



Integrated Demo For Raspberry Pi & Embedded Pi

Rev. 1.0

Release: 2013-04-12

Website: www.coocox.org

Forum: forum.coocox.org

Techinal: master@coocox.com

Market: market@coocox.com

1 Instruction

1.1 Background

Raspberry Pi and Arduino have been well known by many people at present, the two big names have been launching a boom in the world of the open source electronic DIY. Raspberry Pi has a strong processing capacity because of using the ARM11 architecture and Linux-based system. In terms of control and interface, it has 8 GPIO, 1 UART, 1 I2C and 1 SPI, which are basically meet the control requirement. what's more, there are simple and easy-used open source peripheral driver libraries, this is one of the reasons why Raspberry Pi is so popular around the world. However, Raspberry Pi also exists some disadvantages, for example, it does not have analog capture capability and the real-time performance of the system are not high enough. Arduino, as a early open source electronic building block, its flexibility, ease of use and portability are more affected by countless people. With a variety of modules and components, users can develop many amazing interactive work, but the shortcoming is that its main control chip is an 8-bit AVR microcontroller, which means the space of peripherals and program are limited as well as the running speed. Nowadays, the price of Cortex-M3 MCU(32-bit)have fallen closer to 8-bit single-chip, compared with the 8-bit microcontroller, Cortex-M3 MCU has lower power consumption, more memory, more peripherals, more powerful performance and real-time data processing. Besides, it supports online debugging to make it convenient for you to identify a program bug quickly.

The birth of Embedded Pi is made up for the inadequacy of Raspberry Pi and Arduino. Embedded Pi uses STM32F103 series(ARM Cortex-M3 MCU) as the main control chip. It's fully compatible with Arduino and Raspberry Pi's interface, Embedded Pi works in three modes by setting jumpers on the Board:

1) **Ras-Pi Adapter Mode**

In this mode,Embedded Pi works as Raspberry Pi's slave machine. Firstly, Embedded Pi collects data and sends it to Raspberry Pi, then the Raspberry Pi analysis the data and transmits the controlling parameter to Embedded Pi. As a result, Embedded Pi plays the role of controlling the peripherals. This routine will demonstrate this mode.

2) **STM32 Standalone Mode**

In this mode,You can treat Embedded Pi as a separate STM32 development board to learn how

to use the STM32 to develop your own applications. Embedded Pi is fully compatible with Arduino's interface, which can connect and control a variety of Arduino modules directly .

3) Ras-Pi Standalone Mode

This pattern is just output every IO of the Raspberry Pi through Level-conversion,the output interface is the same as the Arduino's, so it can control Arduino module with Raspberry Pi directly, there is no need to take complicated wiring and level compatibility issues into account.

1.2 About the routine

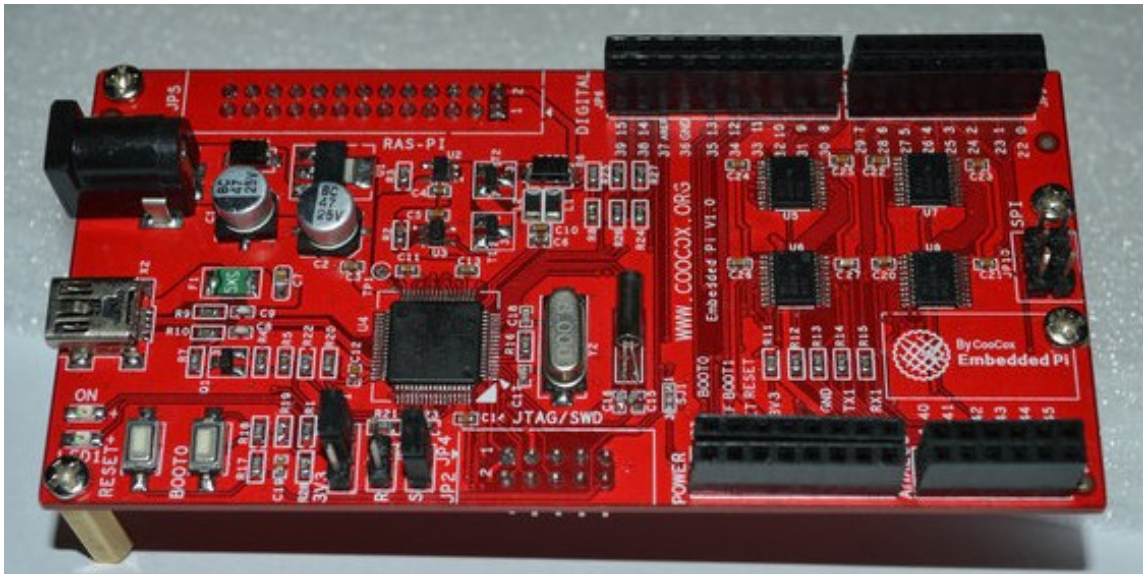
This routine shows how to use Embedded Pi and Raspberry Pi to achieve common control. As a host, Raspberry Pi receives and processes the data from Embedded Pi, while Embedded Pi works as a data collection and control execution terminal, it is used to control Arduino 's motor module and TinkerKit sensor module. Motor module is a stepper motor, here we simulate it as a rudder and add various dials as the indicator of the data size. For the TinkerKit sensor module, we shall connect a sliding rheostat,a ultrasonic distance sensor and a three-axis magnetic sensor respectively to show the demonstration. Raspberry Pi is connected to a computer over the network and the computer is connected to the Raspberry Pi's remote desktop, open command line in Linux system on the remote desktop, we had written a set of open source applications based on the Linux command line,users can send and receive data and instructions through the procedure.Raspberry Pi and Embedded Pi use serial communication.

2 Hardware

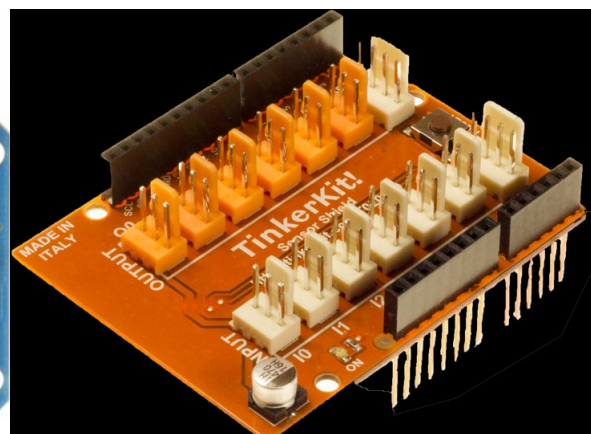
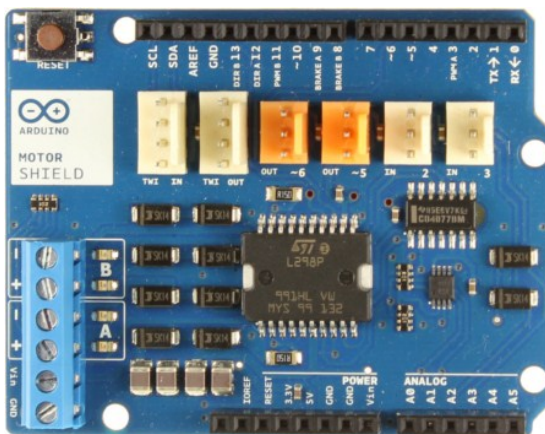
2.1 Hardware Requirement

To realize the demo routines requires the following hardware :

- A piece of Embedded Pi



- One Arduino motor driver module and one TinkerKit module



■ A piece of Raspberry Pi

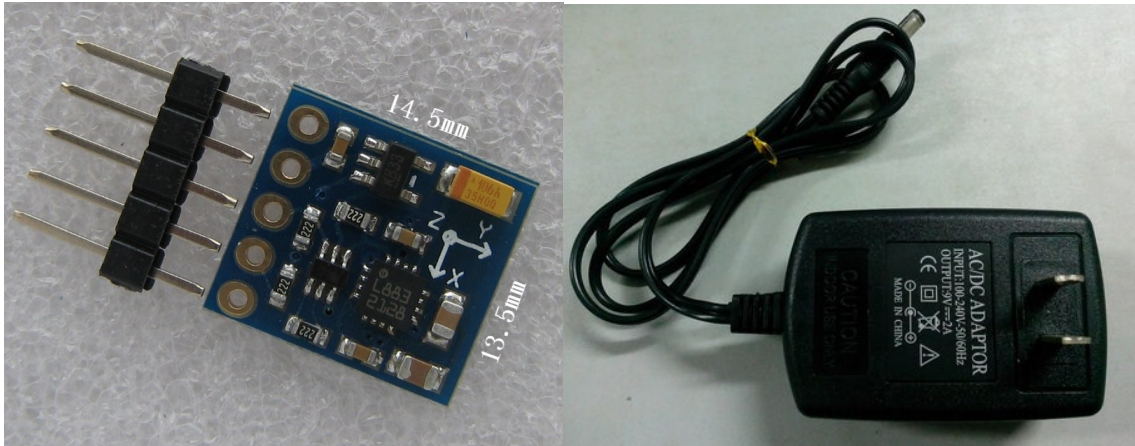


■ Modules and Equipments

- 1) A 42BYGHA stepper motor
- 2) A HC-SR04 ultrasonic distance sensor
- 3) A linear sliding potentiometer



