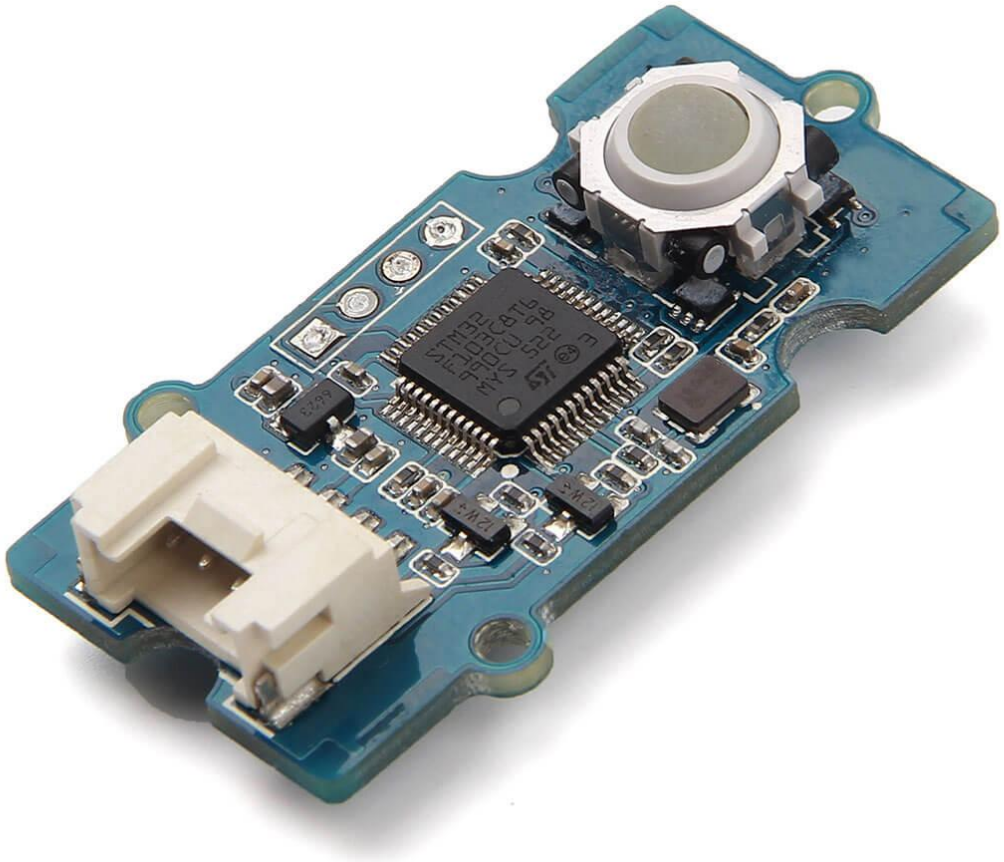
# Grove - Mini Track Ball

Table of contents



Grove - Mini Track ball will give an easy access to prototyping a practical motion-tracking function module for your applications. It has implanted 360° detection and click detection with high accuracy and quick response. With chips **STM32F103C8T6** and **AN48841B** inside, you can turn plenty of your ideas into tangible things. It is also standardized with Grove interface which will save you a lot of work in the prototyping process.

## Features

- 360° and quick detection.
- Translucent click Button.
- Standardized with Grove interface.
- Powerful MCU for you to enrich your applications.

