Dual Digital Pot (100K) SKU: DFR0520

Introduction

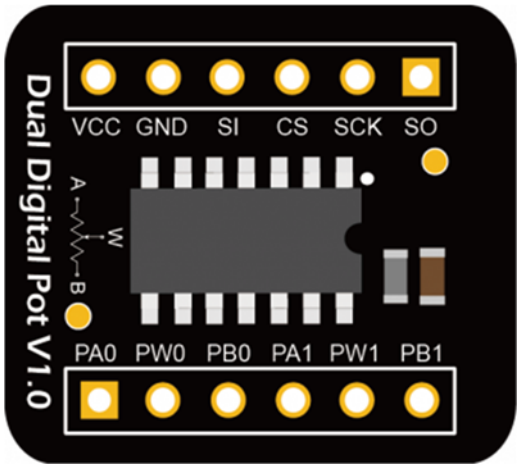
Digital potentiometer is also called "Digital Pot" in short. It is a kind of mixed signal IC, which is able to dynamically change the internal resistors through MCU like Arduino. Compared to the traditional mechanical potentiometer, the digital pot features as flexible (program control), small size (ICs) and high reliability (without mechanical parts). It can replace the tradition one in many applications. Digital pot is usually used to change the sound volume in audio devices, such as smart loudspeaker, cell phone, and music player. In addition, with a proper design op-amp circuit, the digital pot can also be applied to change some key parameters of the circuit dynamically, such as LED DC dimming (output current), linear stable voltage source (output voltage), oscillator (frequency and amplitude), low pass filter (bandwidth) and differential amplifier (gain).

This breakout employs MCP42100 internally manufactured with two individual 100K digital pot POT0 and POT1. Each pot has 256 taps with a resistor of 100KΩ. It supports wide voltage supply (DC 2.7V - 5.5V) compatible with MCU of 3.3V and 5V. The breakout features as small size (20.0mm*18.0mm) and reserves the SO pin for multiple breakouts being configured in daisy-chain connection. If you have a I/O shield in hand, this breakout can be easily connected to it with the attached 5 pin male to male cable.

Specification

- Supply Voltage: 2.7～5.5V DC
- Static Operation Current: < 1 μA
- Potentiometer Value: 100 KΩ
- Resolution: 8 bits, 256 taps for each potentiometer
- Number of Potentiometers: 2
- Interface: SPI
- Operation Temperature: -40℃～85℃
- Dimension: 20.0mm*18.0mm

Pin Definition



Dual Digital Pot（100K）Pin Definition

Pin	Description
VCC	Power supply (DC 2.7V - 5.5V)
GND	Ground
SI	Serial data input
CS	Chip select
SCK	Serial clock input
SO	Serial data output
PAx	Potentiometer terminal A (x=0,1)
PBx	Potentiometer terminal B (x=0,1)
PWx	Potentiometer wiper terminal (x=0,1)

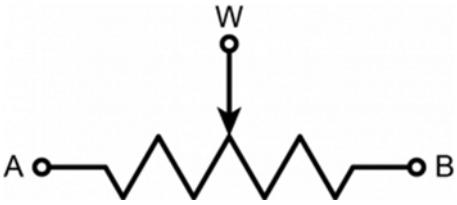
1. Resistor terminals A, B and W have no restrictions on polarity with respect to each other.
2. Current through terminals A, B and W should not excceed ±1mA.
3. Voltages on terminals A, B and W should be within 0 - VCC.

