This is the Revision C verion of the IRProximity2 module. The status of this project is work in progress.

IRProximity2 Module (Revision C)

Table of Contents

This document is also available in PDF format.

1. Introduction

The IRProximity2 module provides the ability to sense the proximity of objects that are reflective at InfraRed (IR) wave lengths. In general, objects that have light colors (e.g. white) tend to be more visible at IR wave lengths than objects with dark colors (e.g. black.) It consists of a single IR receiver module with two IR illumination LED's on either side −− left and right. By alternately lighting the front and back IR illumination LED's, it is possible to determine whether an object is approximatley to the left, center, or right of the the IRProximity2. In addition it is possible to get an extremely rough idea of how far to the left, right, or center the object is. Basically, an object can be 1) out of detection range, 2) barely detectable, 3) intermediate detectable, 4) solidly detectable. The range of each IR illuminatation LED can be independently adjusted by separate trim potentiometers. The detection range can be from about 1cm to 10cm. There are two green output LED's for each side than can be jumpered on or off by the user to suggest what is currently visible.

2. Programming

The IRProximity2 works by sending out a .6ms pulse of IR at 38kHz. It then measures the width of the returned pulse. In general, the wider the returned pulse, the closer (or whiter) the object is. This pulse width comes back as a number between 0 and 63. The repeatability of the pulse width is terrible and is subject to a great deal of jitter. To counter this jitter, a rolling average average is computed to smooth out the jitter. This rolling average is manipulated to provide a number between 0 and 255, where larger numbers represent closer (or whiter). Important: do not go looking for 8−bits of distance feedback; it simply is not there. The best you are going to do is with the high order 3−bit bits and ignore the rest as jitter.

There are two threshold registers for each side. The first threhold register is called the far threshold and the second threshold is called the near threshold. These threshold registers are used to control the lighing of the on−board green feedback LED's and the interrupt system (describe further below.) Each side has two green feedback LED's. The first one is called the far LED and will lit up when the average return value for the side exceeds the far threshold register. The second one is called the near LED and will be lit up when the average exceeds the near threshold register. Thus, the green feedback LED's provide the user with 3 states −− on/on = near, on/off = far, off/off = out of range.

All these green feedback and infraread illumination LED's can burn up battery power. To save power the user

can jumper the green LED's off. To further save power, the user can programmatically enable and dislable the left and right illumination LED's. Furthermore, a user setable delay can be introduced to reduce the sampling frequency and thereby reduce overall power consumption. By default, the illumination LED's are on and the delay is set to 0. The delay can be as high as 255; it is independently setable for the left and right sides.

As usual with input sensors, the IRProximity2 sensor supports the interrupt protocol. With the interrupt protocol, the user can specify conditions at which to generate an interrupt and then leave the IRProximity2 alone. For example, if the user only wants to know when an object comes into range, they can set up an interrupt condition to do so. The two threshold registers for each side partition the average value into 1) between 255 and high threshold, 2) between high and low thresholds , and 3) between low threshold and 0. The user can independently set a 3−bit mask that selects which of the conditions he/she is interested. Since there are both left and a right sides, there are a total of 6−bits in the interrupt mask. By default, the inerrupt enable bit is off and all 6−bits of the interrupt mask are zero.

