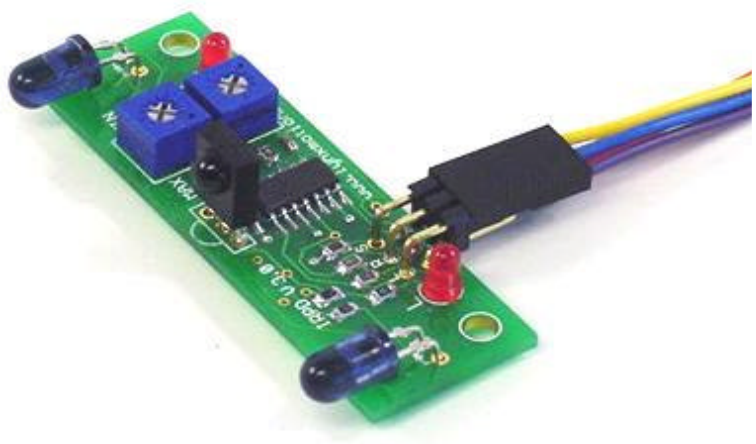
IRPD Ver 3.0

Infrared Proximity Detector



Lynxmotion, Inc.
PO Box 818
Pekin, IL 61555-0818
Tel: 309-382-1816 (Sales)
Tel: 309-382-2760 (Support)
Fax: 309-382-1254
E-m: sales@lynxmotion.com
E-m: tech@lynxmotion.com
Web: http://www.lynxmotion.com

Users Manual IRPD-01 Ver 8.0

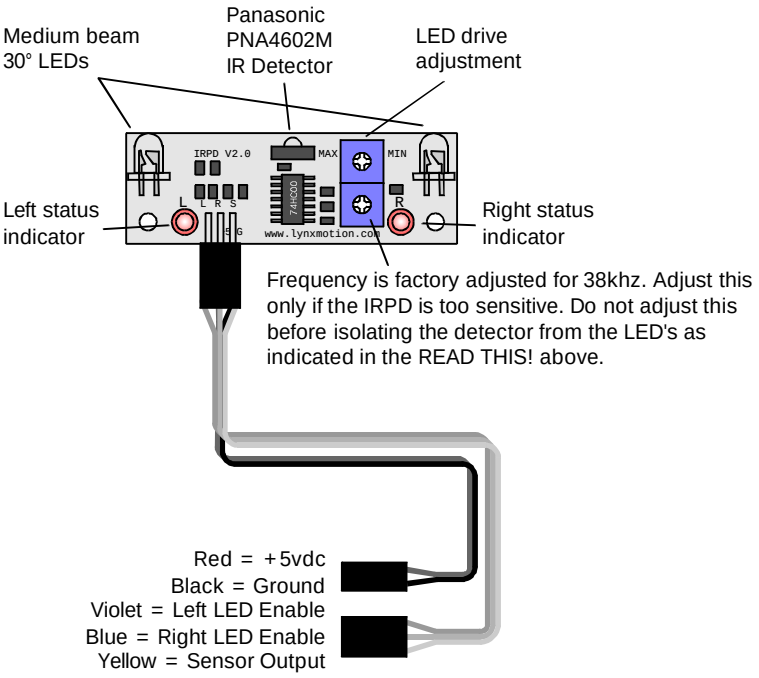
IRPD Ver 3.0

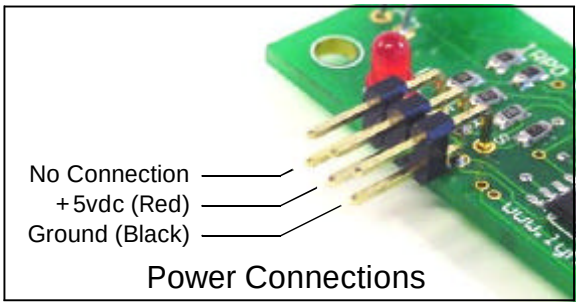
Caution! Read the manual completely before wiring and applying power to the board! Errors in wiring can damage the IRPD or the host microcontroller.

Caution! Do not reverse the power plug as damage to the sensor board will occur. These sensors are fully tested before they leave the factory. If the sensor is damaged by reverse power the warranty is void.

READ THIS!