Tutorial

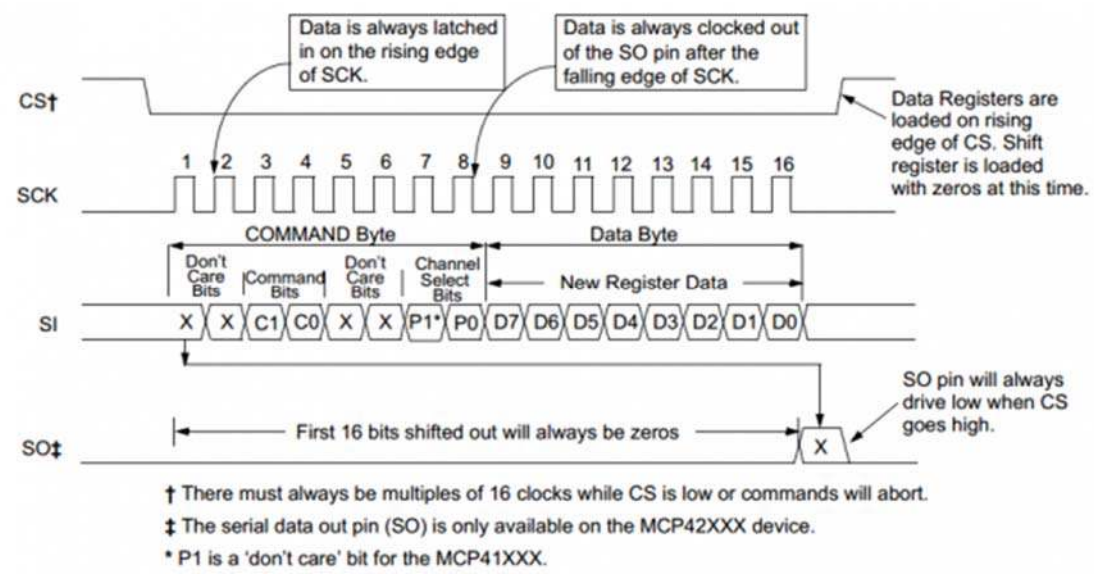
This tutorial generates two triangular waves out of phase by the dual digital pot to demonstrate its basic usage. The two internal digital pot POT0 and POT1 server as voltage divider with terminal A connected to VCC and terminal B connected to GND. We use Arduino UNO R3 to control the terminal W0 of POT0 to change it from min to max (Dn: 0 -> 255) every 1 ms and then in reverse, from max to min (Dn: 255 -> 0). On the contrary, the terminal W1 of POT1 will be changed from max to min (Dn: 255 -> 0) every 1 ms and then in reverse, from min to max (Dn: 0 -> 255). Two channels CH1 and CH2 of the oscilloscope will be used to observe the voltage of W0 and W1 respectively to check whether all the possible taps are available.

Basic Principle

This breakout employs MCP42100, which has two individual digital potentiometer POT0 and POT1 corresponding to two groups of terminals A0, B0, W0 and A1, B1, W1. Similar to the mechanical potentiometer, terminal A and B can be taken as two pins of a resistor (the nominal resistance is Rab=100KΩ) while terminal W is the wiper. The wiper can be changed to one of the 256 positions evenly distributed between A and B. The wiper is reset to the mid-scale position (Dn=128,0x80) upon power-up.



To change the position of the wiper W, two byte should be sent. The first byte is the command to determine which pot to be selected. The second byte is the data to determine the position of the wiper. This byte is also denoted as Dn. When Dn=0, terminal B is connected to W. When Dn=255, W is changed to the closest position to A. For example, to set W0 of POT0 to the position of 100, Arduino should first sends 0x11 (Write data, select POT0) and then 0x64 (=100, decimal) through the SPI.



COMMAND BYTE

X	X	C1	C0	X	X	P1*	P0

Command Selection Bits — Potentiometer Selection Bits

C1	C0	Command	Command Summary
0	0	None	No Command will be executed.
0	1	Write Data	Write the data contained in Data Byte to the potentiometer(s) determined by the potentiometer selection bits.
1	0	Shutdown	Potentiometer(s) determined by potentiometer selection bits will enter Shutdown Mode. Data bits for this command are 'don't cares'.
1	1	None	No Command will be executed.

P1*	P0	Potentiometer Selections
0	0	Dummy Code: Neither Potentiometer affected.
0	1	Command executed on Potentiometer 0.
1	0	Command executed on Potentiometer 1.
1	1	Command executed on both Potentiometers.

