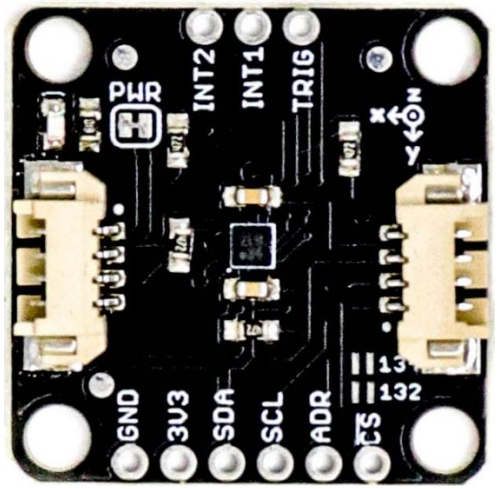
SmartElex Triple Axis Accelerometer Breakout - KX134

The Triple Axis Accelerometer Breakout - KX134 offer high-speed accelerometer 3-axis digital accelerometers from Kionix. The KX134 and KX132 both include a host of accelerometer features including Freefall detection, Directional Tap™ and Double-Tap™ detection, tilt orientation detection and more. The breakouts can interface with controllers using both I^2C and SPI at high speeds.



The KX134 is a low-power, 16-bit resolution 3-axis accelerometer capable of measuring ±8g/16g/32g/64g (user selectable) and has up to a 10kHz (max) output data rate making it ideal for high-g measurements as well as high-speed applications such as vibration sensing. At lower ranges the sensitivity can be set as high as 17367 counts/g (@±2g), so it's a great for applications looking for both high-speed data rates and high-sensitivity measurements at lower acceleration ranges.

KX134 3-Axis Accelerometers

- Four User-Selectable Measurement Ranges
  - KX134: ±8 / 16 / 32 / 64g

- User-configurable 3-stage Advanced Data Path (ADP) with low-pass filter, low-pass/high-pass filter and RMS calculation engine
- User-selectable Low Power or High-Performance Modes
- Configurable Output Data Rate (ODR) up to 25,600Hz
- High resolution Wake-Up / Back-to-Sleep functions with configurable thresholds (as low as 15.6mg on the KX134)
- Free fall detection
- Directional-Tap™/Double-Tap™
- Device Orientation algorithms
- Embedded 512-byte FIFO buffer (continues to record while being read)
- Digital I^2C up to 3.4MHz and Digital SPI up to 10MHz

The KX13x also includes an integrated voltage regulator to maintain consistent performance across its entire supply voltage range (**1.7** to **3.6V**). The table below outlines some of the electrical and functional characteristics of the KX134-1211 from the sensors' datasheet. Refer to the accelerometers' datasheet for a full overview KX134-1211.

Parameter		**Units**	**Min**	**Typical**	**Max**
Supply Voltage (VDD)		V	1.7	2.5 (3.3 for use with connector)	3.6
Current Consumption (Accelerometer Only)	High Performance w/Wake-up Detection (ODR=800Hz)	µA		148	
	Low Power w/Wake-up Detection (ODR=0.781Hz, 2 samples averaged)			0.53	
	Standby			0.50	
Operating Temperature Range		°C	-40	-	105
Output Data Rate		Hz	0.781	50	25600

Sensitivity (16 bit)	±2g[1]	counts/g	14501	16384	17367
	±4g[1]		7700	8192	8684
	±8g		3768	4096	4424
	±16g		1884	2048	2212
	±32g[2]		942	1024	1106
	±64g[2]		471	512	553
Noise[3]	RMS	mg		KX134: 1.9	
	Density	µg/√Hz		KX134: 300	
I²C Address				0x1E (0x1F alternate)	

1. Reminder: ±2/4g ranges are only available on the KX132.
2. Reminder: ±32/64g ranges are only availabe on the KX134.
<=”” a=”” style="box-sizing: border-box; background-color: transparent; color: rgb(224, 49, 29); text-decoration: none;">**3.** Acceleration data noise varies depending on ODR, power mode & Average Filter Control settings. Noise measuring settings: High-Performance Mode (RES=1), ODR=50Hz, IIR Filter Enabled and IIR filter corner frequency set to ODR/2. Refer to Table 1 in the sensors' Datasheets as well as the Technical Reference Manuals for more information.

Pinout

The KX13x Breakouts' I^2C and SPI interface share the same pins so users must select the interface mode by altering the state of the ADR/SDO pin. The ADR jumper sets the state of the ADR/SDO pin (more on that in the Solder Jumpers section). Both breakouts operate in I^2C mode by default. We've labeled these shared pins so I^2C labels are visible from the front and SPI labels are visible when viewed from the back.

SPI Interface

Communicating via SPI on the KX13x is ideal for taking advantage of the maximum Output Data Rate as the Digital SPI interface on the KX13x can operate at speeds up to 10MHz.

The KX13x breaks out the SPI interface to the standard 0.1" spaced header as the I^2C pins. As mentioned above, the board ships with the I^2C interface enabled by default so to switch to the SPI interface users need to **open** the ADR jumper by severing the trace in between the "Center" and "Left" pads and connect the SDO/ADR pin to SDI/CIPO on your microcontroller.

