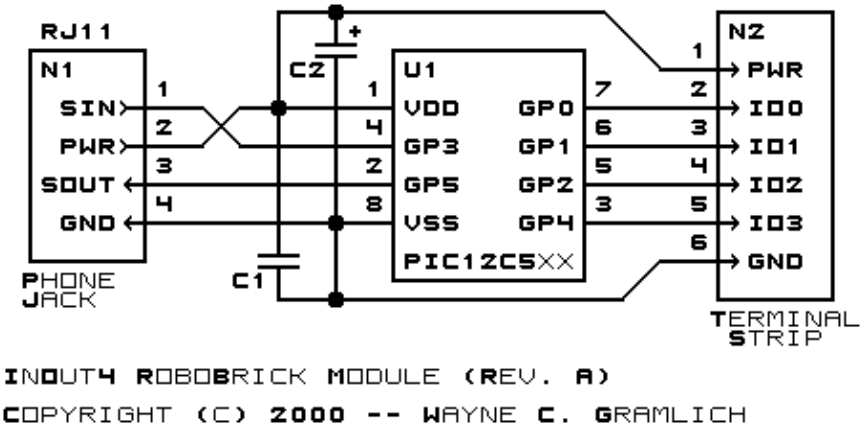
Set Raising Mask	Send	0	0	0	0	1	1	0	0	Set raising mask to *rrrr*
	Send	0	0	0	0	*r*	*r*	*r*	*r*	
Set Falling Mask	Send	0	0	0	0	1	1	0	1	Set falling mask to *ffff*
	Send	0	0	0	0	*f*	*f*	*f*	*f*	
Set Direction Mask	Send	0	0	0	0	1	1	1	0	Set direction mask to *dddd*. 0=output; 1=input
	Send	0	0	0	0	*d*	*d*	*d*	*d*	
Set Outputs	Send	0	0	0	0	1	1	1	1	Set outputs to *oooo*. 0=output; 1=input
	Send	0	0	0	0	*o*	*o*	*o*	*o*	
Bit Clear	Send	0	0	0	1	0	0	*b*	*b*	Clear bit *bb* in outputs.
Bit Set	Send	0	0	0	1	0	1	*b*	*b*	Clear bit *bb* in outputs.
Bit Toggle	Send	0	0	0	1	1	0	*b*	*b*	Toggle bit *bb* in outputs.
Bit read	Send	0	0	0	1	1	1	*b*	*b*	Read bit *bb* from outputs.
	Receive	0	0	0	0	0	0	0	*b*	
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 InOut4 RoboBrick is shown below:



The parts list kept in a separate file −− inout4.ptl.

3.2 Printed Circuit Board

The printed circuit board files are listed below:

inout4_back.png
: The solder side layer.

inout4_front.png
: The component side layer.

inout4_artwork.png
: The artwork layer.

inout4.gbl
: The RS−274X "Gerber" back (solder side) layer.

inout4.gtl
: The RS−274X "Gerber" top (component side) layer.

inout4.gal
: The RS−274X "Gerber" artwork layer.

inout4.drl
: The "Excellon" NC drill file.

inout4.tol
: The "Excellon" tool rack file.

4. Software

The InOut4 software is available as one of:

inout4.ucl
: The µCL source file.

inout4.asm
: The resulting human readable PIC assembly file.

inout4.lst
: The resulting human readable PIC listing file.

inout4.hex
: The resulting Intel® Hex file that can be fed into a PIC12C5xx programmer.

