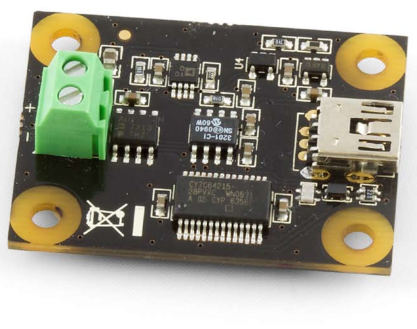
Product Manual

1051 - PhidgetTemperatureSensor 1-Input



Phidgets 1051 - Product Manual

For Board Revision 2

Contents

Product Features

- Supports one thermocouple providing a wide range of temperatures that can be measured.
- Measures temperatures from -200°C to +1250°C when using a K-Type thermocouple.
- Outputs temperature in degrees Celsius.
- Gives the temperature of the Phidget (ambient) in degrees Celsius.
- Automatically compensates for errors introduced by the temperature of the Phidget.
- Uses terminal blocks to interface to inexpensive thermocouples.

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:

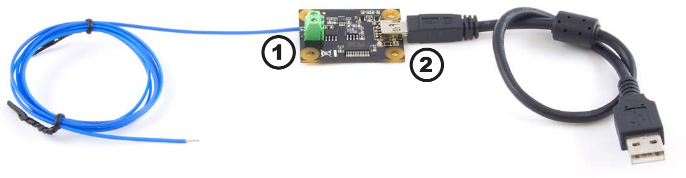
- A PhidgetTemperatureSensor 1-Input
- A Mini-USB Cable

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

- A Thermocouple with stripped end

Connecting all the pieces

1. Connect the thermocouple to the input terminal block on the 1051. Make sure that the thermocouple ground wire goes into the ground terminal (-). If you are using a 2007 - Phidget K-type Thermocouple, the white wire is the ground wire.
2. Connect the 1051 to your PC using the Mini-USB cable.



Testing Using Windows 2000/XP/Vista/7

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)

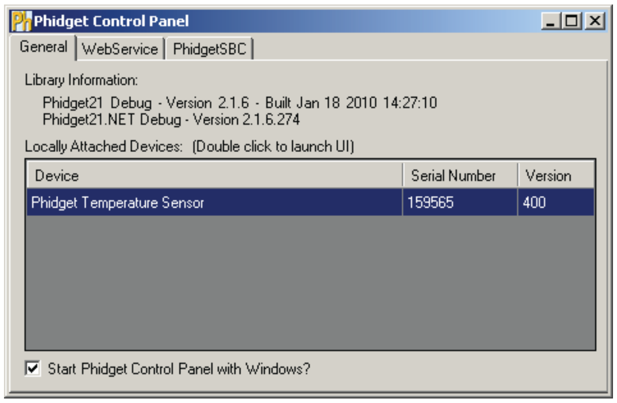
You should see the Ph icon on the right hand corner of the Task Bar.

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 TemperatureSensor-full sample program can be found under C# by clicking on Phidget.com > Programming.

Double Click on the Ph icon to activate the Phidget Control Panel and make sure that the **Phidget Temperature Sensor** is properly attached to your PC.


