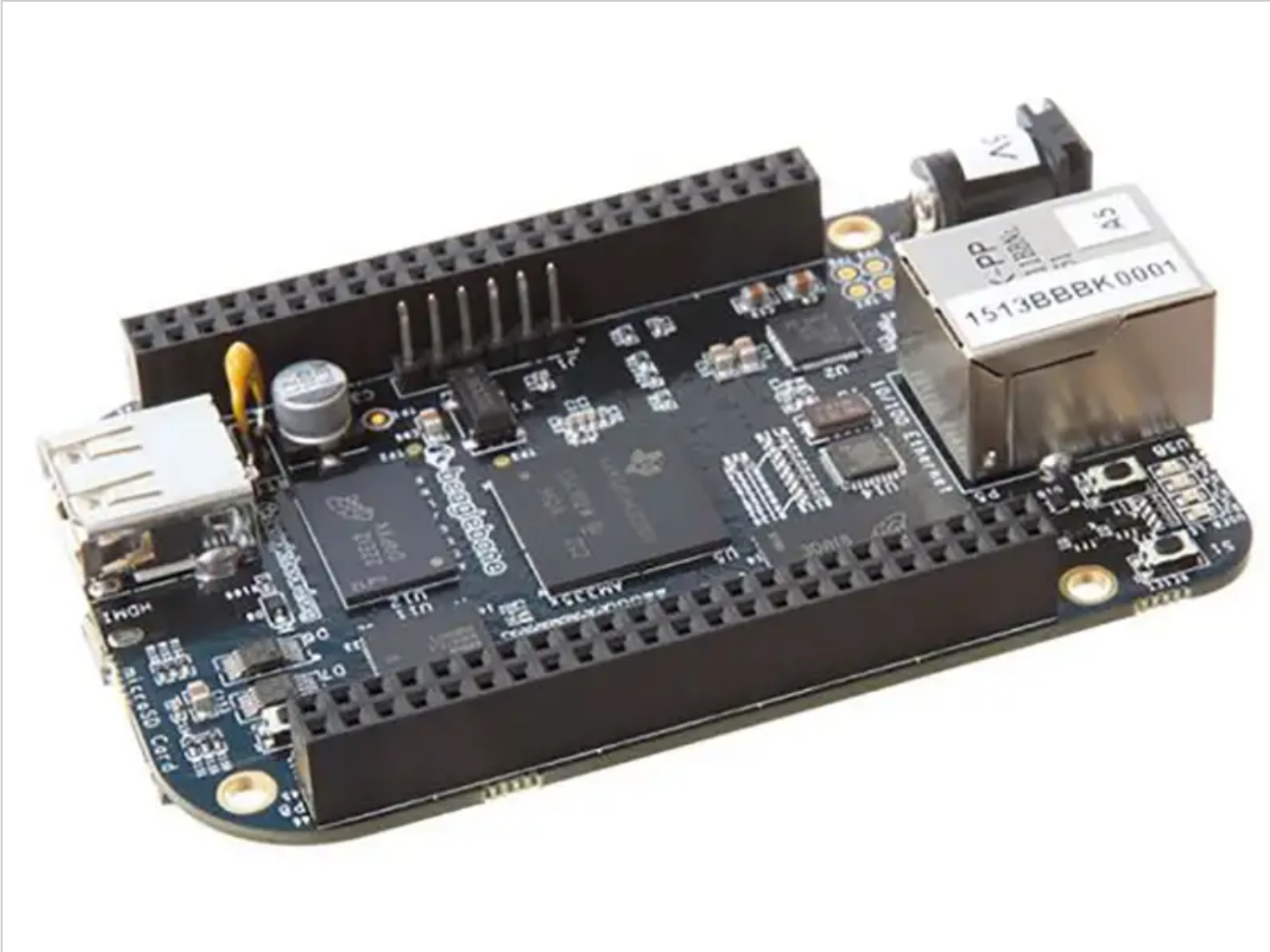
Using the Digital I/O on a BeagleBone Black

BY ALL ABOUT CIRCUITS (/EN/MAKER/PROFILES/E491DB21E4CB468190034A2F2F9901A0)



Courtesy of All About Circuits (https://www.allaboutcircuits.com/projects/how-to-use-the-digital-i-o-on-a-beaglebone/)

A great way to get started with the powerful BeagleBone Black is to learn how to use the general purpose I/O. We'll go over the steps in this project.

