

This is the Revision A version of the IO8 Module. The status of this project is finished.

# IO8 Module (Revision A)

## 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
- 4. Software
- 5. Issues

## 1. Introduction

The IO8 Module is used to provide 8-pins of I/O capability. Each I/O pin can independently be used as a digital input, an analog input, or a digital output. The analog input is at 10-bit resolution based off of an internal 5-volt reference or an externally supplied voltage reference.

## 2. Programming

Module Interrupt Protocol for those lines that are being used as inputs. The interrupt pending bit is set whenever the formula:

$$L \& (\sim I) \mid H \& I \mid R \& (\sim P) \& I \mid F \& P \& (\sim I)$$

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 AnalogIn4 Module supports following commands:

IO8 Module (Revision A)

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 8–bits of input <i>iiii iii</i> (after XOR'ing with complement mask)
	Receive	<i>i</i>	<i>i</i>	<i>i</i>	<i>i</i>	<i>i</i>	<i>i</i>	<i>i</i>	<i>i</i>	
Read Outputs	Send	0	0	0	0	0	0	0	1	Return 8–bits of the outputs <i>oooo oooo</i> (after XOR'ing with complement mask.)
	Receive	<i>o</i>	<i>o</i>	<i>o</i>	<i>o</i>	<i>o</i>	<i>o</i>	<i>o</i>	<i>o</i>	
Read Complement Mask	Send	0	0	0	0	0	0	1	0	Return 8–bits of complement mask <i>cccc cccc</i>
	Receive	<i>c</i>	<i>c</i>	<i>c</i>	<i>c</i>	<i>c</i>	<i>c</i>	<i>c</i>	<i>c</i>	
Read Direction Mask	Send	0	0	0	0	0	0	1	1	Return 8–bits of direction mask <i>dddd dddd</i>
	Receive	<i>d</i>	<i>d</i>	<i>d</i>	<i>d</i>	<i>d</i>	<i>d</i>	<i>d</i>	<i>d</i>	
Read Low Mask	Send	0	0	0	0	0	1	0	0	Return 8–bits of low mask <i>llll llll</i>
	Receive	<i>l</i>	<i>l</i>	<i>l</i>	<i>l</i>	<i>l</i>	<i>l</i>	<i>l</i>	<i>l</i>	
Read High Mask	Send	0	0	0	0	0	1	0	1	Return 8–bits of the high mask <i>hhhh hhhh</i>
	Receive	<i>h</i>	<i>h</i>	<i>h</i>	<i>h</i>	<i>h</i>	<i>h</i>	<i>h</i>	<i>h</i>	
Read Rising Mask	Send	0	0	0	0	0	1	1	0	Return 8–bits of the rising mask <i>rrrr rrrr</i>
	Receive	<i>r</i>	<i>r</i>	<i>r</i>	<i>r</i>	<i>r</i>	<i>r</i>	<i>r</i>	<i>r</i>	
Read Falling Mask	Send	0	0	0	0	0	1	1	1	Return 8–bits of the falling mask <i>ffff ffff</i>
	Receive	<i>f</i>	<i>f</i>	<i>f</i>	<i>f</i>	<i>f</i>	<i>f</i>	<i>f</i>	<i>f</i>	
Read Raw	Send	0	0	0	0	1	0	0	0	Return 8–bits of raw input data <i>rrrr rrrr</i> (without XOR'ing with complement mask)
	Receive	<i>r</i>	<i>r</i>	<i>r</i>	<i>r</i>	<i>r</i>	<i>r</i>	<i>r</i>	<i>r</i>	
Read Analog Mask	Send	0	0	0	0	1	0	0	1	Return the Analog mask <i>aaaa aaaa</i> .
	Receive	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>	
Read Outputs Raw	Send	0	0	0	0	1	0	1	0	Return the raw outputs as <i>oooo oooo</i> with no complement mask.
	Receive	<i>o</i>	<i>o</i>	<i>o</i>	<i>o</i>	<i>o</i>	<i>o</i>	<i>o</i>	<i>o</i>	
Read Analog VRef	Send	0	0	0	0	1	0	1	1	Return the analog Vref at <i>v</i> (0 = 5 volts, 1 = IO1).
	Receive	0	0	0	0	0	0	0	<i>v</i>	
Reset Outputs	Send	0	0	0	1	0	0	0	0	Set all 8 bits of outputs to 0 (then XOR with complement mask).
Set Outputs	Send	0	0	0	1	0	0	0	1	Set output bits to <i>oooo oooo</i> .
	Send	<i>o</i>	<i>o</i>	<i>o</i>	<i>o</i>	<i>o</i>	<i>o</i>	<i>o</i>	<i>o</i>	
Set Complement Mask	Send	0	0	0	1	0	0	1	0	Set 8–bits of complement mask to <i>cccc cccc</i>
	Send	<i>c</i>	<i>c</i>	<i>c</i>	<i>c</i>	<i>c</i>	<i>c</i>	<i>c</i>	<i>c</i>	
Set Direction Mask	Send	0	0	0	1	0	0	1	1	Set 8–bits of direction mask to <i>dddd dddd</i> 1=input; 0=output
	Send	<i>d</i>	<i>d</i>	<i>d</i>	<i>d</i>	<i>d</i>	<i>d</i>	<i>d</i>	<i>d</i>	
Set Low Mask	Send	0	0	0	1	0	1	0	0	Set 8–bits of low mask to <i>llll llll</i>
	Send	<i>l</i>	<i>l</i>	<i>l</i>	<i>l</i>	<i>l</i>	<i>l</i>	<i>l</i>	<i>l</i>	
Set High Mask	Send	0	0	0	1	0	1	0	1	Set 8–bits of the high mask to <i>hhhh hhhh</i>
	Send	<i>h</i>	<i>h</i>	<i>h</i>	<i>h</i>	<i>h</i>	<i>h</i>	<i>h</i>	<i>h</i>	
Set Rising Mask	Send	0	0	0	1	0	1	1	0	Set 8–bits of the rising mask to <i>rrrr rrrr</i>
	Send	<i>r</i>	<i>r</i>	<i>r</i>	<i>r</i>	<i>r</i>	<i>r</i>	<i>r</i>	<i>r</i>	
Set Falling Mask	Send	0	0	0	1	0	1	1	1	Set 8–bits of the falling mask to <i>ffff ffff</i>
	Send	<i>f</i>	<i>f</i>	<i>f</i>	<i>f</i>	<i>f</i>	<i>f</i>	<i>f</i>	<i>f</i>	