**Tip**
More details about Grove modules please refer to Grove System

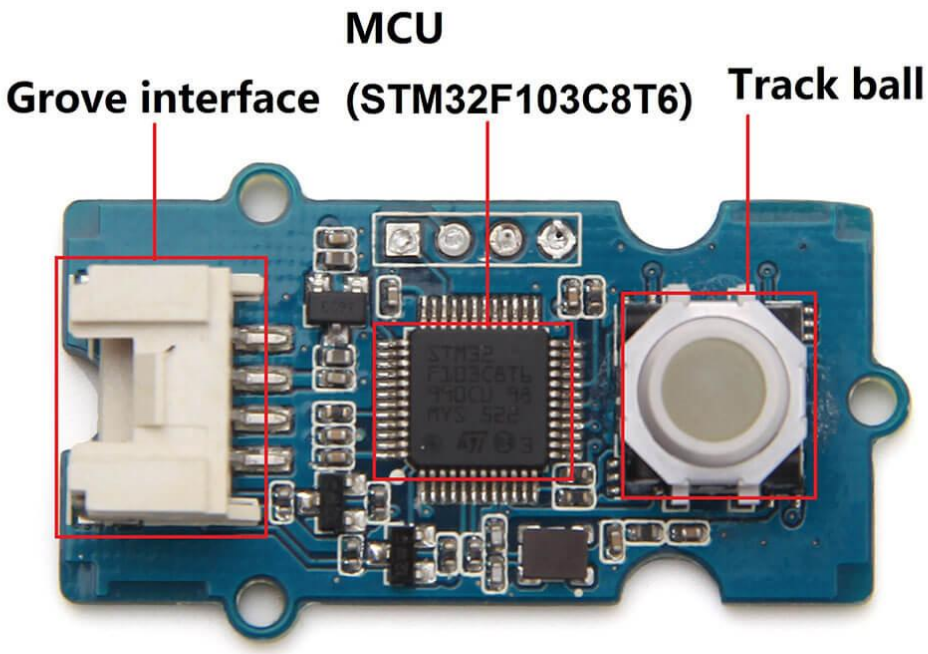
## Application ideas

- Tracking module for a gamepad.
- Tracking module for a haptic controller.
- Tracking module for toys.

## Specifications

| Parameter | Value |
| --- | --- |
| Operating voltage | 3.3V~5.5V (typical at 5V) |
| Operating current | 28 mA (maximum operating current: 40 mA) |
| Operating temperature range | -25 ~ 75 ℃ |
| MCU frequency | 64 MHz |
| Operating frequency | 105±5kHz |
| Hall effect filed strength range | (0.5) ~ (8) mT |
| I2C Address | 0x4A |

## Hardware Overview




- **Grove interface**
  Connect main control board such as **Seeeduino** board with Grove - Mini Track Ball.
- **MCU (STM32F103C8T6)**
  Microcontroller.

- **Track ball**
  Interface to control motions.