Interrupt and Trigger Pins

The KX13x has two physical interrupt pins as well as a trigger pin for FIFO buffer control. Both of the physical interrupts operate as push-pull, enter a high-impedence (high-Z) state during the Power-On-Reset (POR) procedure and are driven LOW after POR. Connect these pins to external interrupt-capable pins on your microcontroller to use the interrupt functionalities. Refer to the Interrupt and Buffer examples in the KX13x Arduino library for a demonstration of using the interrupt pins.

The Trigger pin controls the FIFO buffer. By default, the KX13x ties this pin to ground through the TRIG jumper. Users who wish to use the Trigger pin must **open** that jumper before tying it to a pin on their microcontroller. Refer to the Datasheets and either Technical Reference Manuals for more information on using this pin to control the FIFO buffer.

Solder Jumpers

The KX13x has four jumpers labeled ADR, I2C, TRIG and PWR. In this section we'll cover each jumper's purpose, their default states and how to configure them to alter the functionality of the KX13x Breakouts.

Address (ADR) Jumper

This 3-way jumper selects the I^2C address of the KX13x and also selects the communication interface for the chip by pulling the ADR/SDO pin to either **3.3V**, **0V/Ground** or **No Connect**. By default, the ADR/SDO is pulled to **3.3V** via a **4.7kΩ** resistor to set the KX134 to operate in I^2C mode with the I^2C address as **0x1E**.

To change the I^2C address to **0x1F**, sever the trace between the "Center" and "Left" pads and then connect the "Center" and "Right" pads together to pull the ADR/SDO pin to **0V/Ground**.

Finally, to set the KX13x to SPI mode, sever the trace between the "Center" and "Left" pads of the ADR jumper (default setting) to leave the ADR/SDO pin **Floating/No Connect**. After adjusting the jumper, connect the SDO pin to your controller's SDI/COPI pin.

I^2C Jumper

The I^2C jumper on the KX13x pulls the SDA and SCL lines to **3.3V** via a pair of **4.7kΩ** resistors. The default state of this jumper is **CLOSED**. Open the jumper by severing the traces between the three pads to disable the pullups on these lines.

If you have more than one device on a single I^2C bus, best practices recommend to only maintain a single pair of pullup resistors to avoid creating too strong of a parallel resistance. A strong parallel resistance can lead to communication issues on the bus. Take note that if you are using a single set of pull-up resistors on your I^2C bus, make sure all devices operate at the same logic level or are properly shifted to avoid damage to the device(s).

Trigger (TRIG) Jumper

The Trigger jumper ties the TRIG pin on the KX13x-1211 to **0V/Ground**. The default state of this jumper is **CLOSED**. To use the Trigger pin for FIFO control, open the jumper and connect the TRIG PTH pin to a digital I/O pin on your microcontroller. Refer to section 2.5 in the Technical Reference Manuals for more information on using Trigger Mode.

Power LED (PWR) Jumper

The Power LED jumper (labeled PWR on the board) completes the power LED circuit on the board by tying the anode of the LED to **3.3V** via a **1kΩ** resistor. The jumper is **CLOSED** by default. Disable the power LED by severing the trace between the two pads. Disabling the LED helps reduce the total current draw of the board and is particularly helpful for low-power or battery-powered applications.

