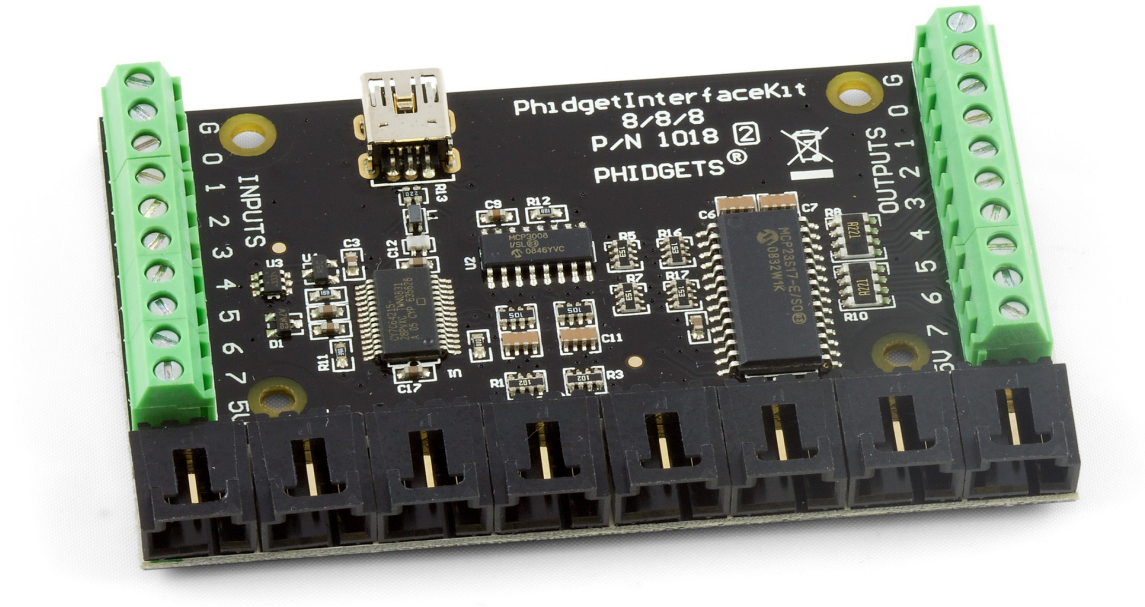
PHIDGETS 1018_2 Interface Kit
Datasheet & Userguide

1.Product Description



1.1 Analog Inputs

The Analog Inputs 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.

Sampling rates can be set at 1ms, 2ms, 4ms, 8ms and multiple of 8ms up to 1000ms.