# Getting started

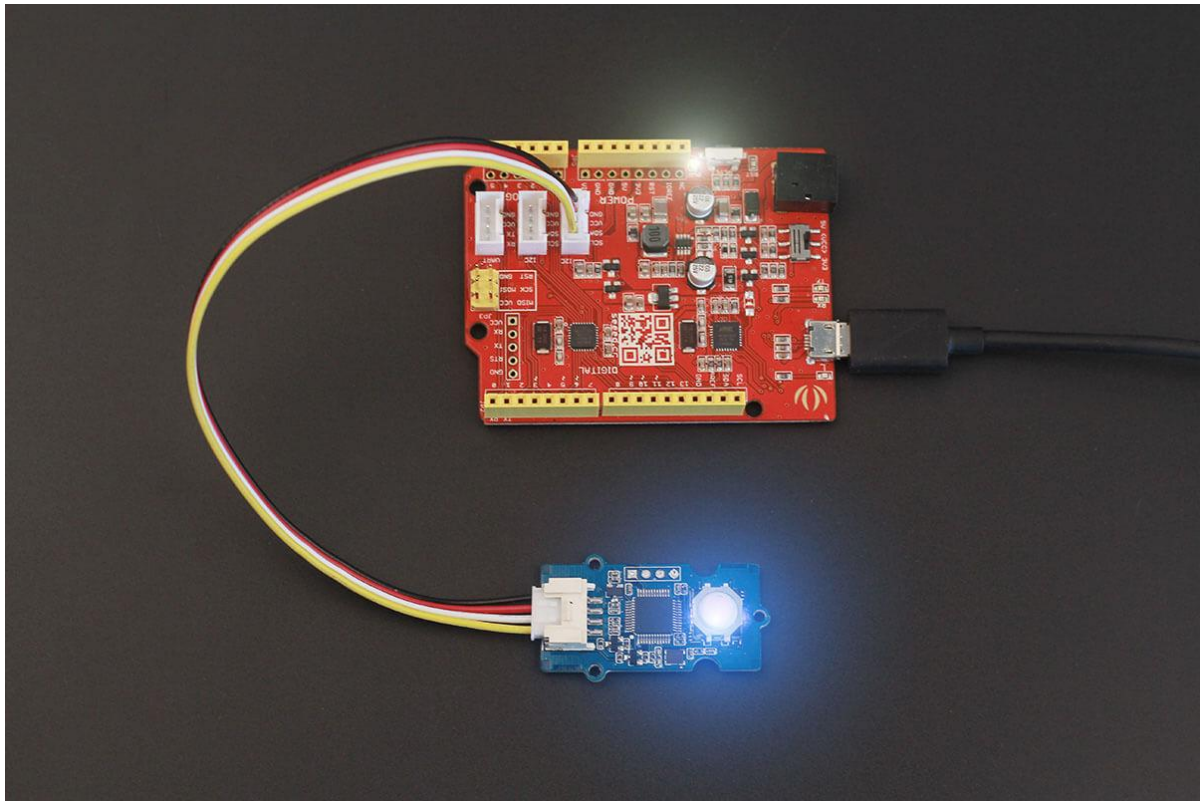
## Play with Arduino

### Hardware

- **Step 1.** Prepare the below stuffs:

| Seeeduino V4.2 | Grove-Mini_Track_Ball |
| --- | --- |
|  |  |
| Get One Now | Get One Now |

- **Step 2.** Connect Grove-Mini_Track_Ball to **I2C** port of Seeeduino.
- **Step 3.** Connect Seeeduino to PC via a USB cable.



