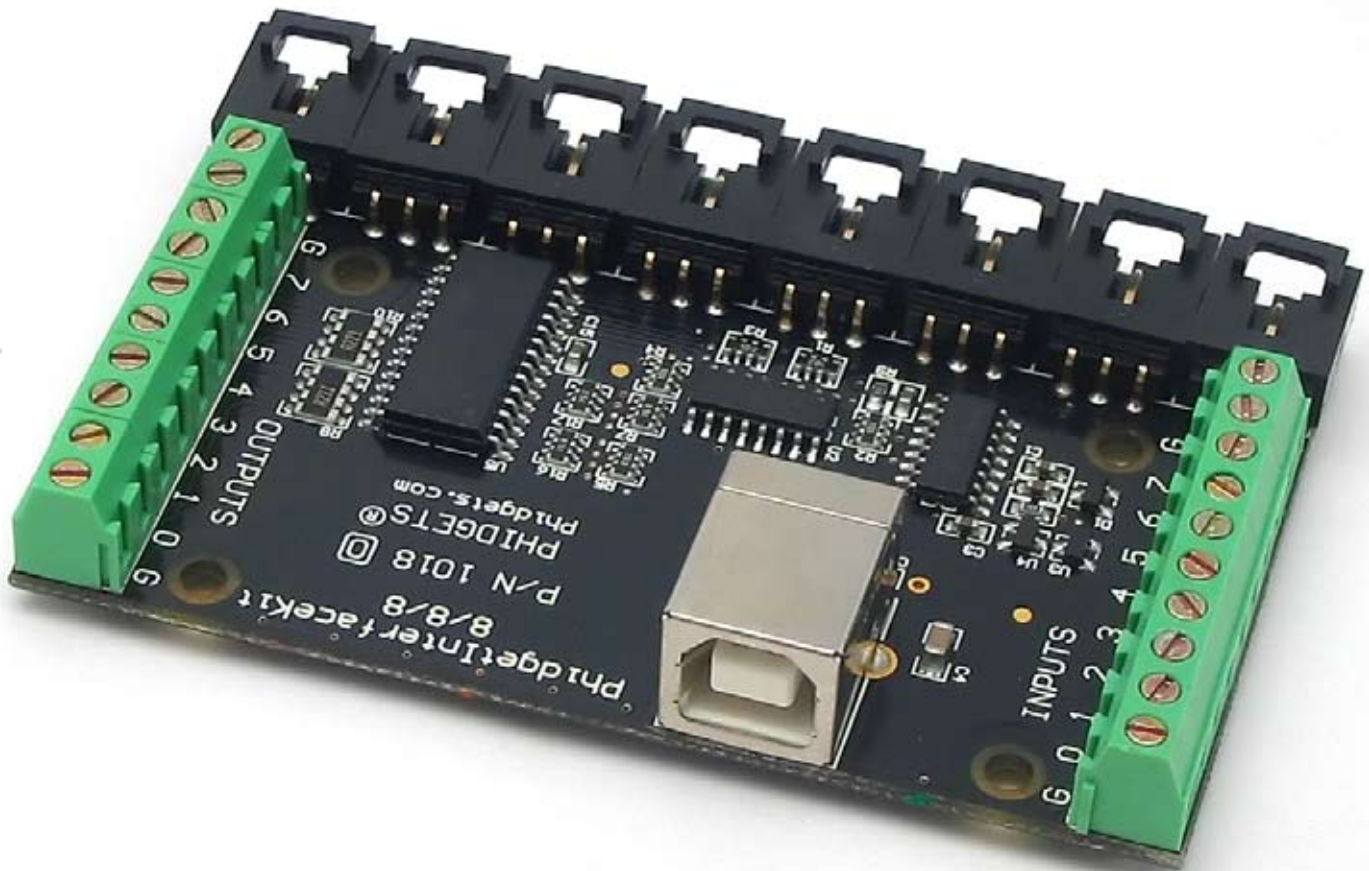# PhidgetInterfaceKit 8/8/8

Operating Systems:

Windows 2000/XP/Vista, Windows CE, Linux, and Mac OS X

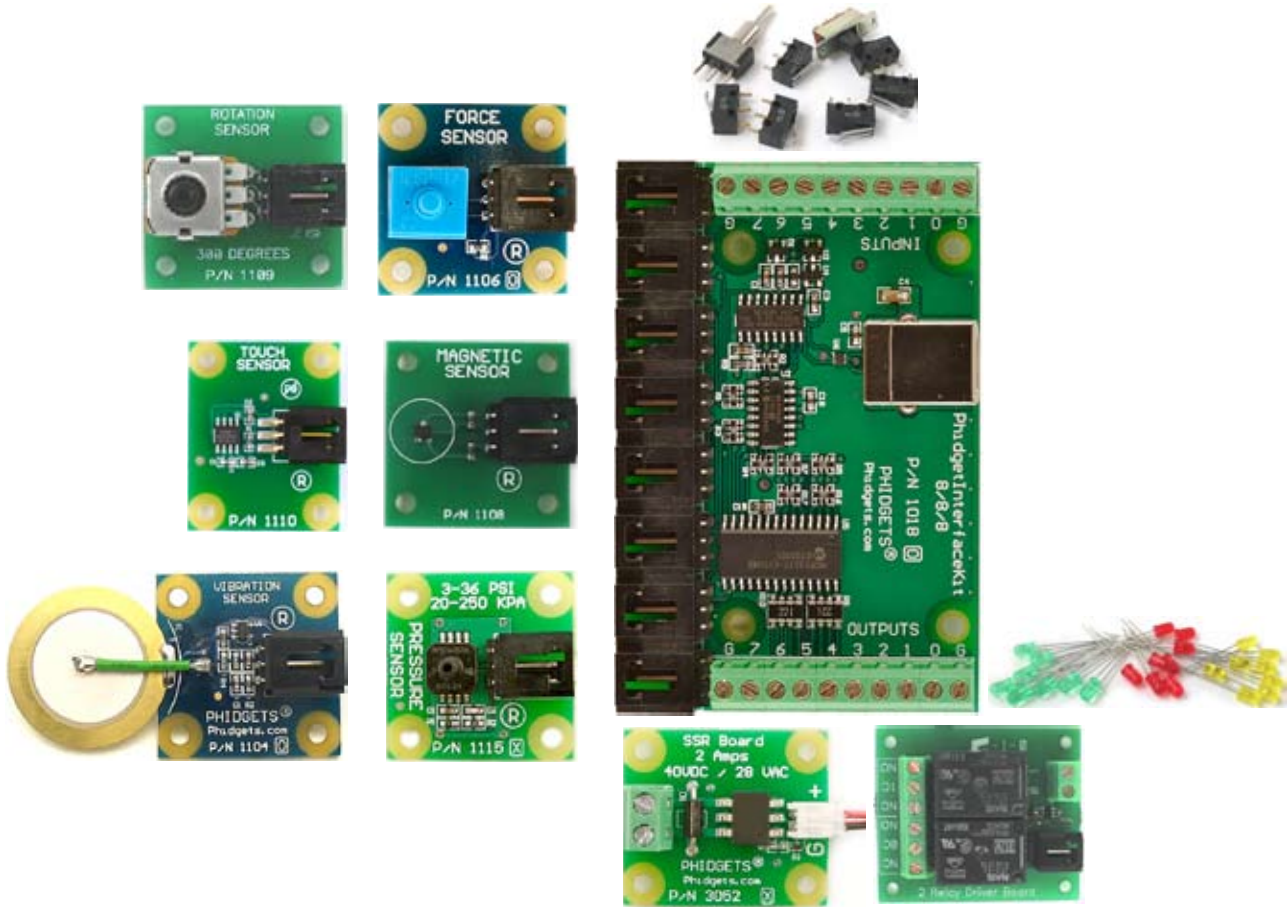## Application Programming Interfaces (APIs):

Visual Basic, VB.NET, C, C++, C#, Flash 9, Flex, Java, LabVIEW, and Matlab

Examples:

You will find program examples in the download section of www.phidgets.com

What Can the PhidgetInterfaceKit 8/8/8 Do?

The PhidgetInterfaceKit 8/8/8 allows you to connect devices to any of 8 analog inputs, 8 digital inputs and 8 digital outputs. It provides a generic, convenient way to interface your PC with various devices.

