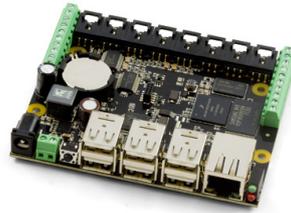


1073 User Guide



Go to this device's product page ^[1]

Getting Started

Checking the Contents

You should have received:

- A Power Supply
- A Cat-5e network cable
- A Mounting kit (4 nuts & bolts, 4 plastic spacers)
- A PhidgetSBC3 Board



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

- A short length of wire to test the digital inputs
- An LED to test the digital outputs
- An Analog Sensor to test the analog inputs
- A UVC compatible Webcam

Connecting the Pieces

1. Connect the analog sensor to the analog input port 4 using a Phidgets sensor cable. The analog input ports are numbered from 0 to 7 starting from the left.
2. Connect one end of a wire to digital input port 0 and the other end to ground (labelled 'G' on the underside of the board).
3. Connect the LED by inserting the long LED wire into the digital output 7 and the shorter wire into Ground.
4. Connect the power supply to the PhidgetSBC3 using the barrel connector.
5. Connect the PhidgetSBC3 to your network with an Ethernet cable. Plug the wall adapter into an appropriate outlet. The red status indicator light located near the USB ports should be lit if the unit is receiving power. The green LED located above the red LED indicates boot status. The green LED will turn on and off once during boot and then turn back on when everything is running.
6. Other Phidgets can also be connected to the 1073 using a USB cable.



Testing Using Windows 2000 / XP / Vista / 7

Make sure you have the current version of the Phidget library installed on your PC. If you don't, follow these steps:

1. Go to the Quick Downloads section on the Windows page
2. Download and run the Phidget21 Installer (32-bit, or 64-bit, depending on your system)
3. You should see the  icon on the right hand corner of the Task Bar.

Running Phidgets Sample Program

Double clicking on the  icon loads Phidget Control Panel; we will use this program to ensure that your new Phidget works properly.

- Make sure that the PhidgetSBC3 is powered and properly connected to your network.
- Click the **PhidgetSBC3** tab in the Phidget Control Panel.
- Double click on the PhidgetSBC3 device to bring up the PhidgetSBC3 configuration panel in your default web browser.
- You can differentiate multiple PhidgetSBC3s by their MAC address, which is printed on the sticker on the back of your board.



This control panel view will tell you (at the bottom):

- The **link local** address of the SBC (called the mDNS address), here it is `phidgetsbc.local`
 - The name `phidgetsbc.local` is the default link local address
 - Although there is a period after the link local address in the Phidget Control panel, the address itself does not include that final period
- The **IP (Internet Protocol)** address of the SBC, here it is 192.168.2.181
 - There is no default for the IP address
 - The SBC will only have an IP address if it successfully gained one through DHCP (Dynamic Host Configuration Protocol) over Ethernet

You will need one of these two pieces of information when you start working directly with the SBC Operating System for tasks such as writing and running code on the SBC, so write these down somewhere while you work through the rest of this Getting Started page.

Basic Use

Basic use of the PhidgetSBC allows the opening of connected Phidgets over the network. Using another Phidget with the PhidgetSBC in this way is almost exactly like using Phidgets over USB, in respect to the API calls and behavior. However, some extra considerations need to be made when working with the PhidgetWebservice.

Phidget Webservice

Support for opening Phidgets over the network is made possible via the Phidget Webservice. This allows a user to write an application in a system and language of their choosing and then operate Phidgets connected to the PhidgetSBC. It is a socket based server that runs on the PhidgetSBC, and allows any attached Phidgets to be seen and opened directly over the network. To use this server go to the "Phidgets->WebService" tab in the web configuration page and enable it.

Opening and controlling a Phidget over the network is nearly the same as opening one locally. The main differences are:

- Different open calls that include server information. New calls `OpenRemote` and `openRemoteIP` (naming depends on language).
- Access to Webservice based properties: Server hostname, port and ID.
- Access to server connect and disconnect events, and network error events.
- Phidgets can be opened by more than one separate application at the same time.
- Reliability is more of a issue because network connections are easily broken

Opening a Phidget over the network is asynchronous and pervasive, just like opening locally. This means that if a connection to the remote server cannot be established right away, it will keep trying indefinitely, and even survive the server being stopped and started, etc. Instances of the Phidget Webservice can be referred to either using hostname (IP Address) and port number, or by Server ID. The advantage of using a Server ID is that it stays consistent compared to IP addresses, and you don't need to know the Port number. A Webservice Server ID is assigned when the Webservice is run - which on the PhidgetSBC defaults to 'phidgetsbc'. In order to use a Server ID, the Bonjour utility also needs to be installed. Refer to the Programming Manual and the API manual for your language for more information about using the Phidget Webservice.

Reliability

Determining reliability needs can become important while opening Phidgets over the network, because the network connection can potentially be interrupted at any time. This can leave the network attached Phidget in an undesirable state. For example - if a motor controller is driving a motor and the connection is lost, there is no way to stop the motor until the connection is re-established. These issues are less important if you are just receiving sensor data from an Interface Kit.

It's generally a good idea to catch server connect and disconnect and Phidget attach and detach events in order to know the state of the connections. It's also a good idea to catch error events - this is where network errors will be reported. If reliability is important, you should consider writing a program to run locally on the PhidgetSBC, and communicate with it through the Dictionary interface. This way, if the connection is broken, the local application will notice and be able to take any appropriate actions. See the advanced chapter for more information.

Finding Phidgets on the Network

Any Phidgets attached to the PhidgetSBC can be identified using the Status >> Phidgets page in the configuration interface, and should be seen on the network through the Webservice. The Phidget Control Panel has a Bonjour tab (under WebService >> Bonjour) that lists all detected network attached Phidgets. The Phidgets connected to the PhidgetSBC should be seen here and can be opened by double clicking its name in the menu. Network attached Phidgets can also be located programmatically with the Phidget Manager. The Phidget Manager is used with either hostname and port, or server ID, just like with 'Open'. The manager can also be used to find all Phidgets on any Webservice through Bonjour, by specifying a NULL Server ID. See your specific language's guide for more information about coding with the Phidget Manager.

Testing Using Mac OS X

The steps are very similar to the Windows process described above:

1. Ensure you have OS X 10.3.9 or later running.
2. Download and run the Installer ^[2] (OS software license ^[3])
3. Click on System Preferences → Phidgets (under Other) to activate the Preference Pane
4. Make sure that the Phidget SBC is properly attached to the network as described in the Getting Started section
5. Double Click on the Phidget SBC in the Phidget Preference Pane to bring up the Web Interface in your default browser
6. Or, click on the Bonjour tab to see attached Phidgets being run over the network and to bring up their examples

The address in the browser that connects to the SBC is either an **IP address** (such as 192.168.2.181) or a **link local address**, (such as `phidgetsbc.local`, which is the default). Note down the address in the browser, as you will need this information for later if you will be working directly with the SBC to perform tasks such as writing or running code. But for now, with the web interface open, we have a section to walk you through it.

Using Linux

With Linux, you have many setup options, but all involve knowing the IP address or link local address of the SBC after you have plugged it in as described in Getting Started section. The IP address can be somewhat difficult to obtain, but the default link local address for all new Phidget SBCs is `phidgetsbc.local` - which is an mDNS address. Wait at least three minutes after booting the SBC to make sure link local addressing is started. You'll also need some form of mDNS (either `avahi` or `mDNSResponder`) installed on your main computer. The `avahi` service is usually installed by default on most Linux machines, try `which avahi-resolve` to make sure. Then, try typing `phidgetsbc.local` into a web browser. The web interface should come up, starting with the Set the Password screen.