- 4) A GY-271 three axis electronic compass
- 5) Power(9V, 2A)



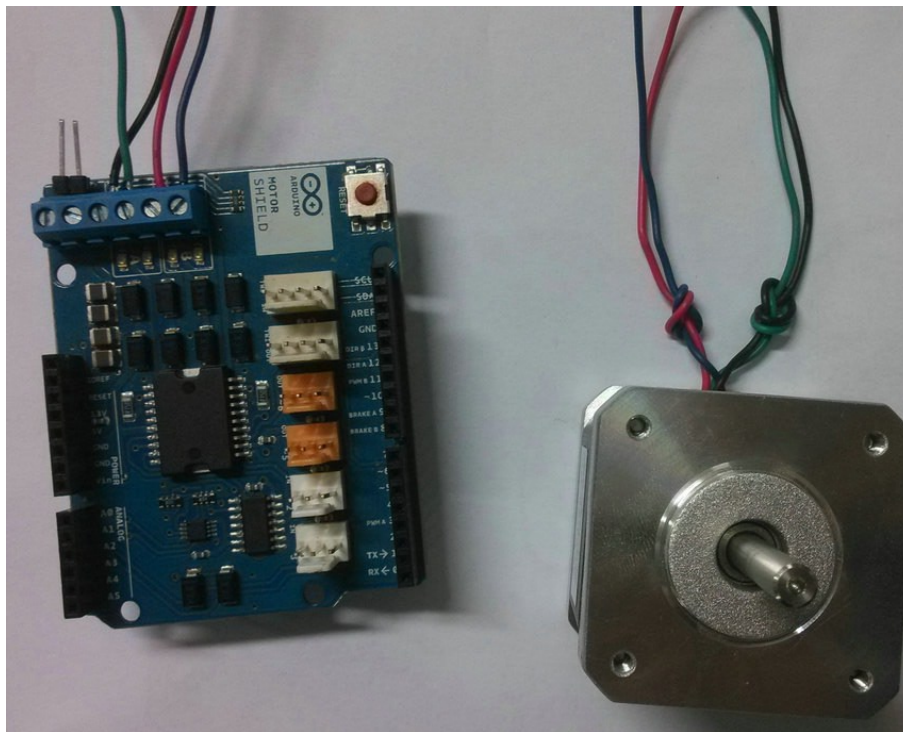
- 6) A 26-pin flat cable (it's used to connect Embedded Pi and Raspberry Pi)
- 7) Some dupont lines
- 8) A pointer
- 9) Some dials(A4 paper printing for showing result only, do not ensure accuracy, **optional**)

The ultrasonic ranging dial is shown as below :

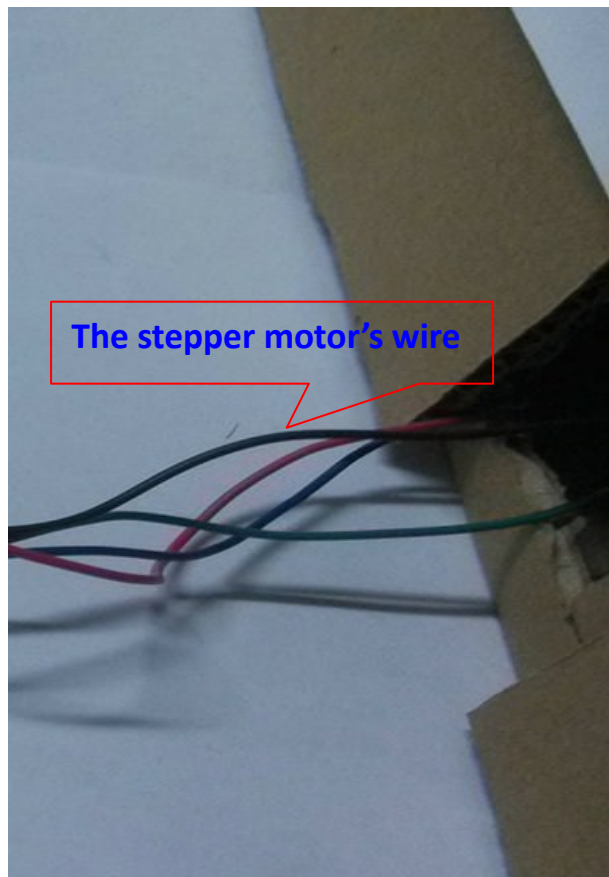
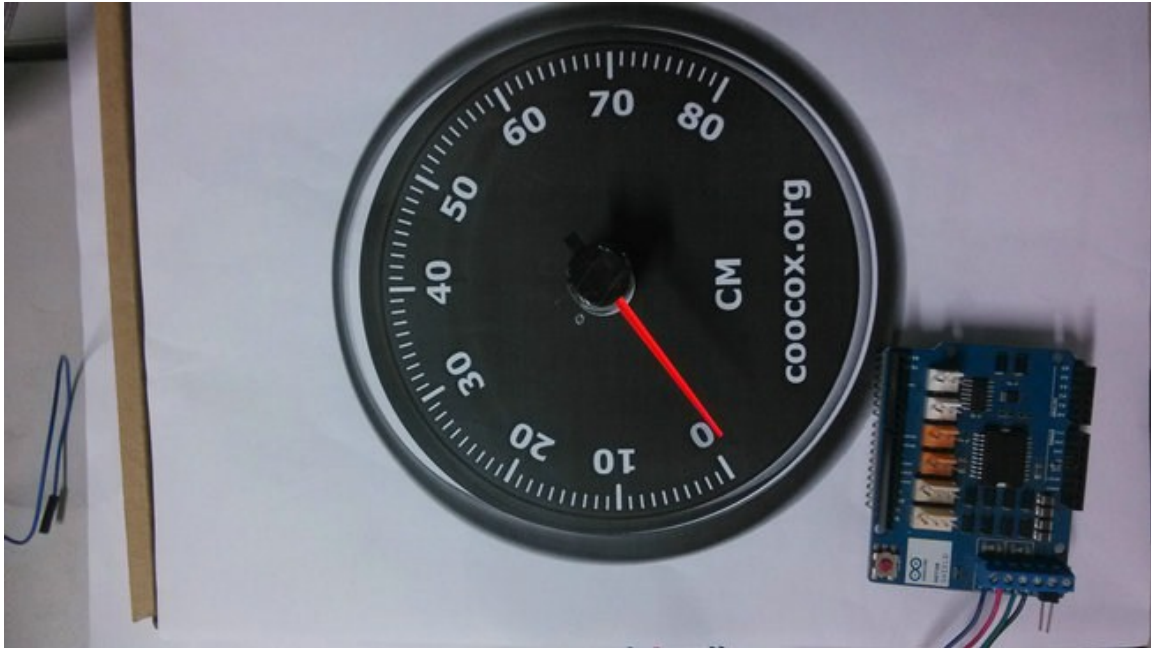


2.2 Equipment Preparation And Hardware Connection

Firstly, print a couple of dials with A4 paper, the scale is "cm" (used for distance measurement), "V" (used for voltage measurement) and "°" (for angle measurement, electronic compass) respectively, the previous part only shows a dial for distance measurement. Secondly, connect the stepper motor's 4 wires to the Motor shields' driver module, note that the sequence of the 4 wires cannot be wrong, the final result as shown in the figure below, the wire sequence of the Motor Shield is black, green, red and blue from left to right.