## IO8 Module (Revision A)

Set Outputs Raw	Send	0	0	0	1	1	0	0	0	Set 8–bits to <i>oooo oooo</i> with no complement mask.
	Send	<i>o</i>	<i>o</i>	<i>o</i>	<i>o</i>	<i>o</i>	<i>o</i>	<i>o</i>	<i>o</i>	
Set Analog Mask	Send	0	0	0	1	1	0	0	1	Set analog mask to <i>aaaa aaaa</i> .
	Send	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>	
Set Analog VRef	Send	0	0	0	1	1	0	1	<i>v</i>	Set A/D converted Voltage reference mode to <i>v</i> (0 = 5 volts, 1 = external).
Reset Everything	Send	0	0	0	1	1	1	1	1	Reset all registers to 0 and set direction bits to 1 (input)
Set Output Bit	Send	0	0	1	0	<i>v</i>	<i>b</i>	<i>b</i>	<i>b</i>	Set output bit <i>bbbb</i> to <i>v</i>
Set Outputs Low	Send	0	1	0	0	<i>l</i>	<i>l</i>	<i>l</i>	<i>l</i>	Set low order 4–bits of Outputs to <i>llll</i> and then XOR complement mask
Set Outputs High	Send	0	1	0	1	<i>h</i>	<i>h</i>	<i>h</i>	<i>h</i>	Set high order 4–bits of Outputs to <i>hhhh</i> and then XOR complement mask
Set Direction Low	Send	0	1	1	0	<i>l</i>	<i>l</i>	<i>l</i>	<i>l</i>	Set low order 4–bits of direction to <i>llll</i> .
Set Direction High	Send	0	1	1	1	<i>h</i>	<i>h</i>	<i>h</i>	<i>h</i>	Set high order 4–bits of direction to <i>hhhh</i> .
Read Analog 8–bits	Send	1	0	0	0	0	<i>b</i>	<i>b</i>	<i>b</i>	Read 8–bits of analog value from <i>bbb</i> and returns result as <i>aaaa aaaa</i> .
	Receive	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>	
Read Analog 10–bits	Send	1	0	0	0	1	<i>b</i>	<i>b</i>	<i>b</i>	Read 10–bits of analog value from <i>bbb</i> and returns result as <i>aaaa aaaa bb</i> .