Note that some browsers (i.e. Google Chrome) combine search and addressing in the same address bar, so you may need to turn off web service and prediction service in Preferences → Under The Bonnet for Chrome to treat `phidgetsbc.local` like a web address rather than a search term. If these steps do not bring you to the initial setup password screen as shown in the SBC Web Interface section, you will probably need to read the internet setup section on the SBC OS Page. That section contains more than just troubleshooting information - it includes in-depth information on how the SBC starts its network, your initial network configuration options, and how to connect to the SBC without mDNS using both DHCP and static IP.

If you want to use the SBC to broadcast data from Phidgets over a network, the SBC is already automatically performing this function with its attached Interface Kit, and will also do so for any Phidgets plugged into its USB ports. If this is your sole intended use of the SBC, you can skip ahead to our example on using the attached Interface Kit.

However, the SBC is much, much more than simply a way to get data from Phidgets over a network. You can use the SBC as an external Linux computer. You'll need to set a password using the SBC web interface, and write down a network address of the SBC (`phidgetsbc.local`, or an IP address if you worked through the internet setup section on the SBC OS Page). But after that, if no other sections in the basic SBC web interface section apply to you (using the webcam, setting up wireless networking, or checking system parameters like memory), you can skip ahead to the OS - Phidget SBC page.

Linux - Attached Interface Kit

As soon as the SBC boots and can connect to the network (either by DHCP or by mDNS/avahi), it begins broadcasting the state of its attached InterfaceKit over the network using the Phidget Webservice. To test the InterfaceKit on your Linux computer over the network, you will need to install the Phidget libraries and the Phidget webservice if you haven't already. (If you have used any Phidget via USB and over a network on your Linux computer, you have already done this.) This process is described on the general Linux OS page, so follow those installation instructions, with the following modifications to the Linux webservice introduction:

- Instead of using a localhost (127.0.0.1) address, use the the SBC's IP or link local (`phidgetsbc`) address,
- Instead of using the function call `openRemoteIP` in your code, use the function call `openRemote`, and the Interface Kit serial number which is on the back of the SBC.

Note that any attached analog sensors in the black ports will not show up over the webservice as individual Phidgets. Rather, they will show up as part of the Interface Kit, through the port number that they are attached to on the SBC board.

Using Windows Mobile / CE 5.0 / CE 6.0

For a step-by-step guide on getting Phidgets running on Windows CE, check the Windows CE page.

SBC Web Interface

Using the operating system sections of this guide, you should have been able to open a web browser with a connection to the SBC using either:

- The Phidget Control Panel (Windows), or
- The Phidget Preference Pane (Mac OS), or
- An mDNS connection (Linux).

Here we cover the basics on the tool that has been opened in the browser, known as the SBC Web Interface.

Set the Password

If this is your first time connecting to the SBC (or after a complete factory reset), you will see a screen asking you to set a password:

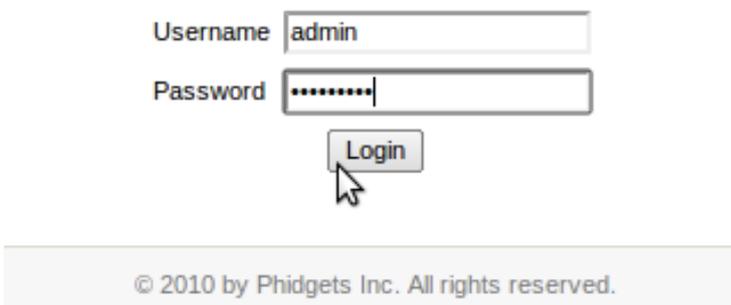


This password will be linked to the user `admin` if you use the web interface in the future, and to the user `root` if you choose to use SSH. These are actually the same user, and the SSH connection is covered in-depth on the SBC operating system page.

Remember this password, as you will need it to connect to the SBC in the future, and the only way to log in if you lose it is to perform a full factory reset on the SBC.

The next time you connect to the SBC after you log out or reboot the SBC, you will be asked to log in with a user name and password:

PhidgetSBC Administration Console



Navigating

The SBC Web Interface has a series of tabs and sub-tabs at the top, and by using these you can access the entire web interface. Note that the web interface only allows you to configure a very small portion of the SBC, please refer to the full OS - Phidget SBC page for an in-depth introduction on the SBC Linux operating system.

The header also has information about how long the SBC had been powered on without a reboot, what the link local address of the SBC is (here it is `phidgetsbc`, so the full link local address is `phidgetsbc.local`) and more:

The starting page of the web interface will also give you information about:

- The name of the SBC (you can rename it under `System` → `General`, although note that this will **also change the link local address of the SBC to that name**)
- The build, kernel, and board revision
- The MAC address (of the Ethernet connection...the wireless MAC address can be found on `Network` → `Status` when the adapter is plugged in)
- The space left in your / filesystem (the amount of space you have for programs, and files not written to an external USB key)
- Any additionally mounted filesystems (such as a USB data key)
- The amount of memory your SBC is currently using. This can be a bit misleading however. To get a better idea of the total memory usage go to the "Processes" subtab and look at memory usage on a per-process basis.

This can also be accessed later from `Status` → `System`:

Board Name	PhidgetSBC2
Board Revision	200
Webif Version	2.0.0
Kernel Version	Linux version 2.6.36.2 (patrick@debian) (gcc version 4.3.2 (Debian 4.3.2-1.1)) #6 PREEMPT Wed Jan 12 10:48:24 MST 2011
Distribution	Debian GNU/Linux 6.0
Phidget Library	Phidget21 - Version 2.1.7 - Built Jan 14 2011 15:31:56
Current Date/Time	Mon Jan 17 20:35:32 UTC 2011
MAC Address	00:1b:0b:00:00:78

Mountpoint	Size	Used	Available	Usage
/	420M	75M	345M	18%
/media/usb0	979M	507M	472M	52%