**Software**

**Step 1.** Download the demo from Github.

**Step 2.** Open the file **Grove - Mini Track ball test.ino**

```
#include <Wire.h>

/*-----------------------------------------------------------------------
```

```
 * define the default data
 */
#define ReadMode 0
#define WriteMode 1
#define DeAddr 0X4A
#define ConfigValid 0x3a6fb67c


/*-------------------------------------------------------------------------
 * define the enum type for Register
 */
enum MOTION_REG_ADDR
{
  MOTION_REG_UP = 0X00,
  MOTION_REG_DOWN,
  MOTION_REG_LEFT,
  MOTION_REG_RIGHT,
  MOTION_REG_CONFIRM,
  MOTION_REG_NUM
};

enum CONFIG_REG_ADDR
{
  CONFIG_REG_VALID = MOTION_REG_NUM,
  CONFIG_REG_I2C_ADDR = CONFIG_REG_VALID + 4,
  CONFIG_REG_I2C_SPEED,
  CONFIG_REG_LED_MODE = CONFIG_REG_I2C_SPEED + 2,
  CONFIG_REG_LED_FLASH_TIME,
  CONFIG_REG_DATA_CLEAR_TIME = CONFIG_REG_LED_FLASH_TIME + 2,  //CONFIC
  CONFIG_REG_DATA_READ_TIME = CONFIG_REG_DATA_CLEAR_TIME + 2,
  CONFIG_REG_NUM = CONFIG_REG_DATA_READ_TIME + 2
};


/*-------------------------------------------------------------------------
```

```
 * define the LED word mode
 */
enum LED_MODE
{
  LED_FLASH_1 = 0X00,
  LED_FLASH_2,
  LED_FLASH_TOGGLE,
  LED_FLASH_ALL,
  LED_ALWAYS_ON_1,
  LED_ALWAYS_ON_2,
  LED_ALWAYS_ON_ALL,
  LED_ALWAYS_OFF,
  LED_BREATHING_1,
  LED_BREATHING_2,
  LED_BREATHING_ALL,
  LED_MOVE_FLASH,
  LED_MODE_NUM
};



/*-------------------------------------------------------------------------
 * Write one byte into register
 */
void WriteByte(uint8_t Reg, uint8_t Value)
{
    Wire.beginTransmission(DeAddr);
    Wire.write(WriteMode);
    Wire.write(Reg);
    Wire.write(Value);
    Wire.endTransmission();
}

/*-------------------------------------------------------------------------
 * Write N byte into register
```

```
 */
void WriteNByte(uint8_t Reg , uint8_t * Value , uint8_t len)
{
    Wire.beginTransmission(DeAddr);
    Wire.write(WriteMode);
    Wire.write(Reg);
    for(int i = 0;i<len;i++)
    {
      Wire.write(Value[i]);
    }
    Wire.endTransmission();
}

/*-------------------------------------------------------------------------
 * Write one word(4 bytes,32 bits) into register ,the register address mu
 */
void WriteOneWord(uint8_t Reg, uint32_t Value)
{
  uint8_t tmp[4]={0};
  tmp[0] = (Value>>0)&0XFF;
  tmp[1] = (Value>>8)&0XFF;
  tmp[2] = (Value>>16)&0XFF;
  tmp[3] = (Value>>24)&0XFF;
  WriteNByte(Reg,tmp,4);
}


/*-------------------------------------------------------------------------
 * Write half word(2 bytes,16 bits) into register ,the register address r
 */
void WriteHalfWord(uint8_t Reg, uint16_t Value)
{
  uint8_t tmp[2]={0};
  tmp[0] = (Value>>0)&0XFF;
  tmp[1] = (Value>>8)&0XFF;
```

```
109     WriteNByte(Reg,tmp,2);
110   }
111
112   /*-------------------------------------------------------------------------
113    * Read one byte from register
114    */
115   uint8_t ReadByte(uint8_t Reg)
116   {
117      Wire.beginTransmission(DeAddr);
118      Wire.write(ReadMode);
119      Wire.write(Reg);
120      Wire.write(1);
121      Wire.endTransmission();
122      Wire.requestFrom(DeAddr, 1);
123      return Wire.read();
124   }
125   /*-------------------------------------------------------------------------
126   * Read half word from register
127   */
128   uint16_t ReadHalfWord(uint8_t Reg)
129   {
130     uint16_t tmp;
131     tmp = ReadByte(Reg);
132     tmp |= ((uint16_t)ReadByte(Reg+1))<<8;
133     return tmp;
134   }
135   /*-------------------------------------------------------------------------
136   * Read one word from register
137   */
138   uint32_t ReadOneWord(uint8_t Reg)
139   {
140     uint32_t tmp;
141     tmp = ReadByte(Reg);
142     tmp |= ((uint32_t)ReadByte(Reg+1))<<8;
143     tmp |= ((uint32_t)ReadByte(Reg+2))<<16;
```

```
144     tmp |= ((uint32_t)ReadByte(Reg+3))<<24;
145     return tmp;
146   }
147 
148 /*-------------------------------------------------------------------------
149   * Set LED mode ,reference to the enum type LED_MODE
150   */
151 void SetLedMode(uint8_t LED_MODE)
152 {
153    WriteByte(CONFIG_REG_LED_MODE,LED_MODE);
154 }
155 
156 /*-------------------------------------------------------------------------
157   * test api ,Set LED mode circularly  ,reference to the enum type LED_MOI
158   */