Export Only

A quick and easy way to manipulate the Beaglebone's I/O is through the bash shell. On Linux, everything is a file, and the bash shell gives you an easy way to interface with files in a... aglebone Black runs Linux, so it's not an exception to this rule, especially since that even the GPIO on Beaglebones are files. The initial step when using the GPIO is to *export* your chosen pin as a GPIO. This step is pretty simple, and involves a file write using the 'echo' command:

```
$ echo 67 > /sys/class/gpio/export
```

Copy Code

Although it looks like this command was pointless because nothing happens to your board, keep going! The Beaglebone is built around TI's Sitara processor. Like many other modern processors, the Sitara family has a slew of pins to interface with the outside world, and most of these pins can conveniently perform multiple functions. These functions range from extremely simple, such as the GPIO function we'll be setting up in this project, to extremely complex, such as a part of a PCIe bus or an SGMII network.

Note that you cannot perform all functions of a pin at once. These pins are *multiplexed* into a port on the processor's internal bus, meaning you must select which function you want a chosen pin to perform. The `echo` command writes the number 67 into the file '/sys/class/gpio/export'. This tells the system we want to use pin 67 on the board as the GPIO, and that the processor should propagate that setting throughout the system. Notice that once you've run this command, the directory /sys/class/gpio/' contains an extra folder marked as:

```
$ ls /sys/class/gpio

export gpio67 gpiochip0 gpiochip32 gpiochip64 gpiochip96 unexport
```

Copy Code

Shifting Directions

By 'echo'-ing 67 into that file, you instructed the system to export the settings for GPIO_67. The system responded by building the folder 'gpio67'. When examining the contents of this, you should see the following structure:

```
:14 .
(/en/maker)
drwxr-xr-x 7 root root 0 Jan 1 00:00 ..
-rw-r--r-- 1 root root 4096 Jan 1 00:42 active_low
-rw-r--r-- 1 root root 4096 Jan 1 00:14 direction
-rw-r--r-- 1 root root 4096 Jan 1 00:43 edge
drwxr-xr-x 2 root root 0 Jan 1 00:42 power
lrwxrwxrwx 1 root root 0 Jan 1 00:41 subsystem -> ../../../../class/gpio
-rw-r--r-- 1 root root 4096 Jan 1 00:14 uevent
-rw-r--r-- 1 root root 4096 Jan 1 00:20 value
```

There are two files in the new folder, 'gpio67', that require attention: the first is the `direction` file. When you run the command `$ cat /sys/class/gpio/gpio67/direction`, this output should be displayed:

```
$ cat /sys/class/gpio/gpio67/direction
```
Copy Code

```
in
```

If you've dealt with any bare metal embedded processor (i.e. PIC, AVR, HCS08), the register called the data direction register should ring a bell, giving you permission to skip the next paragraph.

For those of you who would like a refresher, the *data direction register* dictates which way (either in or out) data can flow out of a GPIO port. Setting the data direction register up for a certain GPIO pin generally involves three steps:

- Finding the right register
- Finding the right bit within that register
- Writing an 'and' statement in C to set or clear that bit

This is not the case for the Beaglebone! When you exported GPIO_67, the Beaglebone created a handy little file to read the processor's data direction register, giving it back to us in an easily read format. Rather than a complex mess of hexadecimal, we get two simple values: 'in' or 'out'. You may have correctly guessed from the earlier 'cat' command that the default state of this register is 'in' - meaning that it has the ability to read in data on