Memory usage: 43% of 59.5MB

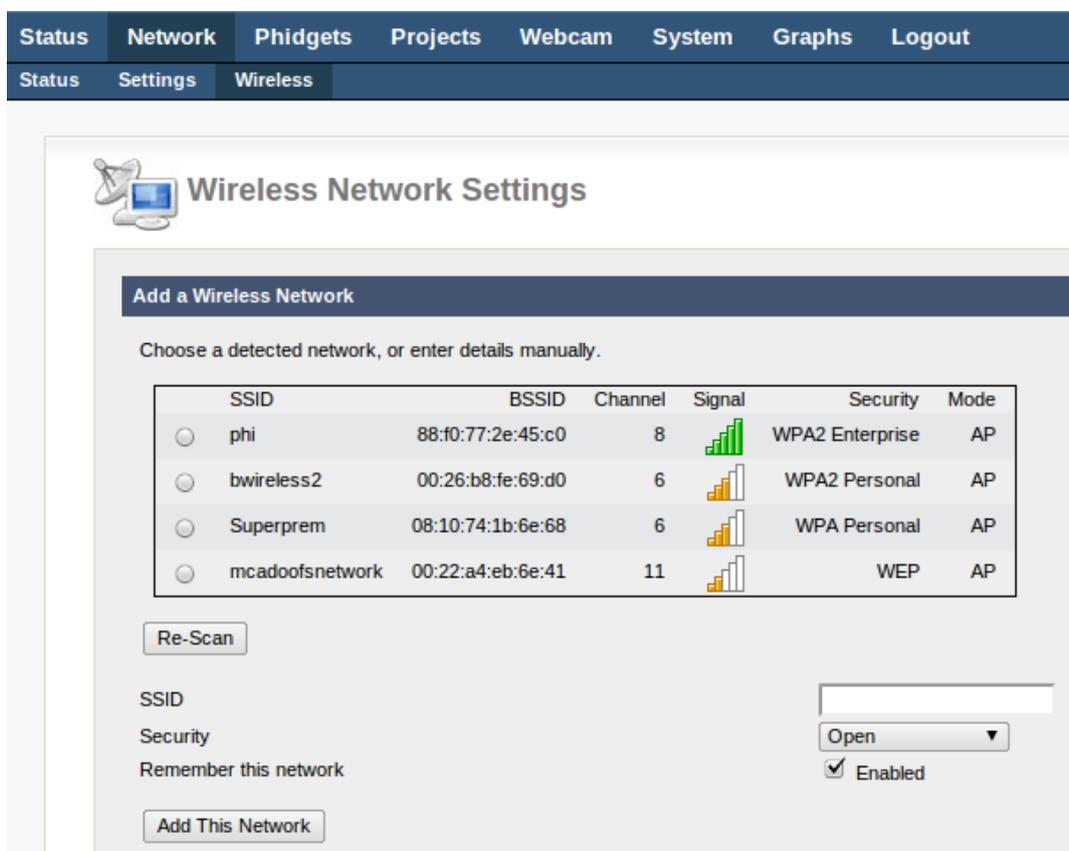
Set up Networking

After using the initial wired Ethernet connection (as described in detail in the hardware section) to access the web interface for the first time, you can use the `Network` tab in the web interface to configure other network settings for use.

The SBC can switch dynamically between Ethernet (what it tries first) and wireless via the Wi-Fi USB Adapter w/SMA Antenna ^[4]. It is also able to dynamically handle both Ethernet and wireless being unplugged and then plugged back in. This is especially useful if you are designing a project where the SBC will operate without a network - when testing you can plug in and remove the network adapters to check on how the SBC is doing.

Wireless

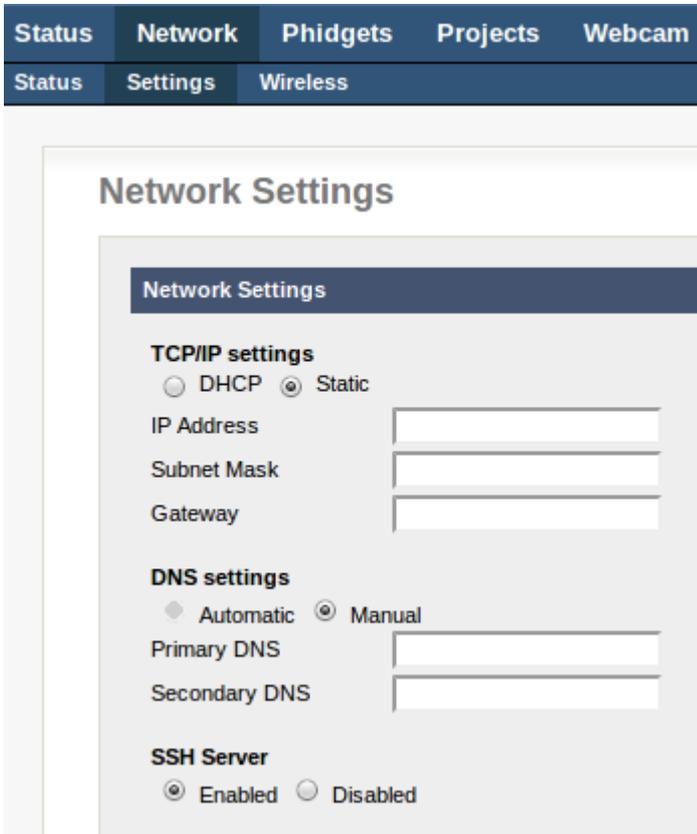
The SBC can only access wireless networks using the Wi-Fi USB Adapter w/SMA Antenna ^[5]. If this WiFi adapter is plugged in to a USB port on the SBC, you will see something like this under `Network` → `Wireless` in the web interface:



Here you can add and remember networks with passwords. These do not necessarily have to be networks that are currently seen - you can add the SSID and password of your home network, for example, and the next time the SBC boots at home with wireless only it will connect to your home wireless network. These SSID settings are only for DHCP networks.

Static IP

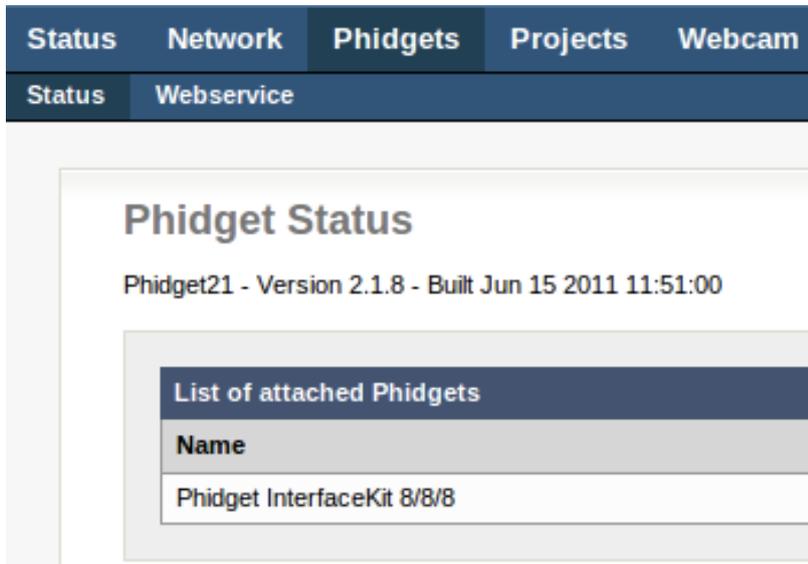
If you do not have DHCP on your main network, and do not want to use link local addressing, you can set the network connection to be static. These settings can be found in the web interface under *Network* → *Settings* for the Ethernet interface, and at the bottom of *Network* → *Wireless* for wireless networking. In the Ethernet settings, for example, you can specify network elements manually in the same way you would for a desktop computer:



This setting is also useful to work entirely without mDNS and link local addressing. After setting it, the SBC will start using that static IP immediately, so if this is done improperly this can make the SBC very hard to re-connect to depending on the routing within the rest of your network. If you are not sure whether this will be a problem, it may help to read the in-depth network troubleshooting section for the SBC.

View Attached Phidgets

You can view Phidgets that are directly attached to the SBC by going to `Phidgets` → `Status` in the web interface:



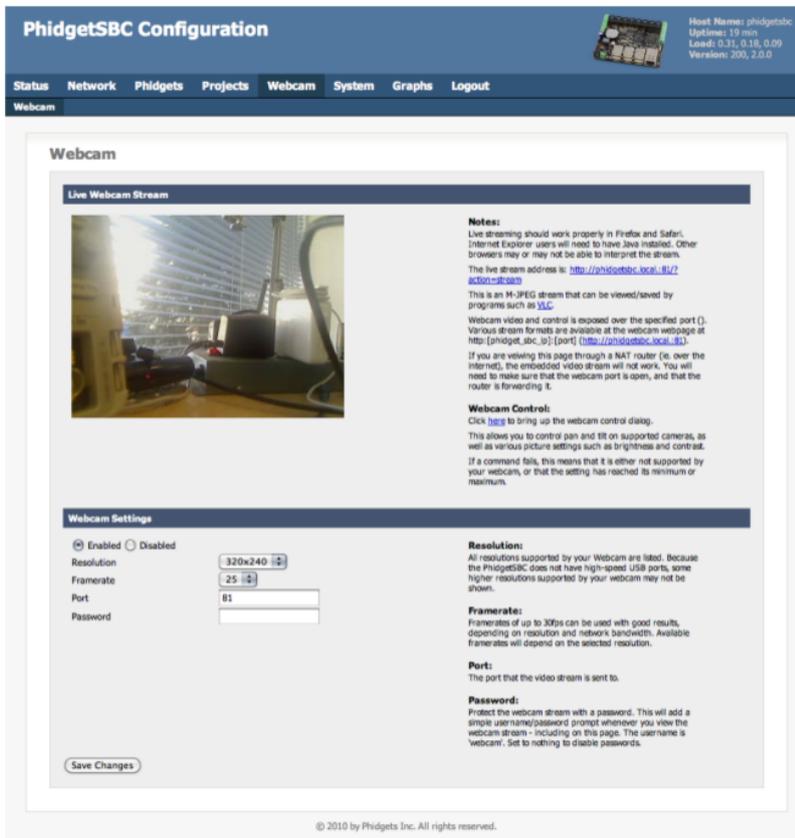
This will always show the attached Interface Kit that is part of the SBC board. It will also show any Phidgets plugged into the SBC's USB ports.

