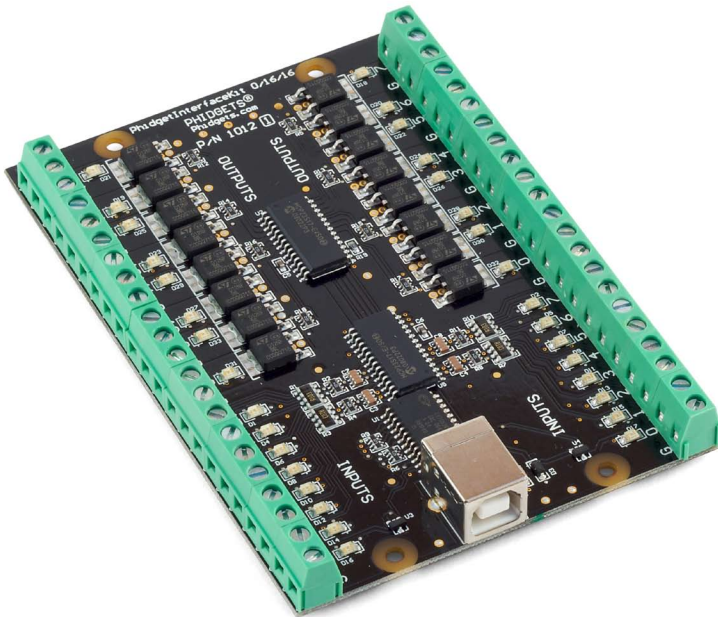


Product Manual

1012 - PhidgetInterfaceKit 0/16/16



Phidgets 1012 - Product Manual
For Board Revision 1
© Phidgets Inc. 2009

Contents

5 Product Features

- 5 Programming Environment
- 5 Connection

6 Getting Started

- 6 Checking the Contents
- 6 Connecting all the pieces
- 6 Testing Using Windows 2000/XP/Vista
 - 6 Downloading the Phidgets drivers
 - 6 Running Phidgets Sample Program
- 7 Testing Using Mac OS X
- 8 If you are using Linux
- 8 If you are using Windows Mobile/CE 5.0 or 6.0

9 Programming a Phidget

- 9 Architecture
- 9 Libraries
- 9 Programming Hints
- 9 Networking Phidgets
- 10 Documentation
 - 10 Programming Manual
 - 10 Getting Started Guides
 - 10 API Guides
- 10 Code Samples
- 10 API for the PhidgetInterfaceKit 0/16/16
 - 10 Functions
 - 10 Events

11 Technical Section

- 11 Digital Inputs
 - 11 Using the Digital Inputs
 - 11 Functional Block Diagram
 - 11 Digital Input Hardware Filter
- 12 Digital Outputs

- 12 Using the Digital Outputs
- 12 Functional Block Diagram
- 12 Ground Protection
- 13 Mechanical Drawing
- 13 Device Specifications

14 Product History

14 Support

Product Features

- 16 Digital Inputs that are activated by an external voltage source, triggering on a wide voltage range - 4 to 30VDC
- They can be used to convey the state of on/off devices, such as push buttons, limit switches, relays
- 16 Digital Outputs switching up to 30VDC at up to 2 Amps
- They can be used to directly control devices requiring substantial power such as incandescent lights, high power LEDs, relays, solenoids, motors
- The Digital Output acts as a switch to ground, and is protected from transient voltages typical when switching inductive devices - relays, solenoids, motors
- LED Indicators on all I/O channels

Programming Environment

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

Programming Languages (APIs): VB6, VB.NET, C#.NET, C++, Flash 9, Flex, Java, LabVIEW, Python, Max/MSP, and Cocoa.

Examples: Many example applications for all the operating systems and development environments above are available for download at www.phidgets.com >> Programming.

Connection

The board connects directly to a computer's USB port.

