Rotary Switch Module V1 (SKU:SEN0156)

Contents

- 1 Introduction
- 2 Specifications
- 3 How to Change Position Number
- 4 Wiring Diagram
- 5 Sample Code 1
- 6 Sample Code 2

Introduction

A rotary switch is a switch operated by rotation. These are often chosen when more than 2 positions are needed, such as multi-speed control. The Rotary Switch uses an analog input to read the 12 state which saves IO resource for the Arduino. This Rotary Switch Module has most 12 position. Each position has the approprite indicator.You can also adjust metal buckle inside the switch to change position number.So, it's very flexible for you to use.

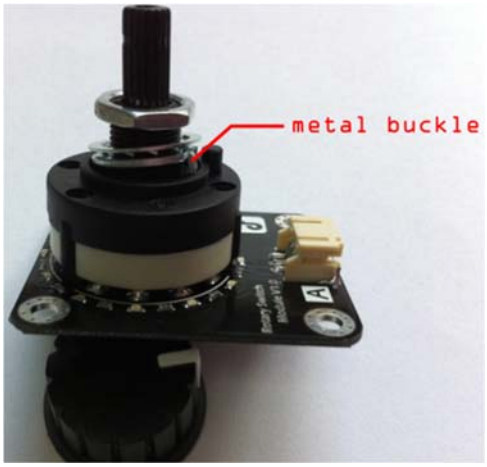
A rotary switch consists of a spindle or "rotor" that has a contact arm or "spoke" which projects from its surface like a cam. It has an array of terminals, arranged in a circle around the rotor, each of which serves as a contact for the "spoke" through which any one of a number of different electrical circuits can be connected to the rotor. Thus a rotary switch provides greater pole and throw capabilities than simpler switches do.

Specifications

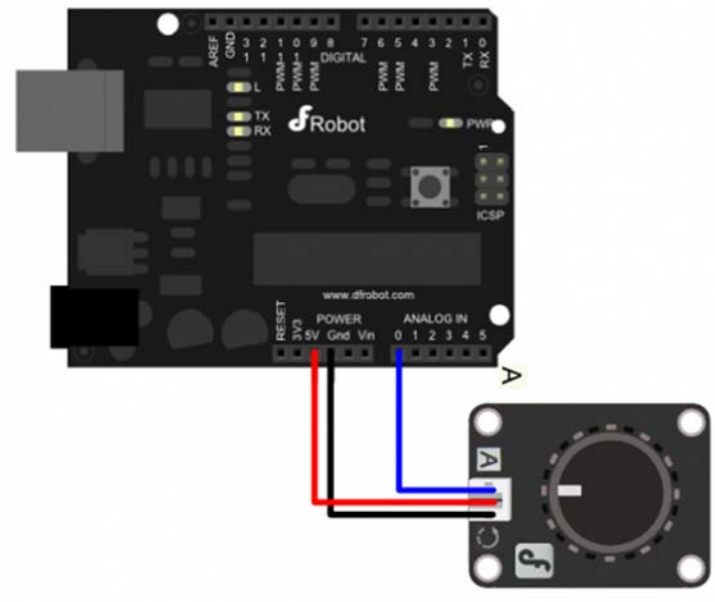
- Supply voltage: 3.3-12V
- Interface: Analog
- Size: 43mm*38mm

How to Change Position Number

Regulating metal buckle among differrnt fixing hole can change position number.



Wiring Diagram



Sample Code 1

```
int sensorvalue;


void setup(){
    Serial.begin(9600);
}
void loop(){
    sensorvalue = analogRead(0);            // read value from different posi
tion
    Serial.println(sensorvalue);
    delay(200);
}
```

Sample Code 2

The same position value of each switch is different.So, by sample code 1, the each position value can be read, and modify the appropriate reference value in the following code.

```
/*
 # This Sample code is reading Switch position.
 */
int adc_key_val[12] ={630,680,750,810,845,860,890,905,920,940,950,980};
int NUM_KEYS = 12;
int adc_key_in;
int key=-1;
int oldkey=-1;
void setup()
{
  Serial.begin(9600);                  // 9600 bps
}
void loop()
{
  adc_key_in = analogRead(0);            // read the value from the sensor
  key = get_key(adc_key_in);             // convert into position
```

```
if (key != oldkey)                    // if a position is detected
 {
  delay(50);  // wait for debounce time
  adc_key_in = analogRead(0);         // read the value from the sensor
  key = get_key(adc_key_in);          //convert into position
  if (key != oldkey)
  {
    oldkey = key;
    if (key >=0){
      switch(key)
      {
         case 0:Serial.println("S1 OK");
                break;
         case 1:Serial.println("S2 OK");
                break;
         case 2:Serial.println("S3 OK");
                break;
         case 3:Serial.println("S4 OK");
                break;
         case 4:Serial.println("S5 OK");
                break;
         case 5:Serial.println("S6 OK");
                break;
         case 6:Serial.println("S7 OK");
                break;
         case 7:Serial.println("S8 OK");
                break;
         case 8:Serial.println("S9 OK");
                break;
         case 9:Serial.println("S10 OK");
                 break;
         case 10:Serial.println("S11 OK");
                break;
         case 11:Serial.println("S12 OK");
```

```
                  break;
        }
      }
    }
  }
 delay(100);
}
// Convert ADC value to key number
int get_key(unsigned int input)
{
    int k;
    for (k = 0; k < NUM_KEYS; k++)
    {
      if (input < adc_key_val[k])
     {
            return k;
        }
   }
       if (k >= NUM_KEYS)k = -1;
       return k;
}
```