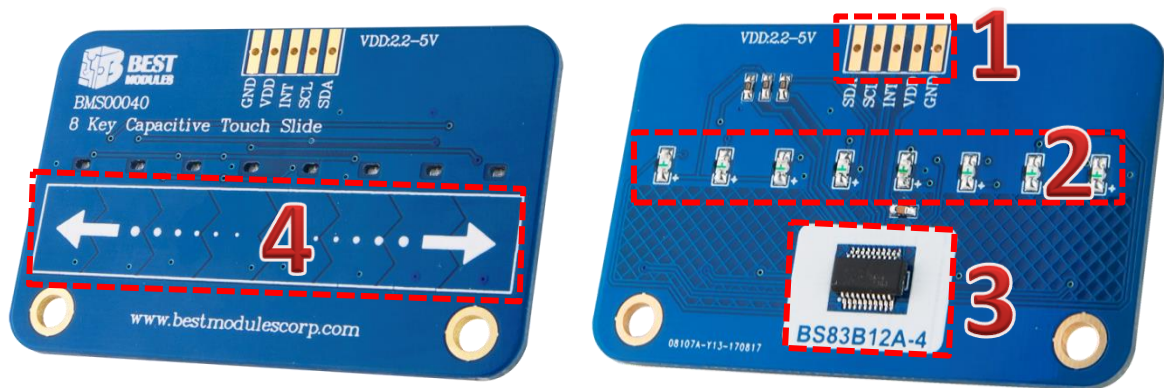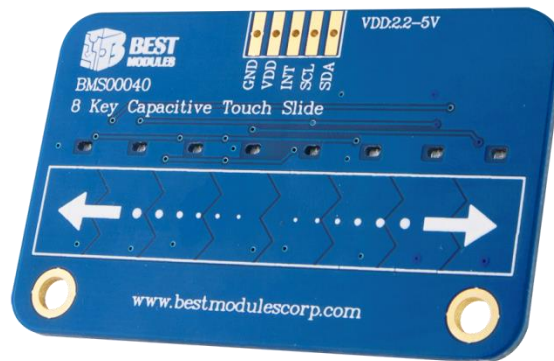# Operation Guide

Layout Diagram

This diagram shows the Slide Capacitive Touch Sensor PCBA layout.

1. Connector : 5-pin connector

2. LED : Touched area indicator on the slide

3. IC : BS83B12A-4 (Holtek Semiconductor Inc. product)

4. Touch Key : Effective touch area is in the shape of rectangular

Pin Function

Name	Function
GND	Ground. This pin should be connected to the host circuit ground.
VDD	Power supply. This pin should be connected to a power supply with a voltage range of 2.2V~5.0V.
INT	Interrupt output. When the slide area is touched, this pin will go low and the associated LED will illuminate. Otherwise it will be in a high state and all LEDs will be off.
SCL	I^2C clock.
SDA	I^2C data.

I²C Command

The touch sensor communicates with an external MCU with the I²C bus. The external MCU reads the touched location from this sensor using an I²C command. The command structure is shown in the following figure.
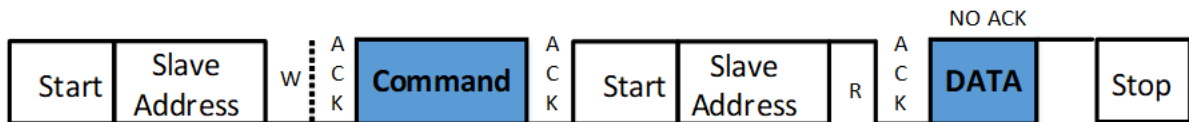
The external MCU (master) transmits the command to the sensor (slave) followed by a read operation from the sensor. Then the sensor will responds to the master MCU with a "DATA" to indication the touched location on the slide. This constitutes a complete I2C command communication.
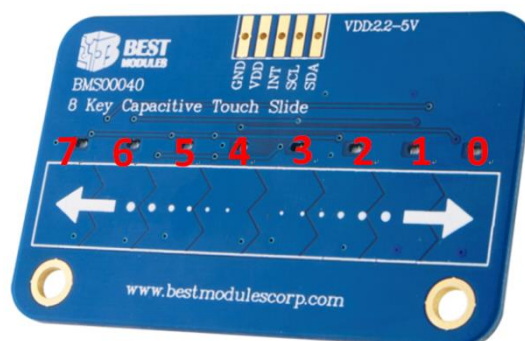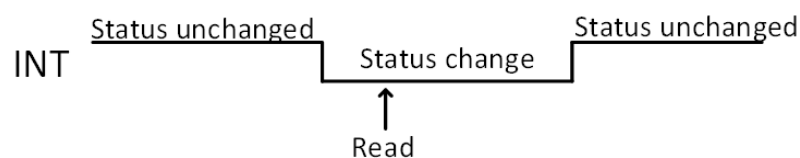
Slave Address (7 bits) = 0x50