Getting Started

Checking the Contents

You should have received:

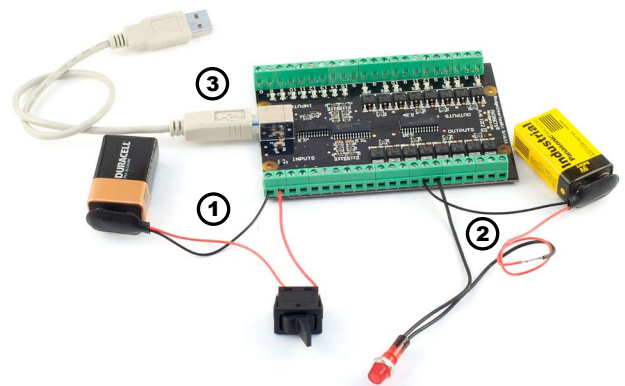
- A Phidget Interface Kit 0/16/16
- A USB cable

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

- Two 9V batteries
- Two 9V battery connectors
- A switch, a piece of wire
- An incandescent bulb

Connecting all the pieces

1. Connect the black wire (-) from the battery to the ground terminal (G). Connect the red wire (+) to one of the switch terminals. Connect the other switch terminal to the digital input terminal block number 0 using a piece of wire.
2. Connect the black wire (-) from the battery to the ground terminal (G). Connect the red wire (+) to one of the bulb wires. Connect the other bulb wire the digital output terminal block number 3.
3. Connect the board to the 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 >> Drivers


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

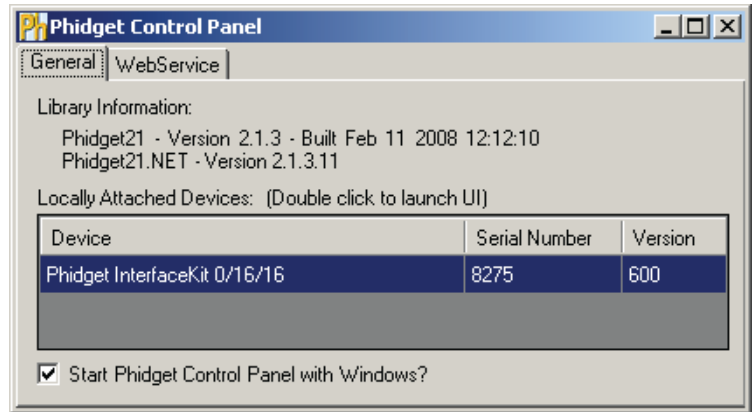
You should see the  icon on the right hand corner of the Task Bar.

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.

The source code for the InterfaceKit-Full sample program can be found under C# by clicking on www.Phidgets.com >> Programming.

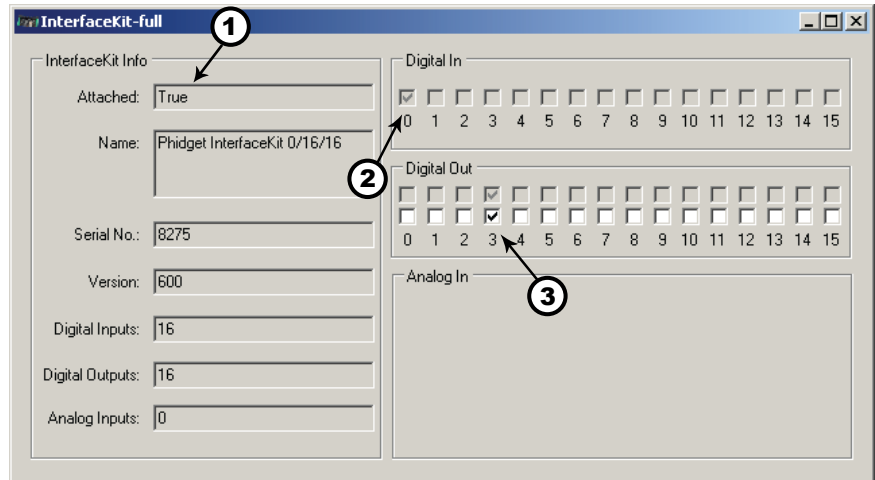
Double Click on the  icon to activate the Phidget Control Panel and make sure that the **Phidget InterfaceKit 0/16/16** is properly attached to your PC.



1. Double Click on **Phidget InterfaceKit 0/16/16** in the Phidget Control Panel to bring up InterfaceKit-full and check that the box labelled Attached contains the word True.

2. To test the digital input, toggle the switch on and off. When on, a tick mark will appear in the Digital In box and The on-board LED will also turn on; the tick mark will disappear when the switch is off and the on-board LED light will go off.

3. To test the digital output, put a tick mark in the digital out box and both the on-board LED and the incandescent bulb will turn on. If you click on the box again the tick mark will go away and both lights will turn off. The bottom row shows the status of the request, while the top row displays the status of the digital output as reported by the device.



Testing Using Mac OS X

- Click on System Preferences >> Phidgets (under Other) to activate the Preference Pane
- Make sure that the Phidget InterfaceKit 0/16/16 is properly attached.
- Double Click on Phidget InterfaceKit 0/16/16 in the Phidget Preference Pane to bring up the InterfaceKit-full example. This example will function in a similar way as the Windows version.