On the right hand side of this screen (not shown in the screenshot) you will see the serial numbers of the attached Phidgets. If you want to use these Phidgets within your code over a network using the webservice, you can use these serial numbers in your programming project.

Note that any analog or digital sensors will not show up here, as they are a 'part' of the Interface Kit and can be accessed through the ports in the Interface Kit API. Taking a look at the User Guide for the `PhidgetInterfaceKit 8/8/8` ^[6] will show you a good example of how this works.

Use the Webcam

Using the webcam (either the Phidget USB Webcam ^[7] or another UVC-compatible webcam ^[8]) should be fairly straightforward. Plug it in, then head to the web interface under Webcam → Webcam:



This will allow you to stream video into the web interface. This stream can also be viewed with any compatible M-JPG viewer, such as VLC. To take pictures with the webcam within your code, see the how-to on the main SBC operating system page.

Webcams that support pan/tilt can be controlled via the web interface. Multiple frame rates and resolutions are supported.

If the video stream is to be exported over the internet, it is recommended that the password be enabled. However, it must be noted that this is a simple HTTP authentication, which is sent unencrypted, and is thus not highly secure.

If prompted for the webcam password, the username is 'webcam'.

If multiple webcams are attached, they will all start up using the same settings, except that each additional webcam will run on the next higher port number. Multiple webcams will generally only work when the resolution and framerate are set to low levels.

At this point, we have covered most of the common uses of the web interface. But the web interface offers many more configuration options. This reference section covers all of the available options. If you want to continue on to tasks such as writing and running code on the SBC, visit the OS - Phidget SBC page.

Can I use multiple webcams?

Probably... In theory you can connect multiple webcams to the SBC, however, the bandwidth of a USB1.1 hub will limit the amount of data you can stream meaning you might have to use reduces resolution/framerate to do it successfully.

Web Interface Reference

This section provides a complete reference to all tabs and sub-tabs within the SBC web interface. Note that the web interface itself also includes help as text on the right side of most tab screens.

Top Bar

On loading the interface, you will see a tool bar along the top of the page. It holds some information across all the configuration pages. The information is as follows:

Host Name

The host name given to the PhidgetSBC on the network.

Uptime

Total time elapsed since the last reboot.

Load

The average CPU utilization in the last minute, 5 minute, and 10 minute durations.

Version

The current board and web interface version.

Tab: Status

The main tab *Status* has options for *System*, *Network*, *Processes*, *USB*, and *Phidgets*.

System

This is the first page you should see after loading the configuration Interface. It contains general information about the SBC.

System Information

Board Name

Name of the device. It should always read "PhidgetSBC2".

Board Revision

Board revision number. This tracks the hardware design.

Webif Version

The version of the web interface currently being used. This will change with updates to the web interface/configuration system.

Kernel Version

The type and version of the loaded Linux kernel.

Distribution

The running Linux distribution name/version. This should read "Debian GNU/Linux 6.0"

Phidget Library

The version of the installed Phidget21 library.

These libraries are included with the firmware, and may need to be updated to use newly released Phidgets.

Current Date/Time

Current date and time, within the SBC

MAC Address

The Ethernet MAC address.

A PhidgetSBC is uniquely identified by its MAC address shown here.

This address is also printed on the label of the underside of the PhidgetSBC.

Other Phidgets, including the integrated InterfaceKit, use a serial number to identify themselves.

Filesystem

All mounted filesystems are listed, along with their size and usage.

Memory

Memory usage is shown. Wired/active memory is shown in red and cached/inactive memory is shown in blue.

Network

See: Tab: Network

Processes

This lists all running processes, along with their Process ID (PID), User, State, CPU usage and memory usage. Advanced users can use this to tell if any application is using too much memory, or has crashed.

USB

This lists all USB devices. The “S3C24XX OHCI” Host Controller, the “General Purpose USB Hub” and the built in Interface Kit 8/8/8 should always be listed, along with any connected devices. Also listed are any mounted USB drives.

All connected devices

A list of all the USB devices present in the system. This includes the main USB, the built in 6 port hub, and all Phidget and non-Phidget devices.

Mounted USB devices

This area lists of all the USB based drives connected to the PhidgetSBC, and their mount points.

USB drives are automatically mounted at /media/usb(0-9) when attached.

Unmount (Button)

Use this button before removing the device to safely disconnect it.

Phidgets

See: Tab: Phidgets

Tab: Network

The main tab *Network* has the sub-tabs *Status*, *Settings*, and *Wireless*.

Status

General network status can be viewed on this page. Modifying these values are done on other pages.

Network

Adapter

Abbreviated name and number of the network interface.

Type

Wired or wireless connection.

Mode

Network protocol used.

IP Address

The IP address of the network interface.

Subnet Mask

The subnet mask of the network interface.

Gateway

The IP address of your gateway.

MAC Address

The MAC address of the interface.

Wireless State

Connection status information for the Wireless link.

This could be 'CONNECTED', 'INACTIVE', 'FAILED', etc.

Wireless SSID

The plaintext name of the wireless connection access point.

Wireless Security

Security protocol used for a wireless link.

DNS Server(s)

List of nameservers being used.

Settings

This is where TCP/IP settings for the wired ethernet are configured. This is also where SSH is enabled.

TCP/IP settings

DHCP will set the system IP Address, Subnet Mask, and Gateway automatically.

In the absence of a DHCP server, Static should be used and filled in manually.

Note that the same TCP/IP settings will be used at all access points.

DNS settings

DNS can be set up automatically if DHCP is enabled.

Under manual settings, up to two DNS servers can be specified.

Note that DNS settings are system-wide and will apply to all interfaces.

SSH Server

This is where the SSH server can be enabled or disabled.

Enabling SSH for the first time can take several minutes as the keys are generated.

Wireless

Wireless networking is supported via a USB wifi adapter ^[4]. When an adapter is plugged in, this wireless configuration page will be available.

Wireless networks are joined based on a list of saved networks. You can join, manually enable and disable, as well as delete these saved networks. To add a wireless network to this list, either choose from the list of detected networks, or enter the details manually. Supported security includes WEP, WPA(2) Personal and WPA(2) Enterprise.

Saved networks will be joined first based on security and secondly based on best signal strength.

Add a Wireless Network

SSID

The SSID of the access point that you wish to add. This is the plaintext name of the access point.

Security

The security system used by this access point.

Remember this network

If enabled, this network will be added to the list of saved networks permanently, and will be available to be automatically joined in the future.

Otherwise, this network will remain in the list of saved networks until the board is reset, or another network is added.

Manage saved networks

Join This Network

Joining a specific network will temporarily disable all other saved networks, so that the specific network will be joined, if available.

The other networks will remain disabled until the board is reset, or another network is added.

Delete This Network

Delete a saved network. There is no confirmation and this cannot be undone.

Enable / Disable

Selected networks that are enabled will be joined automatically.