159 void test_SetLedMode(void)
160 {
161   unsigned char tmp[8]={0};
162   for(int i=0;i<LED_MODE_NUM;i++)
163   {
164      //WriteByte(CONFIG_REG_LED_MODE,(enum LED_MODE)i);
165      tmp[0] = i;
166      WriteNByte(CONFIG_REG_LED_MODE ,tmp , 1);
167      delay(5000);
168   }
169 }
170 
171 /*-------------------------------------------------------------------------
172   * test api,print the track ball data
173   */
174 void test_PrintTrackData(void)
175 {
176   for(int i=0;i<500;i++)
177   {
178     Serial.print(ReadByte(MOTION_REG_UP));
```

```
    Serial.print("-");
    Serial.print(ReadByte(MOTION_REG_DOWN));
    Serial.print("-");
    Serial.print(ReadByte(MOTION_REG_LEFT));
    Serial.print("-");
    Serial.print(ReadByte(MOTION_REG_RIGHT));
    Serial.print("-");
    Serial.println(ReadByte(MOTION_REG_CONFIRM));
    delay(100);
  }
}

/*--------------------------------------------------------------------------
 * test api,Write register
 */
void test_WriteReg(void)
{
  unsigned char tmp[8]={0};
  tmp[0] = 0X4A;
  WriteByte(CONFIG_REG_I2C_ADDR ,tmp[0]);
  delay(100);
  tmp[0] = 0X64;
  tmp[1] = 0X00;
  WriteByte(CONFIG_REG_I2C_SPEED ,tmp[0]);
  WriteByte(CONFIG_REG_I2C_SPEED+1 ,tmp[1]);
  delay(100);
  tmp[0] = 10;
  WriteByte(CONFIG_REG_LED_MODE ,tmp[0]);
  delay(100);
  tmp[0] = 0xc8;
  tmp[1] = 0x00;
  WriteByte(CONFIG_REG_LED_FLASH_TIME ,tmp[0]);
  WriteByte(CONFIG_REG_LED_FLASH_TIME+1 ,tmp[1]);
  delay(100);
  tmp[0] = 0XEA;
```

```
214    tmp[1] = 0X14;
215    WriteByte(CONFIG_REG_DATA_CLEAR_TIME ,tmp[0]);
216    WriteByte(CONFIG_REG_DATA_CLEAR_TIME+1 ,tmp[1]);
217    delay(100);
218    tmp[0] = 0X22;
219    tmp[1] = 0X05;
220    WriteByte(CONFIG_REG_DATA_READ_TIME ,tmp[0]);
221    WriteByte(CONFIG_REG_DATA_READ_TIME+1 ,tmp[1]);
222    delay(1000);
223    Serial.println("Setted Value are over here");
224    Serial.print("valid:0x");Serial.print(ReadByte(CONFIG_REG_VALID+3),HI
225    Serial.print("I2C_ADDR:0x");Serial.println(ReadByte(CONFIG_REG_I2C_AI
226    Serial.print("I2C_SPEED:0x");Serial.print(ReadByte(CONFIG_REG_I2C_SPI
227    Serial.print("LED_MODE:0x");Serial.println(ReadByte(CONFIG_REG_LED_M(
228    Serial.print("LED_FLASH_TIME:0x");Serial.print(ReadByte(CONFIG_REG_LI
229    Serial.print("DATA_CLEAR_TIME:0x");Serial.print(ReadByte(CONFIG_REG_I
230    Serial.print("DATA_READ_TIME:0x");Serial.print(ReadByte(CONFIG_REG_DA
231    Serial.println();Serial.println();Serial.println();
232    delay(3000);
233
234  }
235
236
237  /*-------------------------------------------------------------------------
238   * test api,Set all config to default value
239   */
240  void test_SetDefault(void)
241  {
242    unsigned char Zero[]={0,0,0,0};
243    Serial.println("Setting Default Value");
244    WriteNByte(CONFIG_REG_VALID , Zero , 4);
245    delay(100);
246    Serial.println("Default Value are over here");
247    Serial.print("valid:0x");Serial.print(ReadByte(CONFIG_REG_VALID+3),HI
248    Serial.print("I2C_ADDR:0x");Serial.println(ReadByte(CONFIG_REG_I2C_AI
```

```
  Serial.print("I2C_SPEED:0x");Serial.print(ReadByte(CONFIG_REG_I2C_SPI
  Serial.print("LED_MODE:0x");Serial.println(ReadByte(CONFIG_REG_LED_MO
  Serial.print("LED_FLASH_TIME:0x");Serial.print(ReadByte(CONFIG_REG_LI
  Serial.print("DATA_CLEAR_TIME:0x");Serial.print(ReadByte(CONFIG_REG_I
  Serial.print("DATA_READ_TIME:0x");Serial.print(ReadByte(CONFIG_REG_DA
  Serial.println();Serial.println();Serial.println();
  delay(3000);
}

void setup() {

 Wire.begin();
 Serial.begin(115200);
}

void loop() {

  test_SetLedMode();

  test_PrintTrackData();

 test_WriteReg();

 test_SetDefault();

 delay(3000);
}
```

**Step 3.** Upload your code into Seeeduino board. If uploading process is done, to open Serial Monitor window, Click **Serial Monitor** under menu **Tool**.

**Step 4.** LED indicator under tracking ball will light on in different mode which will last around 50 seconds

**Step 5.** After that you can rotate or "click" the track ball to get information of its trace.



## Resources

- **[Eagle]** Grove-Mini Track ball v1.0 schematic

- **[PDF]** Grove-Mini Track ball v1.0 schematic
- **[Datasheet]** STM32F103C8T6 Datasheet
- **[Datasheet]** AN48841B Datasheet
- **[Library]** Library file in Github

## Tech Support

Please submit any technical issue into our forum.