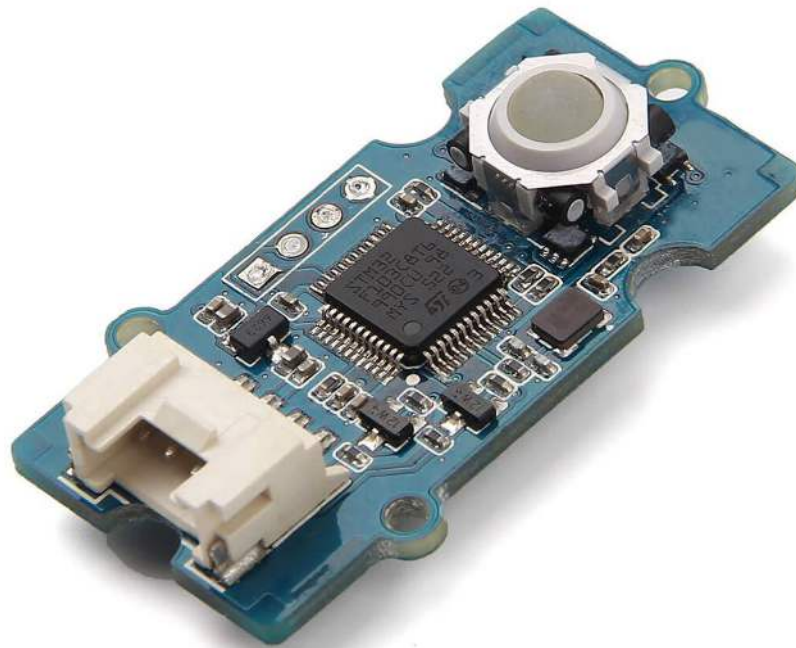


## Grove - Mini Track Ball



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](#)

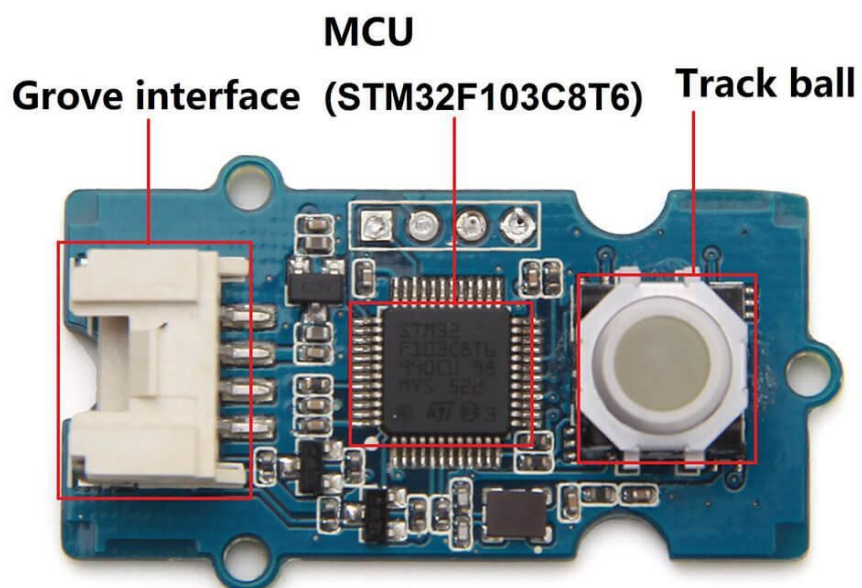
## Application ideas

- 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 °C
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
	

- **Step 2.** Connect Grove-Mini\_Track\_Ball to **I2C** port of Seeeduino.
- **Step 3.** Connect Seeeduino to PC via a USB cable.

```
#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,
    //CONFIG_REG_LED_FLASH_TIME has 2bytes
    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.beginTransaction(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.beginTransaction(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 must be continuous
*/
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 must be continuous
*/
void WriteHalfWord(uint8_t Reg, uint16_t Value)
{
    uint8_t tmp[2]={0};
    tmp[0] = (Value>>0)&0xFF;
    tmp[1] = (Value>>8)&0xFF;
    WriteNByte(Reg,tmp,2);
}

/*-----//
* Read one byte from register
*/
uint8_t ReadByte(uint8_t Reg)
{
    Wire.beginTransaction(DeAddr);
    Wire.write(ReadMode);
    Wire.write(Reg);
    Wire.write(1);
    Wire.endTransmission();
    Wire.requestFrom(DeAddr, 1);
    return Wire.read();
}

/*-----//
* Read half word from register
*/
uint16_t ReadHalfWord(uint8_t Reg)