- The resistance of each digital pot can be calculate by the equations below.

$$R_{WA}(D_n) = \frac{R_{AB}(256 - D_n)}{256} + R_W$$

$$R_{WB}(D_n) = \frac{R_{AB} D_n}{256} + R_W$$

D_n - 8 - bit data of wiper position (0 - 255)

R_W - wiper resistance (= 125 Ω, typical)

R_{WA} - resistance between terminal A and wiper

R_{WB} - resistance between terminal B and wiper

R_{AB} - resistance between terminal A and B (= 100 KΩ, typical)

Requirements

- **Hardware**
  DFRduino UNO Mainboard (or similar) x 1
  Bread board x 1
  IO Expansion Shield for Arduino V7.1 (optional) x 1
  Analog Cable (5Pin male to male) x 1

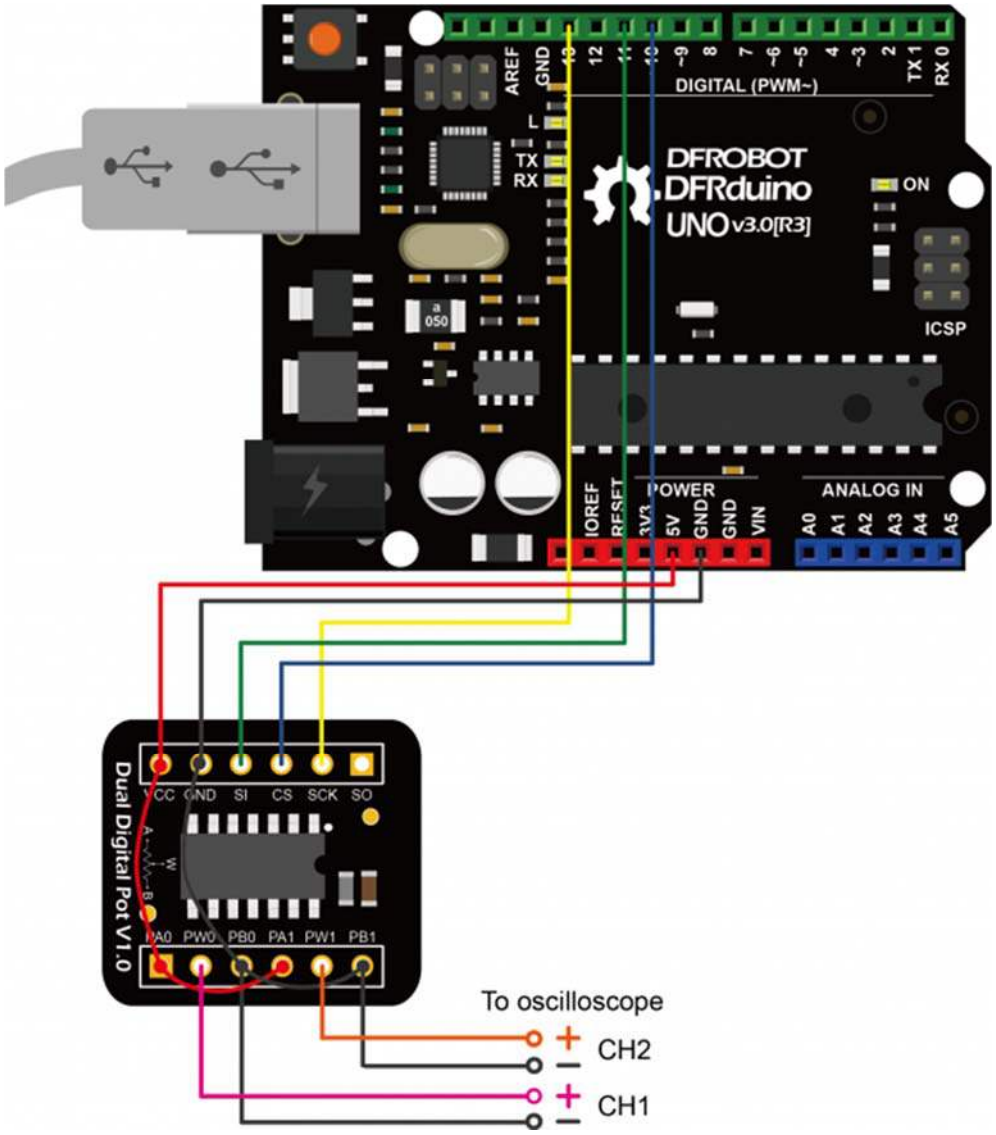
- **Software**

  Arduino IDE (Version requirements: V1.8.x), Click to Download Arduino IDE from Arduino®

