ATTENTION ELC Presenters: Please upload your slide deck for your presentation to the following page: ELC_2017_Presentations

Minnowboard:Physical Computing

From eLinux.org
Revision as of 18:26, 11 August 2013 by Jayneil (Talk | contribs)

(diff) ← Older revision | Latest revision (diff) | Newer revision → (diff)
Jump to: navigation, search

Back to the MinnowBoard home page

Contents

Summary

In this guide, I will describe how to do Physical Computing with the MinnowBoard step by step. This guide is for new users who are just getting started with the MinnowBoard.

Getting Started

For information on 'Setting up a microSD card', 'Booting Angstrom' etc, please refer here

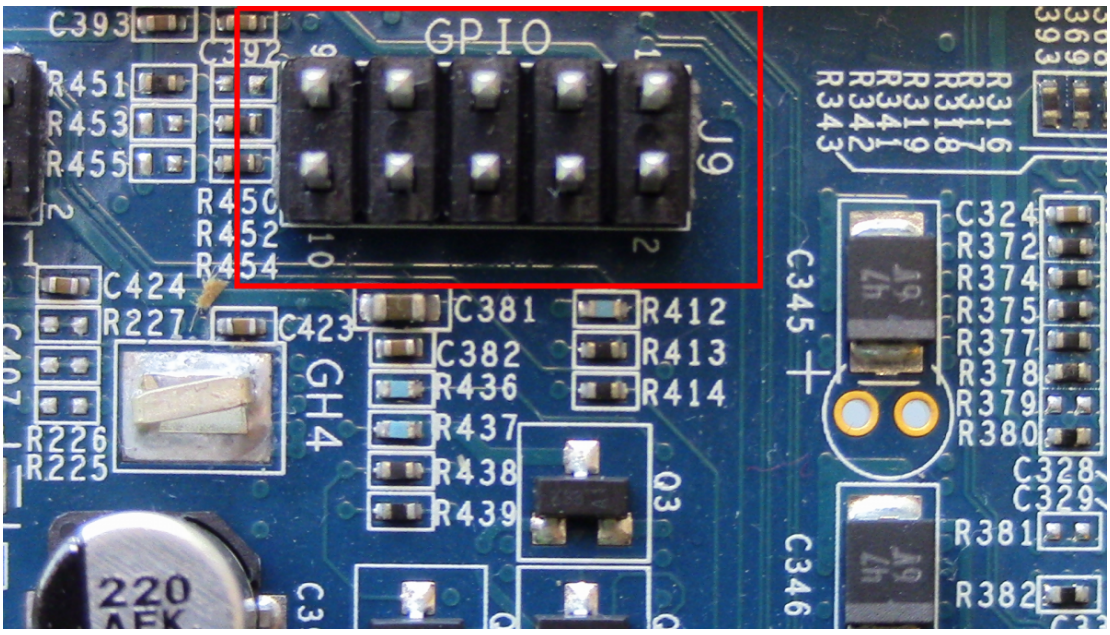
GPIO on the MinnowBoard

The MinnowBoard provides a variety of GPIO(General Purpose Input Output), some with a dedicated purpose. For the purpose of this guide, I will be concentrating only on GPIO(s) which are on the J9 expansion header as shown in Figure - 1 . Please refer the table below to find out how are the GPIO(s) actually referenced in the kernel and their default modes as well as values. Unless otherwise stated the GPIO(s) on J9 expansion header are rated at 3.3V and can source/sink a maximum of 10mA. Also, it can be seen from the table that the GPIO(s) are by default set to be used in INPUT mode and have PULL-UP resistors enabled.

GPIO on the MinnowBoard

Sr. No.	GPIO Reference Number in the kernel*	Default Mode	Default Value
1	N.A.	N.A.	3.3V
2	N.A.	N.A.	GND
3	244	INPUT	HIGH
4	245	INPUT	HIGH
5	246	INPUT	HIGH
6	247	INPUT	HIGH
7	248	INPUT	HIGH
8	249	INPUT	HIGH
9	250	INPUT	HIGH
10	251	INPUT	HIGH

*For example GPIO-3 on J9 header will be referenced as gpio-244 in kernel



As you can see from Figure -1, there are a total of ten pins on the J9 header out of which 8 pins can be used as GPIO. The remaining two pins are GND and 3.3V as mentioned in the table.