```

```

{
  uint16_t tmp;
  tmp = ReadByte(Reg);
  tmp |= ((uint16_t)ReadByte(Reg+1))<<8;
  return tmp;
}
/*-----//
* Read one word from register
*/
uint32_t ReadOneWord(uint8_t Reg)
{
  uint32_t tmp;
  tmp = ReadByte(Reg);
  tmp |= ((uint32_t)ReadByte(Reg+1))<<8;
  tmp |= ((uint32_t)ReadByte(Reg+2))<<16;
  tmp |= ((uint32_t)ReadByte(Reg+3))<<24;
  return tmp;
}
/*-----//
* Set LED mode ,reference to the enum type LED_MODE
*/
void SetLedMode(uint8_t LED_MODE)
{
  WriteByte(CONFIG_REG_LED_MODE,LED_MODE);
}
/*-----//
* test api ,Set LED mode circularly ,reference to the enum type LED_MODE
*/
void test_SetLedMode(void)
{
  unsigned char tmp[8]={0};
  for(int i=0;i<LED_MODE_NUM;i++)
  {
    //WriteByte(CONFIG_REG_LED_MODE,(enum LED_MODE)i);
    tmp[0] = i;
    WriteNByte(CONFIG_REG_LED_MODE ,tmp , 1);
    delay(5000);
  }
}
/*-----//
* test api,print the track ball data
*/
void test_PrintTrackData(void)
{
  for(int i=0;i<500;i++)
  {
    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;
  tmp[1] = 0X14;
  WriteByte(CONFIG_REG_DATA_CLEAR_TIME ,tmp[0]);
  WriteByte(CONFIG_REG_DATA_CLEAR_TIME+1 ,tmp[1]);
  delay(100);
  tmp[0] = 0X22;
  tmp[1] = 0X05;
  WriteByte(CONFIG_REG_DATA_READ_TIME ,tmp[0]);
  WriteByte(CONFIG_REG_DATA_READ_TIME+1 ,tmp[1]);
  delay(1000);
  Serial.println("Setted Value are over here");

  Serial.print("valid:0x");Serial.print(ReadByte(CONFIG_REG_VALID+3),HEX);Serial.print(ReadByte(CONFIG_REG_VALID+2),HEX);Serial.print(ReadByte(CONFIG_REG_VALID+1),HEX);Serial.println(ReadByte(CONFIG_REG_VALID+0),HEX);
  Serial.print("I2C_ADDR:0x");Serial.println(ReadByte(CONFIG_REG_I2C_ADDR+0),HEX);

  Serial.print("I2C_SPEED:0x");Serial.print(ReadByte(CONFIG_REG_I2C_SPEED+1),HEX);Serial.println(ReadByte(CONFIG_REG_I2C_SPEED+0),HEX);
  Serial.print("LED_MODE:0x");Serial.println(ReadByte(CONFIG_REG_LED_MODE+0),HEX);

  Serial.print("LED_FLASH_TIME:0x");Serial.print(ReadByte(CONFIG_REG_LED_FLASH_TIME+1),HEX);Serial.println(ReadByte(CONFIG_REG_LED_FLASH_TIME+0),HEX);

  Serial.print("DATA_CLEAR_TIME:0x");Serial.print(ReadByte(CONFIG_REG_DATA_CLEAR_TIME+1),HEX);Serial.println(ReadByte(CONFIG_REG_DATA_CLEAR_TIME+0),HEX);

```

```

Serial.print("DATA_READ_TIME:0x");Serial.print(ReadByte(CONFIG_REG_DATA_READ_TIME+1),HEX
);Serial.println(ReadByte(CONFIG_REG_DATA_READ_TIME+0),HEX);
  Serial.println();Serial.println();Serial.println();
  delay(3000);
}

/*-----//
 * test api,Set all config to default value
 */
void test_SetDefault(void)
{
  unsigned char Zero[]={0,0,0,0};
  Serial.println("Setting Default Value");
  WriteNByte(CONFIG_REG_VALID , Zero , 4);
  delay(100);
  Serial.println("Default Value are over here");

Serial.print("valid:0x");Serial.print(ReadByte(CONFIG_REG_VALID+3),HEX);Serial.print(ReadByte(CON
FIG_REG_VALID+2),HEX);Serial.print(ReadByte(CONFIG_REG_VALID+1),HEX);Serial.println(ReadByt
e(CONFIG_REG_VALID+0),HEX);
  Serial.print("I2C_ADDR:0x");Serial.println(ReadByte(CONFIG_REG_I2C_ADDR+0),HEX);

Serial.print("I2C_SPEED:0x");Serial.print(ReadByte(CONFIG_REG_I2C_SPEED+1),HEX);Serial.println(
ReadByte(CONFIG_REG_I2C_SPEED+0),HEX);
  Serial.print("LED_MODE:0x");Serial.println(ReadByte(CONFIG_REG_LED_MODE+0),HEX);

Serial.print("LED_FLASH_TIME:0x");Serial.print(ReadByte(CONFIG_REG_LED_FLASH_TIME+1),HEX);
Serial.println(ReadByte(CONFIG_REG_LED_FLASH_TIME+0),HEX);

Serial.print("DATA_CLEAR_TIME:0x");Serial.print(ReadByte(CONFIG_REG_DATA_CLEAR_TIME+1),HE
X);Serial.println(ReadByte(CONFIG_REG_DATA_CLEAR_TIME+0),HEX);

Serial.print("DATA_READ_TIME:0x");Serial.print(ReadByte(CONFIG_REG_DATA_READ_TIME+1),HEX
);Serial.println(ReadByte(CONFIG_REG_DATA_READ_TIME+0),HEX);
  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](#)

## Tech Support

Please submit any technical issue into our [forum](#) or drop mail to [techsupport@seeed.cc](mailto:techsupport@seeed.cc).