MTR-Duino User’s Guide (Rev 5.2)

Table of Contents

[1. Background 2](#_Toc500860041)

[2. Highlights 2](#_Toc500860042)

[3. Specifications 2](#_Toc500860043)

[3.1. Temperature 2](#_Toc500860044)

[3.2. Data 3](#_Toc500860045)

[3.3. Power 3](#_Toc500860046)

[3.4. Real Time Clock 3](#_Toc500860047)

[3.5. Physical Parameters 3](#_Toc500860048)

[4. Overview 3](#_Toc500860049)

[5. Interface 3](#_Toc500860050)

[6. Preparing the Instrument 4](#_Toc500860051)

[6.1. Batteries 4](#_Toc500860052)

[6.2. microSD Card 4](#_Toc500860053)

[6.3. Configuring the Instrument 4](#_Toc500860054)

[6.4. Configuration Instructions 5](#_Toc500860055)

[6.5. Battery Estimates and Sample Intervals 7](#_Toc500860056)

[6.6. A note on Self-Heating Effects 7](#_Toc500860057)

[6.7. Installing Into Pressure Housing 8](#_Toc500860058)

[6.8. Sealing the Pressure Case 9](#_Toc500860059)

[6.9. Deployment Considerations (Dissimilar Metals) 9](#_Toc500860060)

[7. Post-Recovery Procedures 9](#_Toc500860061)

[7.1. Opening the Case 9](#_Toc500860062)

[7.2. Post-Calibration of Clock 9](#_Toc500860063)

[8. Data 10](#_Toc500860064)

[8.1. Recovering Data 10](#_Toc500860065)

[8.2. Data Format 10](#_Toc500860066)

[8.3. Converting Data to Temperature 10](#_Toc500860067)

[8.4. Reference Resistor 10](#_Toc500860068)

# Background

The MTR-Duino was developed by EcoFOCI as a low-cost replacement option for PMEL’s MTR (Mini-Temperature Recorder). The original MTRs were designed by EDD circa 1980 and many units have failed after several decades of use. Repairing old units was not a viable option due to complexity, time, and obsolete components. However, re-using the MTR titanium pressure housings was highly desirable to minimize cost. The goal was to design units which cost ~$100 each with ±0.01°C accuracy - as the least expensive commercial alternative, the SBE56, cost ~$650 each. When replacing several dozen units, as desired, the cost savings are in the tens of thousands of dollars.

** **

# Highlights

* Temperature recording device
* Absolute Accuracy: ±0.01°C or better
* Low Cost: ~$120 per unit (not including housing)
* Utilizes ”Arduino Micro” development board

# Specifications

## Temperature

Absolute Accuracy: ±0.01°C or better (within calibrated range)

Range: -5°C to +70°C

Resolution: 0.0006°C *@ -2°C*

0.001°C *@ 10°C*0.002°C *@25°C*  
0.01°C *@50°C*

## Data

Storage: microSD or microSDHC card: 1GB to 32GB (FAT16 or FAT32)

Sample Limit: 67,897,351 samples\* (4GB micro SD Card)

\*Limited by battery life, not storage capacity

## Power

Primary Batteries: 2 Standard AA (Lithium or Alkaline)

RTC Battery: 12.5mm Lithium Coin Cell (Energizer CR1220=40mAh)

Sampling Current: 70mA @2 seconds/sample

Sleep Current: <15A

## Real Time Clock

Unit: DS3234: RTC with Temperature Compensated Crystal Oscillator

Maximum Error: 1.05 minutes/year (0°C to 40°C)

1.84 minutes/year (-40°C to 85°C)

## Physical Parameters

Housing Material: 6 Al-4V Titanium\*

Depth Rating: 15,000 meters\*

Diameter: 2.0 in (5.1cm)\*

Length: 5.4 in (13.7 cm)\*

Weight in Air: 1.41 lbs (640 g)\*

Weight in Water: 0.78 lbs (358 g)\*

Thermal Time Constant: 4 min\*

\*Info taken from original MTR Manual

# Overview

The MTR-Duino measures temperature with a precision thermistor and a very high accuracy Analog to Digital Converter. It also contains an ultra-precise reference resistor, which is measured during each sample and should be used to remove any drift or inaccuracy in the ADC. The processing is all done via an Arduino Micro development board, which contains an ATmega32u4 microprocessor. The Arduino Micro vastly simplified development and programming, but does bring in some features that are distinctly different than a traditional Serial/RS-232 based instrument.