Notice! The Panasonic IR Detector can sense IR light directly from the LEDs. This will cause the IRPD to report false readings (detect objects when nothing is there.) To prevent this either put a small piece of black electrical tape on the back and sides of the IR Detector, or add a short section of black tubing to the LED's. This is required for all applications of the IRPD!





Power Connections

Using the IRPD

Mounting the IRPD

The PC board can be mounted to the base of the robot using 4-40 hardware and small stand-offs. The height of the sensor and the color of the furniture and flooring will determine the amount of tweaking necessary to get reliable results. The optimum height seems to be from about 4" to 6" from the ground. If the sensor is mounted lower, it may be necessary to point the LEDs higher, or add small directional tubes, such as heat shrink, to the LEDs. Keep the sensor mounted horizontally as close to the front of the robot as possible. There must not be any moving parts of the robot within the field of view, in order to prevent false triggering. Experimentation in the physical setup is the key to reliable operation.

Testing with a Microcontroller

The best way to accurately test the IRPD is to connect it to a microcontroller and write a control program. I have included general purpose Basic programs for the First Step and the Next Step microcontrollers. Follow this procedure to test and verify it's operation. If you have trouble, it will most likely be getting the detector to not "see" the floor. You may need to experiment with the alignment and positioning of the LEDs.

1) Apply power to the IRPD from the First Step or Next Step I/O bus.
2) Adjust the LED drive to the mid position between minimum and maximum.
3) Enter the correct program on page 6 into the Stamp editor and program the Stamp with it.
4) Position the IRPD so it is pointing away from any objects. The LEDs should be off.
5) Place your hand directly in front of the PC board about 4" away from the sensor. The status LEDs should be on steady. If not quickly remove power and double check your wiring!
6) Move your hand about 4" from center to the left. You should see the left LED on only.
7) Move your hand about 4" from center to the right. You should see the right LED on only.
8) Move your hand from side to side to see the field of vision.
9) Now move your hand slowly away from the detector and take note when the LEDs just start to flicker. This is how far the sensor can "see" an obstacle. If it is too far, adjust the LED drive pot to the right for less current to the IR LED, and less distance. If it is too close, adjust the LED drive pot to the left for more current to the LEDs and more distance.