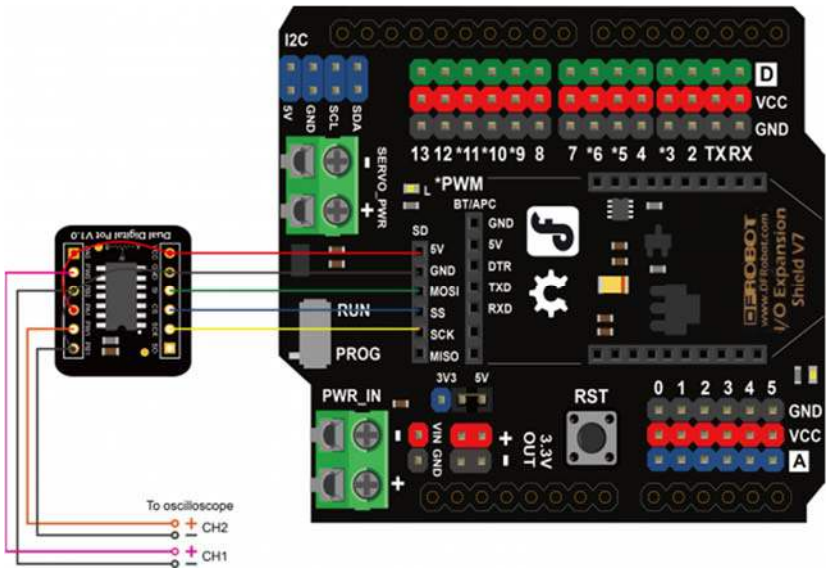
  https://www.arduino.cc/en/Main/Software%7C

Hardware Connection

The module can be connected to Arduino with the attached 5 pin male-to-male cable (one end bonded together and the other end separated). Insert the breakout into the breadboard and plug the cable with the bonded end to the breadboard where pin VCC, GND, SI, CS, SCK lie on (leave pin SO unconnected). The individual end inserts into Arduino shown as follow.



Arduino UNO Connection Diagram



IO Expansion Shield V7.1 Connection Diagram

Sample Code

If the IO Expansion Shield is used for connection, the sentence "const int CS_PIN = 10;" should be change to "const int CS_PIN = 4;" in the sample code. Because the SS pin in the SPI interface of the shield connects to D4 internally.

```
/*************************************************************

    Dual Digital Pot(100K)

     <https://www.dfrobot.com/wiki/index.php/Dual_Digital_Pot_(100K)_SKU:_DFR
0520>

 *************************************************************

   This example generates two triangular waves to demonstrate

   the basic usage of dual digital pot.


   Created 2017-8-31

   By Henry Zhao <Henry.zhao@dfrobot.com>


   GNU Lesser Genral Public License.

   See <http://ww.gnu.org/licenses/> for details.

   All above must be included in any redistribution.

 *************************************************************/
```