# Interface

The MTR-Duino is configured with a terminal program (such as Putty) on a Serial COM Port. Refer to section 6.4 for Serial COM Port Settings. The MTR-Duino connects to a computer using a Micro-USB to USB Cable.

Although a terminal program and Serial comms are used, the Arduino Micro is actually a USB device which installs drivers and mounts to a computer as a virtual COM Port. This requires the user to select the correct COM Port (which may change between units) to communicate with the device properly. Since it is a USB device, the configuration differs slightly from traditional Serial instruments. Refer to Sections 6.3 and 6.4 for details.

The Arduino-Micro (ATmega32u4 microprocessor) is programmed and the MTRDuino’s firmware is installed using a separate program, the Arduino IDE (Version 1.8.5). This should only need to be done if new instruments are built or the firmware is updated. More information is contained in the MTR-Duino Engineer’s Guide if needed.

# Preparing the Instrument

## **Batteries**

Two standard 1.5V AA Batteries are used as the primary power source. The unit has a very efficient boost converter circuit, which will allow the unit to operate as long as the total voltage is at least 1.8V. Lithium batteries may be preferable for full length deployments because they approximately 2.3 times longer, but Alkaline batteries may also be used for calibrations or testing if desired.

**Since the Instrument does not keep track of battery usage, primary batteries should always be replaced before deployment.**

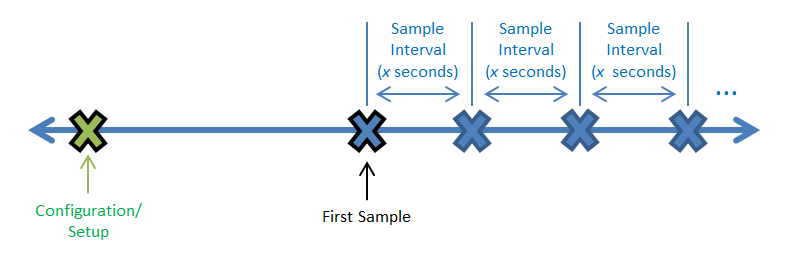
A 12.5mm lithium coin cell battery is used to power the Real Time Clock and wake up the unit at precise intervals when deployed. This battery should last a very long time (40+ years), but should be replaced every 5-10 years for prudence. It can also be measured with a voltmeter – the minimum voltage for operation is 2.3V.

## microSD Card

A standard microSD card is used to store data from deployment. This card can simply be removed from the instrument after deployment to transfer data very quickly. Either SD or SDHC (High Capacity) cards may be used, and as such should not be greater than 32GB in size. SD Cards with a higher capacity than 32GB (SDXC) use a different formatting structure (exFAT) and are not compatible with the firmware (FAT16 or FAT32 is required).

## Configuring the Instrument

The MTR-Duino uses a very simple sampling scheme, waking up to take a sample every *x* seconds. The user must also set a “First Sample” date and time to specify when the instrument should start sampling. **The First Sample time and date must be set for a time/date in the future and the instrument must be unplugged before the First Sample time transpires.** Setting a first sample time/date in the past will not notify the unit to wake up and no data will be collected.



The user can configure and check the functionality of the instrument using a terminal window as described in *Section 5 - Interface*. The commands are displayed in the terminal window as shown below. They are issued with by sending a single character command to the unit and then following the prompts. The command dialog should display again once any prompts have been completed.

**Note that the sample interval is stored on the SDCard. The SD Card must be inserted and functioning correctly in order to set or change the sample interval.**

Once parameters are set properly and the instrument status indicates the proper time, first sample, and sample interval; simply unplug the instrument from the USB cable. **Plugging the instrument back in will turn power on to the unit and reset the first sample time/date, even without manually changing the settings in the terminal window.** This is because the USB cable provides power to the Arduino Micro and the instrument cannot distinguish between USB and battery power – it will simply wake up and begin collecting samples at the programmed interval.