The GPIO(s) are accessible via the user space in Linux at the location below on the filesystem:

```
/sys/class/gpio
```

There is one directory per GPIO, named as shown below(as an example only two GPIO(s) are shown here):

```
/sys/class/gpio/gpio245
```

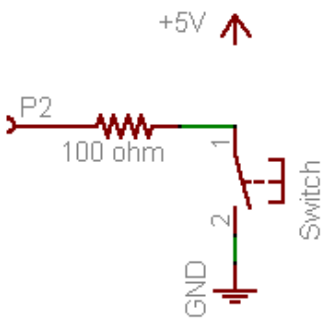
Inside each one of those directories, there are files named "direction" and “value” as shown in Figure - 2. The former is for configuring the mode of GPIO as input(“in”) or output(“out”) while the latter is for the value('1' for HIGH and '0' for LOW) if used in output mode.

```
root@minnow:/sys/class/gpio/gpio244# ls
active_low  device  direction  edge  power  subsystem  uevent  value
```

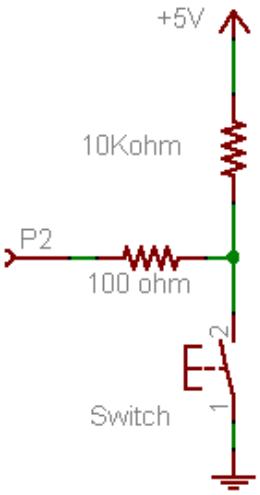
Pullup/Pulldown Resistors

Concept

So, you want to use the GPIO on the MinnowBoard to read values externally connected devices(e.g. push button). The particular GPIO pin is set to be used in input mode. Now, think about the case when no input is being provided to that pin(lets say P2) as shown in Figure - 3 below. This is called a 'floating point' condition and the processor cannot determine what is the value of the input pin. In this case, even the noise in the environment can influence the pin value and give wrong results.

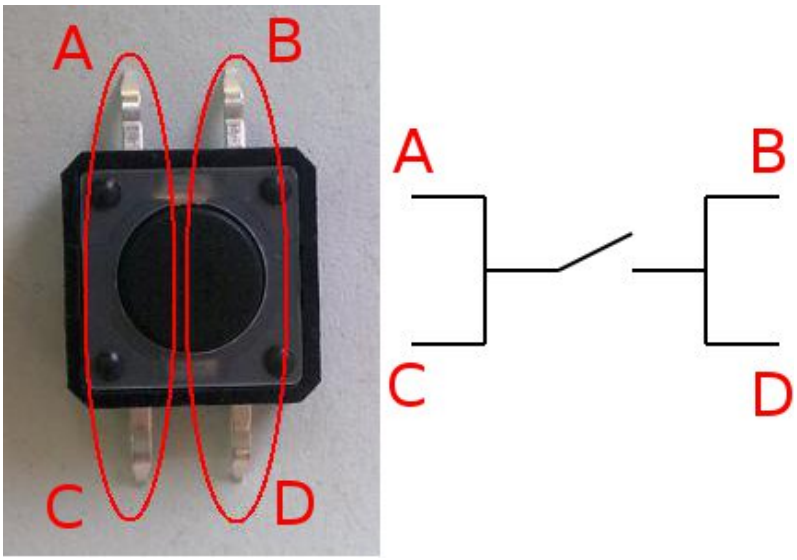


Hence, to avoid this condition the pins have a pullup/pull down resistor connected to them to prevent this from happening . Pullup is used to denote that the pin is connected to Vcc(3.3V in this case) via a resistor and pulldown is used to denote the pin that is connected to GND via a resistor. In Figure - 4 below, a 10k ohms pullup resistor is used. The value is high so that when the switch is pressed and the GND is connected to 5V, the current passing through is very less and does not burn the wire or damage the circuit. MinnowBoard has internal pullup resistors.



