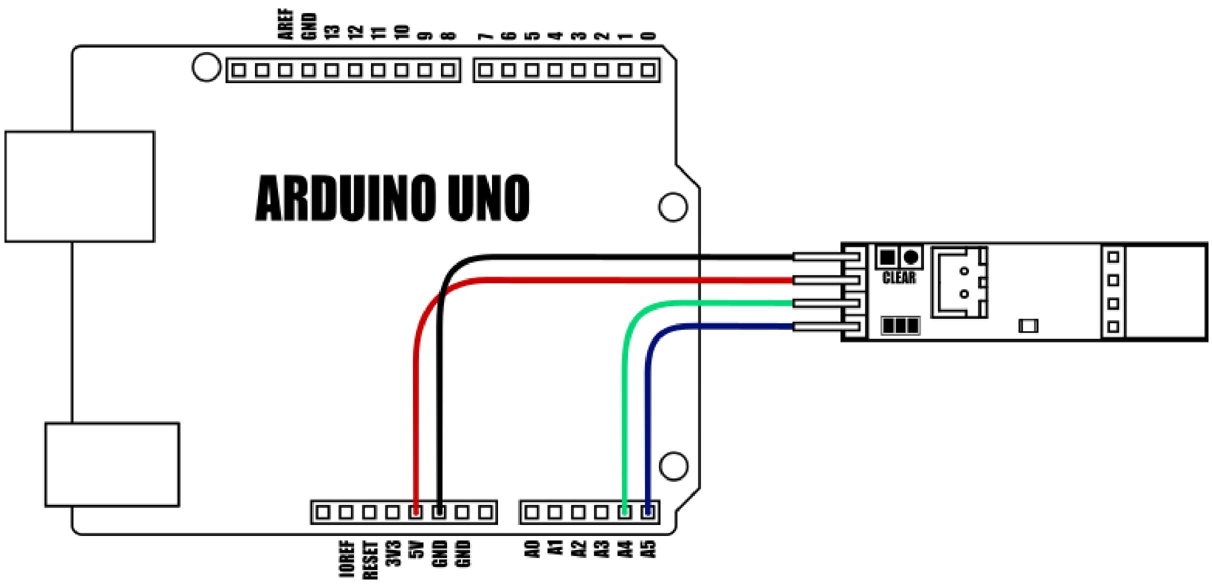
# Arduino Quick Start Guide and Examples for

# mLink NTC Temperature Sensor Module (HCMODU0186)

# Setting up your mLink module in 3 easy steps

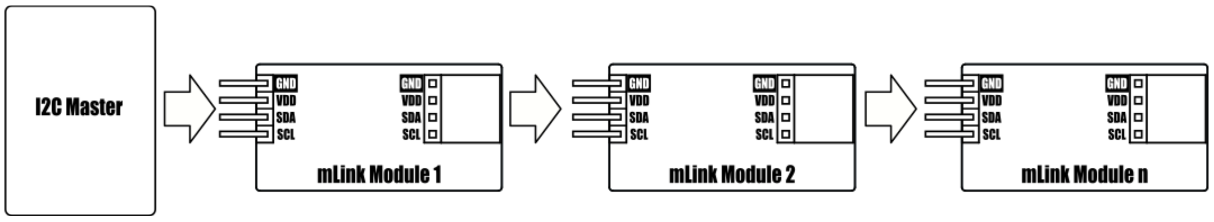
## Step 1: Connecting to your microcontroller



| DEVICE | VDD | GND | SDA | SCL |
|---|---|---|---|---|
| Uno/Nano | 3.3V/5V | GND | A4 | A5 |
| Pro Mini | 3.3V/5V | GND | A4 | A5 |
| Pro Micro | 3.3V/5V | GND | 2 | 3 |
| Mega | 3.3V/5V | GND | 20 | 21 |
| Due | 3.3V/5V | GND | 20 | 21 |
| Other microcontroller | 3.3V/5V | GND | I2C SDA | I2C SCL |

mlink modules can be connected to any microcontroller with an I2C (IIC) serial master interface. The above example shows a mLink module connected to an Arduino Uno's I2C interface. It can be powered via a 5V or 3.3V supply (depending on the logic levels you require the modules digital IO pins to operate at, and voltage requirements of any additional mLink modules connected to the modules I2C output).

