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 The status of this project is that it has been replaced by the Revision B version.

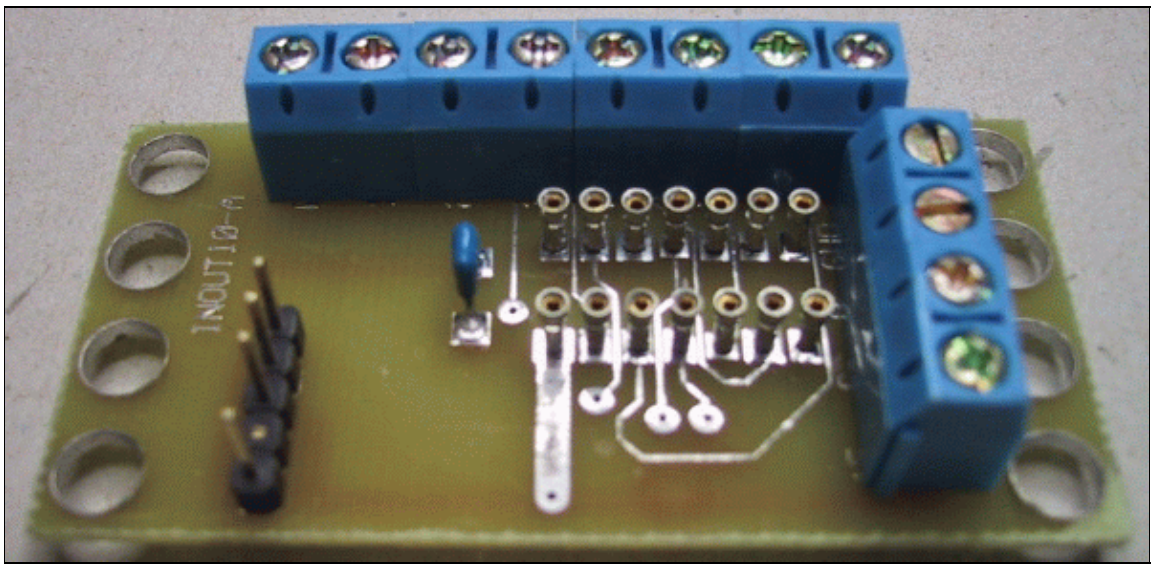
InOut10 Robobrick (Revision A)

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*	
	Send	0	0	0	0	1	0	0	1	

Read Low Mask High	Receive	0	0	0	*L*	*L*	*L*	*L*	*L*	Return high order 5−bits of low mask *LLLLL*
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	Return low order 5−bits of the outputs *ooooo*
	Receive	0	0	0	*o*	*o*	*o*	*o*	*o*	
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*
	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*
	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*	
	Send	0	0	0	1	1	1	1	0	

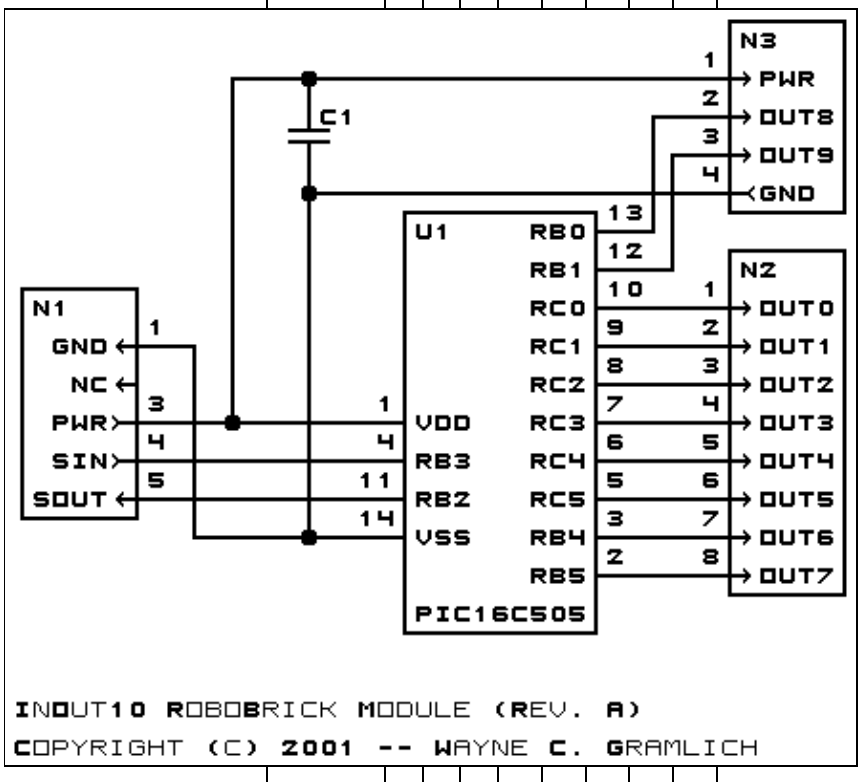
Set Falling Mask Low	Send	0	0	0	*f*	*f*	*f*	*f*	*f*	Set low order 5−bits of the falling mask to *fffff*
Set Falling Mask High	Send	0	0	0	1	1	1	1	1	Set high order 5−bits of the falling mask to *FFFFF*
	Send	0	0	0	*F*	*F*	*F*	*F*	*F*	
Set Outputs Low	Send	0	0	1	o	o	o	o	o	Set low order 5−bits to *ooooo*
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.