Disabled networks will never be joined.

Wireless Network Settings

TCP/IP settings

DHCP will set the system IP Address, Subnet Mask, and Gateway automatically.

In the absence of a DHCP server, Static should be used and filled in manually.

Note that the same TCP/IP settings will be used at all access points.

DNS settings

Switch the DNS settings between Automatic and Manual.

DNS can be set up automatically if DHCP is enabled. Otherwise, up to two DNS servers can be specified.

Note that DNS settings are system-wide and will apply to all interfaces.

Tab: Phidgets

Status

Library Version

The version of the installed Phidget21 library. These libraries are included and are updated along with the firmware.

List of attached Phidgets

A list of all detected Phidgets connected to the PhidgetSBC.

This includes the integrated PhidgetInterfaceKit and displays both the serial number and version.

WebService

The Phidget webservice is a simple server that allows Phidgets connected to the PhidgetSBC board to be opened over the network. This is enabled by default and starts with the SBC.

This page lets you view and modify the Phidget webservice settings, as well as stop and start the server. Please see the Advanced User's Guide for more information on programming with the dictionary.

Enabled/Disabled

Enables or disables the Phidget WebService.

Server ID

Server ID is used when opening a connection to the PhidgetSBC using the mDNS based openRemote calls.

This is by default the same as the PhidgetSBC hostname (phidgetsbc), but can be set to anything (up to 63 characters).

Port

Port is the port that the webservice runs on - default is 5001.

Password

The password is used for securing the webservice.

By default, this option is disabled with a blank password.

Note that while the authentication protocol and password is encrypted during authentication, all following data is sent in the clear.

Start/Stop (Button)

Use this button to start/stop the webservice.

Tab: Projects

This is where user projects are set up. Custom applications can be written in either C (compiled to an ARM architecture) or Java, and then set up to run on the PhidgetSBC at system startup. We offer an in-depth tutorial on creating applications in this manner using Java.

Projects

This sub-tab lists installed applications as well as the controls for creating a new application space.

This is also where you can download the SBC-specific version of the Phidget Java libraries: phidget21.jar

List of Projects

This is a list of all created applications and their current status. Enabled applications will try to run on system boot, and the stopped/running status indicates if the program is currently executing.

Delete applications using the red 'X' near their name. You can click on the application name to launch the Specific Project page.

Create new app (Button)

This button creates a new application space using the input field for its name. Application names should not contain spaces.

Free space remaining on userspace partition

The amount of free space remaining on the user partition in bytes.

Specific Project

On a specific application space page, there are controls to start and stop the program, as well as view the stdout and stderr from the most recent (or current) run.

Start/Stop (Button)

This button is for starting or stopping the execution of the program specified under application settings.

Starting a program will generate stdout and stderr logs.

View stdout

You can view the standard console output of your program through this link.

View stderr

In the event of an error that halts program execution, its corresponding error message is printed here.

Filesystem Browser

This allows viewing, editing and removal of application files, as well as the ability to upload new files. File upload size is limited to 5MiB per file.

Startup Settings

This section configures the application to start at boot. When this enabled, it will start at the end of the system boot process (after things like bringing up the network, starting the Phidget WebService, etc.). The startup order field specifies a start order among the custom applications, with lower numbers being started first.

Executable name is the name of the file to execute. If this filename ends in '.class' or '.jar', the program will be run as a Java program, otherwise it is run as an ARM Binary. We have an detailed walkthrough of this section.

Tab: Webcam

Webcam Settings

Enabled/Disabled

Enable or disable the webcam streaming video.

Video streaming can consume a lot of bandwidth depending on the settings used.

Resolution

The resolution of the capture in pixels.

Only resolutions supported by the webcam are listed.

Framerate

The transmission frame rate of the capture.

Available frame rates will depend on the selected resolution.

Port

The port that the video stream is sent to.

Password

Protect the webcam stream with a password.

This will add a simple username and password prompt whenever you view the webcam stream - including on this page.

The username is 'webcam'.

Set to nothing to disable passwords.

Tab: System

General

This is where general system setting are set up.

System Settings

Host name

The system hostname.

This is used for the system's mDNS (i.e. local link address) hostname, as well as the Phidget WebService default Server ID.

All PhidgetSBCs have a default hostname of `phidgetsbc`. When used as a link local address, the extension `.local` is added

Time Settings

Timezone

Set up your time zone according to the nearest city of your region from the predefined list.

Zoneinfo String

Standard zoneinfo names are defined for different areas of the world. You can supply any zone from the official list as an alternative to selecting a predefined zone from the list.

Logs

This is where the kernel and system logs can be viewed. This also includes the ability to filter the text. We also have a special section on checking the system logs.

Text Filter

Text to Filter

Insert a string that covers what you would like to see or exclude.

You can use the regular expression constants like: `00:[[:digit:]]{2}:[[:digit:]]{2}` or `.debugl.err`

Filter Mode

You will see only messages containing the text in the Include mode while you will not see them in the Exclude mode.

Remove Filter

Clears the filter being used

Filter Messages

Change the filter being used. Including a blank Text to Filter effectively removes the filter.

Password Change

This is where the system password can be changed. The system password is the (which is the `admin` password for the web interface) is also the `root` user password on the SBC when using SSH.

Changes here made will take effect immediately after being saved, without asking for confirmation. The new password must consist of alphanumeric characters and be at least 1 character long.

Password Change

New Password

The first field for a new password.

Confirm Password

The second field for a new password. This must match the first field.

Set Password

This button will commit the changes to your password.

File Editor

This grants access to the full filesystem. Files can be uploaded, edited and deleted, and directories can be created. Some important system directories and files are protected from deletion, but it is still easy to break the system by editing/deleting files.

File upload is limited to a filesize of 5MiB.

File/Directory permission, owner/group and creation time information is available by hovering over the file/directory icon.

Backup & Restore

This is where the system can be backed up to disk and restored. This also allows access to the recovery/upgrade system.

Note that this **only** backs up configuration files directly related to the web interface - such as network configuration, hostname, time settings, and user project setting (but not including project files). For other methods of backup, including installed software, we have a detailed walkthrough.

When restoring from a backup file, the system will check that it is a valid backup before asking the user to continue.

Backup Configuration

Name this configuration

You can give the backup file an arbitrary name. The name is only shown during restore.

Backup

This button creates the backup.

A download link to the backup will be provided and you will be prompted to save the file to an off-SBC location. The download link is **not** valid indefinitely, use it then.

Restore Configuration

Saved backup file

Choose a backup file for this machine type.

The restore system will check the file and ask for confirmation before running the restore.

Restore

This button applies the backup.

The backup file will then be verified.

Clicking restore again will commit the changes and take effect immediately.

Reset Configuration

Reset

This will reset all of the web interface configuration files to their default states. This will **not** reset the system as a whole.

Upgrade / Factory Reset

Go (Button)

This will boot the SBC into the Recovery/Upgrade system.

Refer to our section on the recovery system for more information.

Packages

This is where package updates are performed. From time to time, updates will be made available. These will address any bugs and security issues, add new features, and update the phidget21 library and webservice. There are also

some other package management options available.

Note that this page can take quite a long time to load when 'Include full Debian Package Repository' is enabled.

Upgradable Packages

This lists all available package updates, if any.

Upgrade All Packages

This installs all available package updates.

Selective updating can only be done via SSH.

Refresh Available Packages

This runs apt-get update to refresh the package list and check for new updates.

Installable Packages

This lists some package sets that be installed to add common features to the SBC.

Before trying to install these package sets, enable "Include full Debian Package Repository", and "Refresh available packages" - otherwise the needed packages won't be available and the install will fail.

Java Support (Button)

This installs 'libphidget21-java' and 'default-jre-headless' to support Java based projects.

