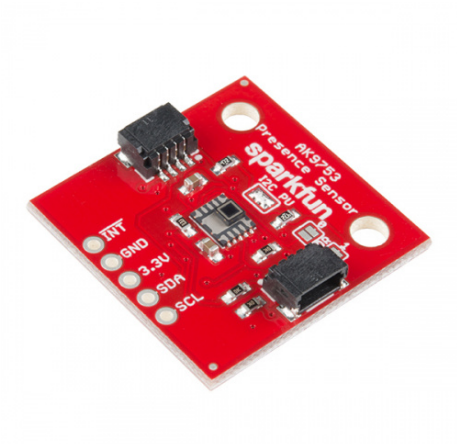
Qwiic Human Presence Sensor (AK9753) Hookup Guide

Introduction

The AK9753 Human Presence sensor is a Qwiic enabled, 4-channel Nondispersive Infrared Sensor (NDIR). Each channel has a different field of view, so not only can the AK9753 detect a human, but it can also tell which direction the person is moving.



SparkFun Human Presence Sensor Breakout - AK9753 (Qwiic)
SEN-14349

This hookup guide will show you how to get started taking basic reading from the sensor. We will cover both a serial output of readings as well as nice graph of the derivative of our readings from a single channel.

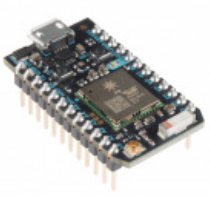
Required Materials

To get started, you'll need a microcontroller to, well, control everything.

SparkFun RedBoard - Programmed with Arduino
DEV-13975

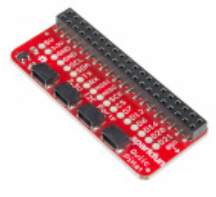
SparkFun ESP32 Thing
DEV-13907

Raspberry Pi 3
DEV-13825



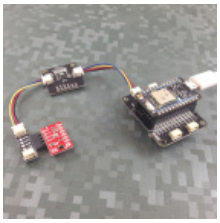
Particle Photon (Headers)
WRL-13774

Now to get into the Qwiic ecosystem, the key will be one of the following Qwiic shields to match your preference of microcontroller:



SparkFun Qwiic HAT for Raspberry Pi
DEV-14459

SparkFun Qwiic Shield for Arduino
DEV-14352



Qwiic Shield for Photon
SPX-14202

You will also need a Qwiic cable to connect the shield to your human presence sensor, choose a length that suits your needs.

Qwiic Cable - 500mm
PRT-14429

Qwiic Cable - 100mm
PRT-14427

Qwiic Cable - 200mm
PRT-14428

Qwiic Cable - 50mm
PRT-14426

Suggested Reading

If you aren't familiar with our new Qwiic system, we recommend reading here for an overview. We would also recommend taking a look at the following tutorials if you aren't familiar with them.

I2C
An introduction to I2C, one of the main embedded communications protocols in use today.

Qwiic Shield for Arduino & Photon Hookup Guide
Get started with our Qwiic ecosystem with the Qwiic shield for Arduino or Photon.

Hardware Overview

Listed below are some of the characteristics and operating ranges of the AK9753 Human Presence Sensor.

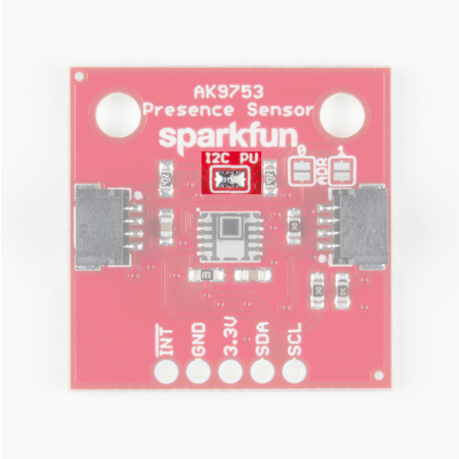
Characteristic	**Range**
Operating Voltage	**3.3V**
Operating Temperature	-30°C to 85°C
Current Consumption	10 µA (typ.), 100 µA (max) (5V)
Spectral Sensitivity	5-7 µm (5V)
Detection Range	3 m (5V)
Temperature Sensor Range	-10° to 60°C