Push button

A pushbutton is a simple switch mechanism which permits user generated changes in the state of a circuit. Pushbutton usually comes with four legs. As seen from the Figure - 5 below, legs are always connected in groups of two. When the pushbutton is pressed all the 4 legs are connected.



The pushbutton used in this guide can be purchased from here:

https://www.sparkfun.com/products/97

Selecting the correct LED

• To make sure that you do not damage the GPIO pins on the MinnowBoard, please use a LED whose rating should not exceed 3.3V/10mA:

http://www.digikey.com/product-detail/en/HLMP-4700-C0002/516-2483-2-ND/1234840

• If you have an LED that is higher than the above mentioned ratings, please refer the refer the link below to select an appropriate current limiting resistor:

http://www.cmiyc.com/tutorials/led-basics

Physical Computing

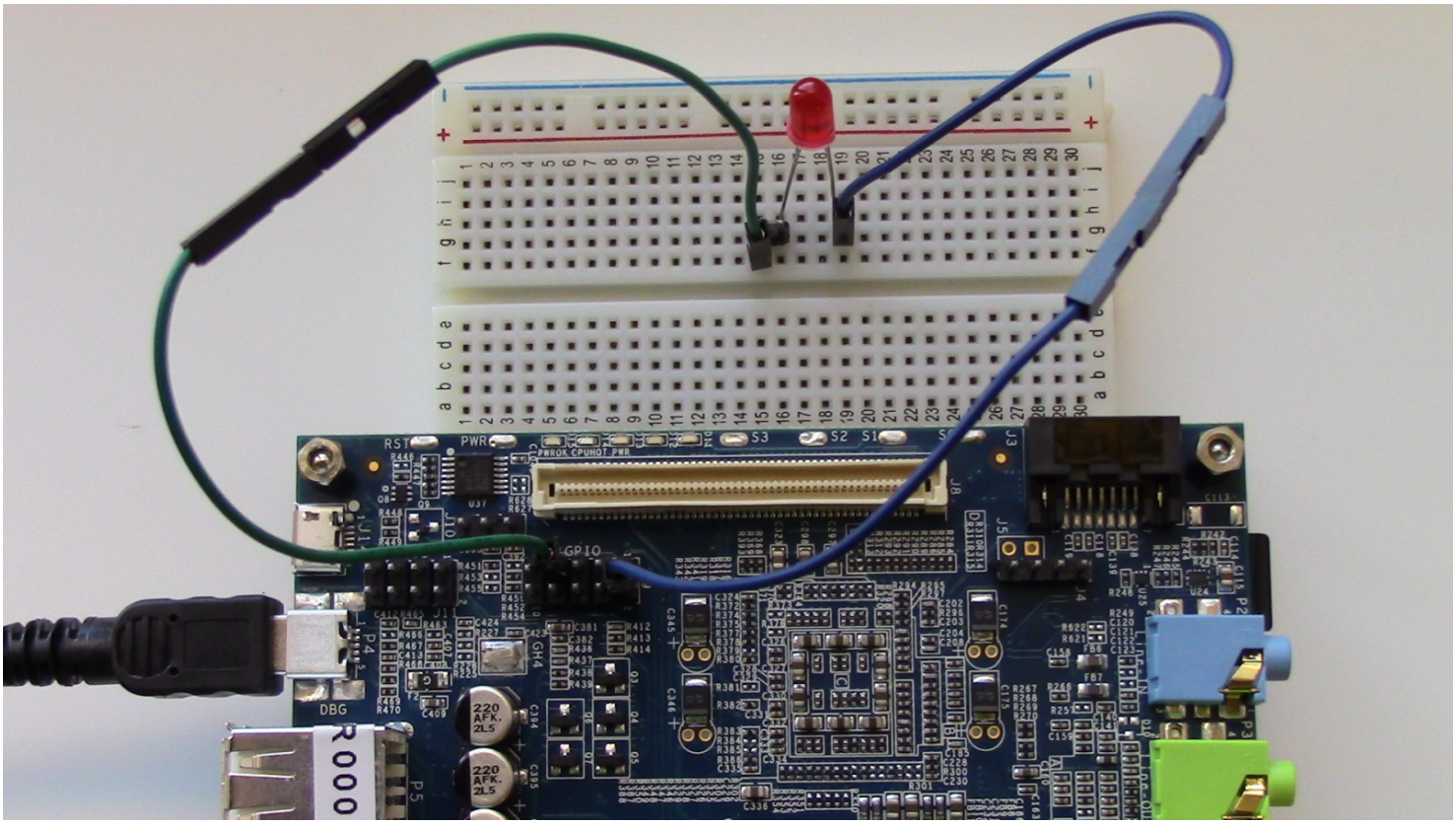
Physical computing, in the broadest sense, means building interactive physical systems by the use of software and hardware that can sense and respond to the outside world. In our case we are trying to take input from the surrounding via the push button and accordingly causing some change in the outside world by turning ON the LED when the push button is pressed.

