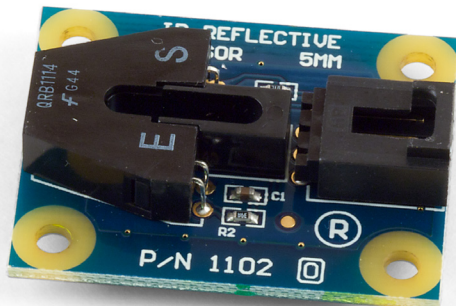




# Product Manual

## 1102 - IR Reflective Sensor 5mm



**Phidgets 1102 - Product Manual**

**For Board Revision 0**

**© Phidgets Inc. 2009**

# Contents

## **4 Product Features**

- 4 Applications
- 4 Connections
- 4 Type of Measurement

## **5 Getting Started**

- 5 Checking the Contents
- 5 Connecting all the pieces
- 5 Testing Using Windows 2000/XP/Vista
- 7 Testing Using Mac OS X

## **7 Programming a Phidget**

- 7 Code Samples
- 7 Coding for your Sensor

## **8 Technical Information**

- 8 Other Interfacing Alternatives
- 8 Analog Input Cable Connectors
- 9 Mechanical Drawing
- 9 Device Specifications

## **9 Product History**

## **9 Support**

# Product Features

---

## Applications

- Based on the QRB1114
- Can detect an object at 5mm
- Can be used to determine the difference between low reflective conditions and high reflective conditions
- Often used with a white/black pattern to detect movement, or to detect if a reflective object is close

## Connections

Designed to connect to a:

- 1018 - PhidgetInterfaceKit 8/8/8
- 1019 - PhidgetInterfaceKit 8/8/8 w/6 Port Hub
- 1070 - PhidgetSBC
- 1202 - PhidgetTextLCD

## Type of Measurement

The sensor uses ratiometric measurement.

# Getting Started

---

## Checking the Contents

### You should have received:

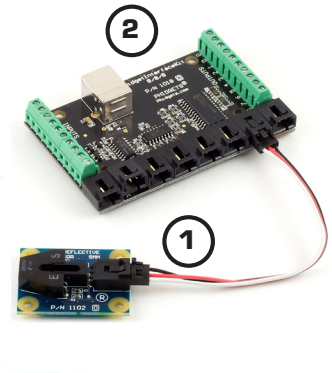
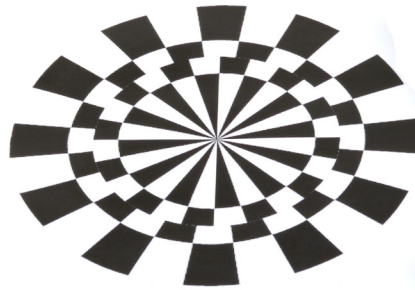
- An IR Reflective Sensor 5mm
- A Sensor Cable

### In order to test your new Phidget you will also need:

- A PhidgetInterfaceKit 8/8/8 or a PhidgetTextLCD
- A USB Cable

## Connecting all the pieces

1. Connect the IR Reflective Sensor 5mm to the Analog Input 6 on the PhidgetInterfaceKit 8/8/8 board using the sensor cable.
2. Connect the PhidgetInterfaceKit to your PC using the USB cable.




## Testing Using Windows 2000/XP/Vista

### Downloading the Phidgets drivers


Make sure that you have the current version of the Phidget library installed on your PC. If you don't, do the following:

Go to [www.phidgets.com](http://www.phidgets.com) >> Drivers


Download and run Phidget21 Installer (32-bit, or 64-bit, depending on your PC)

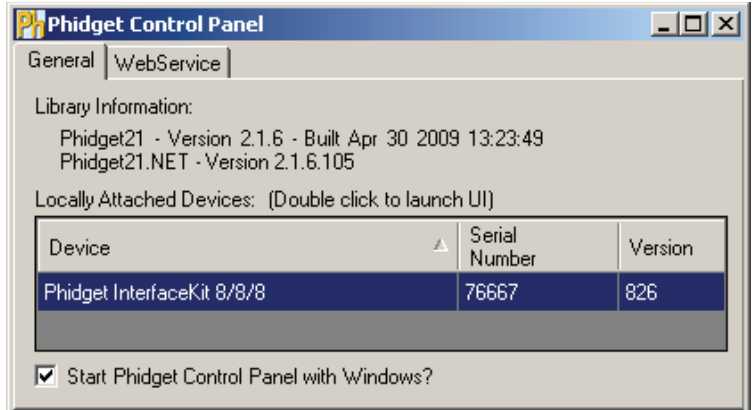
You should see the  icon on the right hand corner of the Task Bar.