	Receive	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>	
	Receive	<i>b</i>	<i>b</i>	0	0	0	0	0	0	
Read Low Threshold	Send	1	0	0	1	0	<i>b</i>	<i>b</i>	<i>b</i>	Read analog low threshold for <i>bbb</i> and returns result as <i>llll llll</i> .
	Receive	<i>l</i>	<i>l</i>	<i>l</i>	<i>l</i>	<i>l</i>	<i>l</i>	<i>l</i>	<i>l</i>	
Read High Threshold	Send	1	0	0	1	1	<i>b</i>	<i>b</i>	<i>b</i>	Read analog low threshold for <i>bbb</i> and returns result as <i>hhhh hhhh</i> .
	Receive	<i>h</i>	<i>h</i>	<i>h</i>	<i>h</i>	<i>h</i>	<i>h</i>	<i>h</i>	<i>h</i>	
Set Low Threshold	Send	1	0	1	1	0	<i>b</i>	<i>b</i>	<i>b</i>	Set the analog low threshold for <i>bbb</i> to <i>llll llll</i> .
	Send	<i>l</i>	<i>l</i>	<i>l</i>	<i>l</i>	<i>l</i>	<i>l</i>	<i>l</i>	<i>l</i>	
Set High Threshold	Send	1	0	1	1	1	<i>b</i>	<i>b</i>	<i>b</i>	Set the analog low threshold for <i>bbb</i> to <i>hhhh hhhh</i> .
	Send	<i>h</i>	<i>h</i>	<i>h</i>	<i>h</i>	<i>h</i>	<i>h</i>	<i>h</i>	<i>h</i>	
<u>Set Interrupt Commands</u>	Send	1	1	1	1	0	<i>c</i>	<i>c</i>	<i>c</i>	Set Interrupt Command <i>ccc</i> .
<u>Shared Commands</u>	Send	1	1	1	1	1	<i>c</i>	<i>c</i>	<i>c</i>	Execute Shared Command <i>ccc</i>

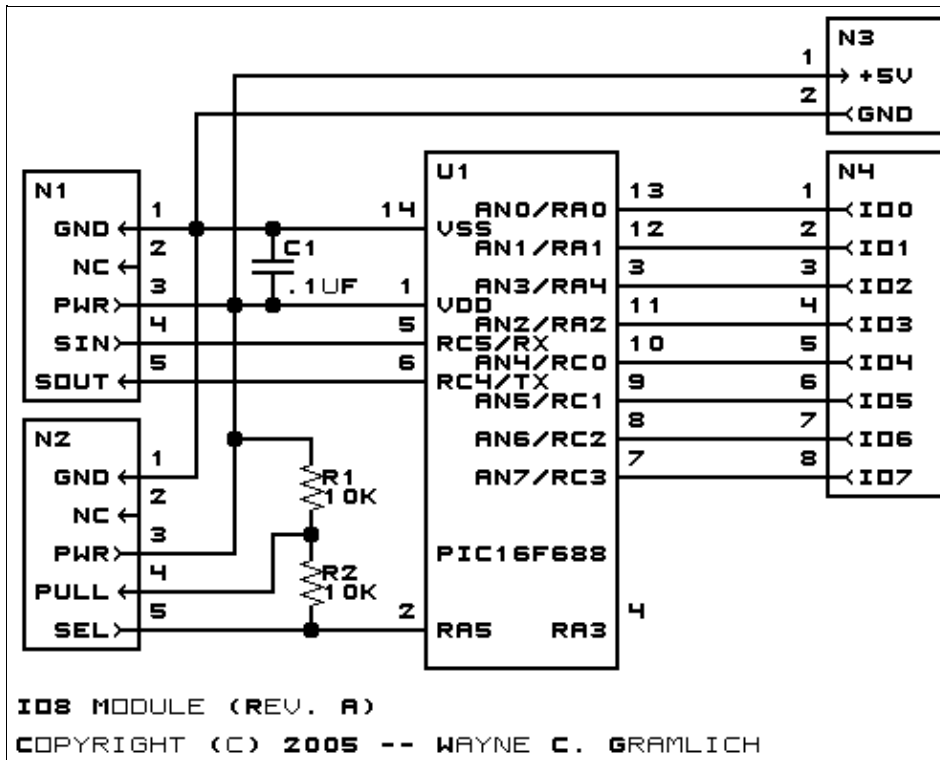
### 3. Hardware

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

#### 3.1 Circuit Schematic

The schematic for the IO8 Module is shown below:

## IO8 Module (Revision A)



The parts list kept in a separate file -- [io8.ptl](#).