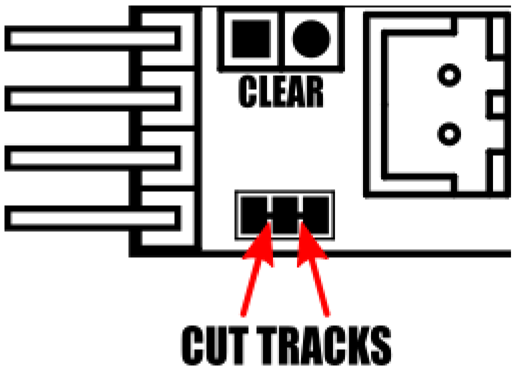
## Connecting multiple mLink modules



Each mLink module includes pullup resistors (10K), which are required for the SDA and SCL data lines. This allows up to 5* mLink modules to be connected directly to an I2C master without any additional hardware or modifications.

*note maximum number of modules will be dependent on data cable lengths and module power requirements.

If more than 5 mLink modules are required to be connected to a single master interface then the built-in 10K resistors will need to be removed from the additional modules.



The two 10K pullups can be removed from the I2C bus by breaking the tracks between the 3 pads shown in the diagram above. Should you need to reconnect the 10K pullups at a later date this can be done by bridging the 3 pads with solder.

The mLink NTC temperature sensor module comes with a preset I2C address of 0x54 (hex). When connecting multiple modules of the same type each module's I2C address must be unique. Therefore you must change the address of any additional modules to a unique address (valid I2C addresses range between 0x08 and 0x77) before linking them together. Changing a module's I2C address can be done via the module's I2C interface. For examples of how to do this, see the Changing I2C address section within this document.