Command	Send/ Receive	Byte Value								Discussion
		7	**6**	**5**	**4**	**3**	**2**	**1**	**0**	
Read LED's	Send	0	0	0	0	0	0	0	0	Return the four bits *abcd* that correspond to the LED's (*a*=left far, *b*=left near, *c*=right far, *d*=right near).
	Receive	0	0	0	0	*a*	*b*	*c*	*d*	
Read Left/Right Averages High Nibble	Send	0	0	0	0	0	0	0	1	Return the high order 4−bits of the left and right averages as *llllrrrr* where *llll* is the left nibble and *rrrr* is the right nibble.
	Receive	*l*	*l*	*l*	*l*	*r*	*r*	*r*	*r*	
Read Enables	Send	0	0	0	0	0	0	1	0	Return the illumination enable bits *l* and *r*.
	Receive	0	0	0	0	0	0	*l*	*r*	
Read Interrupt Mask	Send	0	0	0	0	0	0	1	1	Return the right average as *rrrrrrrr*.
	Receive	0	0	0	0	0	0	*l*	*r*	
Read Left Average	Send	0	0	0	0	0	1	0	0	Return the left average as *llllllll*.
	Receive	*l*	*l*	*l*	*l*	*l*	*l*	*l*	*l*	
Read Right Average	Send	0	0	0	0	0	1	0	1	Return the right average as *rrrrrrrr*.
	Receive	*r*	*r*	*r*	*r*	*r*	*r*	*r*	*r*	
Read Left Latest	Send	0	0	0	0	0	1	1	0	Return the left latest as *llllll*00.
	Receive	*l*	*l*	*l*	*l*	*l*	*l*	0	0	
Read Right Latest	Send	0	0	0	0	0	1	1	1	Return the right latest as *rrrrrr*00.
	Receive	*r*	*r*	*r*	*r*	*r*	*r*	0	0	
Read Left Far Threshold	Send	0	0	0	0	1	0	0	0	Return the left far threshold as *llllllll*.
	Receive	*l*	*l*	*l*	*l*	*l*	*l*	*l*	*l*	
Read Right Far Threshold	Send	0	0	0	0	1	0	0	1	Return the right far threshold as *rrrrrrrr*.
	Receive	*r*	*r*	*r*	*r*	*r*	*r*	*r*	*r*	
Read Left Near Threshold	Send	0	0	0	0	1	0	1	0	Return the left near threshold as *llllllll*.
	Receive	*l*	*l*	*l*	*l*	*l*	*l*	*l*	*l*	
Read Right Near Threshold	Send	0	0	0	0	1	0	1	1	Return the right near threshold as *rrrrrrrr*.
	Receive	*r*	*r*	*r*	*r*	*r*	*r*	*r*	*r*	
Read Left Delay	Send	0	0	0	0	1	1	0	0	Return the left delay as *llllllll*.