### 3.2 Printed Circuit Board

The printed circuit board files are listed below:

[io8\\_back.png](#)

The solder side layer.

[io8\\_front.png](#)

The component side layer.

[io8\\_artwork.png](#)

The artwork layer.

[io8.gbl](#)

The RS-272X "Gerber" back (solder side) layer.

[io8.gtl](#)

The RS-272X "Gerber" top (component side) layer.

[io8.gal](#)

The RS-272X "Gerber" artwork layer.

[io8.drl](#)

The "Excellon" NC drill file.

[io8.tol](#)

The "Excellon" tool rack file.

## 4. Software

The software for the IO8 is listed below:

*io8.ucl*

The  $\mu$ CL file for IO8.

*io8.asm*

The assembly file for IO8.

*io8.hex*

The Intel<sup>®</sup> Hex file.

*io8.lst*

The listing file for IO8.

## 5. Issues

Any fabrication issues will be listed here.

---

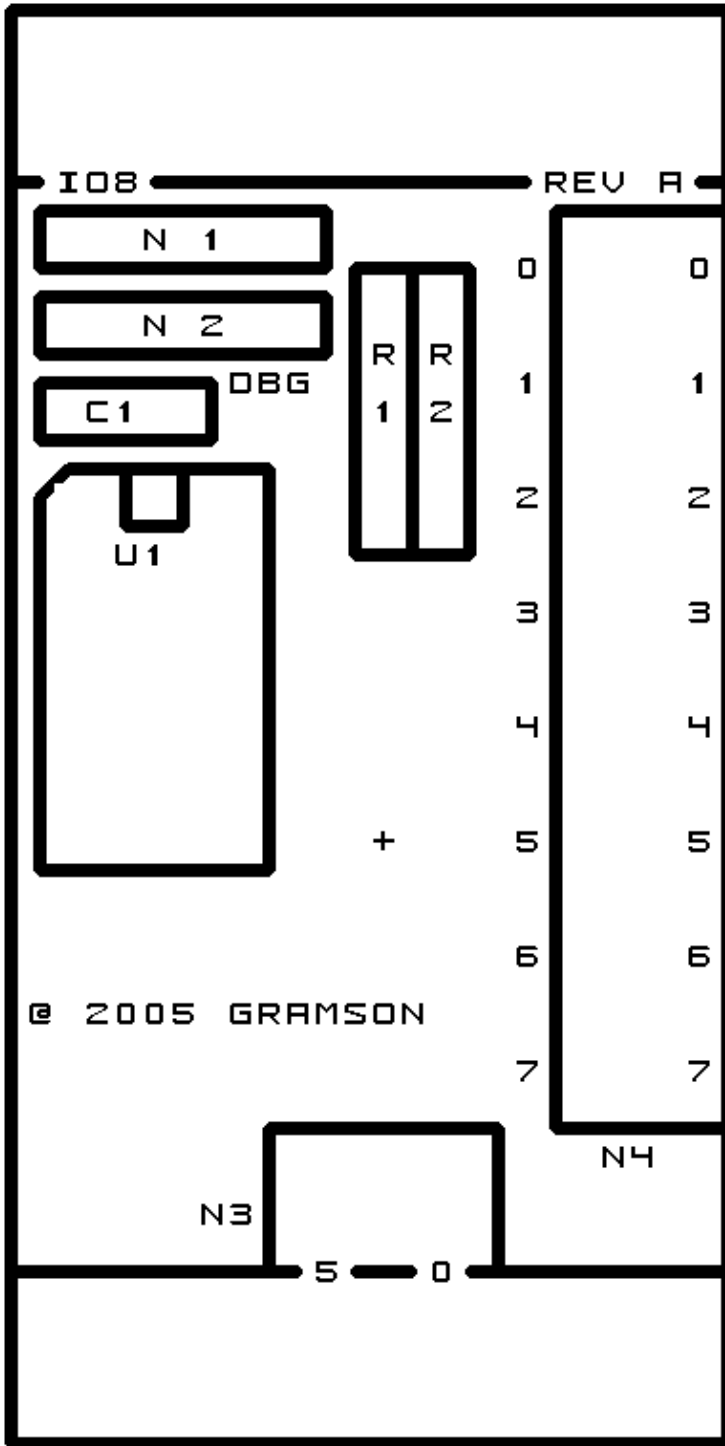
Copyright (c) 2005 by Wayne C. Gramlich. All rights reserved.