5. Issues

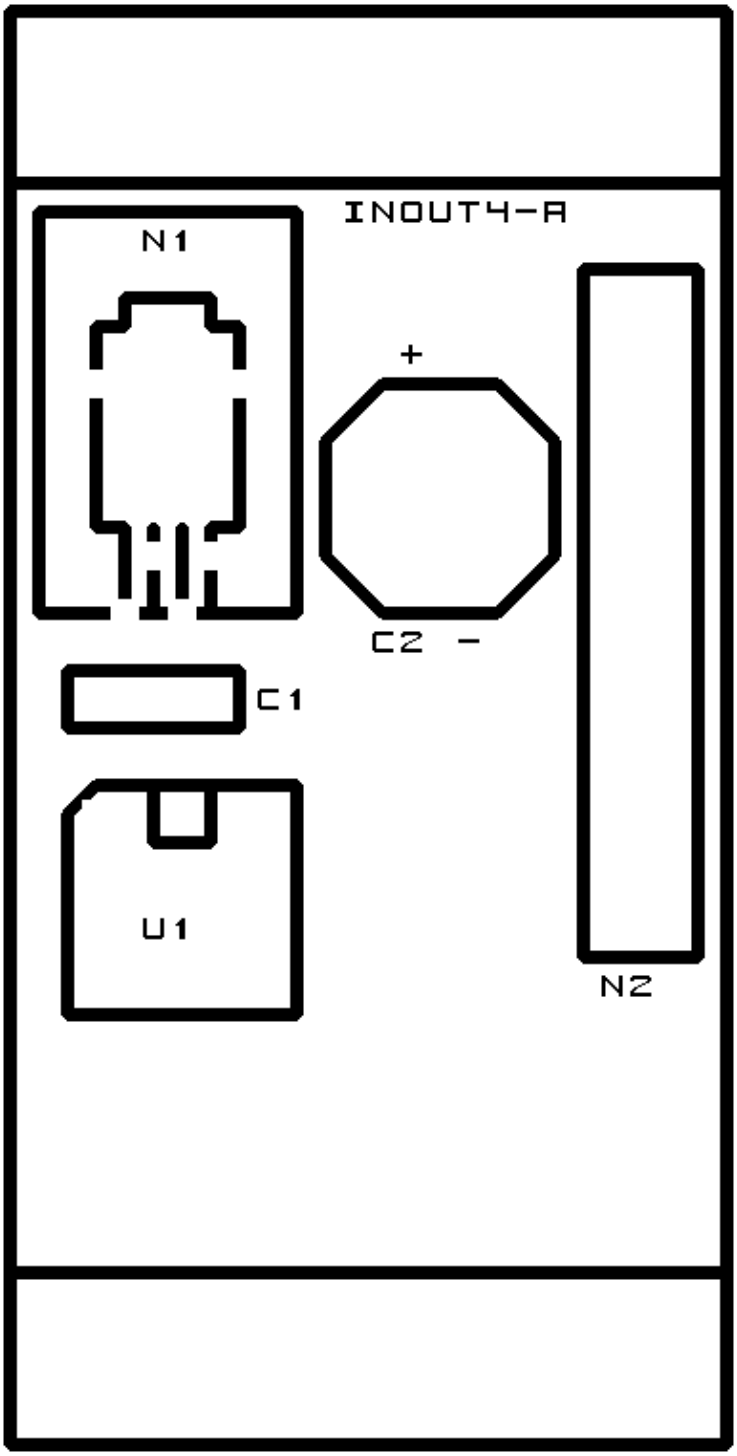
The following issues have come up:

- The 2200 µF capacitor does not fit between the RJ11 and the terminal strip.
- One of the traces has an unnecessary kink in it.
- The 8−pin terminal strip is too close to the the 74LS151.
- The terminal strip holes are too small.
- The Lego holes are not right.
- The RJ11 holes are not right.
- We need to switch over to a 6−wire RJ11 connector.