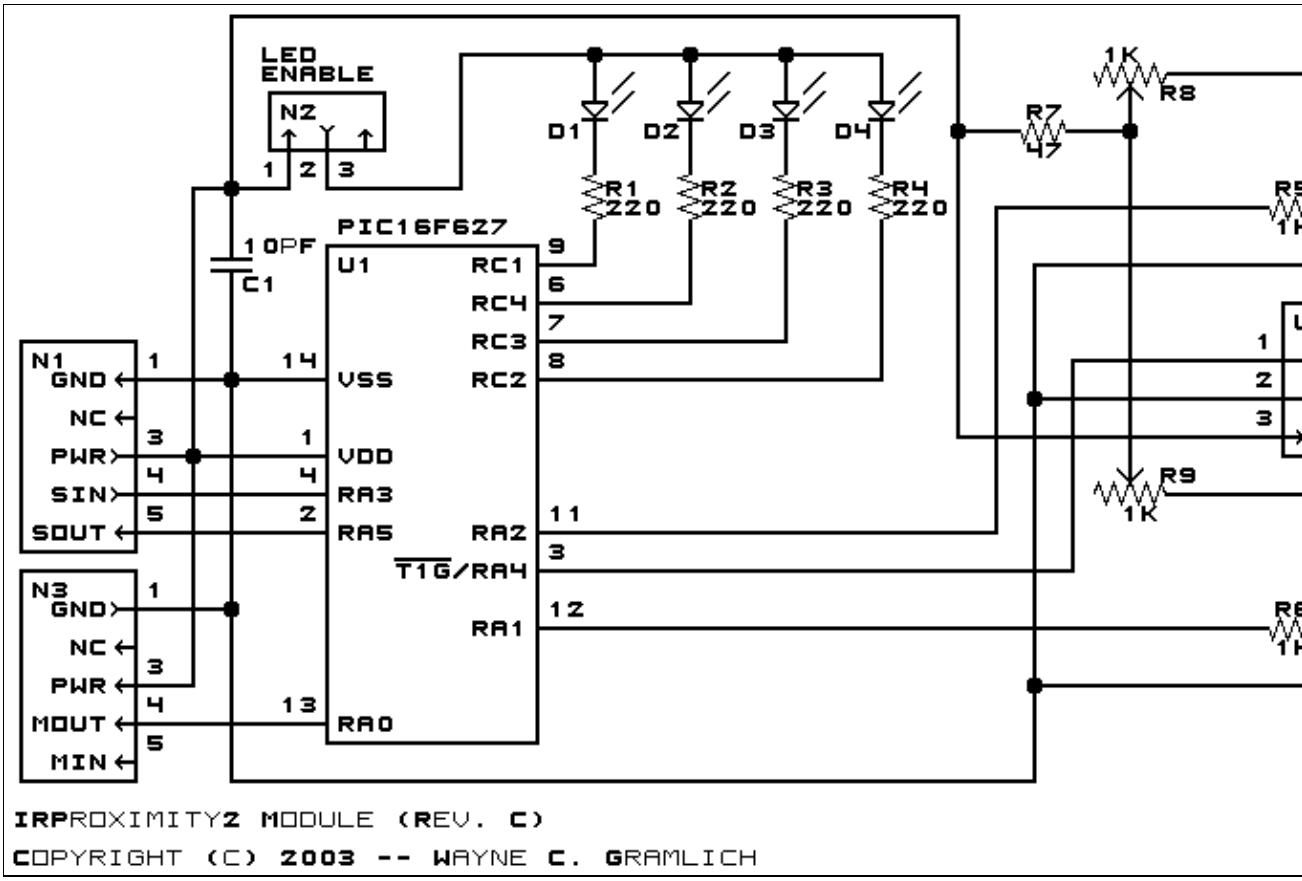
	Receive	*l*	*l*	*l*	*l*	*l*	*l*	*l*	*l*	
Read Right Delay	Send	0	0	0	0	1	1	0	1	Return the right delay as *rrrrrrrr*.
	Receive	*r*	*r*	*r*	*r*	*r*	*r*	*r*	*r*	
Set Enables	Send	0	0	0	1	0	0	*l*	*r*	Set the left enable to *l* and right enable to *r*. (1=on & 0=off)
Set Interrupt Mask	Send	0	0	0	1	0	1	0	0	Set interrupt mask to *lllrrr* where where *lll* is the left interrupt mask and *rrr* is the right interrrupt mask.
	Send	0	*l*	*l*	*l*	0	*r*	*r*	*r*	
Set Left Far Threshold	Send	0	0	0	1	1	0	0	0	Set Left Far Threshold to *llllllll*.
	Send	*l*	*l*	*l*	*l*	*l*	*l*	*l*	*l*	
Set Right Far Threshold	Send	0	0	0	1	1	0	0	1	Set Right Far Threshold to *rrrrrrrr*.
	Send	*r*	*r*	*r*	*r*	*r*	*r*	*r*	*r*	
Set Left Near Threshold	Send	0	0	0	1	1	0	1	0	Set Left Near Threshold to *llllllll*.
	Send	*l*	*l*	*l*	*l*	*l*	*l*	*l*	*l*	
Set Right Near Threshold	Send	0	0	0	1	1	0	1	1	Set Right Near Threshold to *rrrrrrrr*.
	Send	*r*	*r*	*r*	*r*	*r*	*r*	*r*	*r*	
Set Left Delay	Send	0	0	0	1	1	1	0	0	Set left delay to *llllllll*.
	Send	*l*	*l*	*l*	*l*	*l*	*l*	*l*	*l*	
Set Right Delay	Send	0	0	0	1	1	1	0	1	Set right delay to *rrrrrrrr*.
	Send	*r*	*r*	*r*	*r*	*r*	*r*	*r*	*r*	
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 IRProximity2 module is shown below:



The parts list kept in a separate file −− irproximity2.ptl.

3.2 Printed Circuit Board

The printed circuit board files are listed below:

irproximity2_back.png
: The solder side layer.

irproximity2_front.png
: The component side layer.

irproximity2_artwork.png
: The artwork layer.

irproximity2.gbl
: The RS−274X "Gerber" back (solder side) layer.

irproximity2.gtl
: The RS−274X "Gerber" top (component side) layer.

irproximity2.gal
: The RS−274X "Gerber" artwork layer.

irproximity2.drl
: The "Excellon" NC drill file.

irproximity2.tol
: The "Excellon" tool rack file.

4. Software

The following software is available:

irproximity2.ucl
: The µCL file that contains the firmware for the IRProximity2.

irproximity2.asm
: The assembly file for the IRProximity2.

irproximity2.lst
: The listing file for the IRProximity2.

irproximity2.hex
: The Intel® Hex file for the IRProximity2.

5. Issues

The following fabrication issues came up:

- The artwork for D5 is wrong. The "+" should be on the lower pin.
- By design, IR Reciever has an enormous field of view. Without some amount of IR light shielding, no proximity detection occurs.
- The pots are turned in opposite directions to increase and decrease sensitivity. This needs to be fixed.

The following additions should be considered:

- Provide built in shielding.
- Provide the ability to change the IR LED power levels.
- Provide a way to mount the IR LED externally to get a wider separation.