Phidget Control Panel

General | WebService | PhidgetSBC

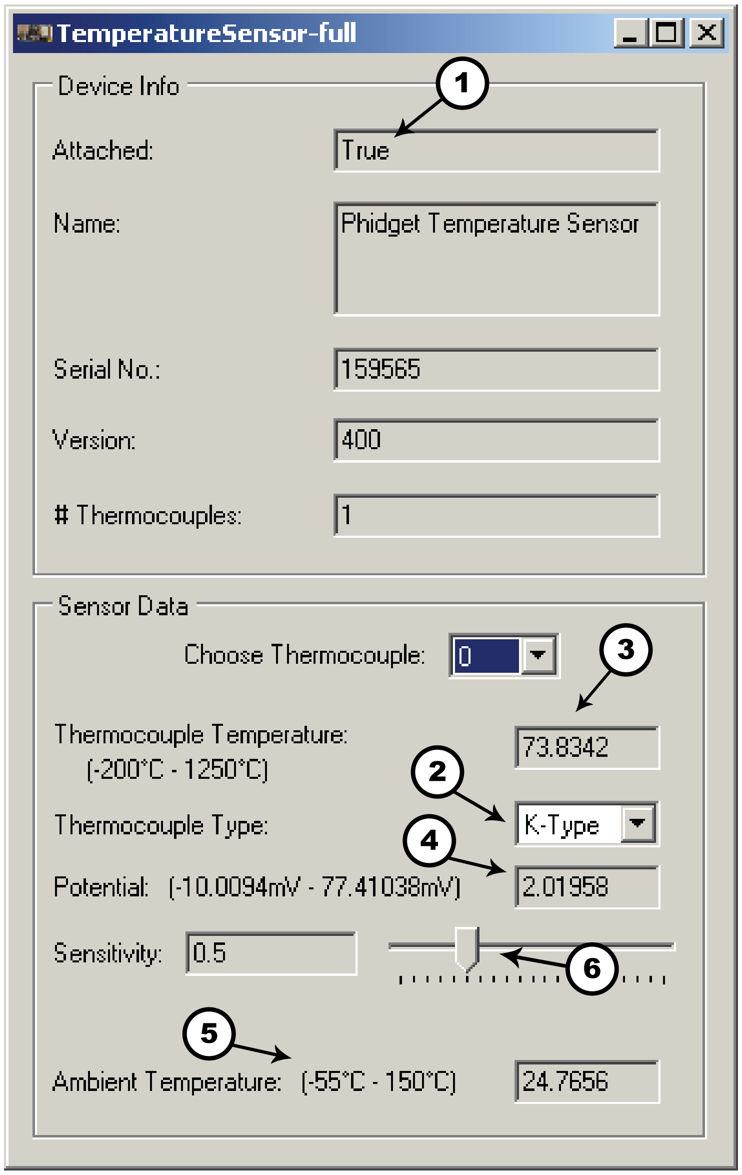
Library Information:

Phidget21 Debug - Version 2.1.6 - Built Jan 18 2010 14:27:10
Phidget21.NET Debug - Version 2.1.6.274

Locally Attached Devices: (Double click to launch UI)

Device	Serial Number	Version
Phidget Temperature Sensor	159565	400

☑ Start Phidget Control Panel with Windows?


TemperatureSensor-full

Device Info

Attached: True

Name: Phidget Temperature Sensor

Serial No.: 159565

Version: 400

Thermocouples: 1

Sensor Data

Choose Thermocouple: 0

Thermocouple Temperature: (-200°C - 1250°C) 73.8342

Thermocouple Type: K-Type

Potential: (-10.0094mV - 77.41038mV) 2.01958

Sensitivity: 0.5

Ambient Temperature: (-55°C - 150°C) 24.7656

1. Double Click on **Phidget Temperature Sensor** in the Phidget Control Panel to bring up TemperatureSensor-full and check that the box labelled Attached contains the word True.
2. Select K-Type thermocouple
3. Touch a source of heat (light bulb, candle or lighter flame) with the thermocouple probe and watch the Thermocouple temperature increase.
4. The potential value will also increase as the thermocouple temperature increases.
5. The ambient temperature gives you the board temperature.
6. You can adjust the data sensitivity by moving the slider pointer.

Testing Using Mac OS X

- Click on System Preferences >> Phidgets (under Other) to activate the Preference Pane
- Make sure that the Phidget Temperature Sensor 1-Input is properly attached.
- Double Click on Phidget Temperature Sensor 1-Input in the Phidget Preference Pane to bring up the TemperatureSensor-full Sample program. This program 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

There is no Control Panel written for Linux, but there are C/C++ and Java code samples available for all Phidgets which will compile and run on Linux without modification.

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.
- For full performance, the Phidget APIs are designed to be used in an event driven architecture. Applications that require receiving all the data streaming from the device will have to use event handlers, instead of polling.