If you are using Linux

There are no sample programs written for Linux.

Go to www.phidgets.com >> Drivers

Download Linux Source

- Have a look at the readme file
- Build Phidget21

The most popular programming languages in Linux are C/C++ and Java.

Notes:

Many Linux systems are now built with unsupported third party drivers. It may be necessary to uninstall these drivers for our libraries to work properly.

Phidget21 for Linux is a user-space library. Applications typically have to be run as root, or udev/hotplug must be configured to give permissions when the Phidget is plugged in.

If you are using Windows Mobile/CE 5.0 or 6.0

Go to www.phidgets.com >> Drivers

Download x86 or ARMV4I, depending on the platform you are using. Mini-itx and ICOP systems will be x86, and most mobile devices, including XScale based systems will run the ARMV4I.

The CE libraries are distributed in .CAB format. Windows Mobile/CE is able to directly install .CAB files.

The most popular languages are C/C++, .NET Compact Framework (VB.NET and C#). A desktop version of Visual Studio can usually be configured to target your Windows Mobile Platform, whether you are compiling to machine code or the .NET Compact Framework.

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.

Architecture

We have designed our libraries to give you the maximum amount of freedom. We do not impose our own programming model on you.

To achieve this goal we have implemented the libraries as a series of layers with the C API at the core surrounded by other language wrappers.

Libraries

The lowest level library is the C API. The C API can be programmed against on Windows, CE, OS X and Linux. With the C API, C/C++, you can write cross-platform code. For systems with minimal resources (small computers), the C API may be the only choice.

The Java API is built into the C API Library. Java, by default is cross-platform - but your particular platform may not support it (CE).

The .NET API also relies on the C API. Our default .NET API is for .NET 2.0 Framework, but we also have .NET libraries for .NET 1.1 and .NET Compact Framework (CE).

The COM API relies on the C API. The COM API is programmed against when coding in VB6, VBScript, Excel (VBA), Delphi and Labview.

The ActionScript 3.0 Library relies on a communication link with a PhidgetWebService (see below). ActionScript 3.0 is used in Flex and Flash 9.

Programming Hints

- Every Phidget has a unique serial number - this allows you to sort out which device is which at runtime. Unlike USB devices which model themselves as a COM port, you don't have to worry about where in the USB bus you plug your Phidget in. If you have more than one Phidget, even of the same type, their serial numbers enable you to sort them out at runtime.
- Each Phidget you have plugged in is controlled from your application using an object/handle specific to that phidget. This link between the Phidget and the software object is created when you call the .OPEN group of commands. This association will stay, even if the Phidget is disconnected/reattached, until .CLOSE is called.
- The Phidget APIs are designed to be used in an event-driven architecture. While it is possible to poll them, we don't recommend it. Please familiarize yourself with event programming.

Networking Phidgets

The PhidgetWebService is an application written by Phidgets Inc. which acts as a network proxy on a computer. The PhidgetWebService will allow other computers on the network to communicate with the Phidgets connected to that computer. ALL of our APIs have the capability to communicate with Phidgets on another computer that has the PhidgetWebService running.

The PhidgetWebService also makes it possible to communicate with other applications that you wrote and that are connected to the PhidgetWebService, through the PhidgetDictionary object.

Documentation

Programming Manual

The Phidget Programming Manual documents the Phidgets software programming model in a language and device unspecific way, providing a general overview of the Phidgets API as a whole. You can find the manual at www.phidgets.com >> Programming.

Getting Started Guides

We have written Getting Started Guides for most of the languages that we support. If the manual exists for the language you want to use, this is the first manual you want to read. The Guides can be found at www.phidgets.com >> Programming and are listed under the appropriate language.

API Guides

We maintain API references for COM (Windows), C (Windows/Mac OSX/Linux), Action Script, .Net and Java. These references document the API calls that are common to all Phidgets. These API References can be found under www.phidgets.com >> Programming and are listed under the appropriate language. To look at the API calls for a specific Phidget, check its Product Manual.

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 >> 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.

API for the PhidgetInterfaceKit 0/16/16

We document API Calls specific to this product in this section. Functions common to all Phidgets and functions not applicable to this device are not covered here. This section is deliberately generic. For calling conventions under a specific language, refer to the associated API manual. For exact values, refer to the device specifications.