Extra Credit

This is an optional section which you can read for further understanding. In a nutshell, we are trying to access the on board USER LED(s) via userspace in Linux. To be more precise we are using the sysfs interface. sysfs is a virtual filesystem which translates the hardware devices and busses attached to the system(board in our case) into a file system hierarchy that can be accessed from userspace. sysfs is generated by the kernel and always mounted at /sys. The GPIO sysfs interface allows users to manipulate any GPIO from userspace. Since it uses gpiolib, it is able to dynamically utilize all GPIOs on the system. The preloaded Angstrom image on the MinnowBoard already has all the GPIO on J9 header exported to the userspace by default.

Setup

Please refer Figure - 7 below to see the connections:



Only the pins on the J9 expansion header are used in this case. Connect the anode of the LED to the resistor(I am using a 330 ohms resistor) which in turn is connected to pin 3(GPIO) and connect the cathode of the LED to pin 2(GND) . You can purchase the male-female type hookup/jumper wires used for connections from the link below:

https://www.sparkfun.com/products/9140

Script

```
#!/bin/bash
#Use Pin-3 as output pin
echo out > /sys/class/gpio/gpio244/direction
#Use Pin-7 as input pin
echo in > /sys/class/gpio/gpio248/direction
# Read forever
while :
do
# Read value of Pin-7
THIS_VALUE=`cat /sys/class/gpio/gpio248/value`
if [ "$THIS_VALUE" = "0" ]
then
echo 1 > /sys/class/gpio/gpio244/value
else
echo 0 > /sys/class/gpio/gpio244/value
fi
done
```

In the above script, '#' is used for comments(except for first line). Write the above script in your favorite text editor, save it(make sure to add the '.sh' extension at the end) and place it in on the microSD card before you boot the MinnowBoard or you can download the script from here

In any case make sure the script is executable by typing the following command in the terminal:

```
$ chmod a+x <script_name>
```

In my case, I named the script 'physicalcomputing.sh'

