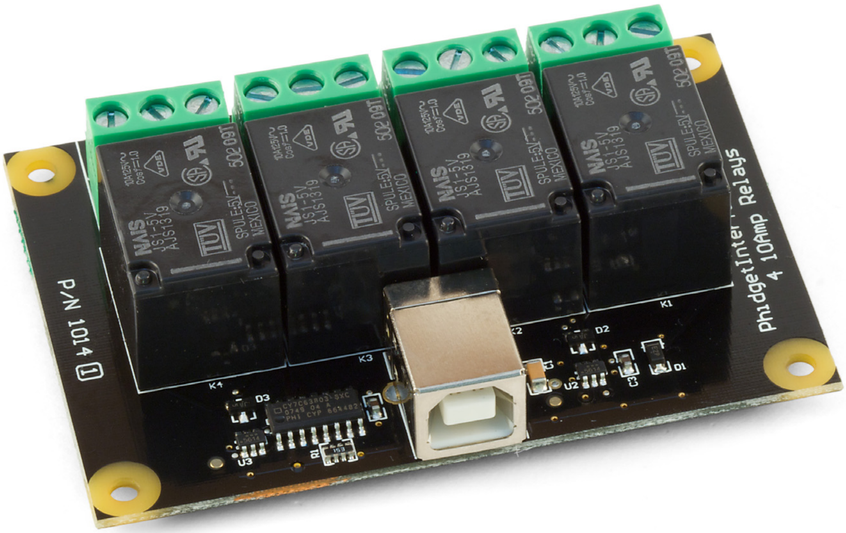
Product Manual

1014 - PhidgetInterfaceKit 0/0/4



Phidgets 1014 - Product Manual

For Board Revision 1

Contents

13 Product History

13 Support

Product Features

- Contains 4 Relay Outputs for switching AC or DC power
- Ratings: 250VAC, 10 Amps or 100VDC, 5 Amps
- Relays are Single Pole Double Throw (SPDT)
- Provides a convenient way to interface your PC with various higher-voltage devices such as incandescent bulbs, high-power relays, and motors

Programming Environment

Operating Systems: Windows 2000/XP/Vista/7, 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:

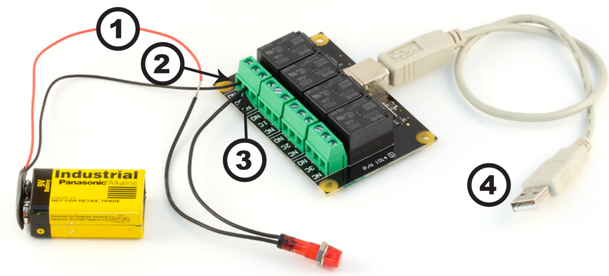
- A PhidgetInterfaceKit 0/0/4
- A USB Cable

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

- A 9V battery
- A battery connector
- A 9V incandescent bulb with wires

Connecting all the pieces

1. Connect the red/positive (+) wire from the battery connector to one of the bulb wires.
2. Connect the black/negative (-) wire from the battery connector to the NO (Normally Open) connector on the InterfaceKit.
3. Connect the other bulb wire to the OC (Common) connector on the InterfaceKit.
4. Connect the InterfaceKit 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 >> Drivers

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

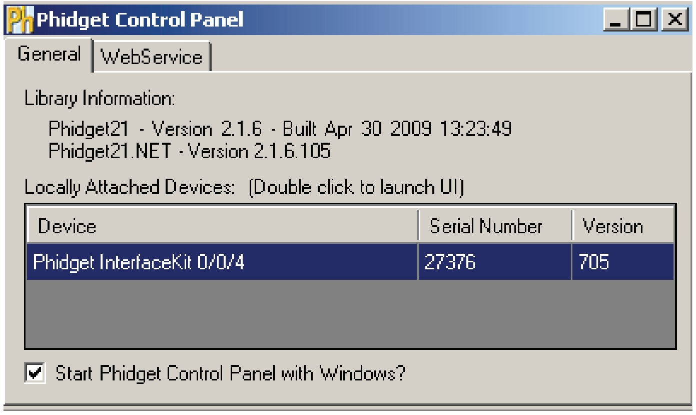
You should see the Ph icon on the right hand corner of the Task Bar.

