  
XC-3704 IR Temperature Sensor Documentation

### Contactless Temperature Monitor

Robbie Andersen, Kai Bushby, Mitch Stevens

## Summary

This documentation is designed to aid you through the setup and testing process of the XC-3704. By the end of this setup process you should be able to read data from the IR sensor and learn how to configure the sensor to more accurately work in specific use case scenarios.

## Equipment

* Arduino Uno
* Laptop loaded with Arduino IDE
* 4 x medium wires
* Small Breadboard
* XC-3704 IR Temperature Sensor

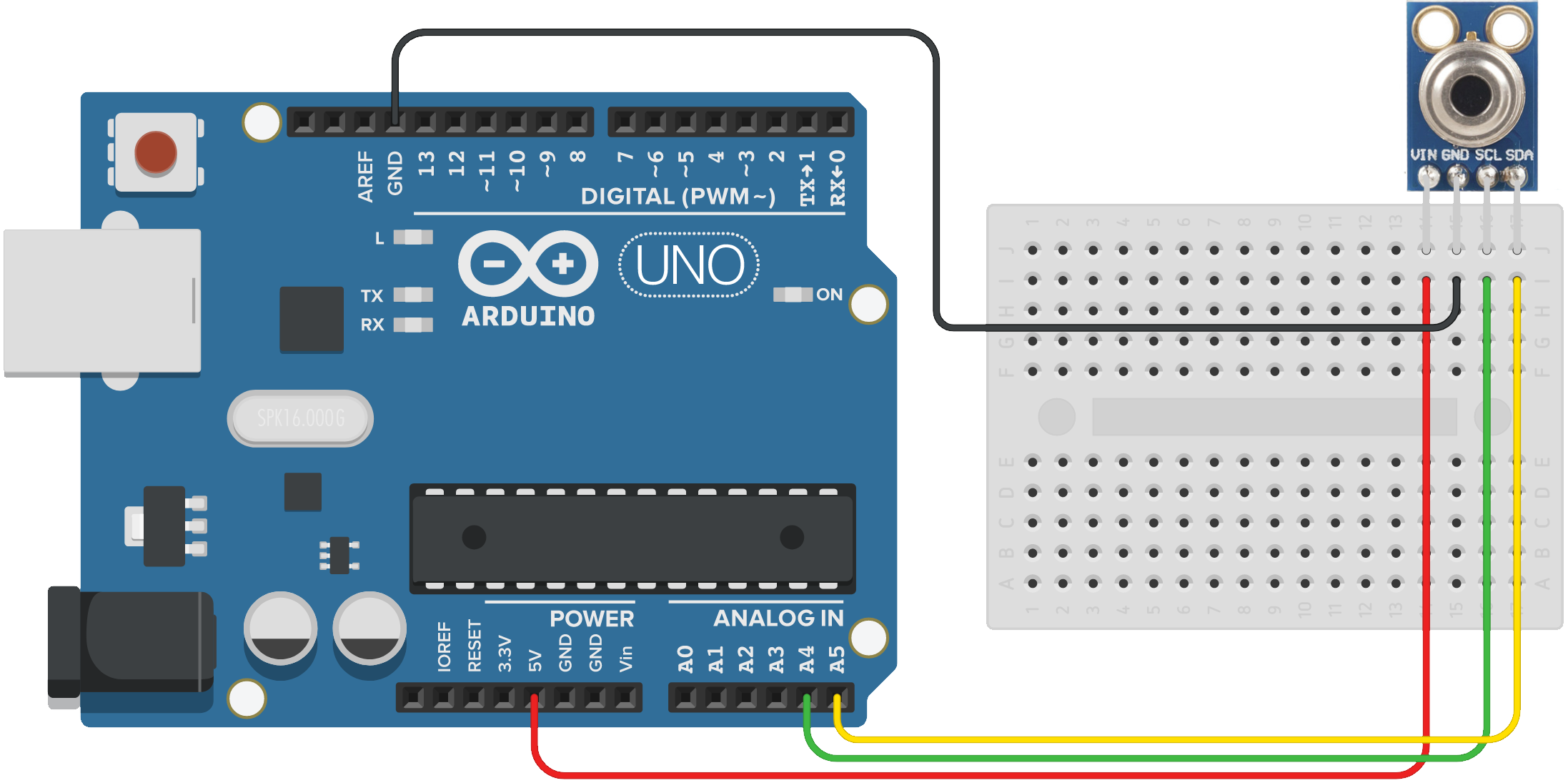
Testing:

* A 15cm or longer ruler
* Thermometer

## Method

1. Connect both the XC-3704 IR Temperature Sensor and Arduino Uno together using the breadboard and the wires.

### Wiring Diagram

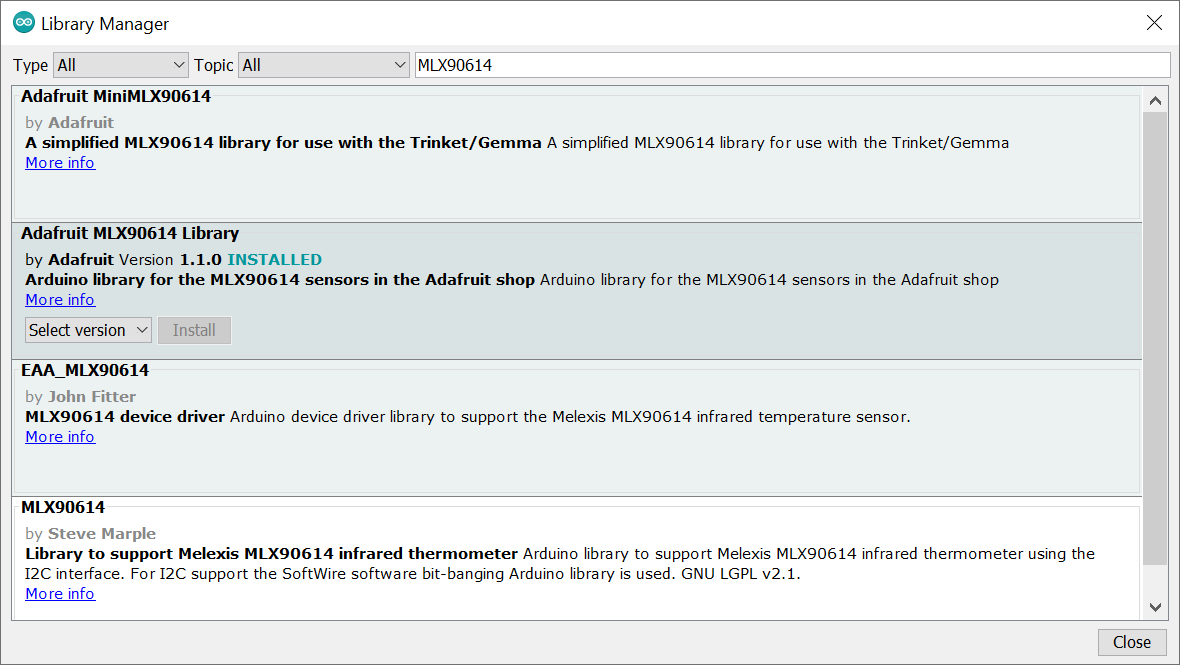


### Wiring Table

|  |  |
| --- | --- |
| **XC-3704** | **Arduino Uno** |
| SLC | A4 |
| GND | GND |
| VIN | 5V |
| SDA | A5 |

1. Search for ‘MLX90614’ under the Library Manager or download directly from https://github.com/adafruit/Adafruit-MLX90614-Library.

To get to the Library Manager: Tools > Manage Libraries...