C/C++ Development Tools/Headers

This installs 'build-essential' which allows for on-board development in C/C++.

We also have a detailed walkthrough of installing and using C, Java, and Python on the SBC.

Settings

Include full Debian Package Repository

This enables the full Debian repository in the apt sources list.

By default, only the Phidgets repository is included.

When enabled, all system packages will show up in the updates list, rather than just Phidgets packages, allowing for full system updates.

This also allows the user to install any packages they like, from the SSH interface.

Reboot

The board can be rebooted remotely from this page. The reboot should take 45-60 seconds depending on network conditions. It tries to shut down all running programs before restarting as opposed to the reset button.

About

The license information and credits for the configuration interface is displayed here. A link is provided to the original sources and the Phidgets web site.

Advanced User's Guide

This section describes use of the PhidgetSBC outside of the web configuration, and basic opening of Phidgets over the network. This includes custom applications, using the included gcc compiler, using SSH, customizing the system, and building your own filesystem and/or kernel from sources. It is recommended that you have some experience with Linux before trying some of these tasks.

Custom Applications

The PhidgetSBC supports custom user application written in either Java or C. Custom applications are set up and managed using the configuration interface. Alternatively applications can also be created using the command line tool 'createapp' - the tool will guide you through the set up process.

Custom applications are created under the /mnt/userspace/userapps/(application name) directory and contain all application files. This must include at least the executable file.

Java applications must be compiled on a separate development machine, where they can also be tested before deployment. When coding, make sure to include the correct version of phidget21.jar file as part of the environment and that the SBC and your development machine are running the same version of Java. You can use the link under Userspace: Applications page in the configuration interface to ensure that your Java program is synchronized with the version of phidget21 on the PhidgetSBC. When using Java packages, make sure to create the appropriate directory for them.

The PhidgetSBC also supports .jar files and you may find it easier to compile and upload a .jar instead of the all the necessary .class files. When your project is completed we recommend to compile the project as a .jar. This reduces the number of extra files created into a single package that is easier to manage and can be executed from the command line, or even by double clicking the file if your operating system environment is configured properly.

Under the command line, you can use the jar utility from the Java SDK to package the .class and .java files. The process starts by going to the directory where your program is located and creating a manifest file. This manifest file tells the java jar compiler the version of the program and the name of the Main-Class which acts as the entry point for the application. The entry point is the class that contains the main method that is run when the program is started. You also want to add a line to specify the class-path which will point to the phidget21.jar file that contains the library for including into the jar.

Let's assume we want to distribute our program MyProgram.java as an executable jar file. First, compile the program to generate the .class files. Now, create the manifest file MyProgram.mf which contains the following lines:

```
Manifest-Version: 1.0
Class-Path: phidget21.jar
Main-Class: MyProgram
```

Save the file and close it. Create the jar by running the following command from the command line:

```
jar cmf MyProgram.mf MyProgram.jar MyProgram.class MyProgram.java SupportClass.class
SupportClass.java.
```

You should now have an executable jar file called MyProgram.jar that you can distribute easily as one package and run from anywhere very easily. To run the executable jar file you can either type the following into the command line:

```
java -jar MyProgram.jar
```

Or, you can double click the file in a visual operating system if your environment is configured properly.

Note: Some IDEs, such as Eclipse and Netbeans, automatically create jar files when you build your project. Simply look in your build output folders for your .jar file. Java applications can also be directly executed through SSH using jamvm.

i.e.: `jamvm -jar MyProgram.ja`

C programs can be compiled on the PhidgetSBC itself, via the SSH interface, or off-board using a cross compiler. Use of a cross compiler is not strictly documented here, but it is possible to build one from the buildroot distribution available on our web site. When developing C applications on the PhidgetSBC, it is recommended that you log in using the 'user' account instead of 'root'. If you need to log data from a custom application, you can either log directly to the application directory with the size limits of the userspace in mind, or to /tmp if the data should be erased on reboot. Alternatively, you can use a flash drive, which are mounted automatically at /media/usb(0-9) when plugged in. Note that custom applications should not try to get user input, as stdin is closed before the application gets run.

GCC

The PhidgetSBC contains full GCC and associated build tools, as well as make and gdb for compiling C source. Use of these is the same as on full linux, just keep in mind that there is no swap space and userspace is limited. Compiling will also be slow for complicated programs. Simple programs are ideal for this environment. The C library used is uClibc. For most uses this should be similar to full libc, just much smaller. Also, when compiling a program that links with libphidget21.so, you need to add '-lphidget21 -ldl -pthread -lm' to the command line. Otherwise, you will get segfaults.

Phidget Dictionary

Communication between a custom application on the PhidgetSBC and the outside world can be facilitated by using the dictionary interface of the Phidget Webservice. The dictionary lets you set and listen for key/value pairs over the network, and take action accordingly. This could be used to post data or listen for commands over the network, while maintaining reliability and ultimate control on the PhidgetSBC itself in case of network failure. See the Phidget Programming manual for its use in your language of choice.

SSH

The built-in SSH Server can be enabled to allow console access to the PhidgetSBC. By default, this server is disabled. SSH access to the PhidgetSBC is enabled in the Network: Network configuration page on the PhidgetSBC. Projects on the PhidgetSBC should be stored in the user home directory (/home/user). Enabling the server for the first time can take several minutes as the encryption keys are generated. Once SSH is enabled, connect to the PhidgetSBC using its hostname or IP address (e.g. 'ssh user@phidgetsbc.local'). You should login using the restricted 'user' account using the initial password set for the 'root' account. The password for the user account can be changed any time either through the unrestricted root account or directly through the user account. Files can be sent to the board using scp or by uploading through the web interface in Userspace: Userspace Browser. The SSH server does not support sftp. Text files (source code, etc.) can be edited using vi or with the web interface. On Windows, we recommend Putty for an SSH client. You can get this at <http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html>.

Customization

If you wish to customize the root filesystem, you have two options: customize locally on the board, or build a custom kernel and filesystem from source. If you mess up the firmware while customizing, you can always perform a board reset and start again. You may also want to add new libraries or kernel extensions, add new Unix tools, update certain aspects, delete others, change the boot process, etc. All of this is possible, but it is also completely unsupported by Phidgets. In order to customize the root file system of a running PhidgetSBC, you will need to remount the filesystem as readwrite with the command: 'mount -o remount,rw /'. At this point you can change any files in /, just keep in mind the amount of free space. Once the SBC is rebooted, / will again be mounted as

read-only.

Custom Kernels and Filesystem

You can build the complete filesystem and kernel from the same codebase as Phidgets Inc. uses. The full Buildroot system can be downloaded from our website. Phidgets Inc. does not provide support for custom filesystems or kernels, nor do we provide advanced tutorials on the buildroot system. However, this basic information should be enough to get started. Buildroot works by building a full cross-compiler for the PhidgetSBC and then using that to build a full set of tools to create a root filesystem from scratch. It also handles building the kernel. You can use the cross compiler that it produces to build your own applications for the PhidgetSBC independently of the system, or integrate your code into the process. Buildroot downloads all of the source code it needs from open-source repositories, so the Buildroot distribution itself is fairly small. Buildroot needs to be run on Linux. Unpack the distribution and run the following commands to set everything up:

- ``touch .config``
- ``make BOARD=phidget_sbc getconfig``

You can build your filesystem by typing 'make' and change configuration options using 'make menuconfig'. Output binaries are located in `binaries/phidget_sbc` - the `.bin` files are accepted by the PhidgetSBC upgrade system. Building the full filesystem relies on some packages which you will need to install into your Linux distribution, including but not limited to: `libacl1-dev`, `zlib1g-dev`, `kaffe`, `liblzo2-dev`. If you make changes to config files, source, etc., run 'rm_root' before running make, otherwise the filesystem may not be updated properly. Kernel patches are stored in: `target/device/PhidgetSBC/`. More information about Buildroot can be found here: <http://buildroot.uclibc.org/docs.html>

Technical Details

Rebooting/Resetting the PhidgetSBC

To reboot the device, quickly press the black reset button found between the USB connectors and the power terminals. Both Ethernet Port LEDs (yellow-connectivity, green-activity), and the green status LED will turn off. The reboot is done when all LEDs come back on in about 25 seconds.