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

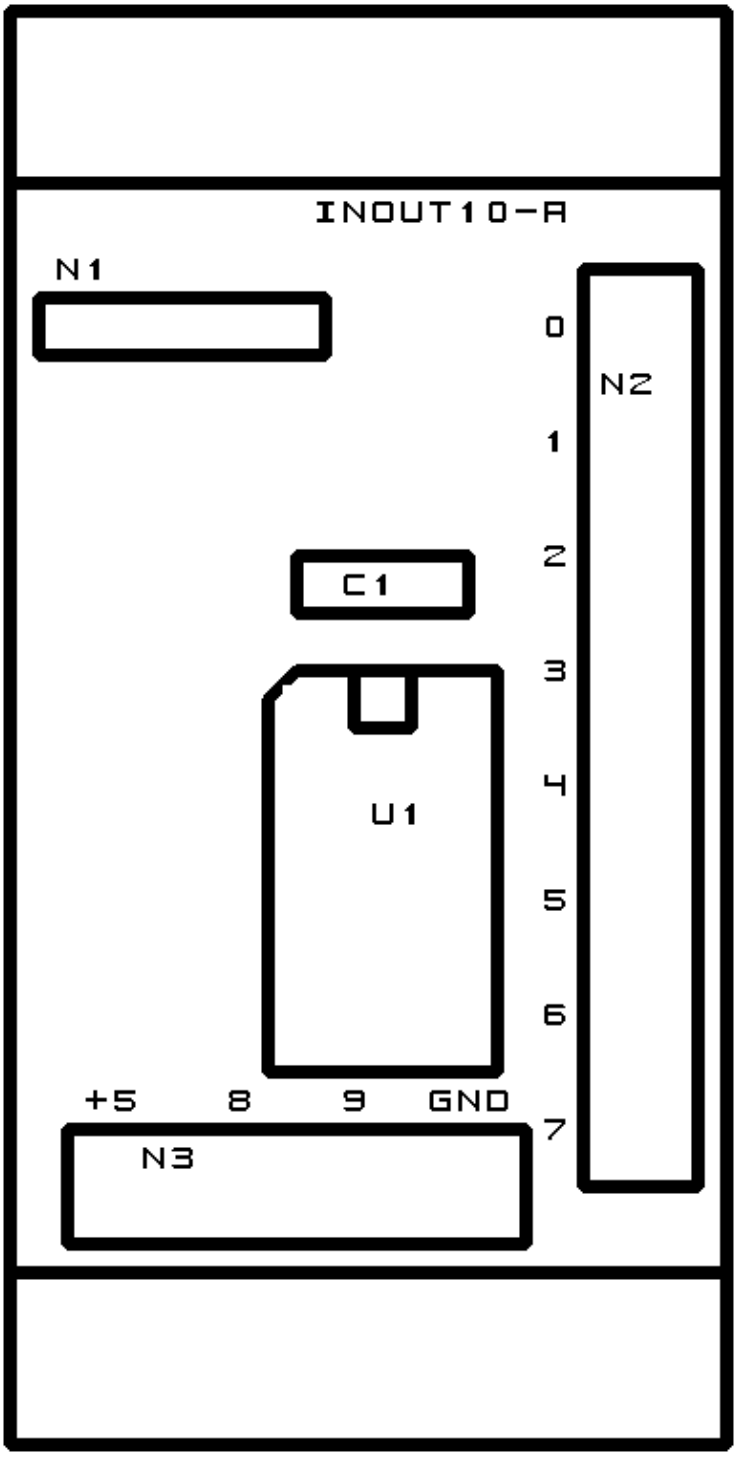
The following issues have come up:

- The holes for N1 (size 3) are too large, make them smaller (size 2).
- The holes for N2 and N3 (size 2) are too small, make them larger (size 4).
- Move the lettering next to N2 to the left a little.
- Move the lettering above N3 up a little.
- Move U1 and C1 up and to the left.
- Move N3 to the left by .05 inches

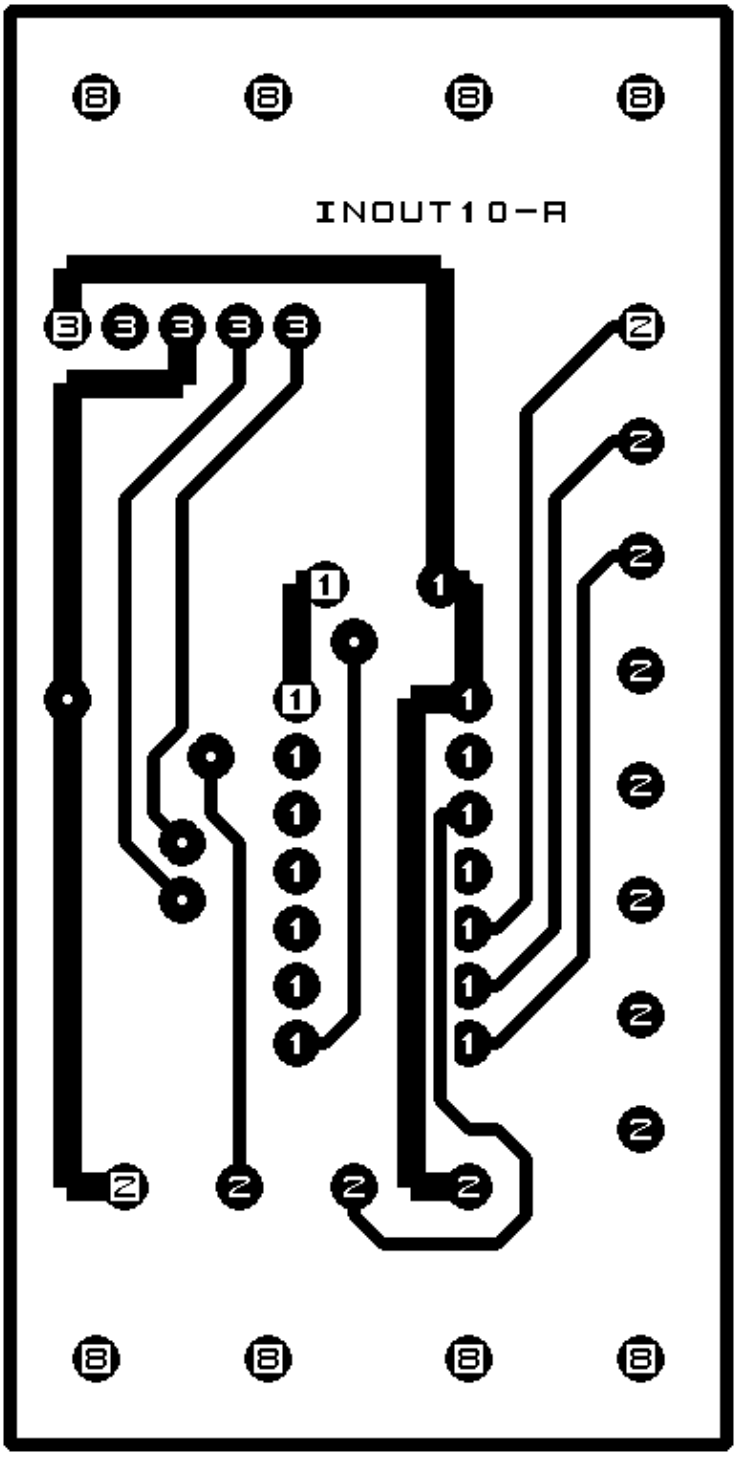
A. Appendix A: Parts List

```
# Parts list for InOut10 RoboBrick (Rev. A)
#
C1: Capacitor10pF − 10 pF Ceramic Capacitor [Jameco: 15333]
N1: Header1x5.RBSlave − 1x5 Male Header [5/40 Jameco: 160881]
N2: TerminalStrip8.InOut10 − 8 Junction Terminal Strip [4 Jameco: 189675]
N3: TerminalStrip4.InOut10 − 4 Junction Terminal Strip [2 Jameco: 189675]
U1: PIC16C505.InOut10 − Microchip PIC16C505 [Digikey: PIC16C505−04/P−ND]
```

B. Appendix B: Artwork Layer



C. Appendix C: Back (Solder Side) Layer



D. Appendix D: Front (Component Side) Layer