1. Load the following code onto the Arduino Uno

|  |
| --- |
| **#include <Wire.h>** **#include <Adafruit\_MLX90614.h>**  Adafruit\_MLX90614 mlx = Adafruit\_MLX90614();  void setup() {  Serial.begin(9600);   Serial.println("Adafruit MLX90614 test");   mlx.begin();   *// emissivity of human skin*  mlx.writeEmissivityReg(0xFAE0);  delay(50);  Serial.println(mlx.readEmissivity()); }  double sampleTemp(int samples) {  double total = 0.0;  for (int i = 0; i < samples; i++) {  total += mlx.readObjectTempC();  delay(5);  }  return total/((double)samples); }  void printDiff(){  double prevTemp = 0.0;  while(true){ Serial.println(mlx.readEmissivity());  double temp = sampleTemp(10);  Serial.println(temp - prevTemp);  Serial.println(mlx.readEmissivity());  prevTemp = temp;  } }  double prevTemp = 0.0;  void loop() {  Serial.print("Ambient = "); Serial.print(mlx.readAmbientTempC());   Serial.print("\*C\tObject = "); Serial.print(mlx.readObjectTempC());  Serial.println("\*C");    Serial.println();  delay(500);  double temp = sampleTemp(10);  Serial.println(mlx.readEmissivity()); } |

1. Start the program on the Arduino Uno and open the serial monitor and begin the testing steps.

## Testing

1. Get the body temperature of your test subject. Place the thermometer probe in the center of the open hand and leave it there for 1 minute or until the temperature is stable. Record this value.
2. Place a ruler standing up right next to the XC-3704 sensor. Then have a test subject hover their hand with their palm above the sensor at the given testing interval for 10-15 seconds. Then record this value.

### Tasting Data Table

|  |  |  |  |
| --- | --- | --- | --- |
|  | Robbie | Mitch | Kai |
| Body Temp | 34.8 | 33.9 | 34.8 |
| 6” | 27.4 | 27.1 | 27.3 |
| 5” | 28.8 | 27.8 | 28.1 |
| 4” | 29.5 | 28.9 | 29.2 |
| 3” | 31.2 | 29.9 | 31.2 |
| 2” | 32.5 | 32.1 | 32.7 |
| 1” | 34.3 | 32.9 | 33.0 |

## Conclusion

As shown in the testing data the temperature is at its most accurate when within 1-2 inches of the XC-3704 IR sensor. This is likely due to the focus of the sensor. Lenses can allow for different focal lengths. This sensor is typically accurate within 2 degrees; this was specified in the official data sheet.

## Installing Kobuki

To control kobuki, a person would need to install ROS (either Kinetic or Melodic, Kinetic is recommended) and a kobuki driver package respective to the ROS distribution. The

**Installation on ROS-Kinetic**

***$ sudo apt-get install ros-kinetic-kobuki***

**Installation on ROS-Melodic**

To install the Kobuki Driver onto Melodic, the package will have to be compiled from source code

***$ sudo apt-get install python-catkin-tools***

Make Workspace

***$ mkdir -p kobukiws/src***

Go to Workspace source folder and clone repo

***$ cd kobuki\_ws/src***

***$ catkin\_init\_workspace***

***$ git clone https://github.com/amateurmaker/Turtlebot2.git***

***$ git clone https://github.com/yujinrobot/kobuki.git***

Go back to workspace and install dependencies

***$ cd ..***

***$ wstool init src src/kobuki/kobuki.rosinstall***

***$ sudo rosdep install --from-paths src --ignore-src -r -y***

Build package

There will be missing dependencies, do apt-get ros-melodic-<missing dependency>

***$ catkin build***

Run there commands or place them into ~/.bashrc

***$ . /kobuki\_ws/devel/setup.bash***

OR

***$ source ~/kobuki\_ws/devel/setup.bash***

## ROS from Remote

Now we have the computer connected to the Raspberry Pi and want to control it.

Terminal 1:

***$ roscore***

Terminal 2:

***$ roslaunch kobuki\_node minimal.launch***

Terminal 3:

***$ roslaunch kobuki\_keyop safe\_keyop.launch***