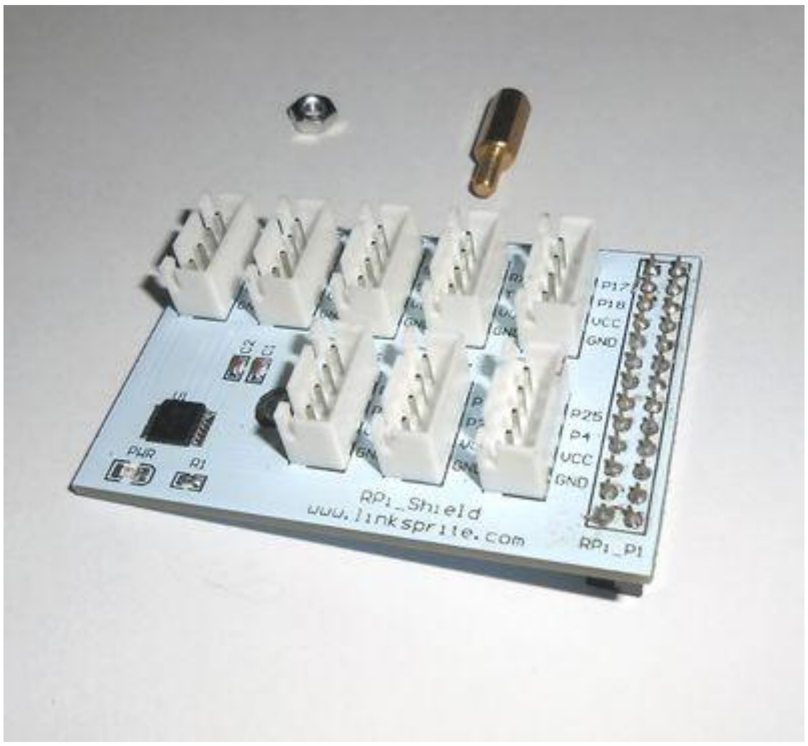
Description

This is a Linker kit base shield for Raspberry Pi with ADC interface. All the Linker kit modules can be plugged onto Raspberry Pi through this base shield. Moreover, this shield has an on-board ADC chip so that analog output modules can be used on Raspberry Pi. The ADC chip used is MCP3004. It talks to Raspberry Pi using SPI interface.



Specification

There are 8 4-pin 2.54mm spacing housing on the base shield:

- JP1: A0, A1, VCC, GND.
- JP2: A2, A3, VCC, GND.
- JP3: SCL, SDA, VCC, GND.
- JP4: RXD, TXD, VCC, GND.
- JP5: P17, P18, VCC, GND.
- JP6: P27, P22, VCC, GND.
- JP7: P23, P24, VCC, GND.
- JP8: P24, P4, VCC, GND.

These connectors cover Analog, I2C, and GPIO.

Install Base Shield on Raspberry Pi

Schematics

- Schematics

Tutorial

- Tutorial

RPI ADC base shield 教程 中文版

The ADC chip used on this base shield is MCP3004 (10 bit resolution, 4 channels, SPI interface). In the following, we are going to cover how to use Python to read the ADC output:

As we use SPI communication, we need to enable SPI module in kernel. To do that, we can edit the blacklist of kernel, and enable spi:

```
sudo vi /etc/modprobe.d/raspi-blacklist.conf
```

Locate the line begins with "blacklist spi-bcm2708" and change it to "#blacklist spi-bcm2708".

After this, we can check the module by "lsmod", and observe the following content:

```
Module Size Used by spi_bcm2708 4421 0
```

Now we will install python-pip (pip is a package used to install and manage python software package, and it is used replace esay_install):

```
sudo apt-get install python-imaging python-imaging-tk python-pip python-dev
git
```

Next, we will install spidev using pip:

```
sudo pip install spidev
```

Then we will install WiringPi (the driver for IOs on Raspberry pi, that can be used in C, shell script or Python, etc):

```
sudo pip install wiringpi
```

The interface python code for MCP3004 is as following:

```
Import spidev
Import time

spi = spidev.SpiDev()
spi.open(0,0)

def readadc(adcnum):
# read SPI data from MCP3004 chip, 4 possible adc's (0 thru 3)
    ifadcnum >3oradcnum <0:
        return-1
    r = spi.xfer2([1,8+adcnum <<4,0])
    adcout = ((r[1] &3) <<8)+r[2]
Returnadcout

whileTrue:
    value=readadc(0)
volts=(value*3.3)/1024
    print("%4d/1023 => %5.3f V" % (value, volts))
    time.sleep(0.5)
```

Save the above code to a file named linker_adc.py, and enter into the directory to execute:

```
$python ./linker_adc.py
```

The following are the results:

```
91/1023 =>0.293V
93/1023 =>0.300V
94/1023 =>0.303V
```