Running Phidgets Sample Program

Double clicking on the Ph 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 Ph icon to activate the Phidget Control Panel and make sure that **Phidget InterfaceKit 0/0/4** is properly attached to your PC.



Phidget Control Panel

General | WebService

Library Information:

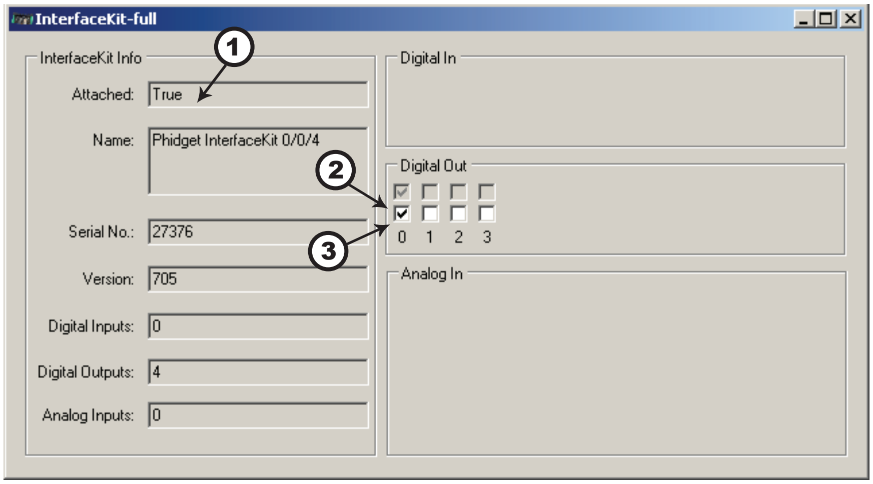
Phidget21 - Version 2.1.6 - Built Apr 30 2009 13:23:49
Phidget21.NET - Version 2.1.6.105

Locally Attached Devices: (Double click to launch UI)

Device	Serial Number	Version
Phidget InterfaceKit 0/0/4	27376	705

☑ Start Phidget Control Panel with Windows?

1. Double Click on **Phidget InterfaceKit 0/0/4** in the Phidget Control Panel to bring up InterfaceKit-full and check that the box labelled Attached contains the word True.
2. Click on the first Digital Out box. A tick mark appears in the box and the bulb lights up. Click on the box again. The tick mark goes away and the light goes out. If you unplug the USB cable while the light is on, it will go 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.



InterfaceKit-full

InterfaceKit Info

Attached: True

Name: Phidget InterfaceKit 0/0/4

Serial No.: 27376

Version: 705

Digital Inputs: 0

Digital Outputs: 4

Analog Inputs: 0

Digital In

Digital Out

☑ □ □ □
☑ □ □ □
0 1 2 3

Analog In

3. Move the positive (+) wire from NO to NC (normally closed). The light is now on when the Digital box has no tick mark. Clicking on the box turns the light off. If you unplug the USB cable when the light is on, it will stay on.

Testing Using Mac OS X

- Click on System Preferences >> Phidgets (under Other) to activate the Preference Pane
- Make sure that the **Phidget InterfaceKit 0/0/4** is properly attached.
- Double Click on **Phidget InterfaceKit 0/0/4** 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, ARMV4I or MIPSII, 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 0/0/4

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

nt OutputCount() [get] : Constant = 4

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

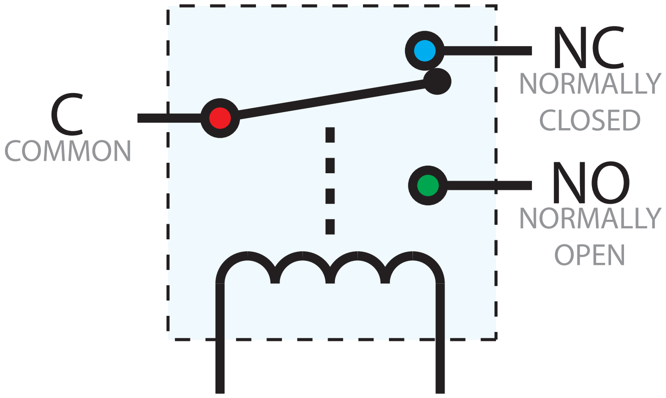
OnOutputChange(int OutputIndex, bool State), [event]

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

Technical Section

Relays

A relay is an electrically-controlled switch. Although many types of electrical switches exist, a relay's mechanical nature gives it the advantage of reliability and current-switching capacity. The main disadvantage to using mechanical relays is their limited life-span, as opposed to solid state relays who do not suffer from this drawback.