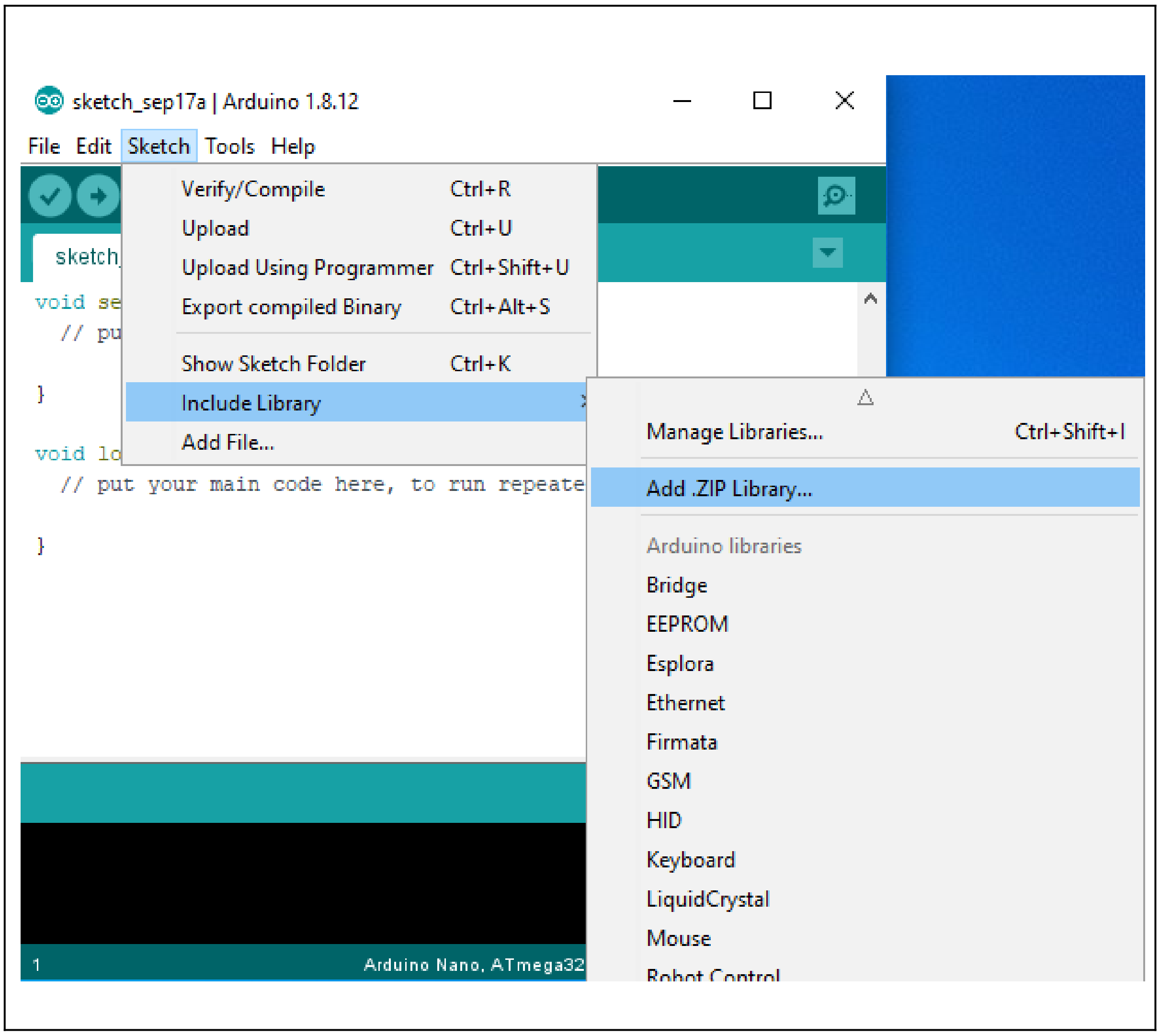
## Step 2) Installing the mLink Library

Adding the mLink library to your Arduino IDE can be done in the same way as any other standard Arduino library:

First download the mLink.zip file from the software section of our support forum here:

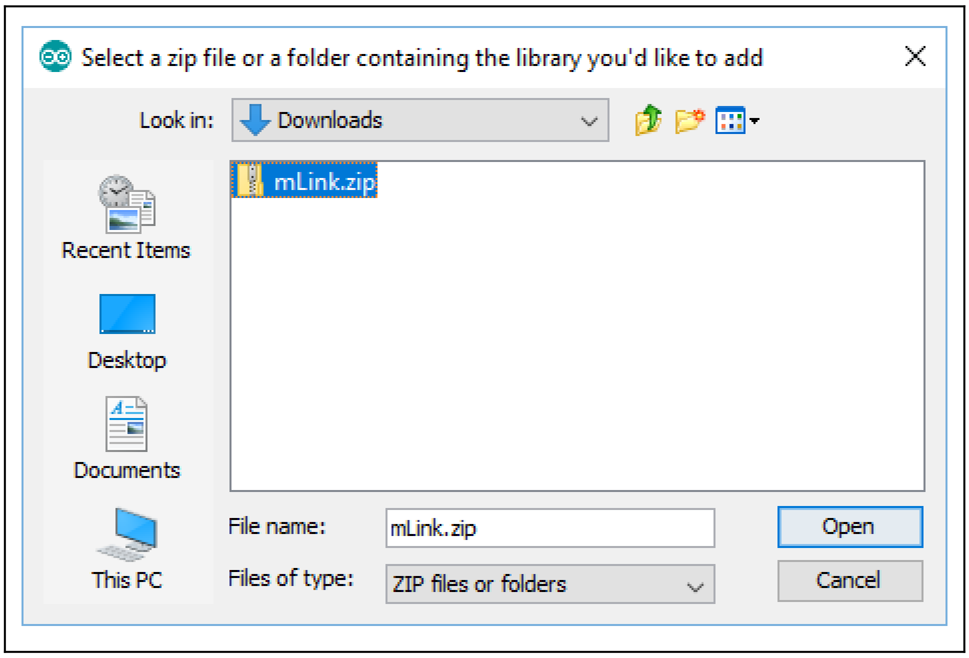
https://hobbycomponents.com/mLink

Once downloaded, open up your Arduino IDE and go to Sketch->Include Library->Add .ZIP Library.



```
void se
  // pu

}

void lo
  // put your main code here, to run repeate

}
```

In the file selection dialogue window that opens up, navigate to wherever you downloaded the mLink .zip file and select it, then click the 'Open' button.



## Step 3) Including the mLink library in your sketch

Adding the mLink library to your sketch consists of 3 steps; Firstly, include the mLink header file (mLink.h) at the top of your sketch, create an instance of the library, then finally initialise the library inside the startup() function:

```
// Step 1: Include the mLink library
#include "mLink.h"


//Step 2: Create an instance of the library
mLink mLink;


void setup()
{
  // Step 3: Initialise the library
  mLink.init();
}


void loop()
{
}
```

# Quick Start Examples

## Reading the temperature

The NTC sensor module will continually take measurements from the NTC sensor whilst it is powered and not in sleep mode. Therefore obtaining current temperature from the module is simply a case of reading the modules temperature registers to get the latest reading in centigrade.