Pins

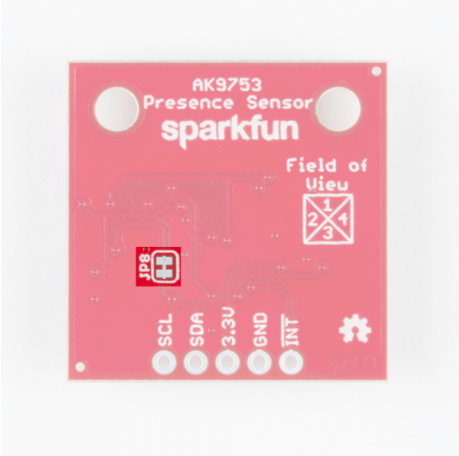
Pin	**Description**	**Direction**
GND	Ground	In
3.3V	Power	In
SDA	Data	In
SCL	Clock	In
$\overline{\text{INT}}$	Interrupt, goes high when data is ready. After data is read, the pin pulls low	Out

Optional Features

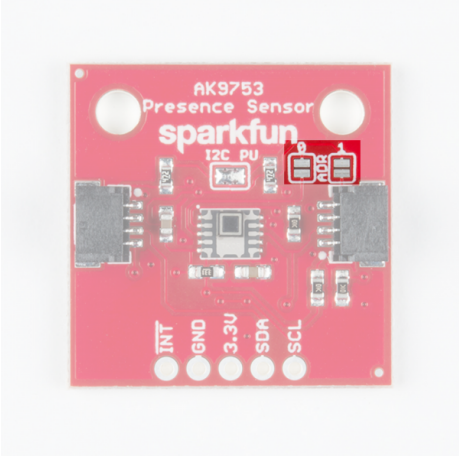
There are several jumpers on board that can be changed to facilitate several different functions. The first of which is the I^{2}C pull-up jumper, highlighted below. If you have your own I^{2}C pull ups, you can remove the solder from this jumper.



The JP8 jumper on the back of the board (highlighted below) can be sliced with a hobby knife to disable the interrupt capability. The “Field of View” text and box shows the field of view of each of the 4 sensors. From the sensor’s point of view, channel 1 is on top, 2 is on left, 3 is on bottom, and 4 is on the right.



Addresses 0 and 1 can be used to change the I^{2}C address of the board in case you have multiple devices using the same address. The below table shows the addresses that correspond to the different combinations of opened and closed jumpers.



Address 0	**Address 1**	**I^{2}C Address**
0	0	0x64
0	1	0x65
1	0	0x67

1	1	Switch Mode

Switch mode is available if you want to avoid using I^{2}C altogether. In switch mode, data is written to the interrupt pin. The pin pulls high when the difference between two outputs (ex. IR1-IR3 or IR2-IR4) is greater than the upper or lower thresholds set in EEPROM by the manufacturer. This mode is a good solution if your project doesn't need much accuracy.

Hardware Assembly

If you haven't yet assembled your Qwiic Shield, now would be the time to head over to that tutorial. With the shield assembled, Sparkfun's new Qwiic environment means that connecting the sensor could not be easier. Just plug one end of the Qwiic cable into the AK9753 Human Presence Sensor, the other into the Qwiic Shield and you'll be ready to upload a sketch and start sensing humans. It seems too easy, but thats why we made it this way!

Software

> **Note:** This example assumes you are using the latest version of the Arduino IDE on your desktop. If this is your first time using Arduino, please review our tutorial on installing the Arduino IDE. If you have not previously installed an Arduino library, please check out our installation guide.

First, you'll need to download and install the SparkFun AK975X Arduino library, this can be done using the button below or by using the Arduino Library Manager.

DOWNLOAD THE SPARKFUN AK975X ARDUINO LIBRARY

Before we get started developing a sketch, let's look at the available functions of the library.