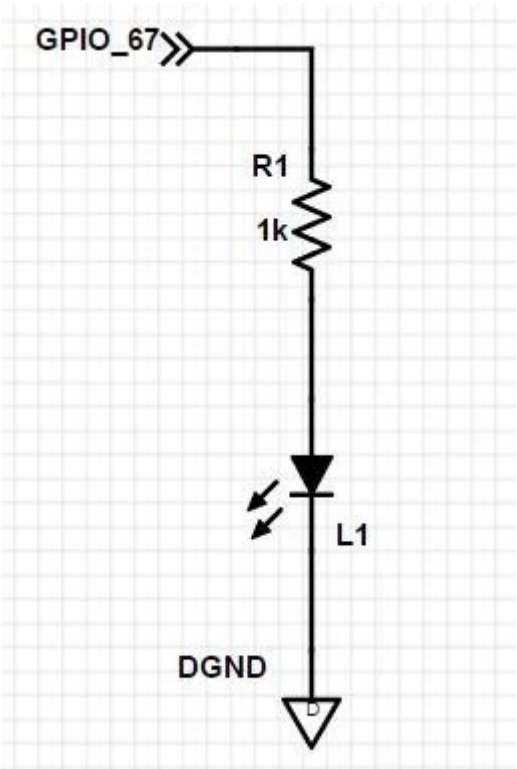
that pin into the processor, but it is unable to affect the state of GPIO_67. Knowing that, let's now change that so we can see that pin's output in the real world! This can be done b ho' command, and using 'cat' to verify it worked:

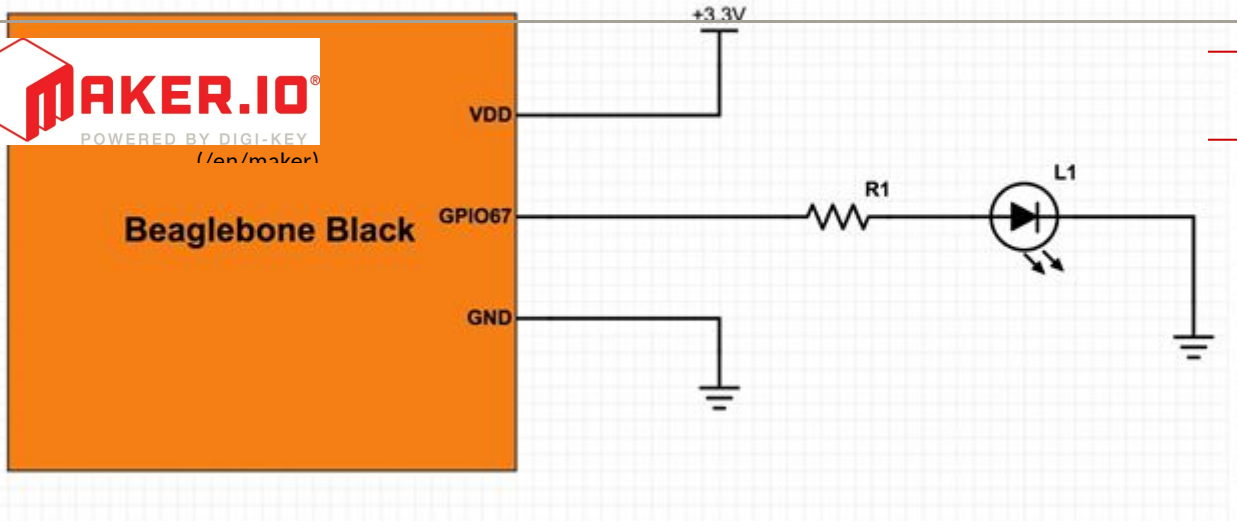

```
$ echo out > /sys/class/gpio/gpio67/direction

$ cat /sys/class/gpio/gpio67/direction

out
```


Just like that, we've changed this I/O's data direction from an input to an output. Now we can move on and really make it do something!

There Is a Light that Sometimes Goes Out

For this next step, you will need to build a simple circuit using a single 1 kOhm resistor and an LED (pick your favorite color!). One pin of the LED needs to connect to pin 2 of header P8 on the Beaglebone, and the other pin into any row of a solderless breadboard. Then, connect one of the resistor's pins to the same breadboard row that the LED is plugged into, and the other pin into GPIO_67 (found on pin 8 of header P).