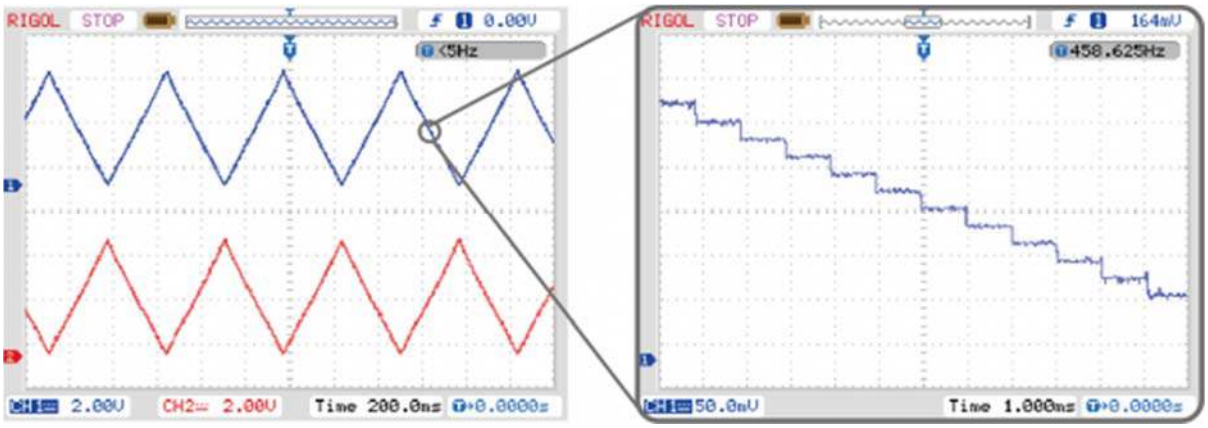
```
/**********************Circuit Connections*******************
      Digital Pot  |   Arduino UNO R3    |   Oscilloscope
          CS       |     D10 (SS)        |
          SI       |     D11 (MOSI)      |
          CLK      |     D13 (SCK)       |
      VCC,PA0,PA1  |       VCC           |
      GND,PB0,PB1  |       GND           |     CH1- CH2-
          W0       |                     |       CH1+
          W1       |                     |       CH2+
 ***********************************************************/

/**********************Notice*********************************
   1.Resistor terminals A, B and W have no restrictions on
     polarity with respect to each other.
   2.Current through terminals A, B and W should not excceed ±1mA.
   3.Voltages on terminals A, B and W should be within 0 - VCC.
************************************************************/

#include <SPI.h>

/**********************PIN Definitions************************/
// set pin 10 as the slave select for the digital pot:
const int CS_PIN = 10;

/**********************MCP42XXX Commands***********************/
//potentiometer select byte
const int POT0_SEL = 0x11;
const int POT1_SEL = 0x12;
const int BOTH_POT_SEL = 0x13;

//shutdown the device to put it into power-saving mode.
//In this mode, terminal A is open-circuited and the B and W terminals are sh
orted together.
//send new command and value to exit shutdowm mode.
```

```
const int POT0_SHUTDOWN = 0x21;
const int POT1_SHUTDOWN = 0x22;
const int BOTH_POT_SHUTDOWN = 0x23;


/**********************Customized Varialbes**********************/
//resistance value byte (0 - 255)
//The wiper is reset to the mid-scale position upon power-up, i.e. POT0_Dn = POT1_Dn = 128
int POT0_Dn = 128;
int POT1_Dn = 128;
int BOTH_POT_Dn = 128;


//Function Declaration
void DigitalPotTransfer(int cmd, int value);     //send the command and the wiper value through SPI


void setup()
{
  Serial.begin(115200);
  pinMode(CS_PIN, OUTPUT);   // set the CS_PIN as an output:
  SPI.begin();     // initialize SPI:
}


void loop()
{
      // change the resistance on the POT0 from min to max:
    for (int POT_Dn = 0; POT_Dn < 256; POT_Dn++) {
      DigitalPotWrite(POT0_SEL, POT_Dn);
      delay(1);
    }


    // change the resistance on the POT0 from max to min:
    for (int POT_Dn = 0; POT_Dn < 256; POT_Dn++) {
      DigitalPotWrite(POT0_SEL , 255 - POT_Dn);
      delay(1);
```

```
    }
}


void DigitalPotWrite(int cmd, int val)
{
  // constrain input value within 0 - 255
  val = constrain(val, 0, 255);
  // set the CS pin to low to select the chip:
  digitalWrite(CS_PIN, LOW);
  // send the command and value via SPI:
  SPI.transfer(cmd);
  SPI.transfer(val);
  // Set the CS pin high to execute the command:
  digitalWrite(CS_PIN, HIGH);
}
```

Experiment Results

Two triangular waves out of phase can be observed from the oscilloscope. If we zoom in a section of the wave, the triangular wave is actually made up of many steps. Each steps correspond to one wiper position. The width of the step is about 1ms, because the program delay for 1ms between every wiper changes. The upstairs half cycle and the downstairs half cycle consist of 256 steps each, therefore the period of the triangular wave is 256*2=512ms. The digital pot can quickly change the wiper position and the settling time can up to micro seconds (18μs, typical).