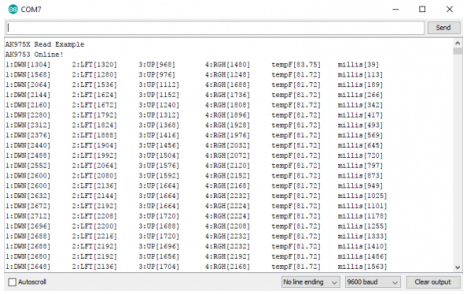
- `int16_t getIR1();` — Returns the value from channel 1, there are also functions `getIR2();` and so on and so forth.
- `void refresh();` — Reads the dummy register telling the sensor to calculate the next reading.
- `boolean available();` — Returns `true` if data is ready.
- `boolean overrun();` — Returns true if the overrun bit is set.
- `void softReset();` — Resets the IC via software.
- `void setMode(uint8_t mode = AK975X_MODE_0);` — Set mode of the sensor. Mode 0 is continuous read mode.
- `void setCutoffFrequency(uint8_t frequency = AK975X_FREQ_8_8HZ);` — Sets the filtering frequency. 8Hz is the fastest and least filtered.
- `float getTemperature();` — Returns sensor temperature in °C.
- `float getTemperatureF();` — Returns sensor temperature in °F.
- `void enableDebugging(Stream &debugPort = Serial);` — Self explanatory, allows the output various extra messages to help with debugging.
- `void disableDebugging();` — Disables debugging messages.
- `uint8_t readRegister(uint8_t location);` — Basic read of I^{2}C register.
- `void writeRegister(uint8_t location, uint8_t val);` — Writes to an I^{2}C register.
- `uint16_t readRegister16(byte location);` — Reads a 16-bit value from an I^{2}C register.

Example 1: Basic Serial Readings

The example code shown below will get you started taking basic serial readings from the Human Presence Sensor. This sketch is relatively simple, pulling values using the `getIRX();` functions and printing them over a serial terminal at 9600 baud.

```
#include <Wire.h>

#include "SparkFun_AK975X_Arduino_Library.h" //Use Library Manager or download here: https://github.com/sparkfun/SparkFun_AK975X_Arduino_Library

AK975X movementSensor; //Hook object to the library

int ir1, ir2, ir3, ir4, temperature;

void setup()
{
  Serial.begin(9600);
  Serial.println("AK975X Read Example");

  Wire.begin();

  //Turn on sensor
  if (movementSensor.begin() == false)
  {
    Serial.println("Device not found. Check wiring.");
    while (1);
  }
}

void loop()
{
  if (movementSensor.available())
  {
    ir1 = movementSensor.getIR1();
    ir2 = movementSensor.getIR2();
    ir3 = movementSensor.getIR3();
    ir4 = movementSensor.getIR4();
    float tempF = movementSensor.getTemperatureF();

    movementSensor.refresh(); //Read dummy register after new data is read

    //Note: The observable area is shown in the silkscreen.
    //If sensor 2 increases first, the human is on the left
    Serial.print("1:DWN[");
    Serial.print(ir1);
    Serial.print("]\t2:LFT[");
    Serial.print(ir2);
    Serial.print("]\t3:UP[");
    Serial.print(ir3);
    Serial.print("]\t4:RGH[");
    Serial.print(ir4);
    Serial.print("]\ttempF[");
    Serial.print(tempF);
    Serial.print("]\tmillis[");
    Serial.print(millis());
    Serial.print("]");
    Serial.println();
  }
  delay(1);
}
```

The output should look similar to the image below, with the values of each channel, temperature, and timestamp of the reading in each row.



Click the image for a closer look.

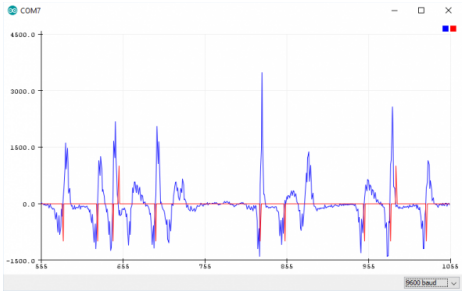
Example 2: Graphing the Human Presence Sensor Serial Data