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 PhidgetTemperatureSensor 1-Input

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 TemperatureInputCount () [get] : Constant

Returns the number of thermocouples that can be read by this PhidgetTemperatureSensor.

double Temperature (int ProbeIndex) [get] : Celsius

Returns the temperature of a thermocouple. This value is returned in degrees Celsius but can easily be converted into other units. This value will always be between TemperatureMin and TemperatureMax. Please refer to the device specifications for noise and accuracy details.

double TemperatureMax (int ProbeIndex) [get] : Constant, Celsius

Returns the maximum temperature that can be returned by a thermocouple input. This value depends on the thermocouple type.

double TemperatureMin (int ProbeIndex) [get] : Constant, Celsius

Returns the minimum temperature that can be returned by a thermocouple input. This value depends on the thermocouple type.

double TemperatureChangeTrigger(int ProbeIndex) [get,set] : Celsius

Sets the change trigger for an input. This is the amount by which the sensed temperature must change between TemperatureChangeEvents. By default this is set to 0.5.

Setting TemperatureChangeTrigger to 0 will cause all temperature updates to fire events. This is helpful for applications that are implementing their own filtering.

double Potential (int ProbeIndex) [get] : Millivolts

Returns the potential of a thermocouple input. This value is returned in millivolts, and will always be between getPotentialMin and getPotentialMax. This is the value that is internally used to calculate temperature in the library.

double PotentialMax (int ProbeIndex) [get] : Constant, Millivolts

Returns the maximum voltage that can be measured by the 1051.

double PotentialMin (int ProbeIndex) [get] : Constant, Millivolts

Returns the minimum voltage that can be measured by the 1051.

double AmbientTemperature () [get] : Celsius

Returns the temperature of the 1051 board, measured near the thermocouple connector. This temperature is used as a reference for the thermocouple voltage. This value will always be between getAmbientTemperatureMin and getAmbientTemperatureMax.

double AmbientTemperatureMax () [get] : Constant, Celsius

Returns the maximum temperature that can be returned by the ambient sensor.

double AmbientTemperatureMin () [get] : Constant, Celsius

Returns the minimum temperature that can be returned by the ambient sensor.

int ThermocoupleType(int ProbeIndex) [get,set]

Returns/Sets the thermocouple type for an input. The possible values are J, K, E, and T, corresponding to K, E, J and T-Type Thermocouples. Support for other thermocouple types, and voltage sources other then thermocouples in the valid range (between getPotentialMin and getPotentialMax) can be achieved using getPotential.

Note that this function will vary widely between APIs. Please refer to the Manual for the API you are programming against for exact calling conventions.

Events

OnTemperatureChange(int ProbeIndex, double Temperature) [event]

Event that fires whenever the temperature changes by more than the TemperatureChangeTrigger.

OnError(int code, String description) [event]

PhidgetTemperatureSensor returns EEPHIDGET_OUTOFRANGE when an input goes out of the range specified by PotentialMin-PotentialMax or TemperatureMin-TemperatureMax. The event is reported once per input per out-of-range type. until the value has gone back into range. The description specifies the input, the type of out-of-range error and the out of range value that was read.

This should only happen for inputs which don't have a thermocouple attached.

Technical Section

Introduction

Thermocouples are widely used to measure extreme temperatures impossible to measure with other types of sensors.

Getting the Temperature value

K, J, T, E-Type Thermocouples

J, K, E and T thermocouples are directly supported in the library; the **getTemperature** call returns the measured temperature in °C.

Typical Temperature Range:

Type	Temperature Range °C	Notes
K	-200 to +1250	The most common sensor, wide range, good in oxidizing atmosphere.
J	0 to +750	More limited range, main application is with old equipment that cannot accept modern thermocouples
T	-200 to +350	Most accurate base-metal thermocouple, non-magnetic, popular choice for strong magnetic fields and subzero environments
E	-200 to +900	Well suited to low temperature (cryogenic) use.

The insulating sheath that surrounds the thermocouple is the limiting factor in the true temperature range. For example, a K-type thermocouple insulated with teflon is good up to +200°C. The same thermocouple using Fiberglass is good to 480°C.

Using Thermocouples other than K, J, E, or T-Type

It is possible to interface other thermocouple types by converting the measured potential (use the **getPotential** call) into Celsius using the appropriate tables[1]. The calculation is as follows:

-Read the voltage in the thermocouple table corresponding to the measured ambient temperature.

-Add this voltage to the thermocouple potential.

-Read the temperature off the thermocouple table corresponding to the summed potential.

