Light Sensor & Port Forwarding Guide

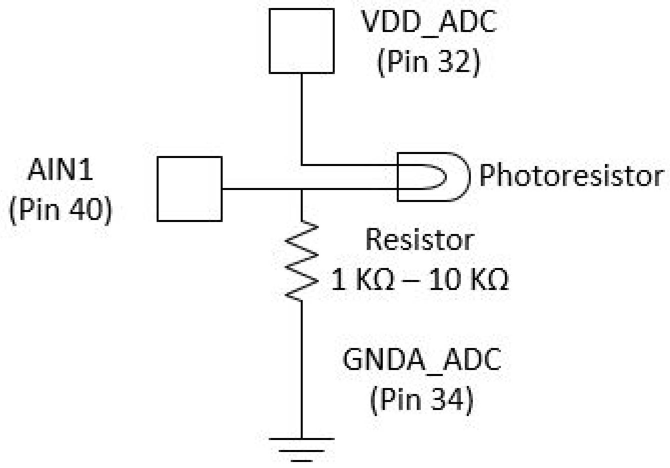
This document describes how to read the value of a GL5539 Photoresistor (Light sensor) and a method of port forwarding to route a connection through an external laptop to the BBB.

Light Sensor

Photoresistors are cheap, durable and simple sensors for measuring the luminance or intensity of light hitting it. It is simply a resistor that changes resistance based on the amount of Light, in Lux, it is exposed to. Resistances commonly range from 10 Ω in direct sunlight to 10,000,000 Ω in complete darkness.

Wiring

The sensor uses Analog-to-Digital Conversion (ADC) to read the value, so the wiring must be done with the ADC voltage(32), ADC Ground(34) and one of the ADC(33, 35-40) input pins.



Since the Light Sensor is basically a resistor, it does not matter which pin is connected to voltage and which to ground. The extra resistor is used as a voltage divider to control whether the sensor reads bright or dark more accurately. A smaller resistor gives more accuracy for bright readings while a larger resistor is better for darkness.

Reading the sensor

1. Load the ADC Cape

```
# echo BB-ADC > /sys/devices/bone_capemgr*/slots
```

2. Read the ADC value (Assuming AIN1, pin 40)

```
# cat /sys/bus/iio/devices/iio:device0/in_voltage1_raw
```

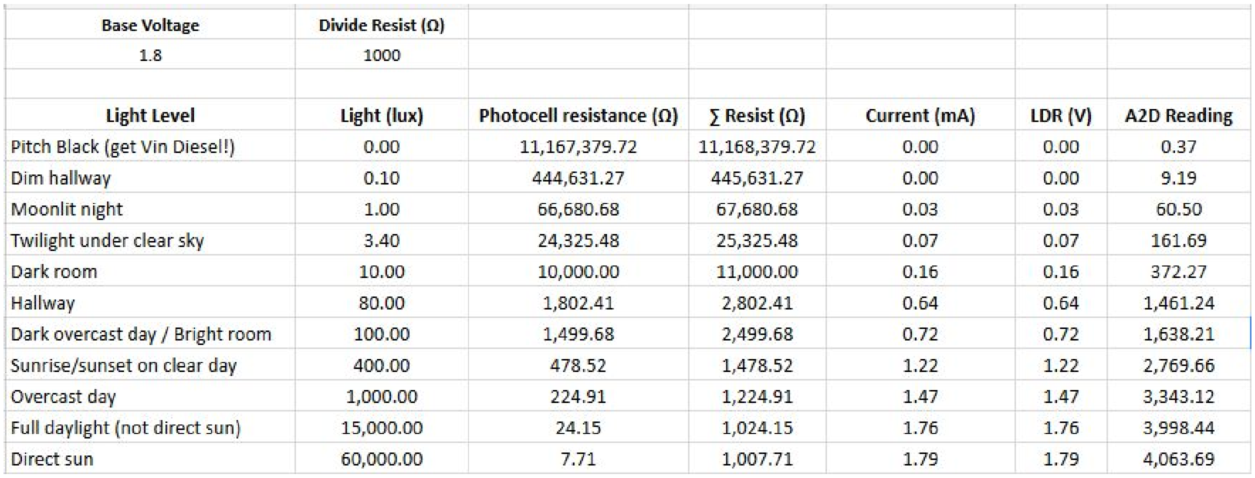
3. Convert to voltage

```
Voltage = (Raw / 4095)
```

4. Convert to photoresistor resistance (Assuming 1000 Ω divider)

```
Resistance = (1000 / Voltage) - 1000
```

At this point you can compare the resistance against the Lux vs Resist graph in the photoresistors spec sheet. It is also possible to write an equation to do the conversion algebraically, but this depends on the specific photoresistor model used (Note that this table is a log-log table and not linear). The following table shows some sample values that were obtained with a GL5539 model. Note that each individual unit can also have differing measurements depending on the quality of the sensor and its calibration.



Base Voltage	Divide Resist (Ω)					
1.8	1000					
Light Level	**Light (lux)**	**Photocell resistance (Ω)**	**∑ Resist (Ω)**	**Current (mA)**	**LDR (V)**	**A2D Reading**
Pitch Black (get Vin Diesel!)	0.00	11,167,379.72	11,168,379.72	0.00	0.00	0.37
Dim hallway	0.10	444,631.27	445,631.27	0.00	0.00	9.19
Moonlit night	1.00	66,680.68	67,680.68	0.03	0.03	60.50
Twilight under clear sky	3.40	24,325.48	25,325.48	0.07	0.07	161.69
Dark room	10.00	10,000.00	11,000.00	0.16	0.16	372.27
Hallway	80.00	1,802.41	2,802.41	0.64	0.64	1,461.24
Dark overcast day / Bright room	100.00	1,499.68	2,499.68	0.72	0.72	1,638.21
Sunrise/sunset on clear day	400.00	478.52	1,478.52	1.22	1.22	2,769.66
Overcast day	1,000.00	224.91	1,224.91	1.47	1.47	3,343.12
Full daylight (not direct sun)	15,000.00	24.15	1,024.15	1.76	1.76	3,998.44
Direct sun	60,000.00	7.71	1,007.71	1.79	1.79	4,063.69

Sample data for GL5539 Photoresistor

Port Forwarding

Rinetd is a small daemon for linux that allows easy configuration and forwarding of TCP traffic on specific ports. An example setup might be a laptop with a wireless connection forwarding traffic on port 8080 to the Beaglebone over an Ethernet cable. The laptop is a passive gateway and does no other work.

1. Install Rinetd

```
    $ sudo apt-get update
    $ sudo apt-get install rinetd
```

2. Open the config file

```
    $ sudo gedit /etc/rinetd.conf
```

3. Forward specific local ip to specific ip

```
    # bindadress    bindport   connectaddress   connectport
    192.168.0.22    8080       192.168.2.2      8080
```

4. Forward all local ips to specific ip

```
    # bindadress    bindport   connectaddress   connectport
    0.0.0.0         8080       192.168.2.2      8080
```

5. Restart the daemon

```
    $ sudo /etc/init.d/rinetd restart
```

Although rinetd is very simple and easy to setup, it does not handle multiple external connections very well. Additionally, there is a possible bug that can occur where Rinetd will use 100% CPU until killed and restarted, and/or changed to a different port.

Possible alternatives (untested):

- Socat: High RAM usage with multiple connections.
- Redir: Medium RAM usage with multiple connections.
- Haproxy: Load balancer, but also has efficient TCP forwarding.
- Iptables: Efficient and powerful, but difficult to configure.