If everything's tied together properly, when you run the following commands, the LED turns on with one, and off with the other.

```
$ echo 1 > /sys/class/gpio/gpio67/value
$ echo 0 > /sys/class/gpio/gpio67/value
```

Copy Code

This works on the same principle as the last set of writes to `/sys/class/gpio/gpio67/value`. The difference between the commands is the value that you write into each file.

A Final Step

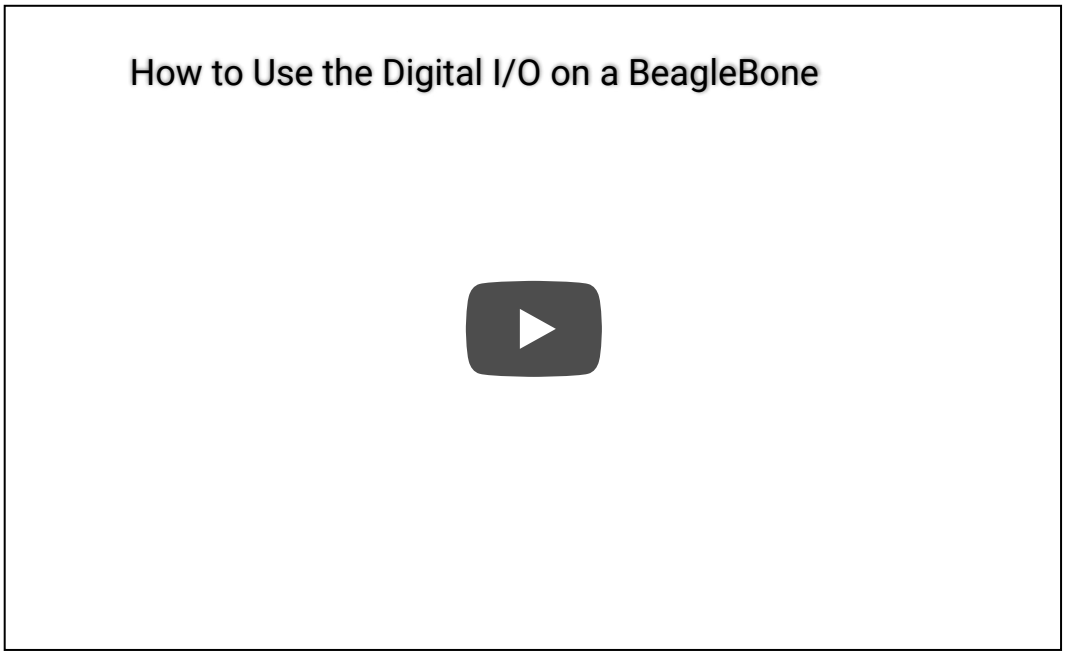
All of these commands can chain together into a simple script to toggle the LED on and off every half second. For example:

```
#!/bin/bash

if  ! -e /sys/class/gpio/gpio67/value
then
  echo 67 > /sys/class/gpio/export
  echo out > /sys/class/gpio/gpio67/direction
fi

while  True
do
  echo 1 > /sys/class/gpio/gpio67/value
  usleep 500000
  echo 0 > /sys/class/gpio/gpio67/value
  usleep 500000
done
```

Copy Code

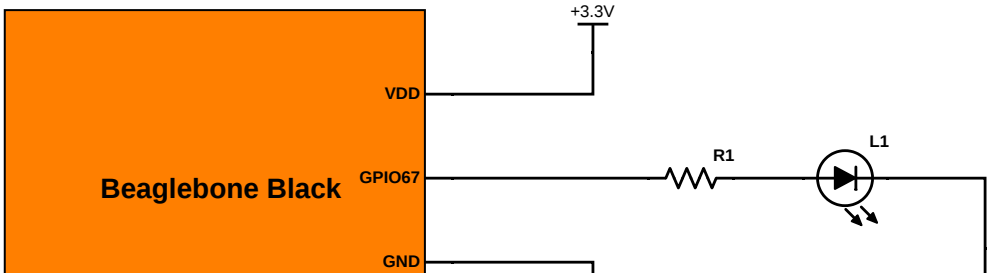
Note that this script will run until you cancel it. In order to cancel the script, press `Ctrl+c` to get back into your Linux terminal. You can copy it from the code excerpt above into your B ull it from git using this repository (https://g ushychicken/bittoggle).