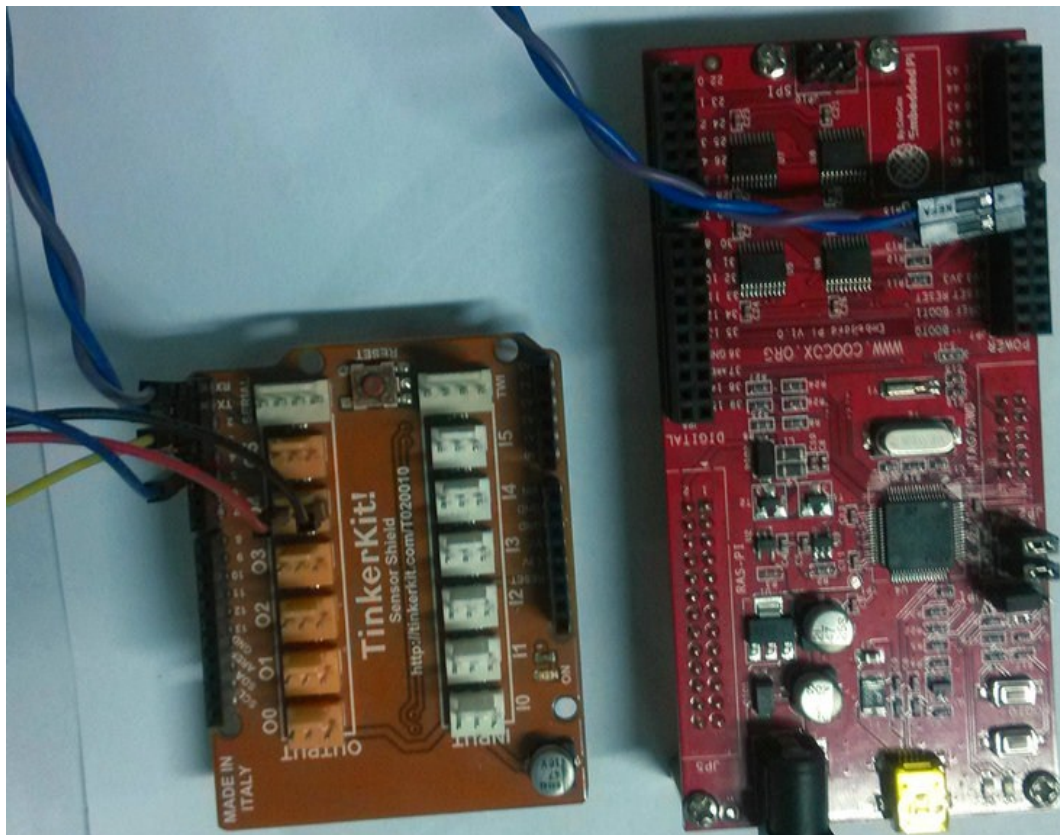
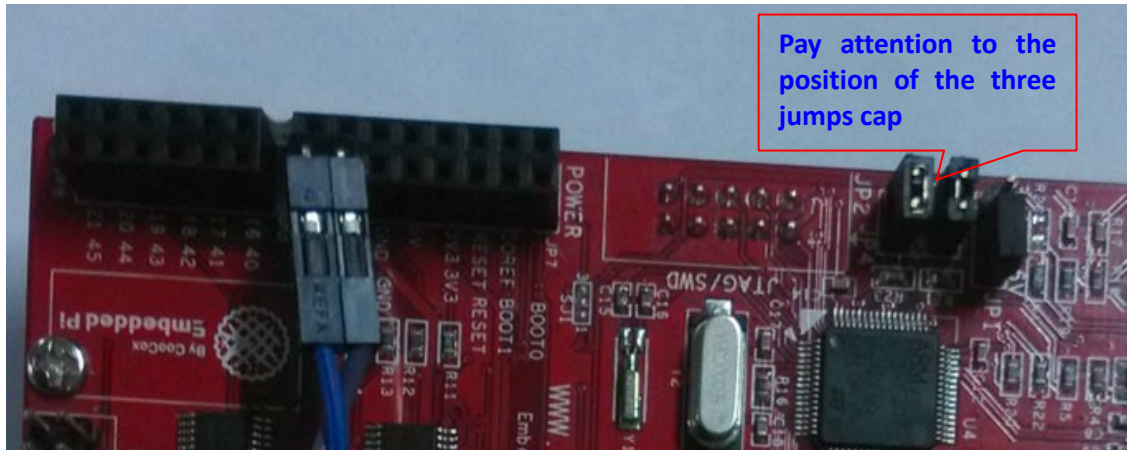


Here we find a paper box of suitable size and fix the stepper motor in it. Then fasten the dial on one side, what we need to do next is to dig two holes on this side: one is in the middle, the stepper motor can get through it; the other provides path for the wire of the stepper motor. Then fix a pointer on the motor's shaft with glue, the pointer is removable so that we can change the dial if necessary, the result is shown in the following figure:



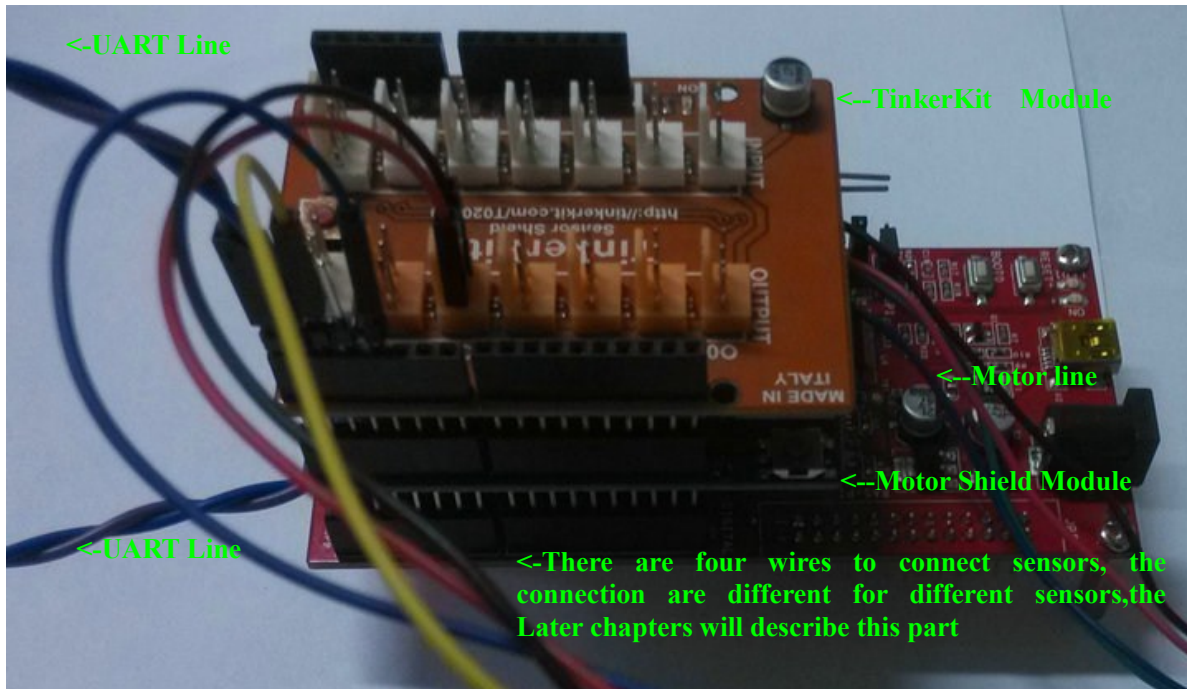
Next link Raspberry Pi to Embedded Pi with flat cable(26 pin), the Raspberry Pi's network interface connects to the network cable. In order to achieve the serial communication between the Embedded Pi and the Ras-Pi, we need to join the Ras-Pi's UART to the STM32's UART.

Although the Ras-Pi's UART on the Embedded Pi board connects to the STM32's UART3 after level conversion, the RXD and TXD did not cross and reverse that's because they are used as outputs. Here we choose the STM32's UART1 instead as sending and receiving data port. So connect the Raspberry Pi's UART and the STM32's UART1 with dupont line, note that the UART's TX on the Ras-Pi connects to the STM32 UART1's RX, while the RX connects to TX, as shown in figure:



Since the TinkerKit module will be placed on the top, the two wires of the UART connect to

TinkerKit. Then plug the Motor Shield module into the Embedded Pi and plug the TinkerKit into the Motor Shield, the final combination as shown in the following figure:



when using 26 pin flat cable to connect Embedded Pi to the Raspberry Pi, note the the "1" pin of the Embedded Pi and the Res-Pi are corresponding. Meanwhile,connect the Raspberry Pi to the computer through a cross Ethernet cable or link the Raspberry Pi to the router within the same local area network (LAN), don't forget that the USB line for power supply is a necessary.