Experiment Results

Template Code for MCP42100

A template code is provided here for user to learn how to control the MCP42100 dual digital pot.

```
/*************************************************************
    Dual Digital Pot (100K)
     <https://www.dfrobot.com/wiki/index.php/Dual_Digital_Pot_(100K)_SKU:_DFR
0520>
 *************************************************************
   This example serves as a template to control the MCP42100 dual
   digital pot through 3-wire SPI.

   Created 2017-8-31
   By Henry Zhao <Henry.zhao@dfrobot.com>

   GNU Lesser Genral Public License.
   See <http://ww.gnu.org/licenses/> for details.
   All above must be included in any redistribution.
 *************************************************************/

/*******************Device Inctrduction**********************
   The MCP42100 has dual potentiometer x (x=0,1).
     Ax - Potenriometer terminal Ax
     Wx - Potenriometer Wiper
     Bx - Potenriometer terminal Bx

     SI - Serial Data Input
     SCK - Serial Clock
     CS - Chip Select

     The MCP42100 is SPI-compatible,and two bytes should be sent to control i
t.
     The first byte specifies the potentiometer (POT0: 0x11, POT1: 0x12, both
: 0x13).
     The second byte specifies resistance value for the pot (0 - 255).
 *************************************************************/
```

```
/**********************Circuit Connections*******************
     Digital Pot | Arduino UNO R3
         CS      |    D10 (SS)
         SI      |    D11 (MOSI)
         CLK     |    D13 (SCK)
         VCC     |      VCC
         GND     |      GND
 ************************************************************/

/**********************Resistances Calculation**************
   Rwa(Dn) =Rab*(256 - Dn) / 256 + Rw
   Rwb(Dn) =Rab*Dn / 256 + Rw

   Rwa - resistance between Terminal A and wiper W
   Rwb - resistance between Terminal B and wiper W
   Rab - overall resistance for the pot  (=100KΩ, typical)
   Rw - wiper resistance  (=125Ω,typical; =175Ω max)
   Dn - 8-bit value in data register for pot number n  (= 0 - 255)
 ************************************************************/

/**********************Notice*********************************
   1.Resistor terminals A, B and W have no restrictions on
     polarity with respect to each other.
   2.Current through terminals A, B and W should not excceed ±1mA.
   3.Voltages on terminals A, B and W should be within 0 - VCC.
************************************************************/

#include <SPI.h>

/**********************PIN Definitions************************/
// set pin 10 as the slave select for the digital pot:
const int CS_PIN = 10;
```

```
/***********************MCP42XXX Commands***********************/

//potentiometer select byte

const int POT0_SEL = 0x11;

const int POT1_SEL = 0x12;

const int BOTH_POT_SEL = 0x13;


//shutdown the device to put it into power-saving mode.

//In this mode, terminal A is open-circuited and the B and W terminals are sh
orted together.

//send new command and value to exit shutdowm mode.

const int POT0_SHUTDOWN = 0x21;

const int POT1_SHUTDOWN = 0x22;

const int BOTH_POT_SHUTDOWN = 0x23;


/***********************Customized Varialbes***********************/

//resistance value byte (0 - 255)

//The wiper is reset to the mid-scale position upon power-up, i.e. POT0_Dn =
POT1_Dn = 128

int POT0_Dn = 128;

int POT1_Dn = 128;

int BOTH_POT_Dn = 128;


//Function Declaration

void DigitalPotTransfer(int cmd, int value);     //send the command and the r
esistance value through SPI


void setup()

{

  pinMode(CS_PIN, OUTPUT);    // set the CS_PIN as an output:

  SPI.begin();     // initialize SPI:

  DigitalPotWrite(BOTH_POT_SHUTDOWN, BOTH_POT_Dn);

}


void loop()

{
```

```
  DigitalPotWrite(POT0_SEL, POT0_Dn);                 //set the resistance of
POT0
  DigitalPotWrite(POT1_SEL, POT1_Dn);                 //set the resistance of
POT1
  //DigitalPotWrite(BOTH_POT_SEL, BOTH_POT_Dn);         //set the resistance
of both potentiometers


  //DigitalPotWrite(POT0_SHUTDOWN, POT0_Dn);              //put POT0 into shu
ntdowm mode, ignore the second parameter
  //DigitalPotWrite(POT1_SHUTDOWN, POT1_Dn);              //put POT1 into shu
ntdowm mode, ignore the second parameter
  //DigitalPotWrite(BOTH_POT_SHUTDOWN, BOTH_POT_Dn);      //put both potentio
meters into shuntdowm mode, ignore the second parameter


}


void DigitalPotWrite(int cmd, int val)
{
  // constrain input value within 0 - 255
  val = constrain(val, 0, 255);
  // set the CS pin to low to select the chip:
  digitalWrite(CS_PIN, LOW);
  // send the command and value via SPI:
  SPI.transfer(cmd);
  SPI.transfer(val);
  // Set the CS pin high to execute the command:
  digitalWrite(CS_PIN, HIGH);
}
```

FAQ

For any questions, advice or cool ideas to share, please visit the **DFRobot Forum**.