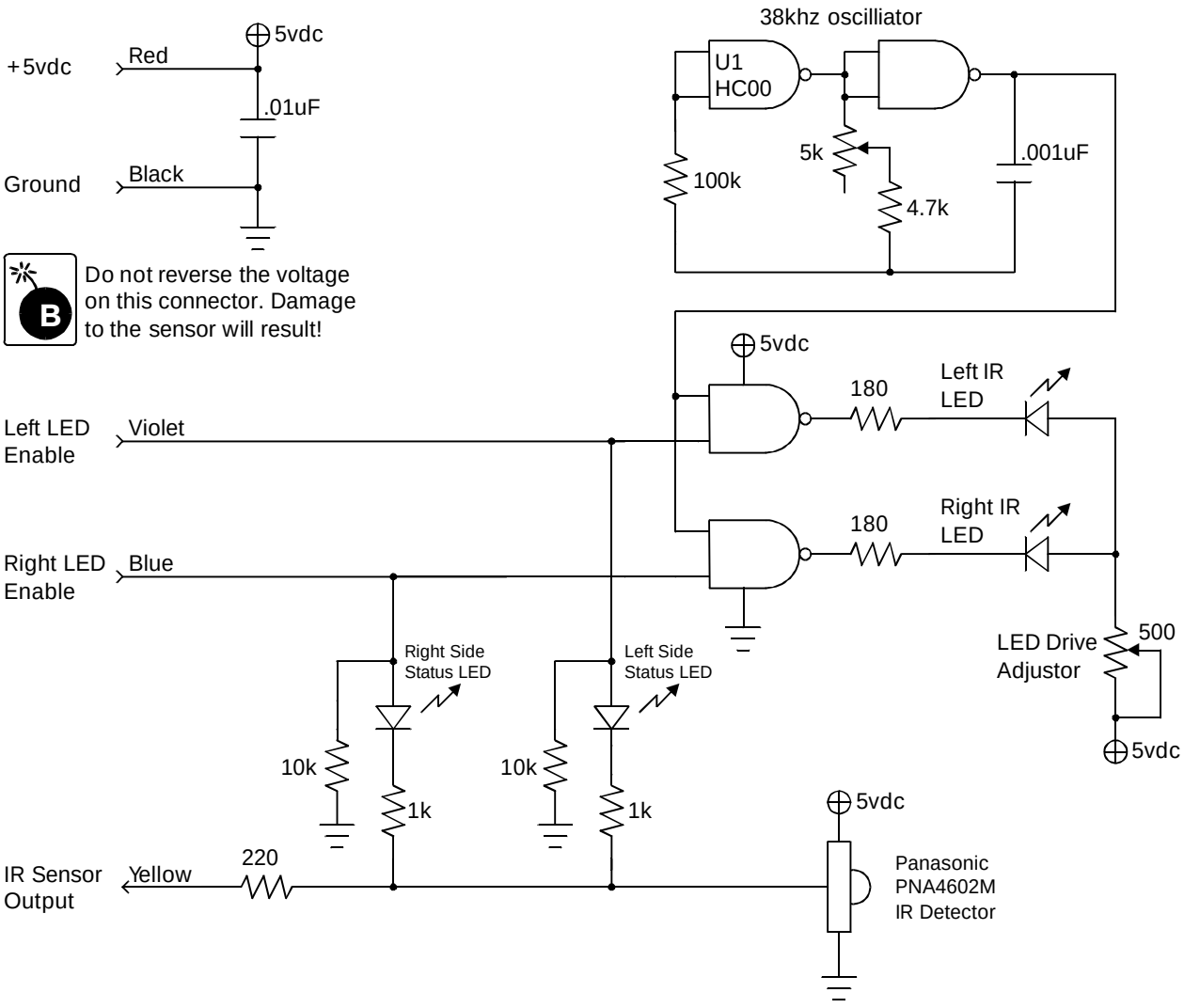
Testing without a Microcontroller

You can test the IRPD without a microcontroller by following these instructions.

1) Apply +5vdc and ground to the red and black wires.
2) Apply +5vdc to the left LED enable (blue) wire.
3) Adjust the LED drive to the mid position between minimum and maximum.
4) Mount the IRPD so it is pointing away from any objects. The LEDs should be off.
5) Move your hand about 4" from center to the left. You should see the left LED on only. The Left Status LED should be on. If not quickly remove power and double check your wiring!
6) Now move your hand slowly away from the detector and take note when the LED just starts to flicker. This is how far the sensor can "see" an obstacle. If it is too far, adjust the LED drive pot to the right for less current to the IR LED, and less distance. If it is too close, adjust the LED drive pot to the left for more current to the LEDs and more distance.
7) Remove the +5vdc to the left LED enable (blue) wire and apply +5vdc to the right LED enable (violet) wire. Repeat steps 5 and 6 for the right side.

There are programs for the IRPD specifically written for the robots we manufacture available from our website. Check the website for the most up to date IRPD programs. If you write an interesting program for the IRPD please submit it to Lynxmotion. We will put it on our website so others may benefit.

Schematic Diagram



Using the IRPD

Additional Information

The IRPD will "see" white objects from farther away than darker objects, so you need to take this into account when adjusting the sensor. If you have the resources, it would be a good idea to include a backup of bumper switches. Experiment with the code and have fun.