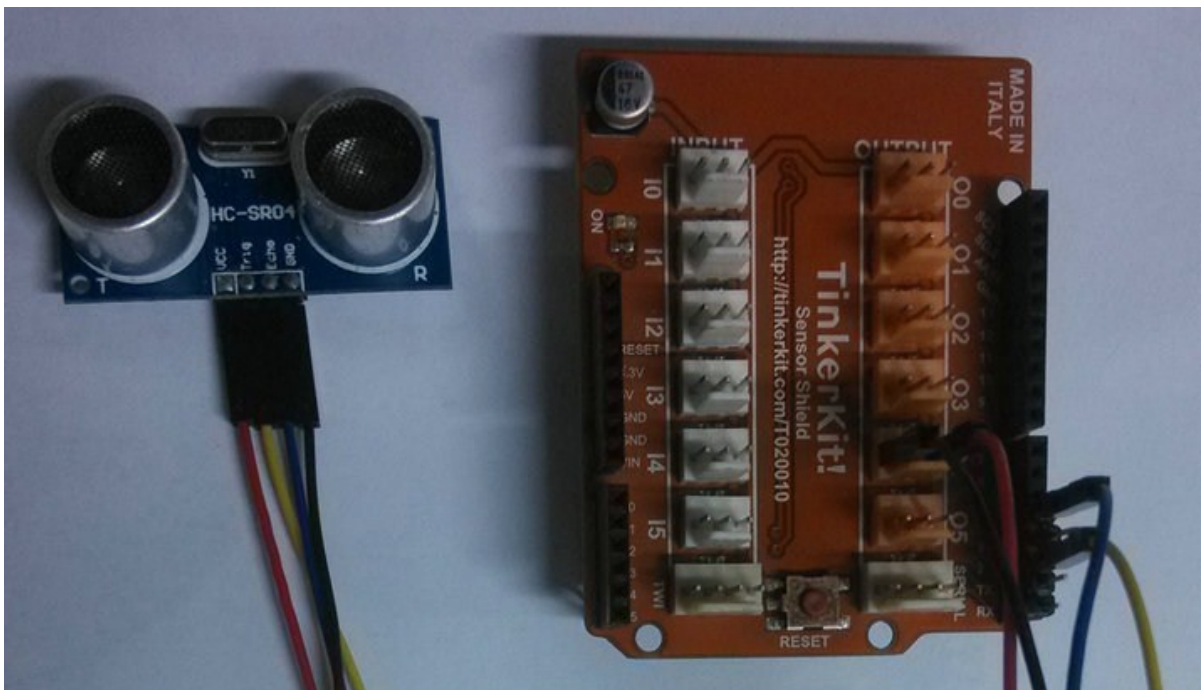
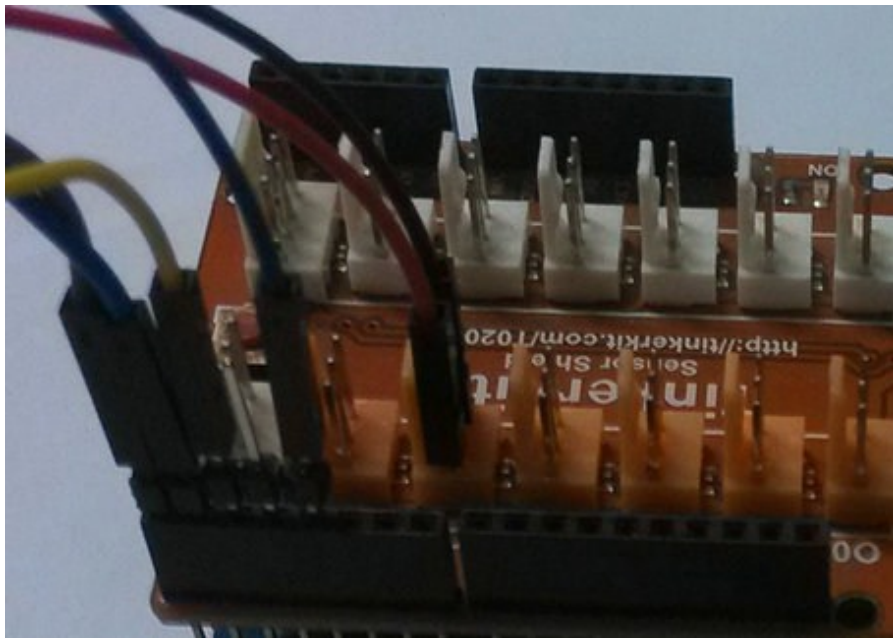


The next part mainly introduces the connection of sensors, there are three sensors in this demo: a linear sliding potentiometer that used to simulate the AD sampling voltage input,a HC-SR04 ultrasonic ranging module and a GY-271 triaxial electronic compass module. Though the three sensors can be used together, we use one module at a time to show the demonstration conveniently in order not to make the connection too messy, of course you can connect the

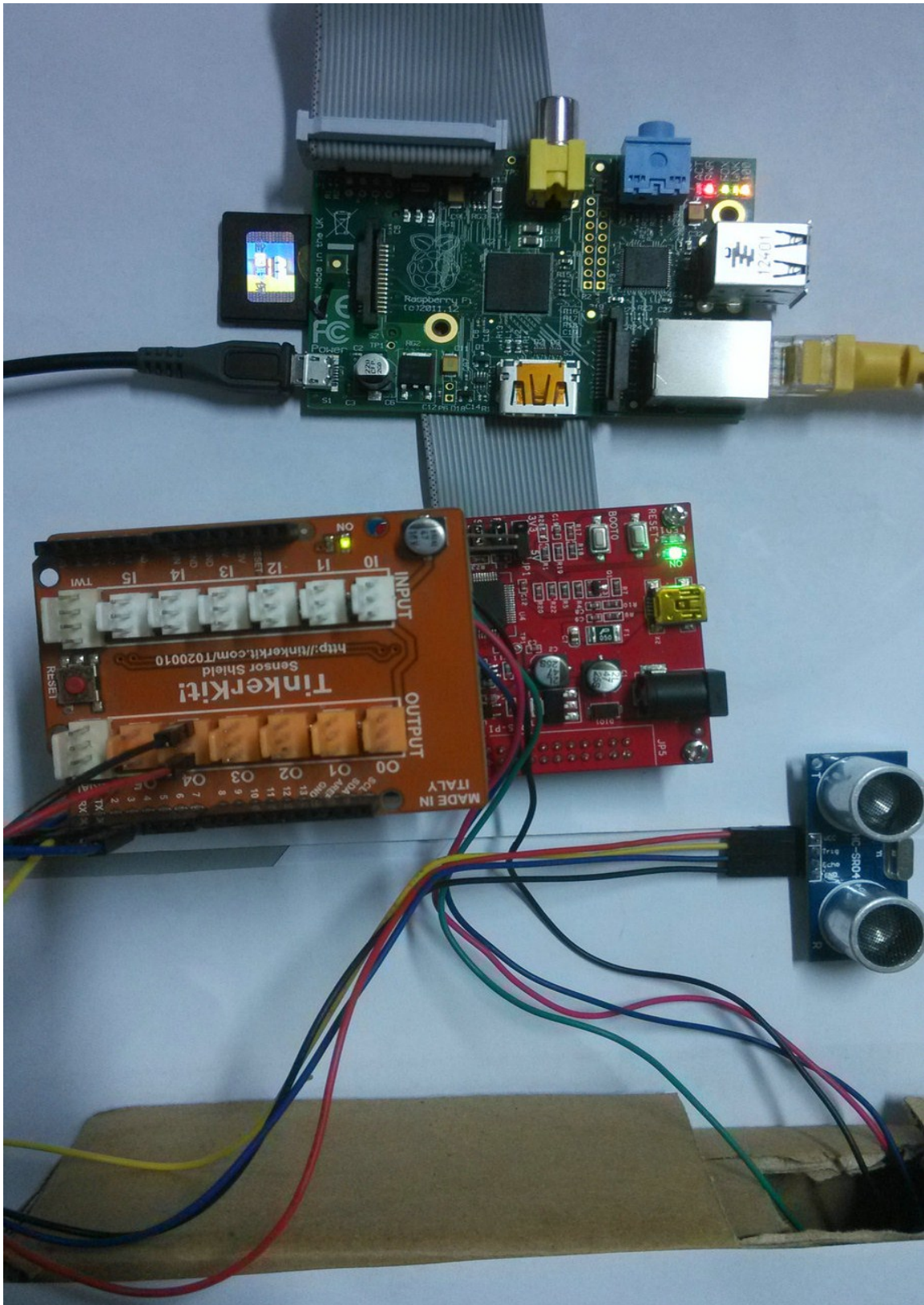
three sensors together at the same time if you think it's more convenient for you.

(1) Demonstrate the ultrasonic ranging module(HC-SR04 sensor)

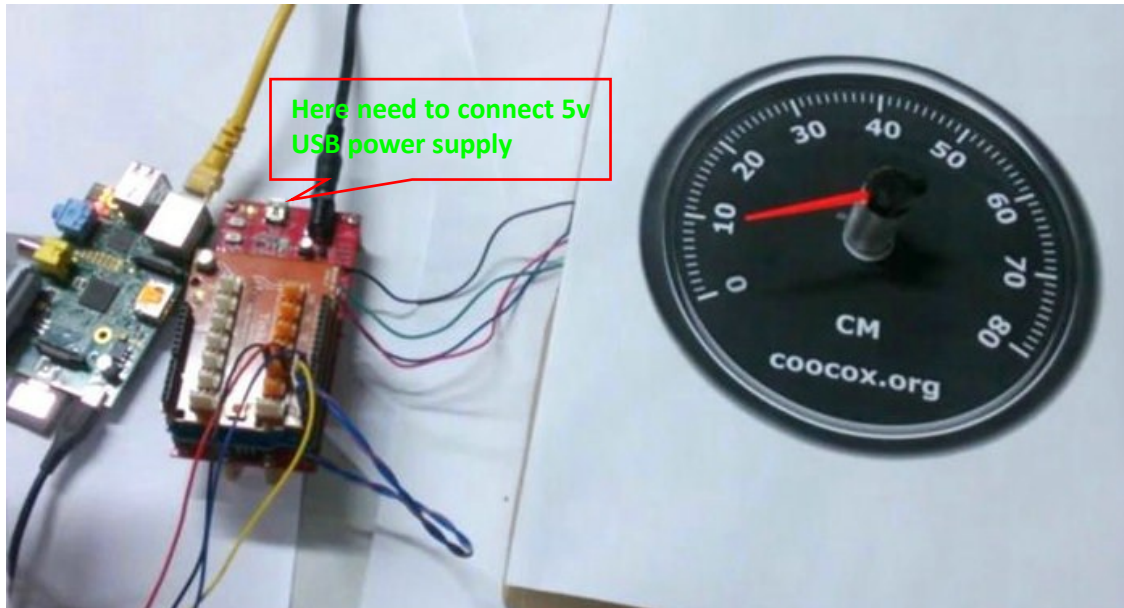
The HC-SR04 module has 4 pins, namely the power supply, ground, feedback signal and trigger signal. So find the power supply (5 v) and ground pins on the TinkerKit module to connect to its corresponding pins. In the program, we define the trigger pin is sD2, feedback signal is pin sD4.