Analog inputs

They are used to measure continuous quantities, such as temperature, humidity, position, pressure, etc. Phidgets offers a wide variety of sensors that can be plugged directly into the board using the cable included with the sensor. Here is a list of sensors currently available:

IR Distance Sensor	IR Reflective Sensor	Vibration Sensor	Light Sensor
Force Sensor	Humidity Sensor	Temperature Sensor	Magnetic Sensor
Rotation Sensor	Voltage Divider	Touch Sensor	Motion Sensor
Mini Joy-Stick	Pressure Sensor	Voltage Sensor	Current Sensor
Slide Sensor			

Non Phidgets Sensors

In addition to Phidgets sensors, any sensor that returns a signal between 0 and 5 volts can be easily interfaced. Here is a list of interesting sensors that can be used with the PhidgetInterfaceKit 8/8/8. Note: these sensors are not "plug & play" like Phidgets sensors.

Manufacturer	Part Number	Description
MSI Sensors	FC21/FC22	Load cells - measure up to 100lbs of force
Humirel	HTM2500VB	Humidity sensors
Measurement Specialties	MSP-300	Pressure sensors - ranges up to 10,000 PSI
Freescale Semiconductor	MPXA/MPXH	Gas Pressure Sensors
Allegro	ACS7 series	Current Sensors - ranges up to 200 Amps
Allegro	A1300 series	Linear Hall Effect Sensors - to detect magnetic fields
Analog	TMP35 TMP36 TMP37	Temperature Sensor
Panasonic	AMN series	Motion Sensors

Note: Most of the above sensors can be bought at www.digikey.com.

Digital Inputs

Digital Inputs can be used to convey the state of push buttons, limit switches, relays (check out our Dual Relay Board), logic levels, etc...

Digital Outputs

Digital Outputs can be used to drive LEDs, solid state relays (have a look at our SSR board), transistors; in fact, anything that will accept a CMOS signal.

Digital outputs can be used to control devices that accept a +5V control signal.

With transistors and some electronics experience, other devices can be controlled, such as buzzers, lights, larger LEDs, relays.
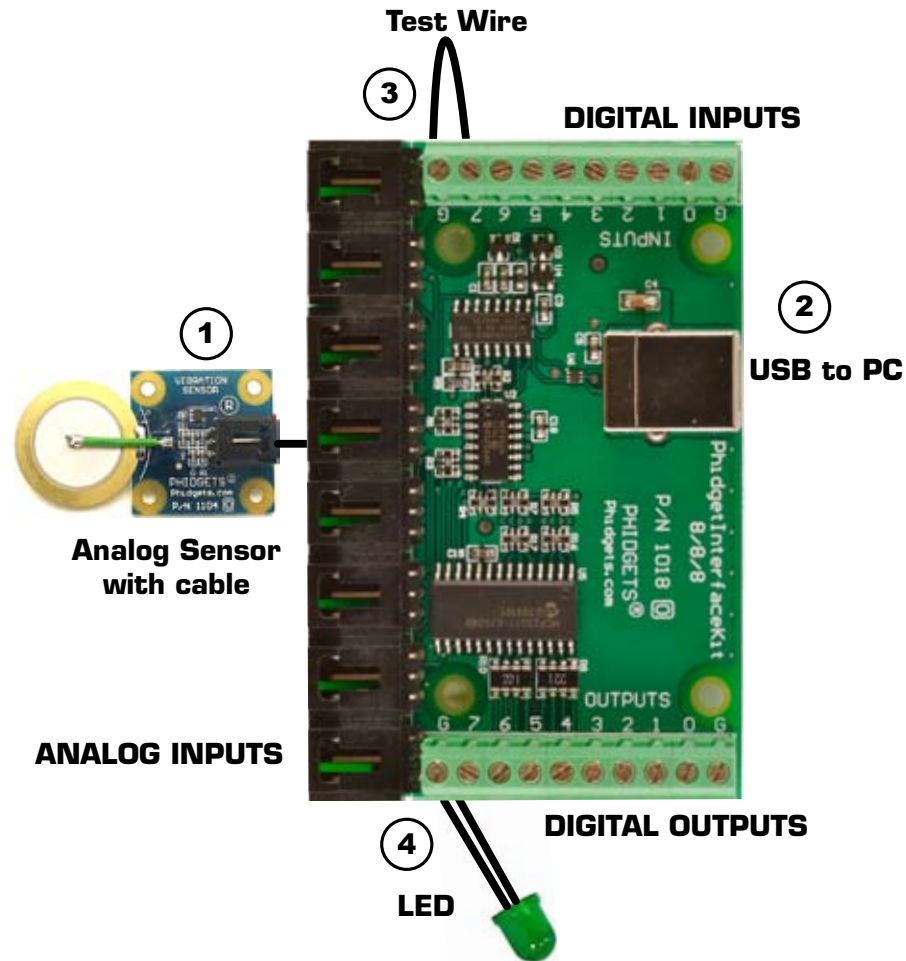
Getting Started