Things to watch for

Electrical Interference

The PhidgetTemperatureSensor measures each thermocouple compared to it's internal ground. If your thermocouple is measuring a conductive device which cannot equalize to the ground of the PhidgetTemperatureSensor, you should isolate the thermocouple wire using thermally conductive epoxy. Thermocouples are conductive, and are very good at shorting electrical systems together. If in doubt, do not allow the thermocouple wire to contact other metal parts.

Keep the PhidgetTemperatureSensor at a stable temperature

Anything that unevenly heats the PhidgetTemperatureSensor board will impact the accuracy of your measurement. We recommend mounting the PhidgetTemperatureSensor a reasonable distance from heat/cold sources, or putting it in an enclosure that will provide a uniform environment.

[1] Conversion tables can be found at this link: http://instrumentation-central.com/pages/thermocouple_reference_table.htm

Wire Size

The maximum temperature varies with the diameter of the wire used in the thermocouple. Although the type of thermocouple dictates the temperature range, the maximum range is also limited by the diameter of the thermocouple wire. That is, a very thin thermocouple may not reach the full temperature range. On the other hand, a thinner thermocouple does reduce the amount of thermal distortion introduced.

Selecting the wire size used in the thermocouple sensor depends upon your application. Generally, when longer life is required for the higher temperatures, the larger size wires should be chosen. When sensitivity is the prime concern, the smaller sizes should be used.

Wire Length

Very long thermocouples are okay (even 100m), but if there is measurement stability issues or USB resets, ferrite beads can be added to the thermocouple wire close to the PhidgetTemperatureSensor. See our tutorial on Addressing EMI issues.

Sheathing material

The insulating sheath along the thermocouple has a temperature limitation which is often much less than the thermocouple type temperature rating. For example, teflon is good to +200C, fiberglass sheath is good to 480C.

How to connect your Thermocouple

Stripped wire

The 1048 is designed to accept stripped wires. We recommend a 5 to 6mm wire strip length; the terminal blocks will accept wires between 16 - 26 AWG. When using thin wires, make sure that you have a solid connection after tightening the terminal screw.

Connectors

Many K-Type thermocouples designed to mate with the Subminiature (flat pin) female connector will work with the PhidgetTemperatureSensor by using to the 3106 - Thermocouple Adapter. It is possible to use this adapter with other Thermocouple types, but the adapter should be kept at the same temperature as the PhidgetTemperatureSensor board.

Listed below are examples of compatible thermocouples from popular sensor manufacturers using this connector.

Sourcing Thermocouples

Manufacturer	Part Number	Description
Omega	TC-PVC-K-24-180	PVC-insulated 4.5m epoxy-coated tip k-type
Omega	88202K	Moving-surface swivel-head handle k-type
Omega	88402K	Flat-leaf magnetized k-type
Nanmac	B8-10	Handheld trident-style ribbon k-type
Cole-Parmer	WU-93631-11	ICONEL-sheathed high-temp 12” k-type

Device Specifications

Characteristic	Value
Electrical	
Device Current Consumption	25mA
USB Device Voltage	4.75 - 5.25 VDC
Operating Temperature	0 - 70°C
Accuracy	
Temperature Update Rate	25 samples/second
Ambient Temperature Resolution	±0.5°C
Thermocouple Resolution (°C)	0.1°C (K-type @ 25°C)
Thermocouple Resolution (μV)	4 μV
Thermocouple Accuracy	2°C (K-type)
Typical Noise	0.02°C standard deviation
Typical Thermocouple Temperature Range	
E-Type	-200°C to +900°C
J-Type	0°C to +750°C
K-Type	-200°C to +1250°C
T-Type	-200°C to +350°C

Product History

Date	Board Revision	Device Version	Comment
October 2003		100	Product Release
January 2005		200	Noise performance improved to 2 Celsius
October 2008	1	300	More accurate ambient temperature sensor. Added support for E, J, and T-type thermocouples in the API library, on-board noise filtering.
April 2010	2	400	Mini USB connector, new thermocouple connector

Support

Call the support desk at 1.403.282.7335 9:00 AM to 5:00 PM Mountain Time (US & Canada) - GMT-07:00

or

E-mail us at: support@phidgets.com