All the parts assembled as shown in the figure below:

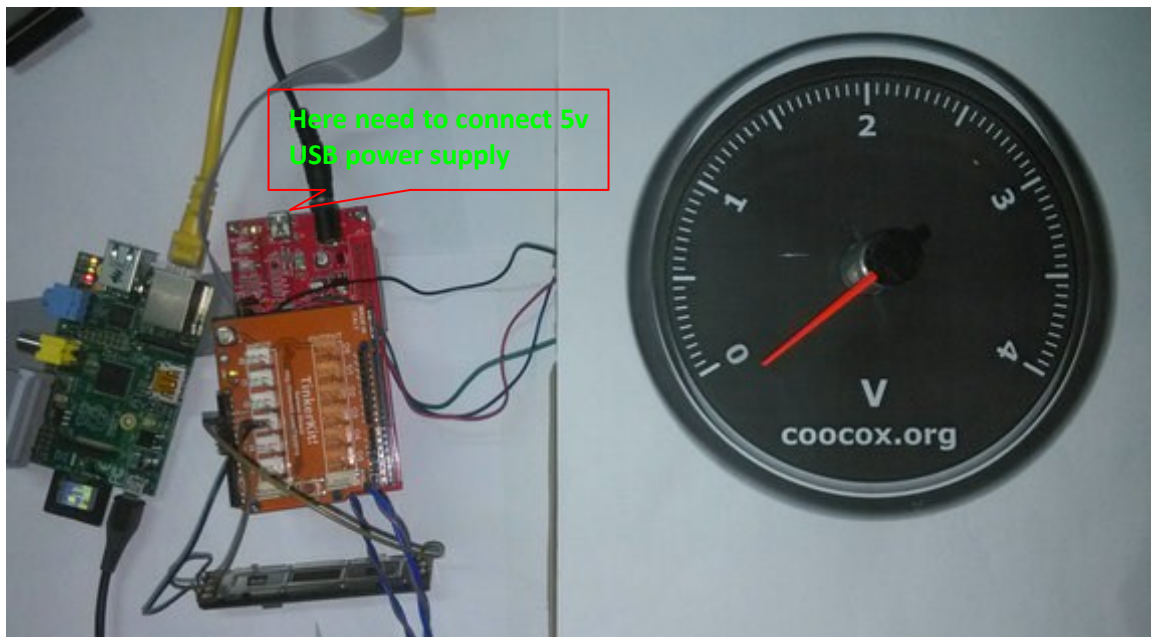
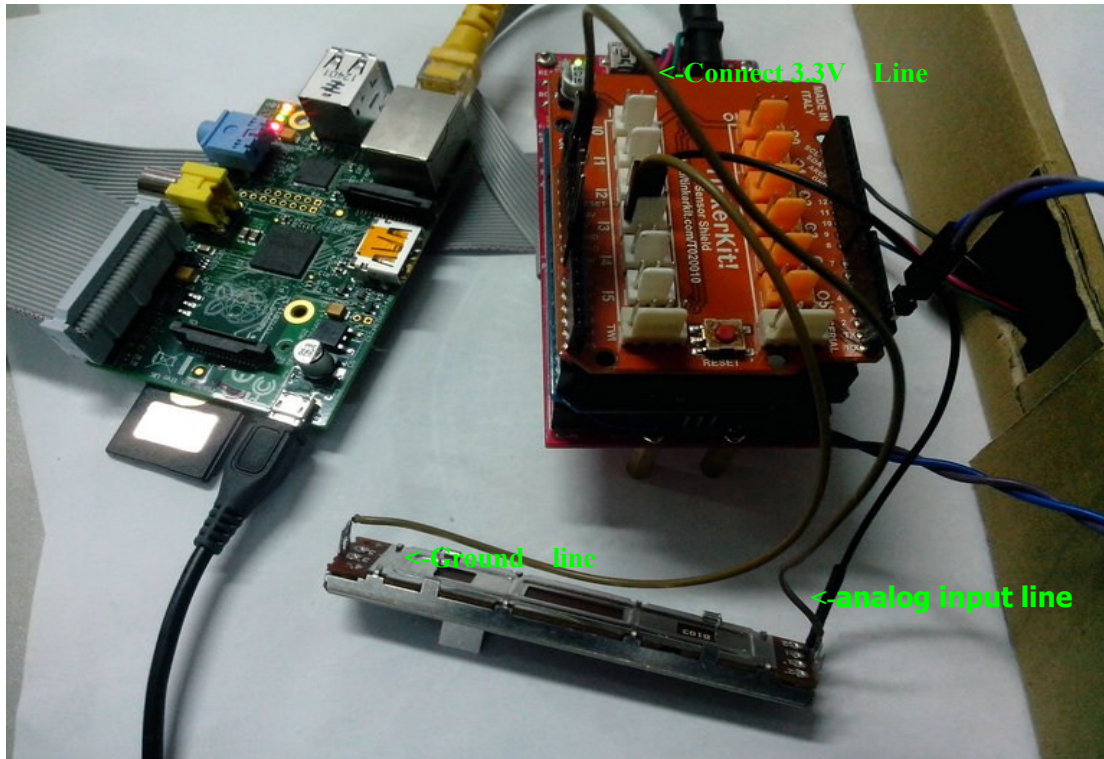


The ultimate effect figure is shown as below:



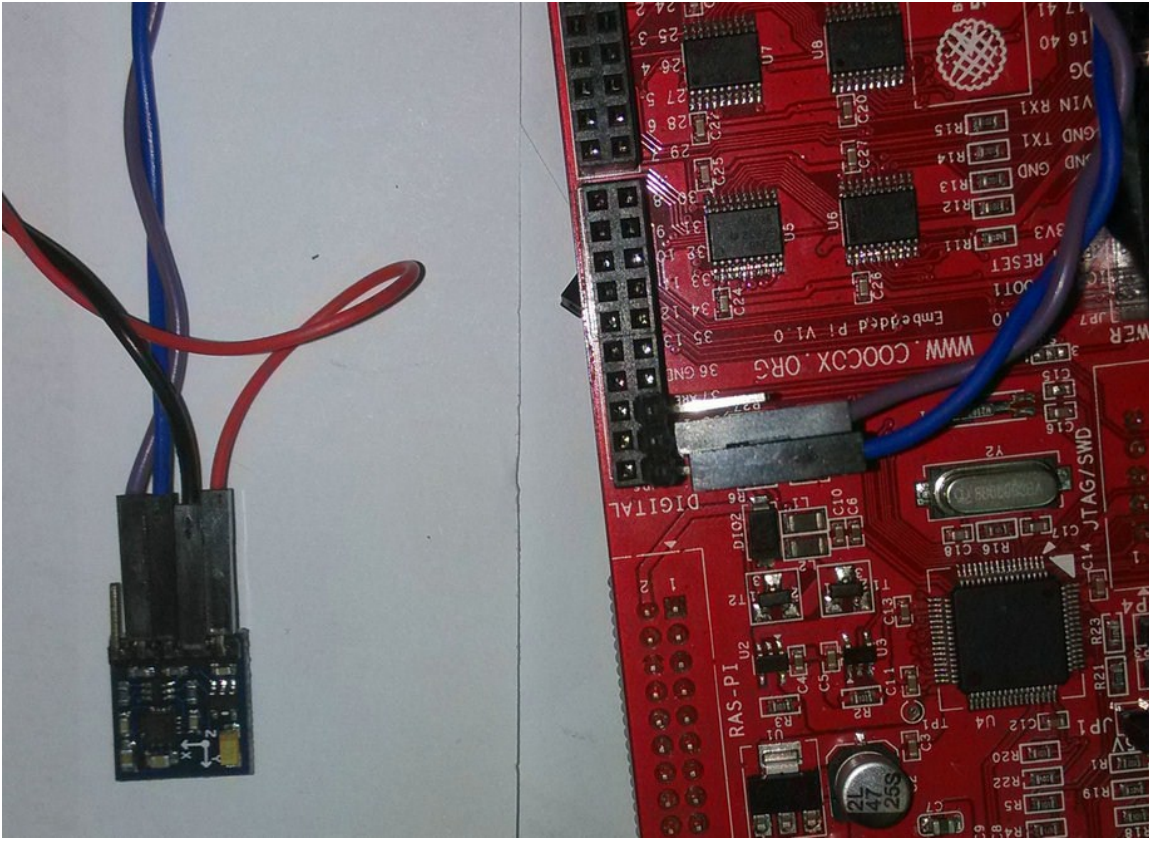
(2) Demonstrate the AD voltage sampling test (linear sliding potentiometer)