Installing the hardware

The kit contains:

• A PhidgetInterfaceKit 8/8/8

• A USB Cable

You will also need:

• A piece of wire to test the digital inputs

• An LED to test the digital outputs

• An Analog Sensor to test the analog inputs

1. Connect the Analog Sensor to any of the analog input ports (labeled 0 to 7) using a Phidgets sensor cable.

2. Connect the InterfaceKit board to the PC using the USB cable.

3. Connect one end of the wire to a digital input port and the other end to the ground connection.

4. Connect the LED to one of the digital output by inserting the long LED wire into any of the digital outputs (labeled 0 to 7) and the shorter wire to Ground.

Download and Install the software

Go to www.phidgets.com
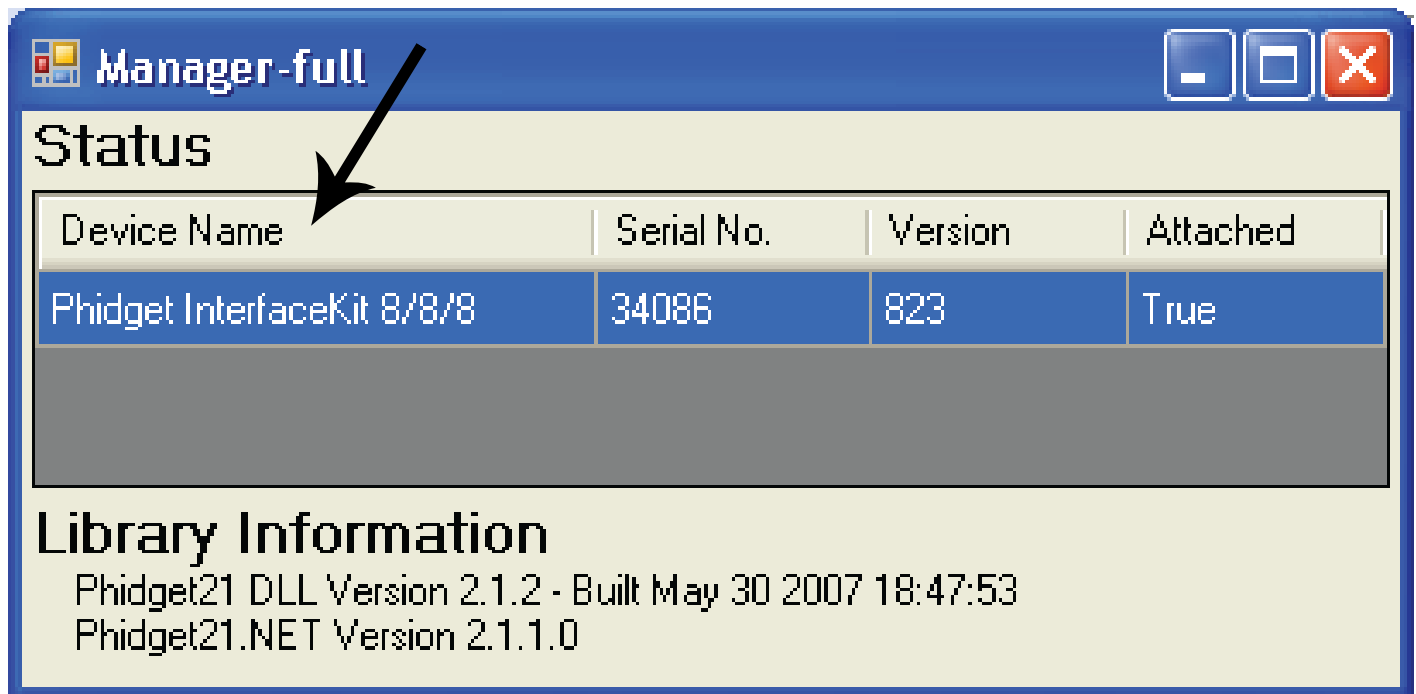
Click on Downloads

Select your operating system (Windows, Linux, MAC OS)