Command (8 bits) = 0x0D
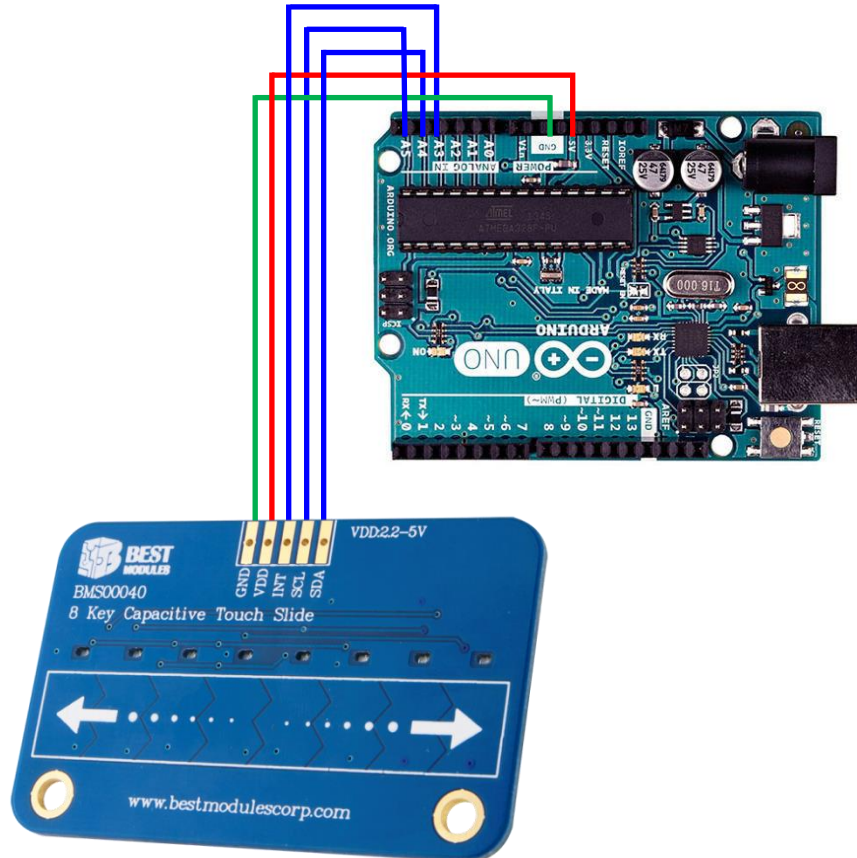
If the slide is not touched, DATA = 0xFF;

If the slide is touched, DATA = 0 ~ 7; each value from 0 to 7 denotes a specific touched location on the slide.

When the slide is touched, the INT pin will go low. It is recommended to read the touched location data from the sensor when the INT pin goes low.

Example Code

The example code and the Arduino board connection are shown below.


```
#include <Wire.h>

#define INT_PIN             digitalRead(A3)

#define INT_PIN_IN          pinMode(A3, OUTPUT)

#define INT_PIN_PU          digitalWrite(A3, HIGH)

#define SLAVE_ADDRESS       0x50

#define COMMAND             0x0D


void setup()
{
  Wire.begin();                             // join the i2c bus as a master
  Serial.begin(9600);                       // Arduino UART test
}
```

```
byte DATA = 0;
void loop() {
    INT_PIN_IN;
    INT_PIN_PU;

    if(INT_PIN == 0)                        //INT trigger
    {
      Wire.beginTransmission(SLAVE_ADDRESS); // transmit to device
      Wire.write(COMMAND);                   // send Command
      Wire.endTransmission();

      Wire.requestFrom(SLAVE_ADDRESS, 1);    // request 1 bytes from slave device
      while (Wire.available()) {             // slave may send less than requested
        DATA = Wire.read();                  // receive a DATA as character
      }
      delay(25);
      Serial.print(DATA);                    // Send DATA to Arduino IDE
      Serial.print("\n");
    }
}
```