The potentiometer is needed to be welded three wires to connect to the TinkerKit module. Since the ADC's reference voltage of the STM32 is 3.3V, the input analog voltage cannot be greater than 3.3V, connect the two fixed ports of the potentiometer to GND and 3.3V respectively and connect the sliding port to the input channel I3 of the TinkerKit (defined by the program). Note that only I0 ~ I3 have ADC function, the I0 and I1 are used to sample the electric current for the motor's AB channels, other input signals can only connect to I3 or I2. Other parts of the connection do not need to change (the dial can be converted into voltage scale as you like), the diagram is as follows:

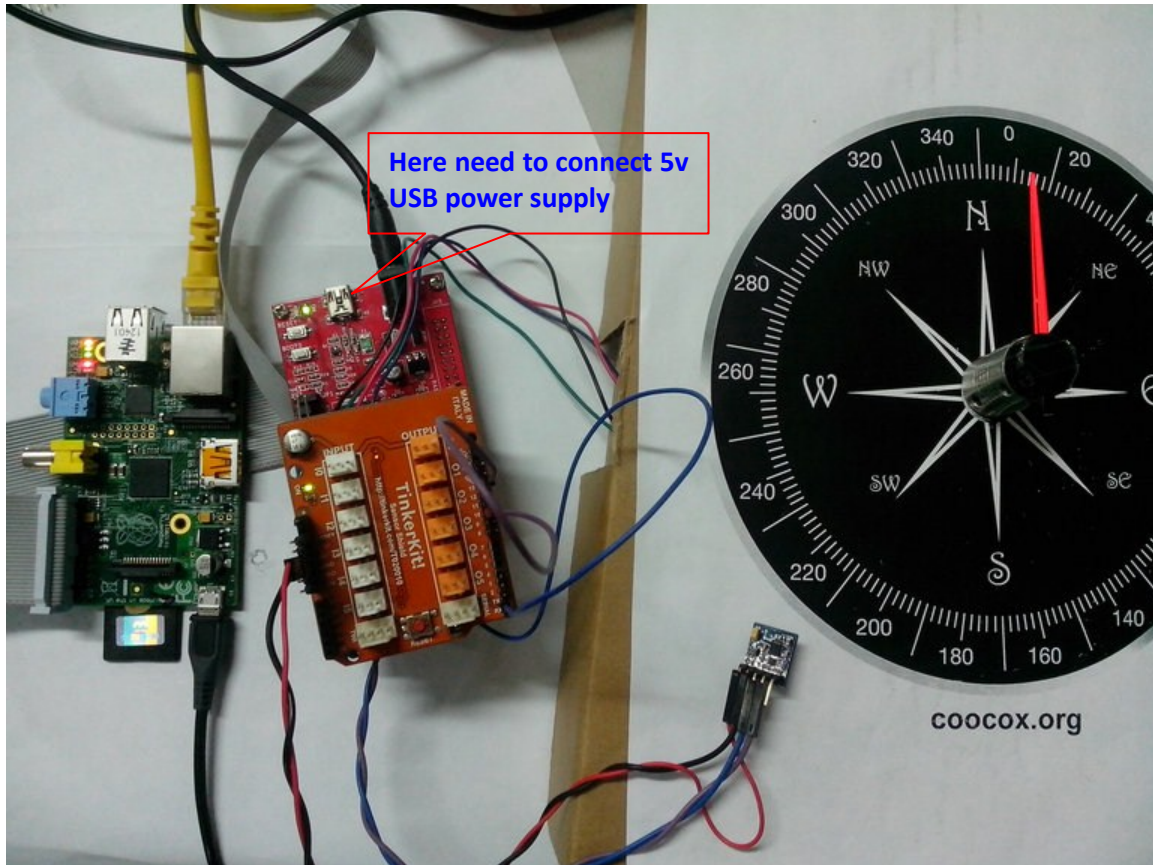


(3) Demo the the experiment with the three triaxial electronic compass modules(GY-271)

This module has 4 wires, namely power supply, ground, SCL clock signal and the SDA data signal. among them, the power supply and ground connect to 3.3 V and GND respectively, SCL and SDA connect to the C_D38 and C_D39 pins, as shown in figure:



The TinkerKit can be unconnected here , the final connection as shown in figure:

**Note:**

The stepper motor needs large current in normal working conditions, if your power supply is not sufficient, the voltage may be pulled down by the motor, which makes the motor driver module and the MCU work improperly. In order to ensure that the hardware works normally, it's strongly advised to provide 5V power for Embedded Pi's USB port and add proper external power supply at the same time.

In addition, the sD13 connects to the LED on Embedded Pi board, the pin's high level output may be pulled down, but it's not a big problem in general TTL circuits, if the power supply is not stable, the motor module probably can't work normally with this pin.

So far, all hardware parts have been introduced.

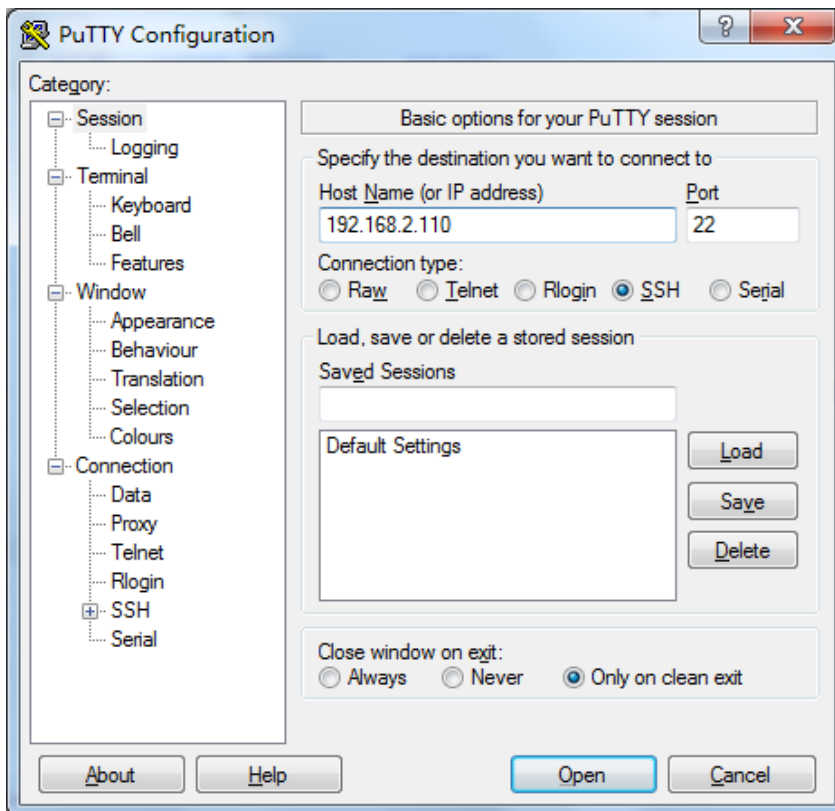
3 Demonstration

3.1 Ras-Pi's Setting and connection

Please refer to the site: http://elinux.org/RPi_Easy_SD_Card_Setup to know how to copy the img