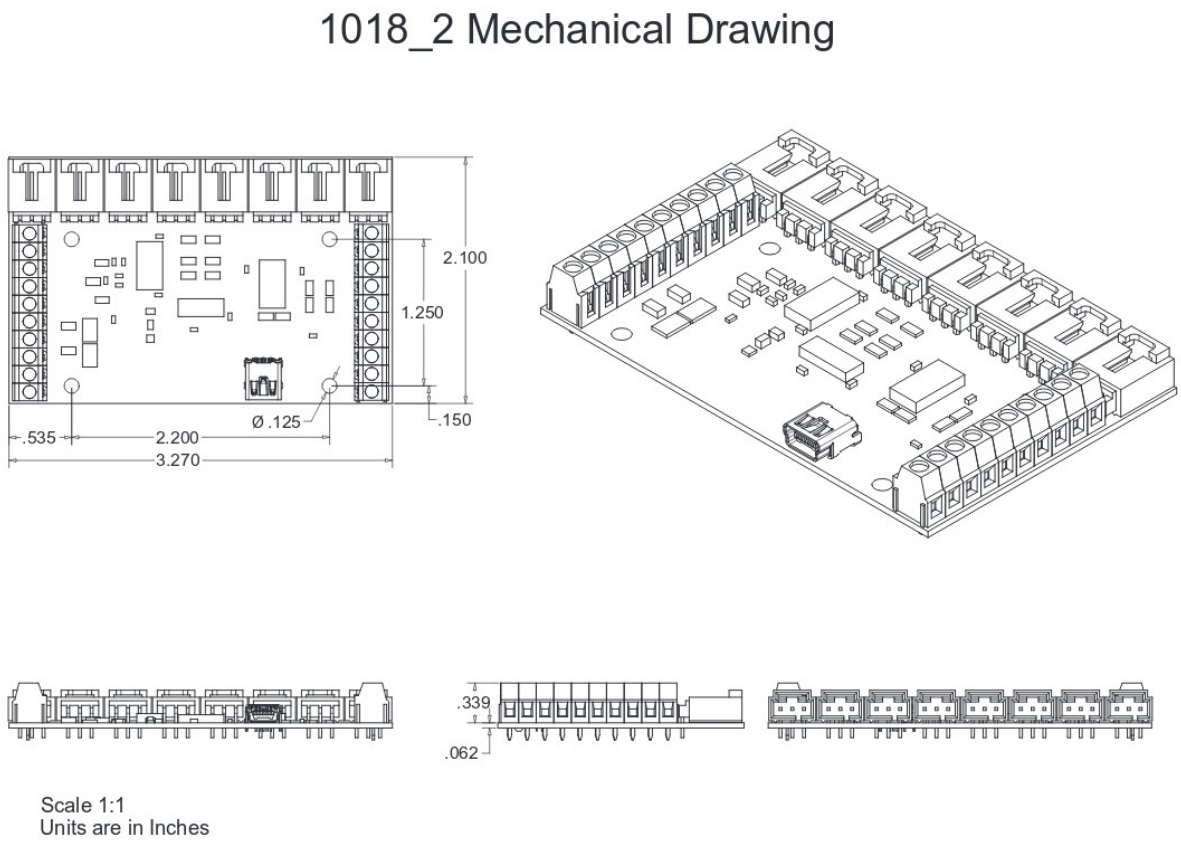
1.2 Digital Inputs

The Digital Inputs have a Digital Input Hardware Filter to eliminate false triggering from electrical noise. They can be used to convey the state of devices such as push buttons, limit switches, relays, and logic levels.

1.3 Digital Outputs

The Digital Outputs can be used to drive LEDs, solid state relays, transistors; in fact, anything that will accept a CMOS signal.

2.Product Specifications



2.1 Board

API Object Name	InterfaceKit
USB Voltage Min	4.6 V DC
USB Voltage Max	5.5 V DC
Current Consumption Min	13 mA
Current Consumption Max	500 mA
Available External Current	487 mA
Recommended Wire Size	16 - 26 AWG
USB Speed	Full Speed
Operating Temperature Min	0 °C
Operating Temperature Max	70 °C

2.2 Analog Inputs

Number of Analog Inputs	8
Analog Input Resolution	10 bit
Input Impedance	900 kΩ
Analog Input Voltage Min	0 V DC
Analog Input Voltage Max	5 V DC
5V Reference Error Max	0.5 %
Analog Input Update Rate Min	1 samples/s
Analog Input Update Rate Max (4 Channels)	1000 samples/s
Analog Input Update Rate Max (8 Channels)	500 samples/s
Analog Input Update Rate Max (WebService)	62.5 samples/s

2.3 Digital Inputs

Number of Digital Inputs	8
Pull-up Resistance	15 kΩ
Low Voltage Max (True)	900 mV DC
High Voltage Min (False)	4.2 V DC
Low Voltage Trigger Length Min	4 ms
High Voltage Trigger Length Min	15 ms
Digital Input Voltage Max	± 15 V DC
Digital Input Update Rate	125 samples/s

2.4 Digital Outputs

Number of Digital Outputs	8
Series Resistance	300 Ω
Digital Output Current Max	16 mA
Digital Output Voltage Min	0 V DC
Digital Output Voltage Max	5 V DC

3.Programing in phidgets

3.1 Installing phidgets

In order to use phidgets you have to set up your system for your device. In this link: http://www.phidgets.com/docs/Operating_System_Support, you can find all the operating system supported by phidgets.

3.2 Basic Functions

In order to use phidgets ,you'll have to:

1. Create a Phidget Software Object, you'll need him to use your device functions.
   Python example: `interfaceKit = InterfaceKit()`

2. Then open phidgets using the object;
   Python example: `interfaceKit.openPhidget()`

3. Detect when a Phidget is attached using the object,the input of your function is time in ms;
   Python example: `interfaceKit.waitForAttach(10000)`

4. Now you can use the functions provided by your object, if you are using an interface kit and wants to get a value from a sensor connect in port 7, you can use the following command;
   Python example: `interfaceKit.getSensorValue(7)`

5. When you are done, close the object
   Python example: `interfaceKit.closePhidget()`

Since you are using an external package, don't forget to include the phidgets library , in this case we are using a interface kit.