### Running Phidgets Sample Program

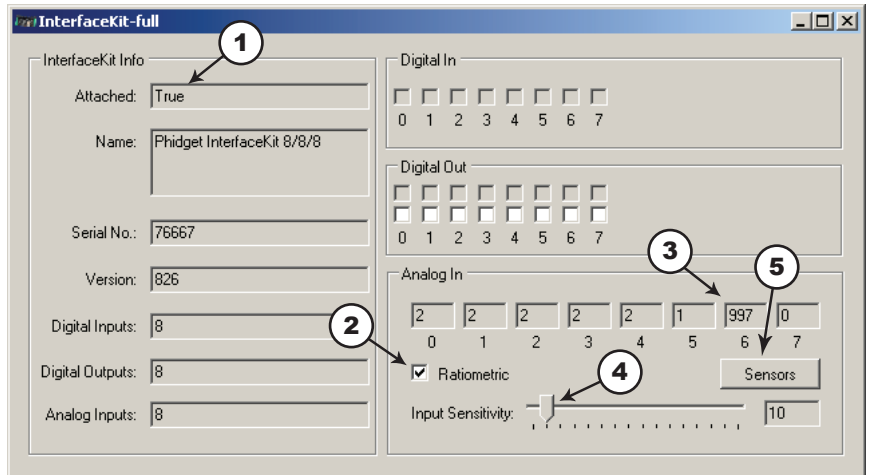
Double clicking on the  icon loads the Phidget Control Panel; we will use this program to make sure that your new Phidget works properly. Since the sensor is connected to a 1018, the computer will see only the 1018. The sensor is providing data through the Analog input it is connected to.

The source code for the InterfaceKit-full sample program can be found under C# by clicking on [www.phidgets.com](http://www.phidgets.com) >> Programming.

Double Click on the  icon to activate the Phidget Control Panel and make sure that the **Phidget InterfaceKit 8/8/8** is properly attached to your PC.

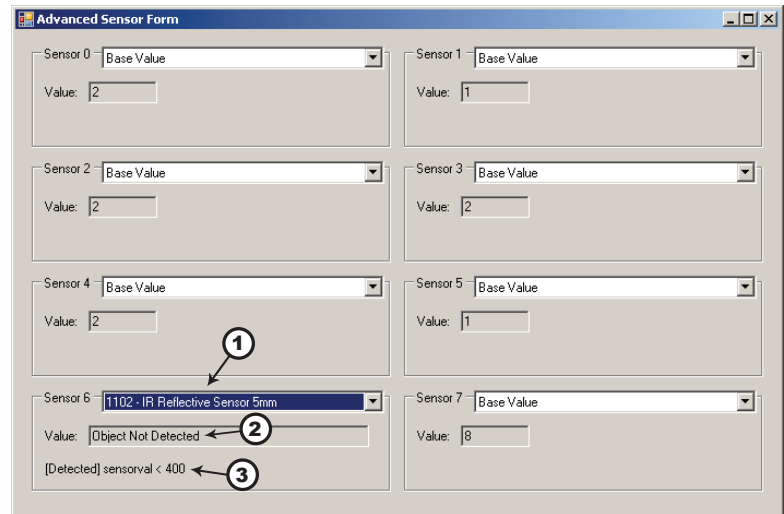


1. Double Click on **Phidget InterfaceKit 8/8/8** in the Phidget Control Panel to bring up InterfaceKit-full and check that the box labelled Attached contains the word True.
2. Make sure that the Ratiometric box is Ticked.
3. Place a piece of reflective material close (5mm) to the sensor and check the value in the Analog In box. Any value less than 400 means that the sensor has detected the object. If you want to use a pattern like the one we used in the Connecting the Hardware picture, do a search on the Web for "optical encoder pattern" and specify images.



4. You can adjust the input sensitivity by moving the slider pointer.
5. Click on the Sensors button to bring up the Advanced Sensor Form.

1. In the Sensor 6 box, select the 1102 - IR Reflective Sensor 5mm from the drop down menu.
2. The sensor's detection state for an object will be shown here.
3. Formula used to convert the analog input SensorValue into a detection state.



## Testing Using Mac OS X

- Click on System Preferences >> Phidgets (under Other) to activate the Preference Pane
- Make sure that the Phidget InterfaceKit 8/8/8 is properly attached.
- Double Click on Phidget InterfaceKit 8/8/8 in the Phidget Preference Pane to bring up the InterfaceKit-Full example. This example will function in a similar way as the Windows version, but note that it does not include an Advanced Sensor Display.

## Programming a Phidget

---

Phidgets' philosophy is that you do not have to be an electrical engineer in order to do projects that use devices like sensors, motors, motor controllers, and interface boards. All you need to know is how to program. We have developed a complete set of Application Programming Interfaces (API) that are supported for Windows, Mac OS X, and Linux. When it comes to languages, we support VB6, VB.NET, C#.NET, C, C++, Flash 9, Flex, Java, LabVIEW, Python, Max/MSP, and Cocoa.

### Code Samples

We have written sample programs to illustrate how the APIs are used.

Due to the large number of languages and devices we support, we cannot provide examples in every language for every Phidget. Some of the examples are very minimal, and other examples will have a full-featured GUI allowing all the functionality of the device to be explored. Most developers start by modifying existing examples until they have an understanding of the architecture.

Go to [www.phidgets.com](http://www.phidgets.com) >> Programming to see if there are code samples written for your device. Find the language you want to use and click on the magnifying glass besides "Code Sample". You will get a list of all the devices for which we wrote code samples in that language.