## A. Appendix A: Parts List

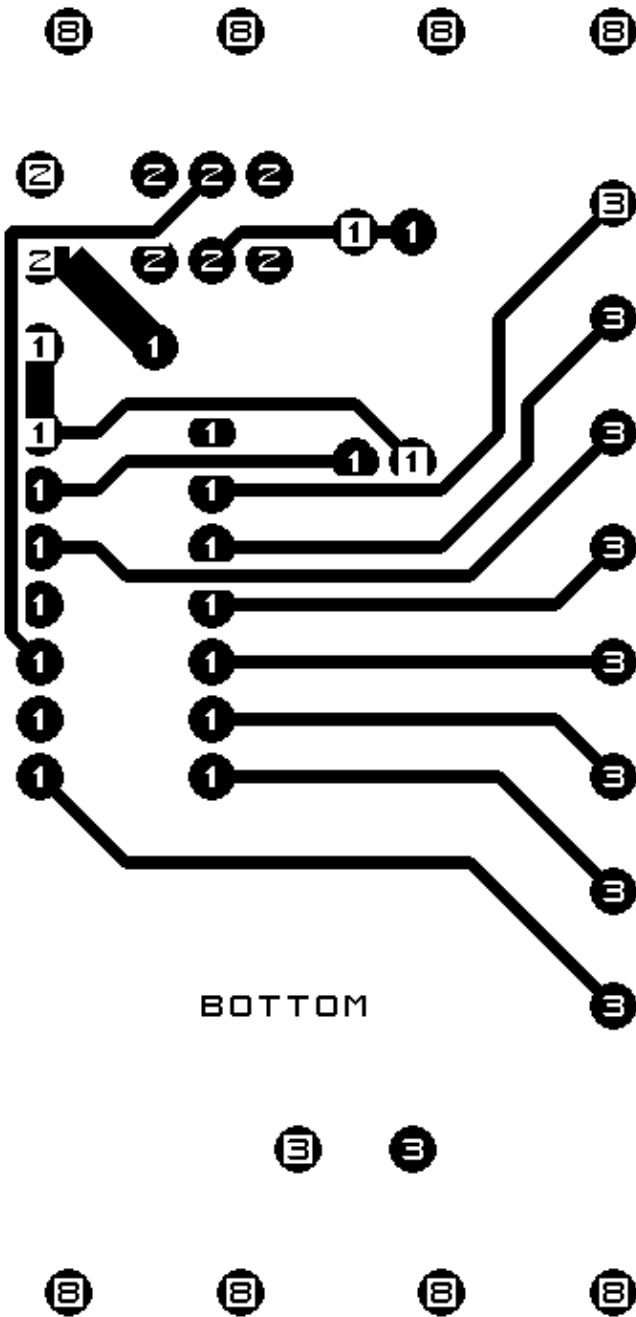
```
# Parts list for IO8 RoboBrix (Rev. A)
#
C1: Capacitor100nF - .1 uF Ceramic Capacitor [Jameco: 25524]
N1: Header1x5.Slave - 1x5 Male Header [5/40 Jameco: 160881]
N2: Header1x5.Debug2 - 1x5 Male Header [5/40 Jameco: 160881]
N3: TerminalStrip2.Power - 2 Junction Terminal Strip [Jameco: 189675]
N4: TerminalStrip8 - 8 Junction Terminal Strip [4 Jameco: 189675]
R1-2: Resistor10K - 10K Ohm 1/4 Watt Resistor [Jameco: 29911]
U1: PIC16F688.IRDistance8 - Microchip PIC16C688 [Digikey: PIC16F688-I/P-ND]
```

## B. Appendix B: Artwork Layer





### C. Appendix C: Back (Solder Side) Layer



### D. Appendix D: Front (Component Side) Layer