The next example takes the derivative of a single channel and displays this on the serial plotter. The code for this example is shown below. Notice how you can change the sensitivity of the Human Presence Sensor using the `sensitivity` value. It is set at 50 by default, and values lower than this will yield higher sensitivity.

```
#include <Wire.h>

#include "SparkFun_AK975X_Arduino_Library.h" //Use Library Manager or download here: https://github.com/sparkfun/SparkFun_AK975X
_Arduino_Library

AK975X movementSensor; //Hook object to the library

unsigned int upValue; // current proximity reading
unsigned int averageValue;   // low-pass filtered proximity reading
signed int fa2;              // FA-II value;
signed int fa2Derivative;     // Derivative of the FA-II value;
signed int fa2DerivativeLast;     // Last value of the derivative (for zero-crossing detection)
signed int sensitivity = 50;  // Sensitivity of touch/release detection, values closer to zero increase sensitivity

#define LOOP_TIME 30  // Loop duration in ms. 30ms works well.

//Exponential average weight parameter / cut-off frequency for high-pass filter
//#define EA 0.3  //Very steep
//#define EA 0.1  //Less steep
#define EA 0.05  //Less steep

void setup()
{
  Serial.begin(9600);

  Wire.begin();

  //Turn on sensor
  if (movementSensor.begin() == false)
  {
    Serial.println("Device not found. Check wiring.");
    while (1);
  }

  upValue = movementSensor.getIR3(); //Get one of the latest IR values
  averageValue = upValue;
  fa2 = 0;
  movementSensor.refresh(); //Read dummy register after new data is read
}

void loop()
{
  unsigned long startTime = millis();

  while (movementSensor.available() == false) delay(1); //Wait for new data

  upValue = movementSensor.getIR3(); //Get one of the latest IR values
  movementSensor.refresh(); //Read dummy register after new data is read

  fa2DerivativeLast = fa2Derivative;
  fa2Derivative = (signed int) averageValue - upValue - fa2;
  fa2 = (signed int) averageValue - upValue;

  //Turn on various variables to see how they respond on the graph
  //Serial.print(upValue);
  //Serial.print(",");
  //Serial.print(fa2);
  //Serial.print(",");
  Serial.print(fa2Derivative);
  Serial.print(",");

  //Look to see if the previous fa2Der was below threshold AND current fa2Der is above threshold
  //Basically, if the sign of the fa2Ders has switched since last reading then we have an event
  if ((fa2DerivativeLast < -sensitivity && fa2Derivative > sensitivity) || (fa2DerivativeLast > sensitivity && fa2Derivative < -
sensitivity)) // zero crossing detected
  {
    if (fa2 < -sensitivity) // minimum
    {
      Serial.print(-1000); //Cause red line to indicate entering presence
      Serial.print(",");
```

```
      //Serial.print("Entered view");
    }
    else if (fa2 > sensitivity) // maximum
    {
      Serial.print(1000); //Cause red line to indicate exiting presence
      Serial.print(",");
      //Serial.print("Exited view");
    }
    else
    {
      Serial.print(0); //Cause red line to indicate no movement
      Serial.print(",");
      //Serial.print("No Movement");
    }
  }
  else
  {
    Serial.print(0);
    Serial.print(",");
  }

  Serial.println();

  // Do this last
  averageValue = EA * upValue + (1 - EA) * averageValue;
  while (millis() < startTime + LOOP_TIME); // enforce constant loop time
}
```

Once again, the output for this example code should look something like the below image. This graph is the derivative of a single channel on our presence sensor, so any variance from 0 shows the rate of change of the signal.



Click the image for a closer look.

Resources and Going Further

Now that you've successfully got your AK9753 up and running, it's time to incorporate it into your own project!

For more information on the AK9753, check out the resources below:

- AK9753 Schematic (PDF)
- AK9753 Eagle Files (ZIP)
- AK9753 Datasheet (PDF)
- Product Showcase: Qwiic Presence Sensor
- Qwiic System Landing Page
- SparkFun AK975X Arduino Library GitHub Repository – Source and example files for the Arduino library used in this tutorial.
- GitHub Repository - Repo for the product.

Need some inspiration for your next project? Check out some of these related tutorials:

RedBoard Santa Trap

A fun holiday project to try for anyone looking to catch Santa on Christmas!

PIR Motion Sensor Hookup Guide

An overview of passive infrared (PIR) motion detecting sensors, and how to hook them up to an Arduino.

ZX Distance and Gesture Sensor SMD Hookup Guide

How to connect and use the SparkFun ZX Distance and Gesture Sensor with an Arduino.

OpenPIR Hookup Guide

How to use and customize the SparkFun OpenPIR motion sensor.