Three Ch Temperature Monitor App.

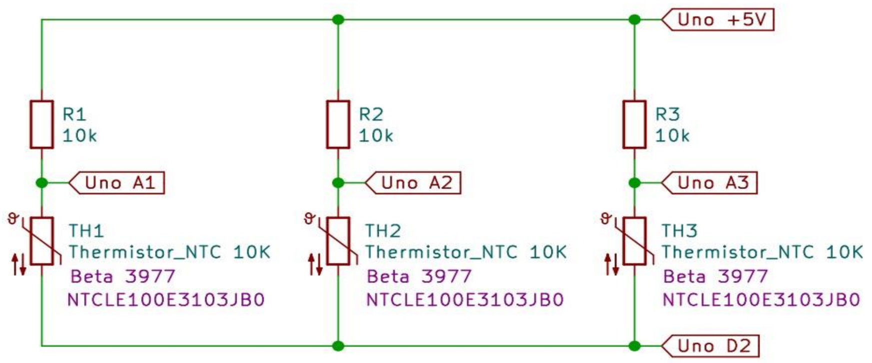
Parts used:

10K thermistor x 1 - 3

10K 1% or better resistor x 1 - 3

Arduino Uno

Connect the above parts as following:



Do not worry if the parts you have are different, as the App allows for variations.

The reason for the connection to D2 is to minimise thermistor self-heating.

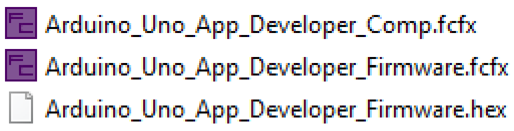
D2 only goes low when measuring temperature.

You do not need to use all three channels.

You could only use 1 x thermistor and resistor.

Before any of the free Apps for Arduino can be run for the first time, you will need to load the Arduino Uno App developer Firmware on to your Arduino Uno.

You can manually find The firmware for different target devices by going to the Wiki and searching for scada slaves. Or you can click on the link from Free-apps Website The zip file will contain three files:


Arduino_Uno_App_Developer_Comp.fcfx
Arduino_Uno_App_Developer_Firmware.fcfx
Arduino_Uno_App_Developer_Firmware.hex

Once loaded on to Arduino, any of the Arduino apps can be run without reloading the firmware.

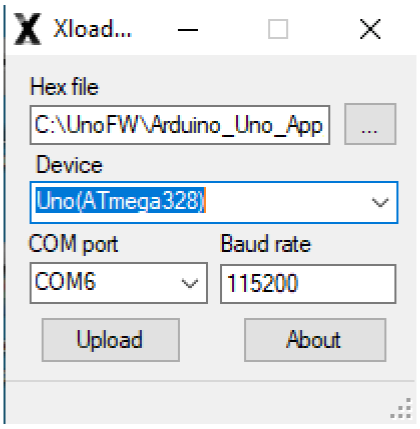
Download the firmware from the Temperature logger App website or the first link.

Extract the zip file into a folder of your choice.

Of course, you can use Flowcode to load the firmware then send it to the Arduino.

If you don't have Flowcode, then download XLoader from here.

XLoader enables you to upload hex files very easily:



Hex file: C:\UnoFW\Arduino_Uno_App
Device: Uno(ATmega328)
COM port: COM6
Baud rate: 115200

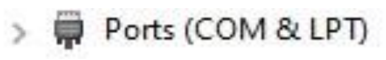
Use the three dots on XLoader to browse to where you extracted the Arduino_Uno_App_Developer_Firmware.hex zip file.

Select **Arduino_Uno**_App_Developer_**Firmware.hex**

Make sure Device is showing **Uno(ATmega328)**

Run **Device Manager** by right-clicking on the windows key and left-click on Device manager.

Expand:



Look for either Arduino Uno or USB Serial Device (depends on which drivers are used) and remember the Com port number.

Select correct com port in XLoader, then click on Upload.

You should see the RX and TX LEDs on the Arduino flash while data is being transferred.

If it is not, then make sure no other software has taken the port over.

After the transfer is complete, you can run any of the App

developer's software.

Run the App by double-clicking on the Three Ch Temperature

Monitor v1.6.bat file. You can also right-click on the bat file and

select **Send to,** then **Desktop (create a short-cut)**

The mouse scroll wheel can zoom in & out.

When happy with the size, right-click, select **Look at the origin**, Which will centralise the App.

Right-click on an empty background area and select properties.

The properties can be undocked by right-clicking and selecting Floating or Left-click and dragging.

Measure and enter the resistor values for the channels you are using before placing them on your hardware.

Use the datasheet for the thermistor/s you must obtain and enter the beta values.

Make sure the correct Resistor values and thermistor values at 25C are entered in the properties.

Finally, make sure the correct **COM port** is selected.

Left-click on **Go**, then click on 4 of the keypad to save all the settings.

That will generate a config.cfg file that contains all the property settings values.

When you next load the App, the settings will be automatically loaded.

To start measuring and logging temperature, left-click on Start/Stop

If the App Developer runs successfully, then both RX and TX LEDs on the Arduino will constantly flash, and the temperature will be displayed and graphed.

When the App starts to log the temperatures, after clicking Start/Stop, a new data1.csv file with the logged data is saved on the App's root directory.

Therefore, if you require this data, copy and paste the file as soon as you have stopped the logging.

All the logged data can also be viewed within the console (**View** menu, select **Consoles**).

Undock the console so the App Developer panel can be made larger.

Data can be viewed by selecting the **Console Writer** tab.

To rerun the App Developer, select the **Contr**ol tab.

Troubleshooting.

When App is logging, and you get unexpected results.

Unexpected results would mainly be down to a communication issue.

Other causes could be hardware issues, e.g. faulty potentiometer or Arduino or a software bug.

There are two ways of determining if it is a communication issue.

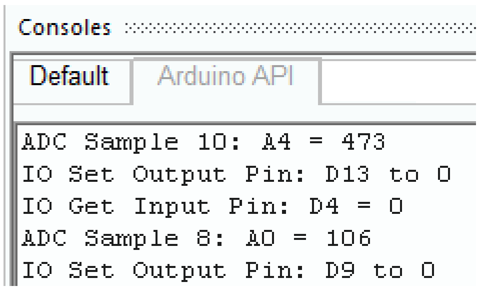
If both TX and RX LEDs are not flashing while App Developer is logging.

Or open the console (View menu, select Consoles).

Undock the console so the App Developer panel can be made larger.

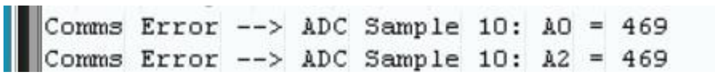
Select **Arduino API** tab.

If communication OK, you will see something like:



```
ADC Sample 10: A4 = 473
IO Set Output Pin: D13 to 0
IO Get Input Pin: D4 = 0
ADC Sample 8: A0 = 106
IO Set Output Pin: D9 to 0
```

If there is an issue with communication or firmware, then you will see:



```
Comms Error --> ADC Sample 10: A0 = 469
Comms Error --> ADC Sample 10: A2 = 469
```

If it is a communication issue:

Is the correct comport selected within properties?

Has some other software taken over the port?

Is only one instance of App developer run at a time?

With the hardware issue, so long as communication is confirmed working, measure the pin's voltage that is giving unexpected results.

It should be steady and not floating.

If the voltage is correct, then make sure the Arduino has the Arduino App developer firmware programmed.

For further advice, there will be help available via the forums.