To reset the firmware, press and hold the button for 10 seconds until the green status LED begins to blink, then release. Both Ethernet Port LEDs will turn off (yellow-connectivity, green-activity) for 80 seconds; the green status LED will then turn off; then all LEDs will come back on in 20 seconds. All data will be lost and the operating system will be reset to a factory state.

To boot into the Recovery / Upgrade system, hold the button for 20 seconds until the green status LED switches from blinking slowly to blinking quickly, then release. The recovery system allows for factory reset, full system upgrades and recovery of the main system.

Recovery / Upgrade System

The recovery / upgrade system is a small system from which the main system can be reset/upgraded, or recovered.

Entering the Recovery System

The recovery system can be entered in two ways.

1. From the 'System: Backup & Restore' web interface page.
2. By holding down the reset button for 20+ seconds - until the green light has switched from flashing slowly to flashing quickly.

Upgrades

Generally, you should not need to do a full system upgrade. Upgrades are meant to be performed via the package system, and Phidgets maintains it's own package repository from which to push out updates.

Occasionally, you may wish to go back to a clean install and upgrade to the latest rootfs/kernel from Phidgets. Phidgets will not be creating these images with every release of phidget21 as we did with PhidgetSBC1, rather they will be released several times a year, as needed for major changes not easy to push out via packages.

You can also flash your own custom kernel or root filesystem image.

Factory Reset

This restores the kernel and root filesystem from backup, overwriting any changes that may have been made. This is equivalent to holding down the reset button for 10 seconds.

Recovery

If the main filesystem has been damaged/misconfigured in such a way that it won't boot, you may be able to fix the issue / recovery important files before running a reset. The recovery system runs an SSH server that you can loginto for console access. Username/password is: root/root.

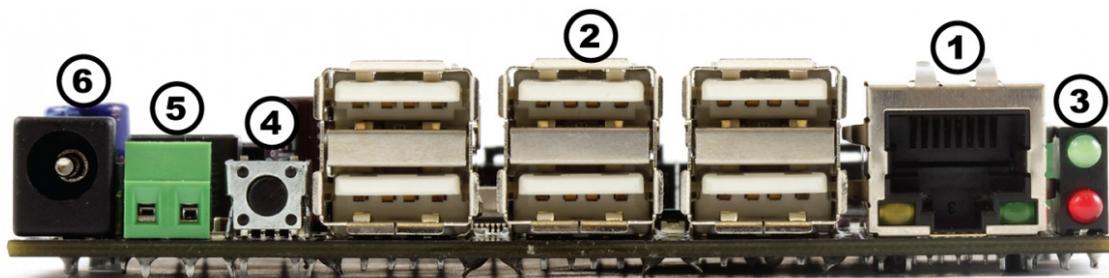
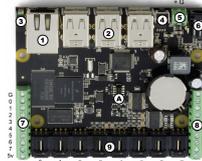
You can mount the main root filesystem with the following commands (assuming it's not damaged):

1. ubiattach /dev/ubi_ctrl -m 6
2. mount -t ubifs /dev/ubi0_0 /mnt

Ports and Connectors

Numbered in the circles on the diagram:

1. 10/100baseT Ethernet
 2. Six USB High-Speed Ports
 3. Indicator LEDs
 4. Reboot / Reset Button
 5. Power input terminal
 6. Power input jack
 7. Eight Interface Kit Digital Inputs (Indexed 0 to 7)
 8. Eight Interface Kit Digital Outputs (Indexed 0 to 7)
 9. Eight Interface Kit Analog Inputs (Indexed 0 to 7)
- A. JTAG connector, for internal testing purposes only



G +

1. This Ethernet port is used for network connectivity to the PhidgetSBC. This enables access to the PhidgetSBC as well as any connected Phidgets through the webservice. Alternatively, the USB Wireless adapter can be used for network connectivity.
2. These USB ports can be used for connecting Phidgets, Wi-Fi adapters, flash drives, webcams, USB hubs, etc.

3. These LEDs indicate the status of the PhidgetSBC. The Red LED indicates that the power supply is on and running properly. The green LED indicates boot status. The green LED will turn on and off once during boot and then turn back on when everything is running.

4. This will reboot the board if pressed once. Note that this is a forced reboot. Any user programs that were running may leave their data in a inconsistent state, but this is safe for the base system. A soft reboot can be performed remotely from the configuration interface. If held for more then 10 seconds, the red LED will start to blink and enter emergency Reset mode. Once the button is released, the onboard memory will revert to a factory-fresh state. This includes overwriting the kernel and root file system, and erasing all configuration, user data, and applications. If held for more then 20 seconds, the Recovery/Upgrade system will be booted, from which a Factory Reset/Full filesystem upgrade can be performed.

5,6. The PhidgetSBC can be powered from either the terminals or the barrel connector.

7,8,9. The Interface Kit I/O is explained in the Interface Kit section of the manual.

Power Distribution

The 12V power supply is stepped down to 5V and distributed in the following way:

- Each USB port has 500 mA available
- All analog inputs share a total of 500 mA
- The digital outputs, +5V terminals, USB controller, and pull-ups all share a total of 500 mA

Power Over Ethernet

Power over Ethernet can be used to provide both a network connection and power to a device when a power outlet is not available. This means that with the proper adapters, you can run the PhidgetSBC entirely off an Ethernet source. The PhidgetSBC does not draw power directly from a powered Ethernet line, but instead can use a setup where the power is split to a separate line again near the PhidgetSBC. The board has been tested and will work with Power Sourcing Equipment that can output 6-12VDC.

Hardware Layout

The PhidgetSBC is based around the i.MX28 processor. This is an ARM926EJ-S based microprocessor from Freescale, which runs at 454MHz. Connected to this is 128 MiB of DDR2 SDRAM, 1 GiB of very large page NAND and a 10/100baseT Ethernet controller. The microprocessors USB Host port is connected to a 7-port USB 2.0 High Speed Hub chip. The integrated PhidgetInterfaceKit 8/8/8 is connected to one of the hub ports, with the other 6 ports being brought out to the user.

Software Layout

The PhidgetSBC runs Debian/GNU Linux 7.0 as its operating system and gets booted with U-Boot. The kernel is 3.6.3 and generally kept up to date with the latest releases. The root filesystem is created using debootstrap and is mounted in a ~924MiB nand partition using the UBIFS filesystem, in Read-Write mode.

Configuration data is located at '/etc/webif'. This is where all configuration that can be set through the website is located.

User applications are stored in '/usr/userapps', each is their own directory.

The kernel is stored on bare Nand in it own 4MiB partition, in the uImage format.

Date and Time

The date and time are set using ntp (network time protocol) at boot. The ntp daemon continues to run in the background and will periodically update the clock, keeping it very close to real time.

There is a real-time clock with battery backup which will preserve date/time across reboots, power removal. The real-time clock is synced to system time during reboot/shutdown. If power is unplugged suddenly, the real-time clock may not have the correct time.

Wireless Networking System

Wireless networking is supported using the available adapter and is configured through the configuration interface.

Configuration System

The configuration system used by the website is stored in '/etc/webif'. These files should generally not be changed manually, but there is no reason why they could not be. It's very easy to enter invalid data that could cause the system to behave unexpectedly or not boot.