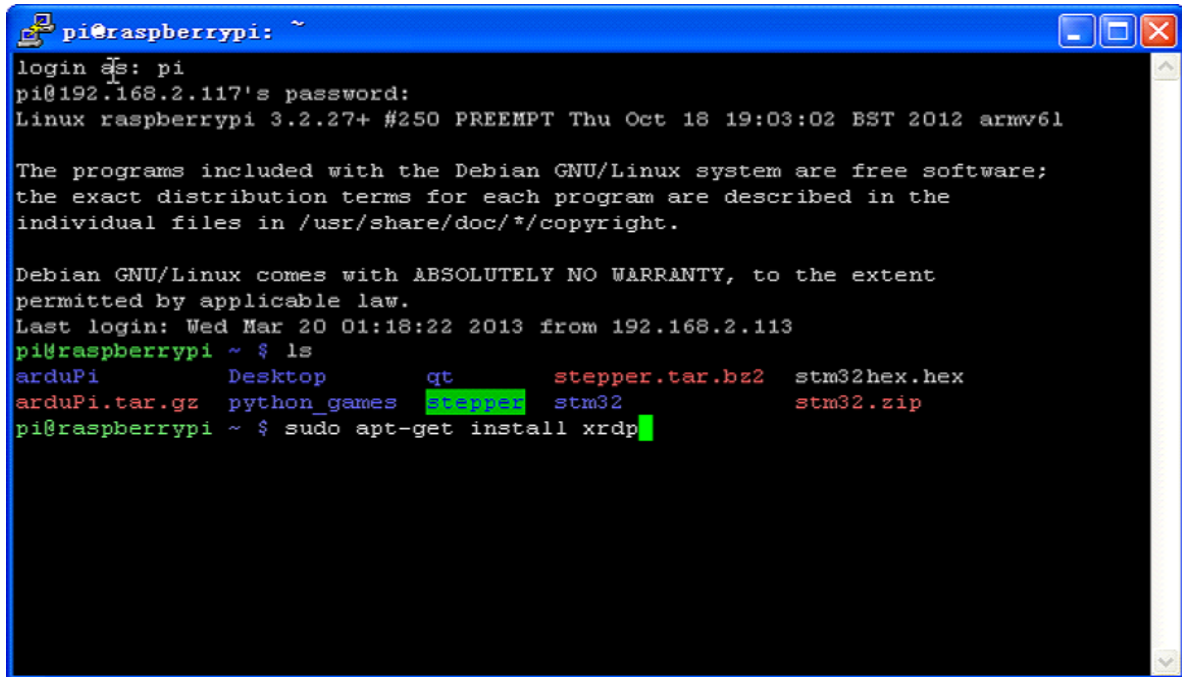
file of the Raspbian OS to SD card and the steps of starting the Raspberry Pi, we do not introduce the detail here.

Raspberry Pi can connect to a display via HDMI interface, it can also control remote login through the network. Considering the ease of operating, here we adopt the method of the remote control. The first step is to install the PuTTY software on Windows by its default configuration, after the success of the installation, click start -> programs -> PuTTY -> the PuTTY, the startup screen as follows, then input the Raspberry Pi's default IP address "192.168.2.110" in the "Host Name (or IP address)", if the Raspberry Pi connects to a computer directly, it's necessary to set the computer's IP address as 192.168.2.XXX, XXX can be arbitrary value in the range of 1 ~ 254, but not the same as the Raspberry Pi's IP. Then click on the "Open".



If it pops up a safety warning, select "Y".

Then input the login user name "pi" and the password "raspberrypi" in the pop-up interface, press enter, it will enter into the Raspberry Pi control terminal, execute the command "sudo apt-get install XRDP", the system will install the XRDP.



```
pi@raspberrypi: ~  
login as: pi  
pi@192.168.2.117's password:  
Linux raspberrypi 3.2.27+ #250 PREEMPT Thu Oct 18 19:03:02 BST 2012 armv6l  
  
The programs included with the Debian GNU/Linux system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*/copyright.  
  
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent  
permitted by applicable law.  
Last login: Wed Mar 20 01:18:22 2013 from 192.168.2.113  
pi@raspberrypi ~ $ ls  
arduPi      Desktop      qt           stepper.tar.bz2  stm32hex.hex  
arduPi.tar.gz  python_games  stepper      stm32            stm32.zip  
pi@raspberrypi ~ $ sudo apt-get install xrdp
```

3.2 Demonstrate The Ultrasonic Ranging Module

1) Download the "RasPi.Tar.bz2" files and unzip, compiler and execute the "make" command, then it will generate an executable file

```
$tar -xjvf RasPi.tar.bz2
```

```
$cd RasPi.tar.bz2
```

```
$make
```

2) Input "\$sudo ./main <CR>" in the console to run the program, it will initialize the stepper motor by default and configure the speed for indicating the sensor's value. Then select the demo you need in the main menu and input the corresponding numbers into the corresponding Demo program, select "0", return to the console.


```
File Edit Tabs Help
pi@raspberrypi ~/mcy/proj $ ls
arduPi.cpp arduPi.h main.cpp Makefile
pi@raspberrypi ~/mcy/proj $ make
sudo g++ -lrt -lpthread arduPi.cpp main.cpp -o main
pi@raspberrypi ~/mcy/proj $ ls
arduPi.cpp arduPi.h main main.cpp Makefile
pi@raspberrypi ~/mcy/proj $ sudo ./main
motor init is ok
motor speed is ok
Select the demo you want to run
1. Ultrasonic Wave Demo
2. Gravity Sensor Demo
3. Magnetic Field Sensor Demo
4. AD Measurement Demo
0. Quit
```

3) Input "1", then it will enter into the ultrasonic wave process and appear the run menu for "Ultrasonic wave".

1:Init ultrasonic wave 2:Start run 3:Stop run 0:Back to top

For any module, before running the test, it is needed to perform initialized command, so the first step is initialization:

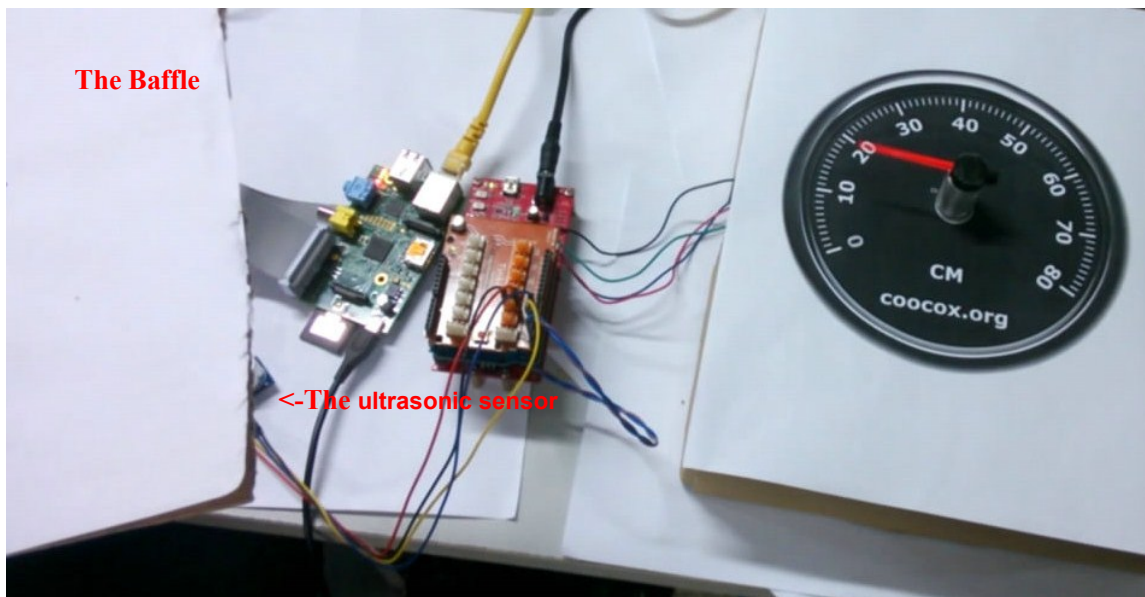
Input "1" to initialize the Ultrasonic wave, after the initialization is completed, the menu will pop up again to prompt the next step. Then use a big baffle to cover the probe of the ultrasonic range sensor, the distance between baffle and the probe is not less than 1 cm, otherwise the measurement result are inaccurate if the ultrasonic module cannot receive echo; meanwhile, the distance is not more than 3 cm. Next, keep the height of the baffle and move the dial pointer to zero by hand. Then input "2" on the command line to start distance measurement. At this point, the baffle can be moved up and down, the RasPi will continuously display the distance between the ultrasonic range module and the baffle. At the same time, the control motor is rotating accordingly to show the approximate distance.

```
pi@raspberrypi ~/mcy/proj $ sudo ./main
motor init is ok
motor speed is ok
Select the demo you want to run
1. Ultrasonic Wave Demo
2. Gravity Sensor Demo
3. Magnetic Field Sensor Demo
4. AD Measurement Demo
0. Quit
1
1: Init ultrasonic wave 2: Start run 3: Stop run 0: Back to top
1
ultrasonic wave init ok
1: Init ultrasonic wave 2: Start run 3: Stop run 0: Back to top
2
MeasureValue =15 angle = 90
MeasureValue =15 angle = 90
MeasureValue =15 angle = 90
MeasureValue =15 angle = 90
```

In theory, the furthest distance of the module is 4 m. For demonstration only, here we ignore the values that exceeds 50cm, so if the distance between the baffle and the module is more than 50cm the pointer will stop.

- Input "3", stop measuring the distance, it will pop up the menu, users can input again. When the program is running distance, you can click on the "3" on keyboard to stop running the program.
- Input "0", return to the upper directory and start again.