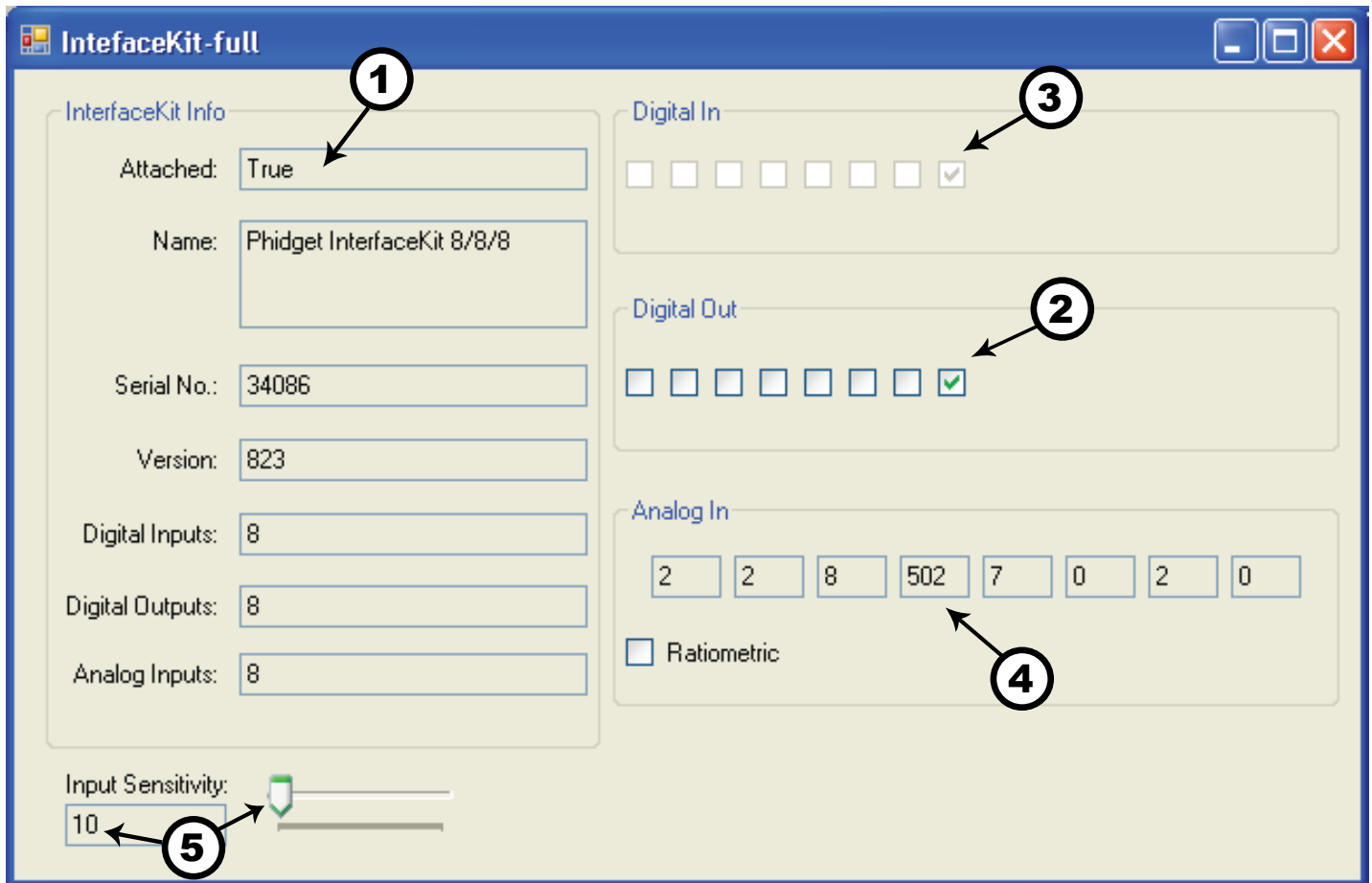Select the language you want to use and download the appropriate examples and Libraries.

Install the Libraries and decompress the Example file.

Testing the PhidgetInterfaceKit 8/8/8 using Windows

• Note that some examples are not available for Linux, Mac OSX or Windows CE.

• Make sure that you have installed the libraries and decompressed your example file.

Run the program **Manager-full** to make sure that the **PhidgetInterfaceKit** is properly connected to your PC.

1. Run the program ***InterfaceKit- full*** and check that the box labelled Attached contains the word True.

2. Test the digital output by clicking on the box to turn on the LED. Clicking again will turn the LED off.

3. Test the digital input by disconnecting the wire end connected to the digital input. connector. The tick mark in the box will go away.