Python example: `from Phidgets.Devices.InterfaceKit import InterfaceKit`

3.3 Example Program

In the following example, this script will be used to detect any variation in the sensors connected in the interface Kit;

```
#!/usr/bin/env python
#The packages that will be used
from Phidgets.Devices.InterfaceKit import InterfaceKit
import sys
#This function will receive the device as argument
def interfaceKitSensorChanged(e):
    source = e.device
    print("InterfaceKit %i: Sensor %i: %i" % (source.getSerialNum(), e.index, e.value))
#First you'll have to create the device object.
interfaceKit = InterfaceKit()
#Then open the device
print("Opening phidget object....")
interfaceKit.openPhidget()
#Wait for the device to be attached
print("Waiting for attach....")
interfaceKit.waitForAttach(10000)
```

```
#This method will call the funcion used in the argument everytime any sensor value
changes
interfaceKit.setOnSensorChangeHandler(interfaceKitSensorChanged)
#This method will maintain the program running unless you press ENTER
print("Press Enter to quit....")
chr = sys.stdin.read(1)
print("Closing...")
#Closing the program
interfaceKit.closePhidget()
print("Done.")
exit(0)
```

3.3.1 The code explained

First, we'll have to declare the packages that will be used, in this case we'll use the Phidgets InterfaceKit and the package sys.

```
from Phidgets.Devices.InterfaceKit import InterfaceKit
import sys
```

Then we declare the function **interfaceKitSensorChanged** that will be used to print in the screen the sensor value, index and the device serial number

```
def interfaceKitSensorChanged(e):
source = e.device
print("InterfaceKit %i: Sensor %i: %i" % (source.getSerialNum(), e.index, e.value))
```

Now, we'll create the software object, open phidgets and wait for device to be attached.

```
#First you'll have to create the device object.
interfaceKit = InterfaceKit()
#Then open the device
print("Opening phidget object....")
interfaceKit.openPhidget()
#Wait for the device to be attached
print("Waiting for attach....")
interfaceKit.waitForAttach(10000)
```

The method **interfaceKit.setOnSensorChangeHandler** will call the function used as argument everytime any sensor changes its value, in this case the function will be: **interfaceKitSensorChanged.**

```
#This method will call the funcion used in the argument everytime any sensor value
changes
interfaceKit.setOnSensorChangeHandler(interfaceKitSensorChanged)
```

Then , we have the command that will 'hold' your program until you press enter, and after that, close phidgets and exit.

```
print("Press Enter to quit....")
chr = sys.stdin.read(1)
print("Closing...")
#Closing the program
interfaceKit.closePhidget()
print("Done.")
exit(0)
```

4.Important functions and libraries

interfaceKit.getSensorRawValue(self,index)

This method is useful if you want maximum accuracy, you will have to change the sensor formula when using the **getSensorRawValue,** you can find the new formula in the device datasheet.

PhidgetException

Phidget Exception is the error given by the Phidgets when something goes wrong, in order to use this you will have to include the Phidgets.PhidgetException in your code, like the example above.

```
from Phidgets.PhidgetException import PhidgetErrorCodes, PhidgetException
```

Generally in Python, you can use the exception with **try**:and **except** statements. In this script we'll try to open phidgets except we have an error:

```
try:
        interfaceKit.openPhidget()
except PhidgetException as e:
        print("Phidget Exception %i: %s" % (e.code, e.details))
        print("Exiting....")
        exit(1)
```

4.Recommended readings

Phidgets 1018_2 product page:
http://www.phidgets.com/products.php?category=0&product_id=1019_1

Phidgets Python library documentation
http://www.phidgets.com/documentation/web/PythonDoc/Phidgets.html

Genneral Phidgets Programming
http://www.phidgets.com/docs/General_Phidget_Programming#Do_Things_with_the_Phidget

Phidgets operating system support
http://www.phidgets.com/docs/Operating_System_Support

5.REFERENCES

All the images and tables cointained in this document were download from phidgets 1018_2 InterfaceKit Product Page.