Functions

int InputCount() [get] : Constant = 16

Returns the number of digital inputs supported by this PhidgetInterfaceKit.

bool InputState(int InputIndex) [get]

Returns the state of a particular digital input. Digital inputs read True where they are activated and false when they are in their default state.

int OutputCount() [get] : Constant = 16

Returns the number of digital outputs supported by this PhidgetInterfaceKit.

bool OutputState (int OutputIndex) [get,set]

Sets/returns the state of a digital output. Setting this to true will activate the output, False is the default state. Reading the OutputState immediately after setting it will not return the value set - it will return the last state reported by the Phidget.

Events

OnInputChange(int InputIndex, bool State) [event]

An event that is issued when the state of a digital input changes.

OnOutputChange(int OutputIndex, bool State), [event]

An event that is issued when the state of a digital output changes.

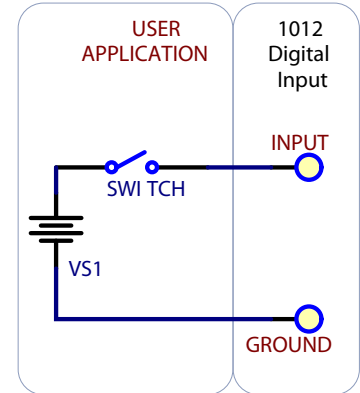
Technical Section

Digital Inputs

Using the Digital Inputs

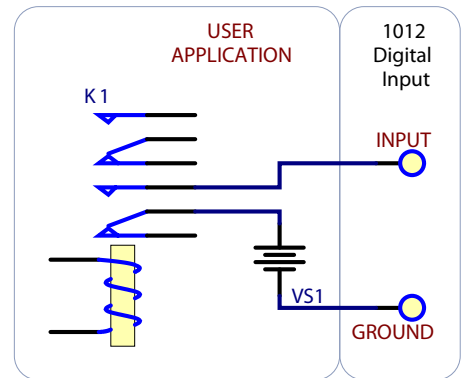
Wiring a switch to a Digital Input

VSI should be 4-30V.



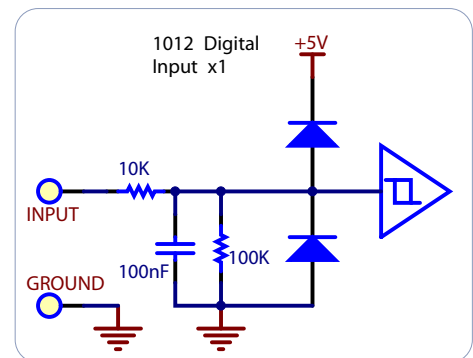
Monitoring the Position of a Relay

The relay contact causes the Digital Input to report TRUE. VSI should be 4-30V.



Functional Block Diagram

The Digital Inputs are capable of sensing up to 30VDC. A voltage of 4VDC to 30VDC will be read as True or logical 1; below 1VDC will be read as a False or logical 0. The input is high impedance, which means current flowing into the Phidget device will be limited. Ground terminals are provided in multiple locations along the input terminal strip; it is recommended that the ground terminal located nearest the input terminal be used.



Digital Input Hardware Filter

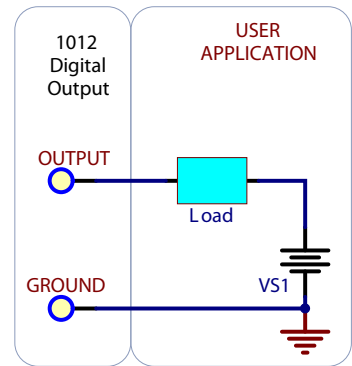
There is built-in filtering on the digital input, to eliminate false triggering from electrical noise. The digital input is first RC filtered by a 10K/100nF node, which will reject noise of higher frequency than 1Khz. This filter generally eliminates the need to shield the digital input from inductive and capacitive coupling likely to occur in wiring harnesses.

Digital Outputs

Using the Digital Outputs

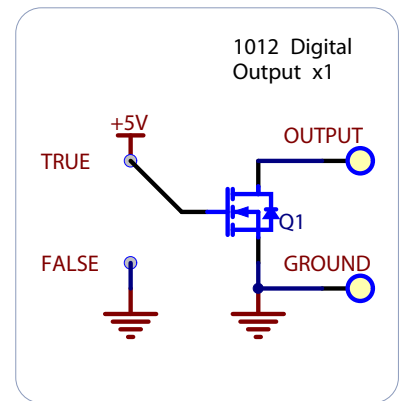
Using Digital Output as a Lowside switch for a load

Maximum Continuous current sink is 2 Amps. Load can be Relays, Solenoids, current limited LEDs, Bulbs, Motors, etc.