Steps

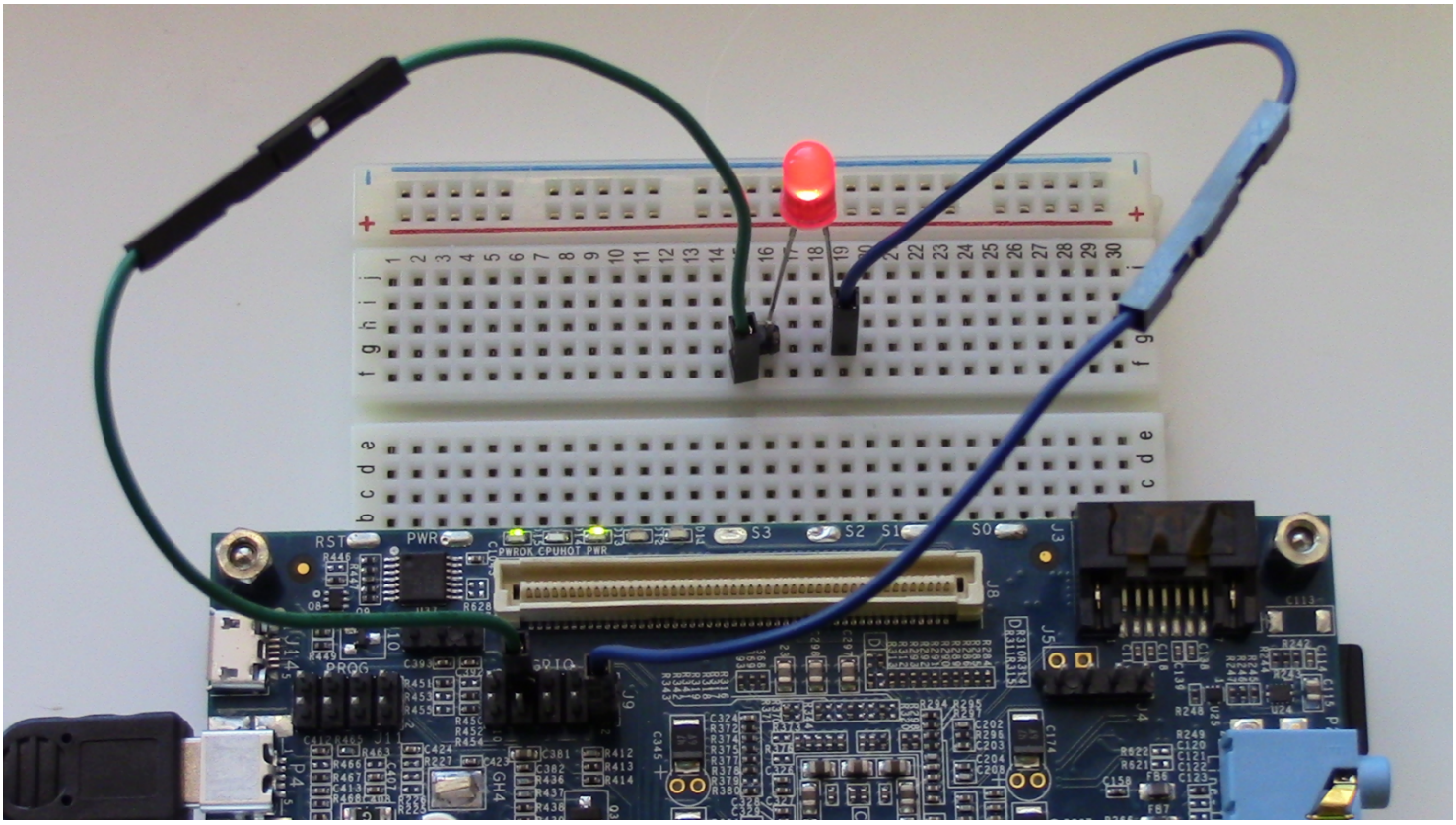
So, now run the script created earlier by typing the following command in the terminal as shown in Figure - 8:

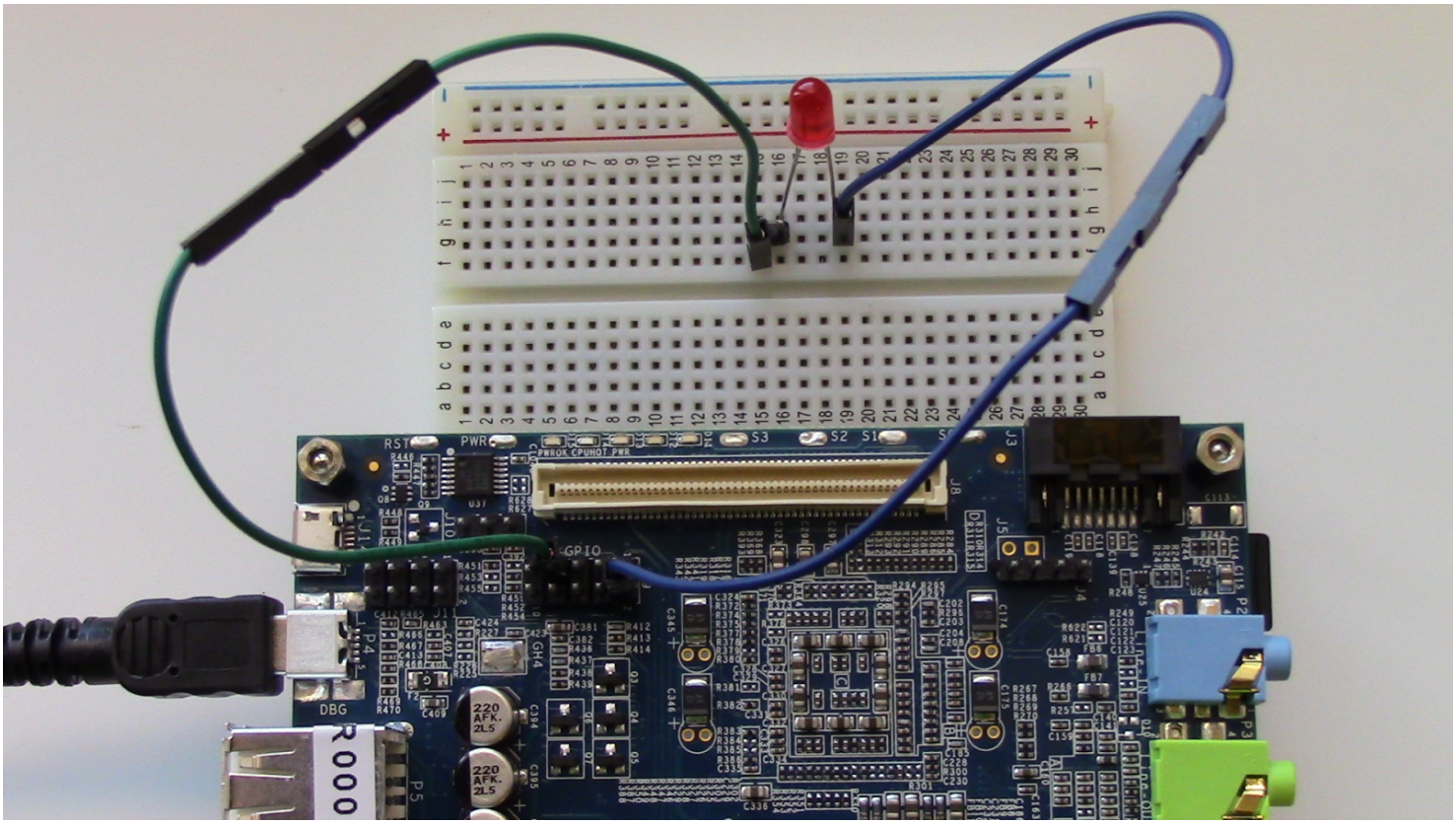
```
$ ./physicalcomputing.sh
```

```
root@minnow:~/Desktop# ls
physicalcomputing.sh
root@minnow:~/Desktop# ./physicalcomputing.sh
```

Output

You should get an output as shown below:





Navigation menu

Personal tools

- Log in
- Request account

Namespaces

- Minnowboard
- Discussion

Variants

Views

- Read
- View source
- View history

More

Search

Search Search Go

Navigation

- Main Page
- Community portal
- Current events
- Recent changes
- Help
- Volunteering
- Bug Tracker

Where else to find us

- Google+ Community
- Twitter (@elinux)
- #elinux on Freenode
- Facebook (@elinux.org)
- Mailing Lists

Tools

- What links here
- Related changes
- Special pages
- Printable version
- Permanent link
- Page information

- Privacy policy
- About eLinux.org
- Disclaimers