If this is your first time writing a program to control a Phidget, you should read the Getting Started Guide for the language you are planning to use.

### Coding for your Sensor

Phidget analog sensors do not have their own API, but instead their output is a voltage that is converted to a digital value and accessed through the SensorValue properties and events on a PhidgetInterfaceKit. It is not possible to programmatically identify which sensor is attached to the Analog Input. Your application will need to apply any formulas from this manual to the SensorValue to translate it into usable data.

See the PhidgetInterfaceKit product manual for an overview of its API and a description of our architecture.

# Technical Information

---

The Infrared sensor can detect an object at 5mm. It measures the amount of energy from the object and returns a value between 0 and 1000. A returned value between 0 and 400 signifies that the object has been detected. Values over 400 indicate that there is no reflective object within range, or that an object is too close.

The 1102 sensor consists of an infrared emitting diode and an NPN silicon phototransistor mounted side by side on a converging optical axis in a black plastic housing. The phototransistor responds to radiation from the emitting diode only when a reflective object passes within its field of view. The area of the optimum response approximates a circle 5mm in diameter.

The amount of reflectivity is measured by Infrared and some materials that look very reflective to the human eye might not be as reflective in the infrared spectrum. The sensor can only detect objects that are between 3 to 7 mm away; it cannot see any objects outside that range.

The 1102 is not a digital sensor. The returned value is inversely proportional to the amount of reflectivity of the object (0 being more reflective, and 400 being less reflective), but in practice the variation between sensors is broad enough that the 1102 should not be used to measure the reflectivity of an object - only to detect if the object exists.

When working with sensors that are not digital, it is helpful to filter out noise by implementing a simple hysteresis in your code. By interpreting any `SensorValue < 400` as a detection, and not releasing the detection until `SensorValue` goes above some higher threshold, such as 500, multiple triggering can often be avoided.

Finally, the 1102 works best in a constrained environment, where objects can be mechanically guaranteed to be within the 3-7mm range, or not present at all. The 1103 is a better choice as a general purpose "bump" sensor or human interface.

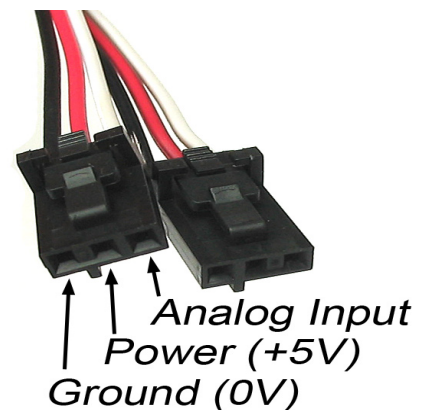
## Other Interfacing Alternatives

If you want maximum accuracy, you can use the `RawSensorValue` property from the `PhidgetInterfaceKit`. To adjust a formula, substitute `(SensorValue)` with `(RawSensorValue / 4.095)`

If the sensor is being interfaced to your own Analog to Digital Converter and not a Phidget device, our formulas can be modified by replacing `(SensorValue)` with `(Vin * 200)`. It is important to consider the voltage reference and input voltage range of your ADC for full accuracy and range.

## Analog Input Cable Connectors

Each Analog Input uses a 3-pin, 0.100 inch pitch locking connector. Pictured here is a plug with the connections labeled. The connectors are commonly available - refer to the Table below for manufacturer part numbers.



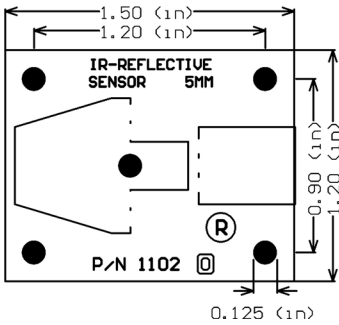


Cable Connectors		
Manufacturer	Part Number	Description
Molex	50-57-9403	3 Position Cable Connector
Molex	16-02-0102	Wire Crimp Insert for Cable Connector
Molex	70543-0002	3 Position Vertical PCB Connector
Molex	70553-0002	3 Position Right-Angle PCB Connector (Gold)
Molex	70553-0037	3 Position Right-Angle PCB Connector (Tin)
Molex	15-91-2035	3 Position Right-Angle PCB Connector - Surface Mount

Note: Most of the above components can be bought at [www.digikey.com](http://www.digikey.com)

## Mechanical Drawing

1:1 scale



**Note:** When printing the mechanical drawing, "Page Scaling" in the Print panel must be set to "None" to avoid re-sizing the image.

## Device Specifications

Characteristic	Value
Current Consumption	15mA
Output Impedance	10K ohms
Supply Voltage	3.0VDC to 5.25VDC

## Product History

Date	Board Revision	Comment
October 2005	n/a	Product Release

## Support

Call the support desk at 1.403.282.7335 8:00 AM to 5:00 PM Mountain Time (US & Canada) - GMT-07:00

or

E-mail us at: [support@phidgets.com](mailto:support@phidgets.com)