4. Test the analog input sensor by observing the sensor value as you activate the Phidget sensor.

5. You can adjust the input sensitivity by moving the slider pointer.

Programming a Phidget

Where to get information

- Go to www.phidgets.com >> Downloads

- Select the Operating System and the language you want to use.

- Download the appropriate API manual and read the section under the ***InterfaceKit*** heading.

- Have a look at the source code of the ***InterfaceKit-full*** program.

- Have a look at the C# example below.

- Modify an existing program or write your own program from scratch.

Simple example written in C#

```
// - InterfaceKit simple -
// This simple example creates an Interfacekit object, hooks the event handlers, and opens it for connections to InterfaceKit
// Phidgets. It will then wait for user input to terminate, and in the meantime, display event generated data from the InterfaceKit.
// For a more detailed example, please see the InterfaceKit-full example.
//
// Please note that this example was designed to work with only one Phidget InterfaceKit connected.
// For an example using multiple Phidget InterfaceKits, please see a "multiple" example in the InterfaceKit Examples folder.
//
// Copyright 2007 Phidgets Inc. All rights reserved.
using System;
using System.Collections.Generic;
using System.Text;
using Phidgets; //needed for the interfacekit class and the phidget exception class
using Phidgets.Events; //needed for the event handling classes
namespace InterfaceKit_simple
{
class Program
{
//Declare an InterfaceKit object
static InterfaceKit ifKit;
static void Main(string[] args)
{
try
```

```csharp
{
//Initialize the InterfaceKit object
ifKit = new InterfaceKit();
//Hook the basica event handlers
ifKit.Attach += new AttachEventHandler(ifKit_Attach);
ifKit.Detach += new DetachEventHandler(ifKit_Detach);
ifKit.Error += new ErrorEventHandler(ifKit_Error);
//Hook the phidget spcific event handlers
ifKit.InputChange += new InputChangeEventHandler(ifKit_InputChange);
ifKit.OutputChange += new OutputChangeEventHandler(ifKit_OutputChange);
ifKit.SensorChange += new SensorChangeEventHandler(ifKit_SensorChange);
//Open the object for device connections
ifKit.open();
//Wait for an InterfaceKit phidget to be attached
Console.WriteLine("Waiting for InterfaceKit to be attached…");
ifKit.waitForAttachment();
//Wait for user input so that we can wait and watch for some event data frm the phidget
Console.WriteLine("Press any key to end…");
Console.Read();
//User input was rad so we'll terminate the program, so close the object
ifKit.close();
//set the object to null to get it out of memory
ifKit = null;
//If no expcetions where thrown at this point it is safe to terminate the program
Console.WriteLine("ok");
}
catch (PhidgetException ex)
{
Console.WriteLine(ex.Description);
}
}
//Attach event handler…Display the serial number of the attached InterfaceKit to the console
static void ifKit_Attach(object sender, AttachEventArgs e)
```
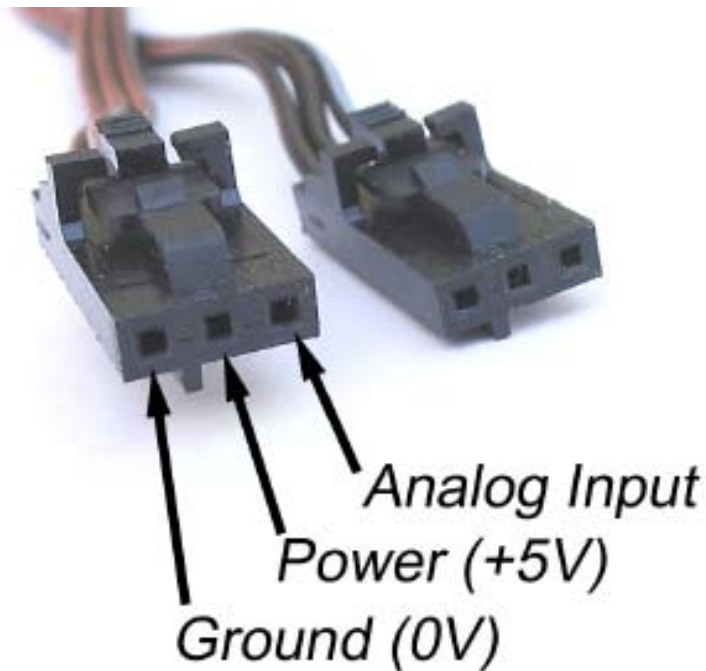
```
{

Console.WriteLine("InterfaceKit {0} attached!", e.Device.SerialNumber.ToString());

}

//Detach event handler...Display the serial number of the detached InterfaceKit to the console

static void ifKit_Detach(object sender, DetachEventArgs e)

{

Console.WriteLine("InterfaceKit {0} detached!", e.Device.SerialNumber.ToString());

}

//Error event handler...Display the error description to the console

static void ifKit_Error(object sender, ErrorEventArgs e)

{

Console.WriteLine(e.Description);

}

//Input Change event handler...Display the input index and the new value to the console

static void ifKit_InputChange(object sender, InputChangeEventArgs e)

{

Console.WriteLine("Input index {0} value {1}", e.Index, e.Value.ToString());

}

//Output change event handler...Display the output index and the new valu to the console

static void ifKit_OutputChange(object sender, OutputChangeEventArgs e)

{

Console.WriteLine("Output index {0} value {0}", e.Index, e.Value.ToString());

}

//Sensor Change event handler...Display the sensor index and it's new value to the console

static void ifKit_SensorChange(object sender, SensorChangeEventArgs e)

{

Console.WriteLine("Sensor index {0} value {1}", e.Index, e.Value);

}

}

}
```