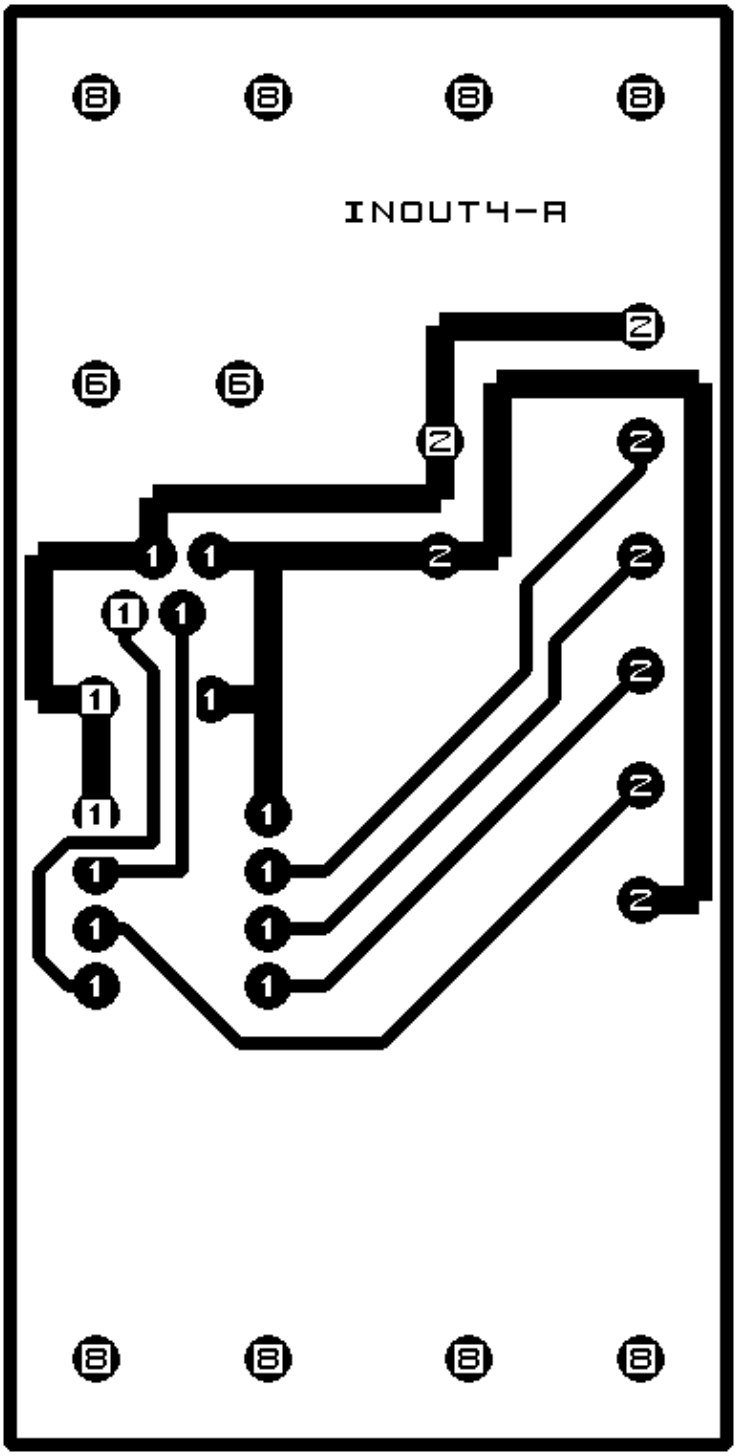
A. Appendix A: Parts List

```
# Parts list for InOut4 RoboBrick (Rev. A)
#
C1: Capacitor10pF − 10 pF Ceramic Capacitor [Jameco: 15333]
C2: Capacitor2200uF − 2200 uF 6.3V Electrolytic Capacitor [Jameco: 133145]
N1: RJ11Female4_4.RBSlave − Female RJ11 (4−4) Phone Jack [Digikey: A9071−ND]
N2: TerminalStrip6.InOut4 − 6 Junction Terminal Strip [2 Jameco: 189667]
U1: PIC12C509.InOut4 − Microchip PIC12C509 [Digikey: PIC12C509A−04/P−ND]
```

B. Appendix B: Artwork Layer



C. Appendix C: Back (Solder Side) Layer



D. Appendix D: Front (Component Side) Layer