Using a Digital Output Relay

Relays have a connection scheme determined by the arrangement of contacts within the relay. Because relays are a type of switch, they are defined in the same way other electromechanical switches are defined.

In switch schemes, the number of poles represents the number of common terminals a switch has, and the number of throws represents the number of switchable terminals that exist for each pole. The relays used in the InterfaceKit 0/0/4 are SPDT relays: single pole, double throw. The internal construction of this type of relay is depicted in the diagram above. Many other types of relays exist: SPST, DPDT, and DPST, to name a few.

In an SPDT relay, one of the throw terminals is labelled Normally Closed (NC), and the other is labelled Normally Open (NO). As the name indicates, the normally closed terminal is the terminal connected to common when the relay coil is not powered. When the relay coil is energized by the relay control circuit, the electromagnetic field of the coil forces the switch element inside the relay to break its contact with the normally closed terminal and make contact with the normally open terminal. The switch element would then connect the normally open terminal and the common terminal.

Wetting Current

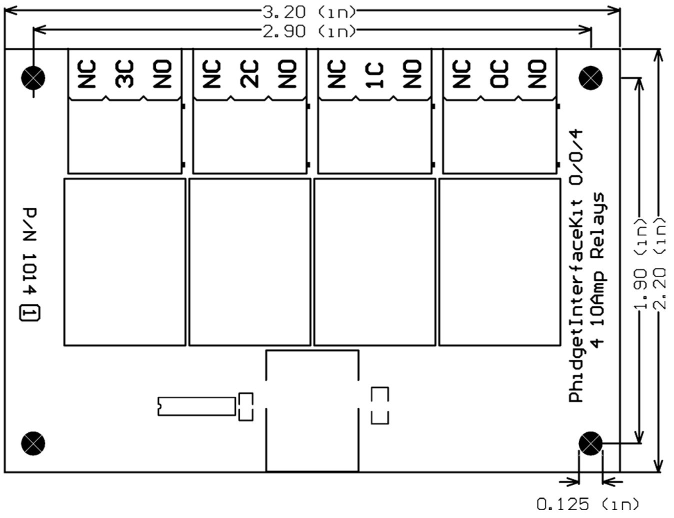
When a relay is in one switch position for a period of time, oxidation of the open contact(s) can occur. Depending upon the internal coating material of the contacts, oxide films of varying density will be displaced upon the surface of open contacts; this film acts as an insulator to current flow. When the relay is switched, a certain amount of current flowing through the contacts, known as the wetting current, is required to remove the film of oxides and ensure proper conduction. Because of this requirement, these relays are not reliable for signal switching. See the device specification on page 10 for detailed requirements.

Load Noise

If highly inductive loads are used with the InterfaceKit, it is recommended that a noise limiting component be used to prevent damage to the device. An MOV, TVS diode, or kickback diode (for DC applications) shunted across the load will assist in dissipating voltage transients.

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**
Type of Relay	Single Pole Double Throw (SPDT)
Contact Resistance (max)	120 mohms
Minimum Switching Current (Wetting Current)	100mA @ 5VDC
Maximum DC Switching Voltage	100 VDC
Maximum DC Switching Current	5 A
Maximum AC Switching Voltage	250 VAC
Maximum AC Switching Current	10 A
Operate Time	10 ms
Switching Speed (Contacts Per Minute)	20 cpm
Recommended Terminal Wire Size	12 - 24 AWG
Terminal Wire Strip Length	5 - 6mm (0.196" - 0.236")
Device Quiescent Current Consumption	14mA
Device Active Current Consumption	320mA max
Maximum Ambient Temperature	70°C

Product History

Date	Board Revision	Device Version	Comment
August 2002		700	Product Release
January 2004		704	Added State Echoing
May 2008	1		Terminal Blocks now accept 12-24 AWG wire, PCB increased to accommodate larger connectors.

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