Learning more ...

- check out the forums

- check out the Phidgets projects

Technical Section

Analog Input
Power (+5V)
Ground (0V)

Using the Analog Inputs

The Analog Input can measure a voltage between 0V and 5V. The analog measurement is represented in the software as a value between 0 and 1000, so a sensor value of 1 unit represents a voltage of approximately 5 millivolts.

Each analog input uses a 3-pin, 0.100 inch pitch locking connector. Pictured here is a plug with the connections labeled. If this is wired backwards, damage to your sensor may result. The Interface Kit provides + 5VDC, ground, and an analog input with a range of 0 to 5V.
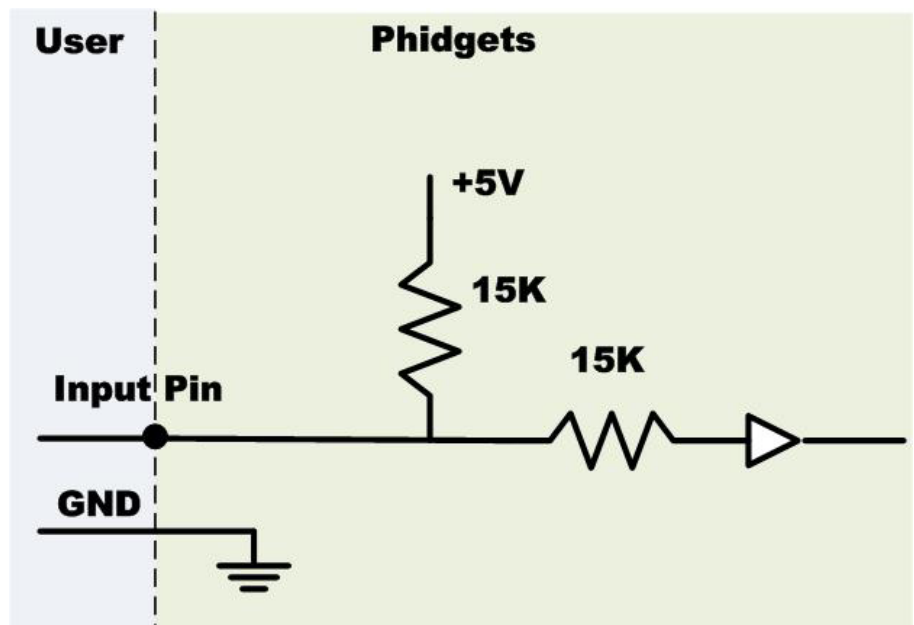
Ratiometric Sensors

If you are using a sensor whose output changes linearly with variations in the sensor's supply voltage level, it is said to be ratiometric. Most of the sensors sold by Phidgets are ratiometric (this is specified in the manual for each sensor).