And there you have it! An easy way to toggle GPIOs on a Beaglebone Black (http://www.teachmemicro.com/beaglebone-black-blink-led-using-c/).



beaglebone-digital-io
Share
Notes
Open Project

-
-



Key Parts and Components

MAKER.IO
POWERED BY DIGI-KEY

(/product-detail/en/ghi-electronics-llc/BBB01-SC-505/BBB01-SC-505-ND/6210999?WT.mc_id=frommaker.io)

GHI ELECTRONICS, LLC

BEAGLEBONE BLK REV C AM3358BZCZ (/product-detail/en/ghi-electronics-llc/BBB01-SC-505/BBB01-SC-505-ND/6210999?WT.mc_id=frommaker.io)

BBB01-SC-505

$62.38 (/product-detail/en/ghi-electronics-llc/BBB01-SC-505/BBB01-SC-505-ND/6210999?WT.mc_id=frommaker.io)

(/product-detail/en/broadcom-limited/HLMP-4700-C0002/516-2483-1-ND/2744909?WT.mc_id=frommaker.io)

BROADCOM LIMITED

LED RED DIFFUSED T-1 3/4 T/H (/product-detail/en/broadcom-limited/HLMP-4700-C0002/516-2483-1-ND/2744909?WT.mc_id=frommaker.io)

HLMP-4700-C0002

$0.58 (/product-detail/en/broadcom-limited/HLMP-4700-C0002/516-2483-1-ND/2744909?WT.mc_id=frommaker.io)

Add all Digi-Key Parts to Cart (/classic/ordering/fastadd.aspx?newCart=true&part0=BBB01-SC-505-ND&qty0=1&part1=516-2483-1-ND&qty1=1)

- BBB01-SC-505-ND
- 516-2483-1-ND

RECENT COMMENTS(0)

Please sign up (https://www.digikey.com/MyDigiKey/Register?ReturnUrl=https%3A%2F%2Fwww.digikey.com%2Fen%2Fmaker%2Fprojects%2Fusing-the-digital-io-on-a-beaglebone-black%2F97242e7f8e5f4480b97ea35ed78c882a) or log in (https://www.digikey.com/MyDigiKey/Login?ReturnUrl=https%3A%2F%2Fwww.digikey.com%2Fen%2Fmaker%2Fprojects%2Fusing-the-digital-io-on-a-beaglebone-black%2F97242e7f8e5f4480b97ea35ed78c882a) to comment.

Project Details

Platforms

Login (https://www.digikey.com/MyDigiKey/Login?ReturnUrl=http%3a%2f%2fwww.digikey.com%2fen%2fmaker%2fprojects%

latforms/b/beaglebone)

MAKER.IO® POWERED BY DIGI-KEY

MENU

Tags (/en/maker)

BeagleBone (/en/maker/search-results?t=51386b17348d4b74bf481d7f14a45b82&y=13825c8674444e22884d8d26197819d1)
Breadboard (/en/maker/search-results?t=1b015617e5d24e49b361485333538d57&y=13825c8674444e22884d8d26197819d1)
LEDs (/en/maker/search-results?t=50b78b6cc0bf4e8db7cd0d273734c07c&y=13825c8674444e22884d8d26197819d1)
Linux (/en/maker/search-results?t=31738e4f737d4f9494de22b52ab97273&y=13825c8674444e22884d8d26197819d1)
mbed (/en/maker/search-results?t=dafb2cbe0b96488aaa3cddd459214dcc&y=13825c8674444e22884d8d26197819d1)
Networking (/en/maker/search-results?t=483d5b3cad214a55a5fd3adc85bb8147&y=13825c8674444e22884d8d26197819d1)
Processing (/en/maker/search-results?t=1fb41260395841248cf26ec6d88217ec&y=13825c8674444e22884d8d26197819d1)
Resistors (/en/maker/search-results?t=e0e544a86bbb4b6cbfda03d099ab7ffb&y=13825c8674444e22884d8d26197819d1)

License

Get Involved