The following shows the actual results:



Input "q", it will return to the directory for selecting the demo unconditionally, then the user can choose to enter another demo program.

```
MeasureValue =15  angle = 90
MeasureValue =15  angle = 90
MeasureValue =15  angle = 90

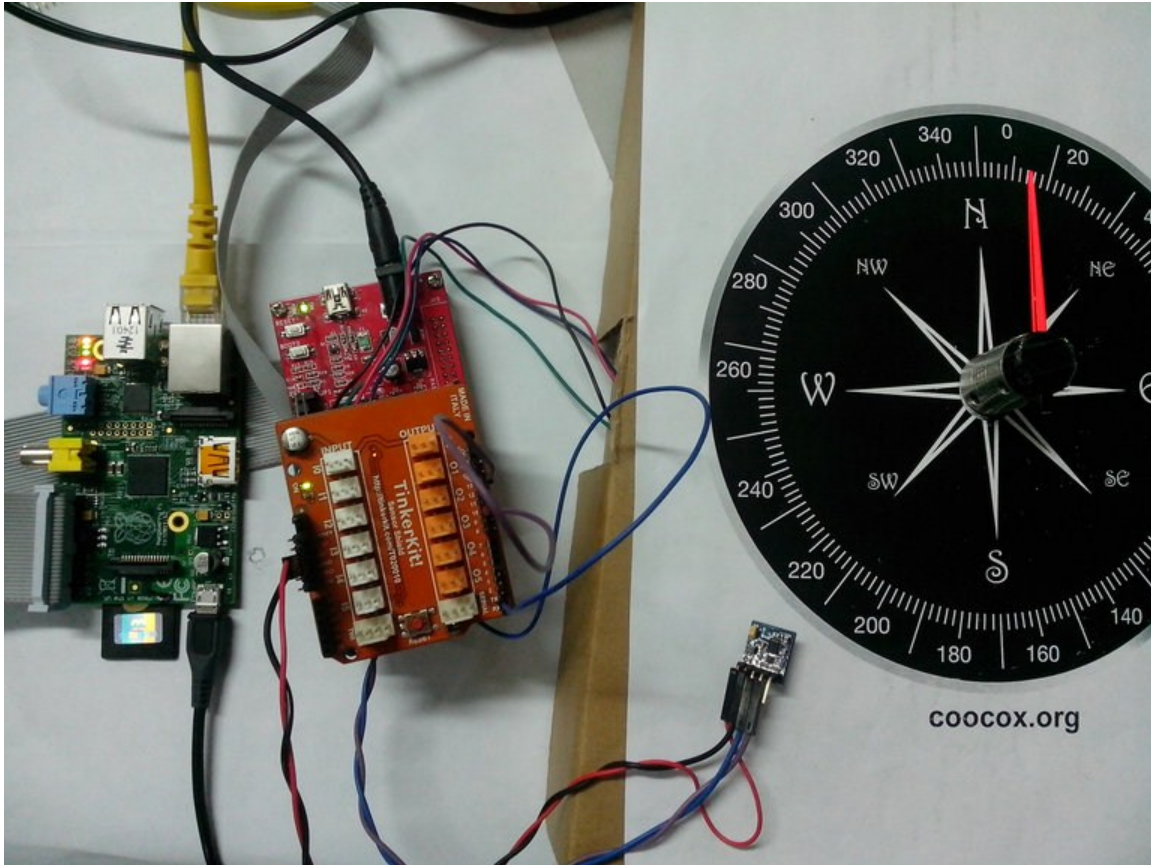
pressed
3 pressed
ultrasonic wave stop ok
1: Init ultrasonic wave  2: Start run  3: Stop run  0: Back to top
0
Select the demo you want to run
1. Ultrasonic Wave Demo
2. Gravity Sensor Demo
3. Magnetic Field Sensor Demo
4. AD Measurement Demo
0. Quit
█
```

3.3 Demonstrate The Three Triaxial magnetic Field Sensor

The first two steps are the same as chapter 3.1, including initialization, start, stop, and return to the upper menu. The third step is basically the same, the only difference exists in the content of the menu. We use the magnetic field sensor's X-axis and Y-axis to measure the angle between the Y-axis and the geomagnetic field. After the initialization, keep the magnetic field sensor flat (reduce errors), try to make the Y-axis parallel to the north and south direction, then select "2" to start the measurement.

```
Select the demo you want to run
1.Ultrasonic Wave Demo
2.Gravity Sensor Demo
3.Magnetic Field Sensor Demo
4.AD Measurement Demo
0.Quit
3
1:Init MR  2:Start MR  3:Stop MR  0:Back to top
1
Magnetic Field Sensor init ok
1:Init MR  2:Start MR  3:Stop MR  0:Back to top
2
```

Keep the magnetic field sensor flat, rotate the sensor left and right, you can see the pointer is turning at the same time, as shown in the following figure:



3.4 Demonstrate The ADC voltage Acquisition

1) The first two steps are the same as the chapter 3.1, after choosing the AD testing example, it will prompt the user to input the number of pin which is connected with the AD. When doing the test, we connect the variable end of the potentiometer to the I3 on the TinkerKit, so here input "3".

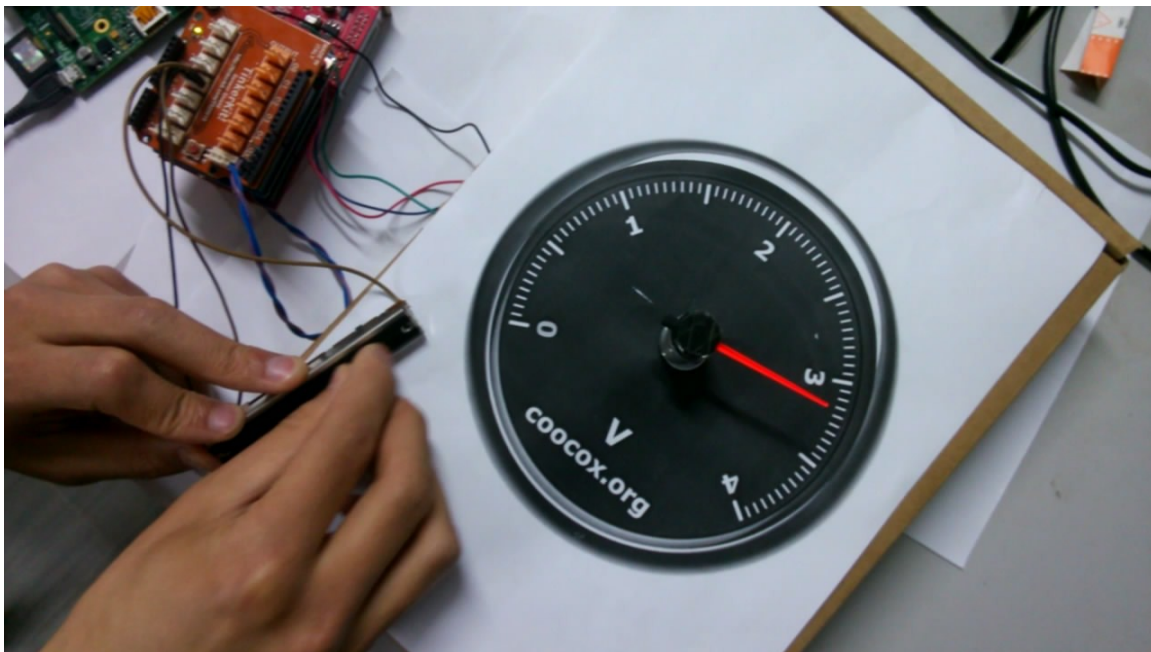
Note that despite the TinkerKit has six input ports (I0~I5), the ports connect to the STM32 ADC port are I0~I3 actually, so I4 and I5 have no ADC function, they can only be used as a TTL level input.

2) Firstly, slide the potentiometer to the ground terminal, at this time the output voltage is "0", then enter "23" to start the ADC conversion.

```
pi@raspberrypi ~/mcy/proj $ sudo ./main
motor init is ok
motor speed is ok
Select the demo you want to run
1.Ultrasonic Wave Demo
2.Gravity Sensor Demo
3.Magnetic Field Sensor Demo
4.AD Measurement Demo
0.Quit
4
You should enter the analog input pin number first
3
1:Change AIN pin number and init  2:Start AD  3:Stop AD  0:Back to top
2
```

After starting the console, it will show the ADC values and control the rotation of the motor according to the sampling data. At this movement, slid the potentiometer and the pointer will point to the current AD sampling voltage.

The actual result as follow:



Note: if the EPI have been reset, the program of the Ras-Pi also have to be performed again, otherwise the motor will not be able to work without initialization and other modules cannot run at the same time. The second demonstration in the Demo menu is gravity sensor, since the steps are very similar to the triaxial magnetic field sensor in chapter 3.3, here we won't describe the item in this demo.