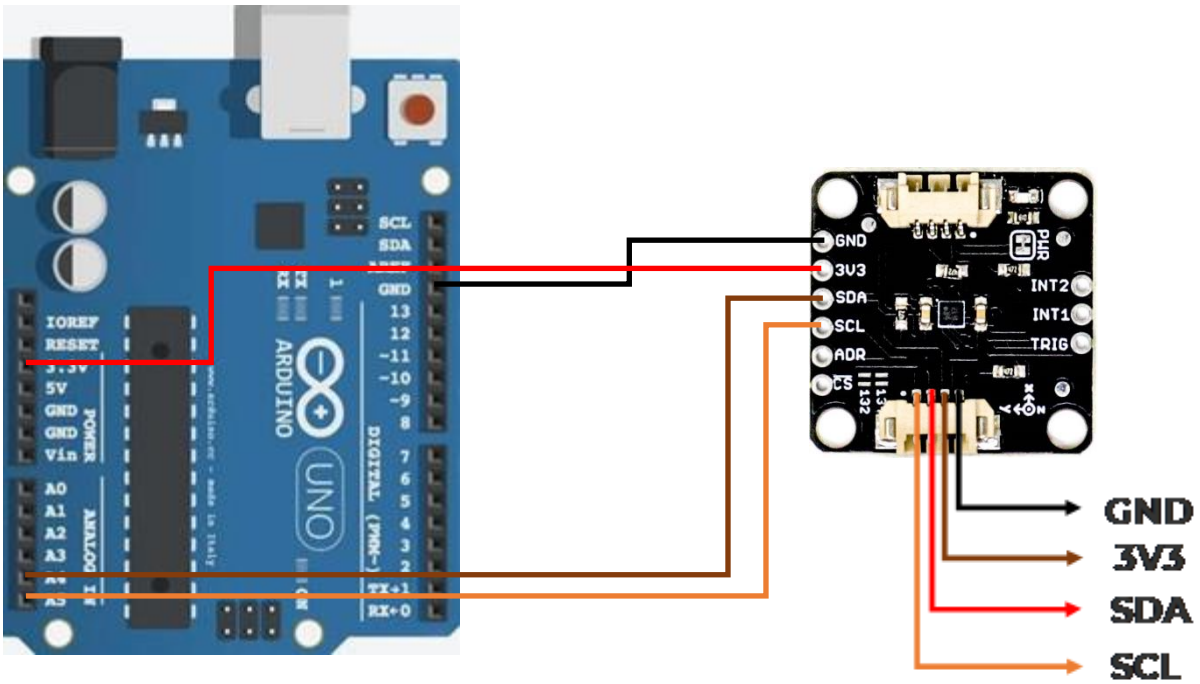
SPI Assembly

If you'd prefer to take advantage of the max output data rate of the KX13x, you'll want to use the SPI interface instead of the I^2C interface. Assembling the KX13x Breakout in SPI mode requires some through-hole soldering.

The Address (ADR) Jumper must be opened by severing the trace between the "Center" and "Left" pads to switch to SPI mode. After opening this jumper, connect the SDO pin to your controller's SDI/CIPO pin.

With the KX13x Breakout set to SPI mode, solder headers or wire to the PTH header on the board and make the SPI connections with your controller. Remember the KX13x operates at **3.3V** logic so make sure to connect to a board running at the same logic level or use a level shifter to adjust it to a safe voltage.

Wiring:



Arduino	KX134
A5(SCL)	SCL
A4(SDA)	SDA
3.3V	3V3
GND	GND

KX13x Arduino Library

The SparkFun KX13x Arudino library makes it easy to get started measuring acceleration data from the sensor. Install the library through the Arduino Library Manager by searching for **"SparkFun KX13x Arduino Library"**.

Example - Basic Readings

Example 1 is a basic example to demonstrate how to read data from the accelerometer. Open the example by navigating to **"File > Examples > SparkFun Qwiic KX13x Library > Example1BasicReadings"**. Next, open the **Tools** menu and select your board (in this case, Arduino Uno) and correct Port your board enumerated on. Upload the code, open the serial monitor and set the baud rate to **115200**.

The example defaults to use the KX132 so if you are using the KX134, make sure to comment/uncomment the appropriate line:

```
SparkFun_KX132 kxAccel;
//SparkFun_KX134 kxAccel; // For the KX134, uncomment this and comment line above
```

The setup initializes the sensor and performs a software reset, configures it to operate at the 16g range and enable the accelerometer. Just as above, make sure to select the appropriate option for either the KX132/134 as the code defaults to the KX132.

After initializing the IC, the code prints out data for all three axes every 20ms. The delay here is important as it should be 1/ODR (Output Data Rate) and the default setting is 50Hz.

```
#include <Wire.h>
```

```ino
#include <SparkFun_KX13X.h> // Click here to get the library:
http://librarymanager/All#SparkFun_KX13X

SparkFun_KX132 kxAccel;
// SparkFun_KX134 kxAccel; // For the KX134, uncomment this and comment line above

outputData myData; // Struct for the accelerometer's data

void setup()
{

  Wire.begin();

  Serial.begin(115200);
  Serial.println("Welcome.");

  // Wait for the Serial monitor to be opened.
  while (!Serial)
    delay(50);

  if (!kxAccel.begin())
  {
    Serial.println("Could not communicate with the the KX13X. Freezing.");
    while (1)
      ;
  }

  Serial.println("Ready.");

  if (kxAccel.softwareReset())
    Serial.println("Reset.");

  // Give some time for the accelerometer to reset.
  // It needs two, but give it five for good measure.
  delay(5);

  // Many settings for KX13X can only be
  // applied when the accelerometer is powered down.
  // However there are many that can be changed "on-the-fly"
  // check datasheet for more info, or the comments in the
  // "...regs.h" file which specify which can be changed when.
  kxAccel.enableAccel(false);

  kxAccel.setRange(SFE_KX132_RANGE16G); // 16g Range
  // kxAccel.setRange(SFE_KX134_RANGE16G);         // 16g for the KX134

  kxAccel.enableDataEngine(); // Enables the bit that indicates data is ready.
```

```
  // kxAccel.setOutputDataRate(); // Default is 50Hz
  kxAccel.enableAccel();
}

void loop()
{
  // Check if data is ready.
  if (kxAccel.dataReady())
  {
    kxAccel.getAccelData(&myData);
    Serial.print("X: ");
    Serial.print(myData.xData, 4);
    Serial.print(" Y: ");
    Serial.print(myData.yData, 4);
    Serial.print(" Z: ");
    Serial.print(myData.zData, 4);
    Serial.println();
  }
  delay(20); // Delay should be 1/ODR (Output Data Rate), default is 1/50ODR
}
//////////////////////////////////////////////////////END//////////////////////////////////////////////////////
```