The following example will repeatedly read the current temperature from the module and output the result to the serial port.

```
#include "mLink.h"

mLink mLink;

#define I2C_ADD 0x54

void setup()
{
  mLink.init();

  Serial.begin(9600);
}


void loop()
{
  float temp = mLink.NTC_Temp(I2C_ADD);      // Get the temperature in oC

  Serial.print("Temperature: "); Serial.println(temp);

  delay(1000);
}
```

## Low power mode

If you are using the module in a low power application you can reduce the amount of power the module consumes whilst a temperature measurement is not required. This can be done by using the mLink libraries mLink.sleep() command to put the module into a low power mode between measurements. Once in sleep mode making any I2C access to the module will automatically wake the device up from sleep.

Note that after waking the module out of sleep mode the busy flag in the status register will be set and will automatically reset once a new temperature measurement is made. Therefore to ensure that you have an up to date temperature measurement after waking the device from sleep mode you should first check that the busy bit is low before reading the modules temperature registers.

The following example will perform a temperature as per the previous example but will put the module into low power sleep mode in between measurements.

```
#include "mLink.h"

mLink mLink;

#define I2C_ADD 0x54

void setup()
{
  mLink.init();

  Serial.begin(9600);
}


void loop()
{
  while(mLink.busy(I2C_ADD));                    // I2C access will wake the module from sleep
                                                  // Next wait until busy is low before reading a
                                                  // new temperature.

  float temp = mLink.NTC_Temp(I2C_ADD);          // Get the temperature in oC

  mLink.sleep(I2C_ADD);                          // Put the module back to sleep

  Serial.print("Temperature: "); Serial.println(temp);

  delay(10000);                                  // Wait 10 seconds before reading again
}
```

## Changing the I2C address

To change the module's address you can use the libraries write() function.

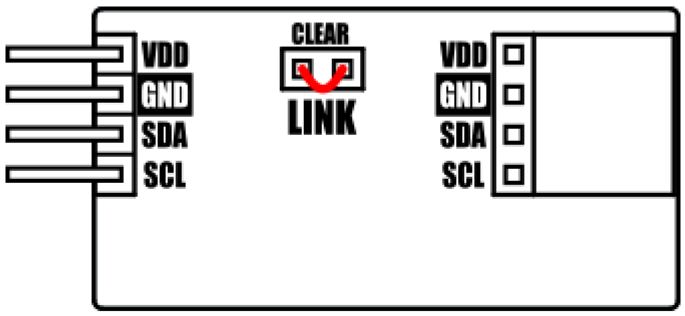
The following example changes the module's I2C address from the default 0x51 to 0x52. The new address will automatically be saved into non-volatile memory and so will retain the new address even after power has been removed from the module.

Before the module's address can be changed, the address register must first be unlocked by writing the byte value 0x55 followed by the byte value 0xAA to the register. The new address must then be written within 100ms of sending the 0xAA byte otherwise the unlock process will timeout and the process will then have to be restarted.

```
#include "mLink.h"

mLink mLink;

void setup()
{
  mLink.init();

  // Unlock the address register by writing 0x55 followed by 0xAA
  mLink.write(0x54, MLINK_ADD_REG, 0x55);
  mLink.write(0x54, MLINK_ADD_REG, 0xAA);

  // Change the I2C address from 0x54 to 0x55
  mLink.write(0x54, MLINK_ADD_REG, 0x55);
}

void loop()
{
}
```

## Factory Reset

Should you wish to restore the module back to its factory default configuration, this can be done by manually forcing a factory reset. All mLink modules include a set of pads labelled clear:



Note, exact location of clear jumper may vary on your module

To perform a factory reset carefully short the two pads together with a piece of wire or with something conductive such as a paperclip.

Whilst shorted, connect power to the module via the VCC and GND connections.

Wait a few seconds and then remove the short from the pads.

The module's settings, including its I2C address, should now be restored back to factory defaults.

## DISCLAIMER

The mLink range is a series of modules intended for the hobbyist and educational markets. Where every care has been taken to ensure the reliability and durability of this product it should not be used in safety or reliability critical applications.

This document is provided "as is". Hobby Components Ltd makes no warranties, whether express, implied or statutory, including, but not limited to, implied warranties of merchantability and fitness for a particular purpose, accuracy or lack of negligence. Hobby Components Ltd shall not, in any circumstances, be liable for any damages, including, but not limited to, special, incidental or consequential damages for any reason whatsoever.

## COPYRIGHT NOTICE