If the analog sensors you are using are ratiometric, enable the *ratiometric* property in software. This causes the reference to the internal Analog to Digital Converter to be set to the power supply voltage level. If ratiometric is not enabled, the ADC reference is set to a 5.0V 0.5% stable voltage reference.
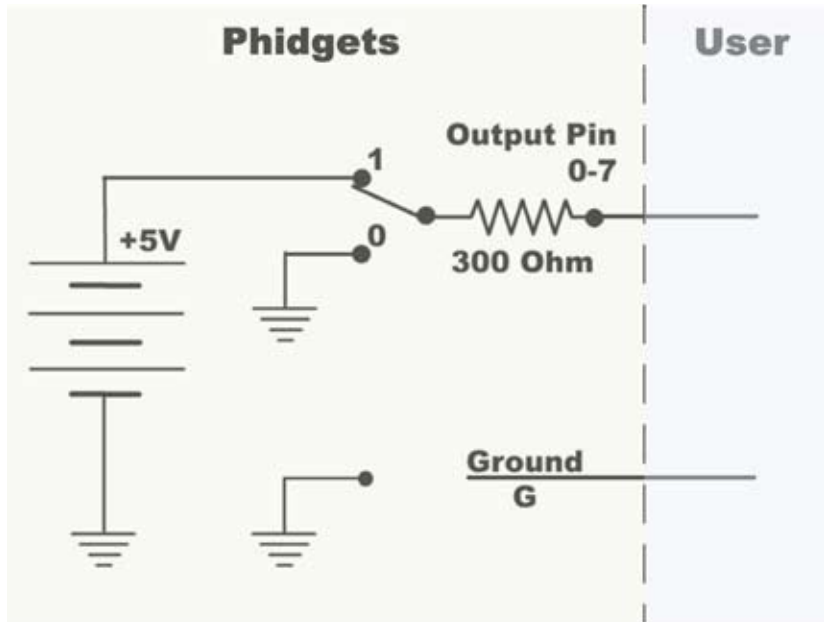
Using the Digital Inputs

To wire a switch to a digital input, connect the switch between an input, labeled 0 to 7, and a provided ground, labeled G. The default state of the Digital Input in software is False (the switch state is open and the input pin is pulled to 5V by an internal resistor). When the switch is closed, the input pin is pulled to ground and the Digital Input is set to True.

Using the Digital Outputs

Connecting an LED or other circuit to a digital output is simple. In the case of an LED, wire the anode to a digital output labeled 0 to 7 on the Interface Kit, and the cathode to a supplied ground, labeled G.
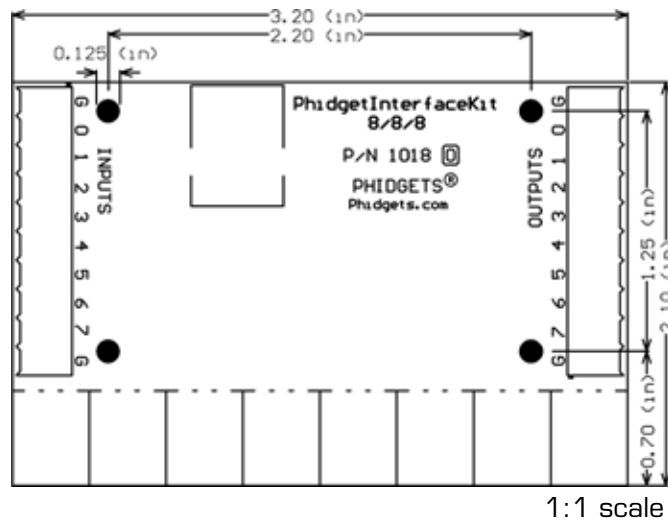
The 300 ohm resistance is internal to the PhidgetInterfaceKit 8/8/8, and limits the current that can flow through the output. This is intended to protect the device from being damaged if there

is a short to ground or if an LED is used. The output is intended to drive TTL or CMOS inputs; it is not designed to provide power to an external circuit.

The digital outputs can be used to switch larger electrical currents and voltages using devices such as power transistors, or logic level MOSFETs. You can also use the 3051 or 3052 to control a larger load.

Mechanical Drawing

1:1 scale

Device Specifications

Analog Input Impedance	900K ohms
Digital Output Series Resistance	300 ohms
Digital Input Pull-Up Resistance	15K ohms
Analog Input Update Rate	~65 samples / second
Digital Output Update Rate	~125 samples / second
Digital Input Update Rate	~125 samples / second
Digital Input Recommended Wire Size	16 - 26 AWG
Digital Output Recommended Wire Size	16 - 26 AWG
Digital Input Wire Stripping	5-6mm strip
USB-Power Current Specification	Max 500mA
Quiescent Current Consumption	20mA
Available External Current (source)	480mA

Cable and Connector Components for Analog Inputs

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.

Product History

Date	Product Revision	Comment
July 2007	1018_0	Product Release