Functional Block Diagram

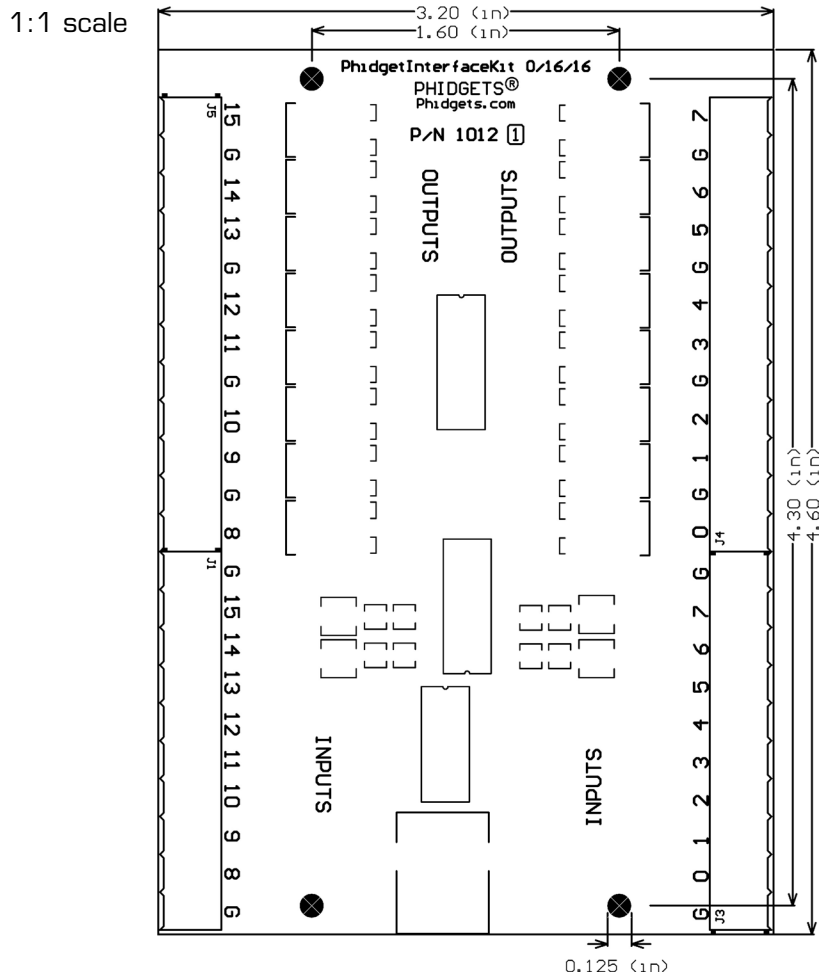
The Digital Outputs require an external voltage source to supply power, and act as a switch to ground. The outputs can sink up to 2A at 30V, and are designed to operate with DC voltage only. This type of switching output is often referred to as a low-side switch. It is common to use the Digital Outputs with electrical devices such as motors, lamps, relays, and solenoids. When using ground terminals, it is recommended that the ground terminal located nearest the output terminal be used. If highly inductive loads are used with the Digital Outputs, the use of a clamping diode is recommended to reduce noise.



Ground Protection

Ground terminals on the InterfaceKit share a common ground with USB ground. Because they are not internally isolated, these terminals will expose the USB ground potential of the PC to which they are connected. Be sure you are completely familiar with any circuit you intend to connect to the InterfaceKit before it is connected. If a reverse voltage or dangerously high potential is applied to the input or output terminals, damage to the Phidget or the PC may result. Limit input and output voltages to 30VDC, and always observe correct polarity.

Mechanical Drawing



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
Digital Input Impedance (-0.5V to +5.5V)	110kΩ
Digital Input Impedance (>5.5V)	10kΩ
Digital Output Impedance (on)	0.2Ω
Digital Input Update Rate	125 Updates/second
Digital Input Minimum Event Detection Time	3mS
Digital Output Update Rate	125 Updates/second
Digital Output Current Sinking (30V)	2A max
USB Power Current Specification	500mA max
Device Quiescent Current Consumption	18mA
Device Active Current Consumption	120mA max

Product History

Date	Board Revision	Device Version	Comment
January 2003		600	Product Release
January 2004		601	Added State Echoing
September 2008	1		Added Digital Input Filtering

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