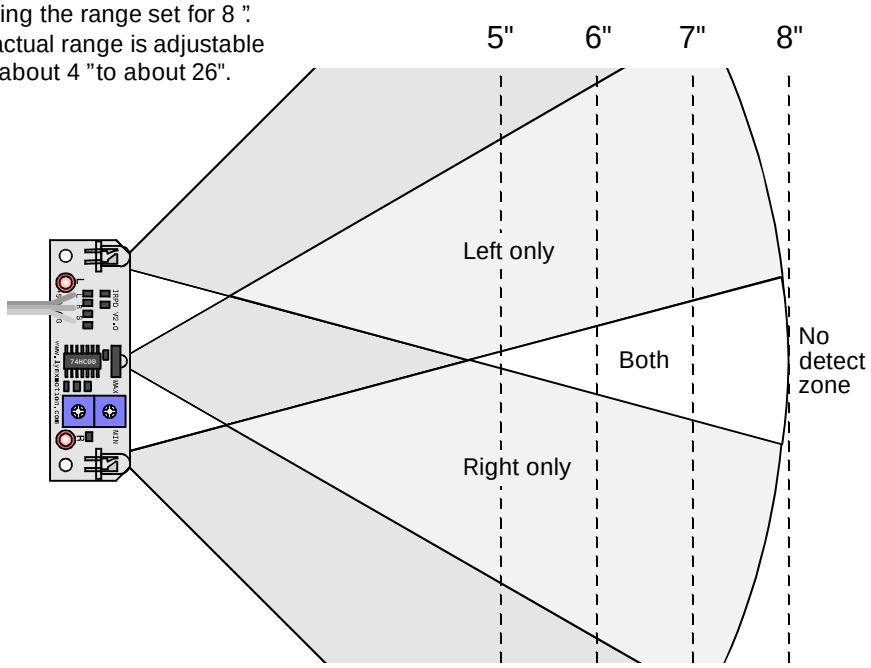
Other Uses

The IRPD can be used for purposes other than robotics. This sensor can be used whenever it is necessary to detect the presence of an object in a small area. Many people use it for automating their Halloween props. A little ingenuity will go a long way in creating an interesting display.

IRPD Truth Table

IRPD I/O pins / LEDs	Not Allowed	With obstacle			Without obstacle		
Left enable (blue)	~~High~~	Low	High	Low	Low	High	Low
Right enable (violet)	~~High~~	Low	Low	High	Low	Low	High
Left status LED	????	Off	On	Off	Off	Off	Off
Right status LED	????	Off	Off	On	Off	Off	Off
IR detector (yellow)	????	High	Low	Low	High	High	High

Note: This is an illustration showing the range set for 8 ". The actual range is adjustable from about 4 "to about 26".



Programming Examples

```
' Program name: IRPD.BS2 for Basic Stamp 2
' This is the demo program to verify proper
' operation of the IRPD.

'Connections;
'IRPD left LED to I/O pin10
'IRPD right LED to I/O pin11
'IRPD IR Sensor output to I/O pin12

'Initialize Symbols
templ   var bit
tempr   var bit
action  var nib

pause 1000          'wait for IR to settle.
start:

high 11 : pause 1
tempr = in12
low 11

high 10 : pause 1
templ = in12
low 10

action = 0
   if templ=1 then cont1 'If 1 then nothing
   action = action + 1   'is on the left.
cont1:
   if tempr=1 then cont2 'If 1 then nothing
   action = action + 2   'is on the right.
cont2:

'action = 0, nothing in the way
'action = 1, something in front of left side
'action = 2, something in front of right side
'action = 3, something in front of both sides

branch action,[forward,turn_r,turn_l,backup]
goto start

forward:
'Place code to make robot go forward here.
goto start

Turn_r:
'Place code to make robot go right here.
goto start

turn_l:
'Place code to make robot go left here.
goto start

backup:
'Place code to make robot go backward, then
'turn here.
goto start
```

```
'Program name: SMOOTH01.BS2 for Basic Stamp 2
'This program controls the Carpet Rover with a
'very fluid motion. The routine constantly
'checks the IRPD and increases or decreases the
'speed accordingly. The program operates as a
'single loop and is extremely efficient. This
'code assumes that the robots left servo has
'the internal motor leads reversed so both
'servos have the same response to similar
'values. Servos must be calibrated to stop with
'1.5mS pulses applied.

'Pinout
'pin0 = Left servo motor
'pin1 = Right servo motor
'pin10 = IRPD left LED enable
'pin11 = IRPD right LED enable
'Pin12 = IRPD sensor output

templ var bit
tempr var bit
left var byte
right var byte

left = 150  'Begin with a stopped vehicle.
right = 150

Start:    'Check the IRPD for obstacles
  high 10: pause 1
Templ=in12 'a 0 means an obstacle is there.
  low 10: pause 1
  high 11: pause 1
Tempr=in12 'a 1 means an obstacle is not there.
  low 11

'Each pass adjusts servo speed slightly.
'Loop time is approx. 20mS.
'Adding to value builds toward forward.
'Subtracting from value builds towards reverse.

branch tempr,[sub_left, add_left]

sub_left:
    left = left - 1 min 130
    goto done_left
add_left:
    left = left + 1 max 170
done_left:

branch templ,[sub_right, add_right]

sub_right:
    right = right - 1 min 130
    goto done_right
add_right:
    right = right + 1 max 170
done_right:

'Output the servo pulses.
pulsout 0, (left * 5)
pulsout 1, (right * 5)
pause 10
goto start'Back to the top.
```