Nand Layout

The board contains 1GiB of Nand. This nand is split into 8 partitions as follows:

0: bootloader	size: 4MiB	Read Only
1: environment	size: 512KiB	Read Only
2: device_tree	size: 512KiB	Read Only
3: recovery_system	size: 4MiB	Read Only
4: recovery_kernel	size: 4MiB	Read Only
5: recovery_fs	size: 83MiB	Read Only
6: kernel	size: 4MiB	Writable
7: rootfs	size: 924MiB	Writable

U-Boot and recovery kernel and filesystem cannot be written from Linux - this is a safety measure.

Boot Process

This describes the boot process from power on.

1. CPU loads the bootstream from Nand
2. Bootstream initializes memory, power and GPIO, then runs u-boot from Nand
3. u-boot turns the green LED on and then checks to see if the reset button is being held down
4. if the reset button is held, u-boot either restores the kernel/filesystem, or boot into the recovery system, depending on the hold time
5. u-boot then loads the device tree and Linux kernel into Ram and then runs the kernel
6. Linux reads in the device tree from ram, and turns off the green LED
7. Linux then boots, bringing up usb, ethernet, etc.
8. init gets run as the parents of all processes, and uses the /etc/inittab script to bring up the system. This includes mounting other filesystems, settings the hostname, running the scripts in /etc/init.d, and running user applications
9. inittab then turns the green LED on. The system has finished booting

Drivers for USB to Serial adapters

The SBC kernel contains driver support for the following USB to Serial Adapters. Please consult the kernel documentation for details into the driver support for the USB to Serial adapters.

Company	Product
ConnectTech	WhiteHEAT
Keyspan	USA-18X, USA-28, USA-28X, USA-28XA, USA-28XB, USA-19, USA-19W, USA-19QW, USA-19QI, USA-49W, USA-49WLC
FTDI	Single Port Serial Adapter
Cypress	M8 CY4601 Family
Digi International	AccelePort USB Serial
Belkin	USB Serial Adapter F5U103
MCT	USB Single Port Serial Adapter U232
Inside Out Networks	Edgeport Serial Adapter
Prolific	PL2303

Drivers for USB Wifi Adapters

The SBC kernel contains driver support for the following USB Wireless Adapters. Please consult the kernel documentation for more details.

Company	Module	Chipsets
Atheros	ath9k_htc	AR9271, AR7010
Atheros	carl9170	AR9170
Intersil	r8712u	ISL3877, ISL3880, ISL3886, ISL3887, ISL3890
Realtek	r8712u	RTL8188SU, RTL8191SU, RTL8192SU
Ralink	rt2500usb	RT2500USB/RT2571
Ralink	rt2800usb	RT2070, RT2770, RT2870, RT3070, RT3071, RT3072, RT3370, RT3572, RT5370, RT5372
Ralink	rt73usb	RT2501USB/RT2571W
Realtek	rtl8187	RTL8187L, RTL8187B
Realtek	rtl8192cu	RTL8188CUS, RTL8192CU
ZyDAS / Atheros	zd1211rw	ZD1211/ZD1211B, AR5007UG

U-Boot

U-Boot is used for setting up the processor and booting Linux, and is only accessible by the serial port. Normal users will not need to use it. If you are connected to the serial port, you will see the U-Boot prompt shortly after power up. You can view the environment variables for information on how to properly boot Linux on the PhidgetSBC.

Be very careful when modifying the u-boot partition. If it is damaged or overwritten, it is difficult to fix.

Refer to U-Boot documentation here: www.denix.de/wiki/DULG/Manual ^[9] for more information on using U-Boot.

Ad-hoc Networks

The SBC can be configured as a device in an ad-hoc network.

For more information, visit the Ad-Hoc Networks page.

Further Reading

Check the Phidget SBC page for more details about using the Phidget SBC.

Check the 1018 User Guide for more information about the InterfaceKit on this board.

API

The Phidget SBC has the same API functions and events as the PhidgetInterfaceKit.

Check the 1018 User Guide for API details.

Product History

Date	Board Revision	Device Version	Comment
January 2013	0	300	Product Release

References

- [1] http://www.phidgets.com/products.php?product_id=1073
- [2] <http://www.phidgets.com/downloads/libraries/Phidget.dmg>
- [3] http://www.phidgets.com/Drivers_Info.html#Mac
- [4] http://www.phidgets.com/products.php?product_id=3702
- [5] http://www.phidgets.com/products.php?product_id=3072
- [6] http://www.phidgets.com/products.php?product_id=1018
- [7] http://www.phidgets.com/products.php?product_id=3402
- [8] <http://linux-uvc.berlios.de/#devices>
- [9] <http://www.denx.de/wiki/DULG/Manual>

Article Sources and Contributors

1073 User Guide *Source:* http://www.phidgets.com/wiki/index.php?title=1073_User_Guide *Contributors:* Kat, Mparadis

Image Sources, Licenses and Contributors

Image:1073.jpg *Source:* <http://www.phidgets.com/wiki/index.php?title=File:1073.jpg> *License:* unknown *Contributors:* Mparadis

File:1072_0_Package_Contents.jpg *Source:* http://www.phidgets.com/wiki/index.php?title=File:1072_0_Package_Contents.jpg *License:* unknown *Contributors:* Mparadis

File:1072_0_Connecting_The_Hardware.jpg *Source:* http://www.phidgets.com/wiki/index.php?title=File:1072_0_Connecting_The_Hardware.jpg *License:* unknown *Contributors:* Mparadis

File:Ph.jpg *Source:* <http://www.phidgets.com/wiki/index.php?title=File:Ph.jpg> *License:* unknown *Contributors:* Mparadis

File:1073_0_Control_Panel_Screen.jpg *Source:* http://www.phidgets.com/wiki/index.php?title=File:1073_0_Control_Panel_Screen.jpg *License:* unknown *Contributors:* Mparadis

File:sbc_gs_set_passwd.png *Source:* http://www.phidgets.com/wiki/index.php?title=File:Sbc_gs_set_passwd.png *License:* unknown *Contributors:* Cora

File:sbc_gs_login.png *Source:* http://www.phidgets.com/wiki/index.php?title=File:Sbc_gs_login.png *License:* unknown *Contributors:* Cora

File:sbc_gs_web_header.png *Source:* http://www.phidgets.com/wiki/index.php?title=File:Sbc_gs_web_header.png *License:* unknown *Contributors:* Cora

File:sbc_gs_web_interface.png *Source:* http://www.phidgets.com/wiki/index.php?title=File:Sbc_gs_web_interface.png *License:* unknown *Contributors:* Cora

File:sbc_gs_wireless.png *Source:* http://www.phidgets.com/wiki/index.php?title=File:Sbc_gs_wireless.png *License:* unknown *Contributors:* Cora

File:sbc_gs_static_ip.png *Source:* http://www.phidgets.com/wiki/index.php?title=File:Sbc_gs_static_ip.png *License:* unknown *Contributors:* Cora

File:sbc_attached_phidgets.png *Source:* http://www.phidgets.com/wiki/index.php?title=File:Sbc_attached_phidgets.png *License:* unknown *Contributors:* Cora

File:sbc_gs_webcam.png *Source:* http://www.phidgets.com/wiki/index.php?title=File:Sbc_gs_webcam.png *License:* unknown *Contributors:* Cora

File:1073_0_Layout_Top.jpg *Source:* http://www.phidgets.com/wiki/index.php?title=File:1073_0_Layout_Top.jpg *License:* unknown *Contributors:* Mparadis

File:1072_0_Layout_Front.jpg *Source:* http://www.phidgets.com/wiki/index.php?title=File:1072_0_Layout_Front.jpg *License:* unknown *Contributors:* Mparadis