## Configuration Instructions

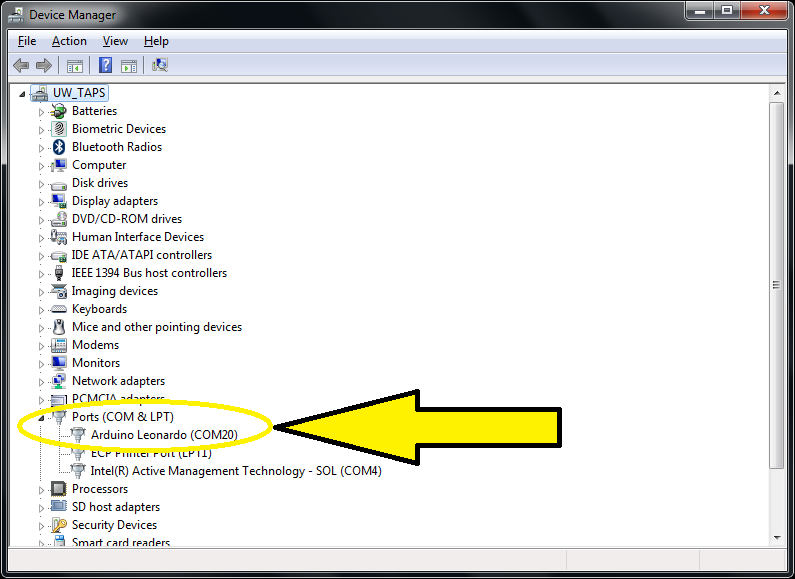
1. **Connect the Instrument**

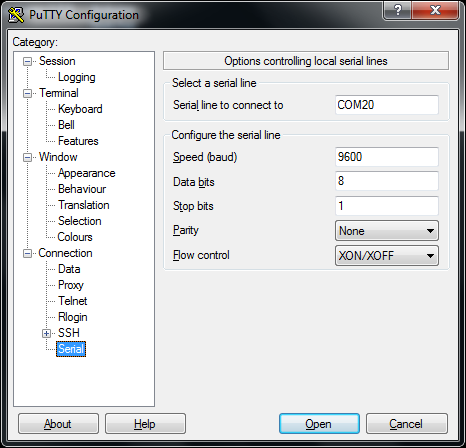
* Connect the MTR-Duino to the computer with a Micro-USB Cable. You should see a steady yellow LED lit on the top of the Arduino Micro and a steady Blue LED lit on the underside of the Arduino Micro
* Use only Anker brand or other high quality USB Cables.

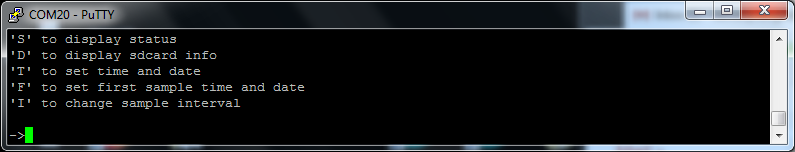
 

1. **Open the Serial Terminal**

* Open a terminal program (such as Putty) using the settings below.
  + COM Port: Check “*Device Manager🡪Ports*” to find correct COM Port Number
  + Baud Rate: 9600
  + Data Bits: 8
  + Stop Bits: 1
  + Parity: None
  + Flow Control: None

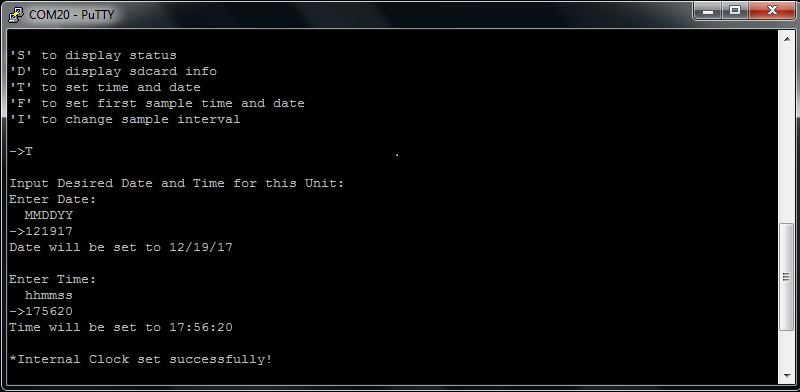


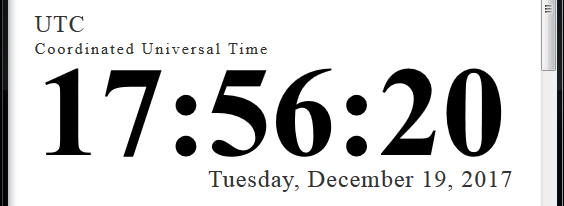




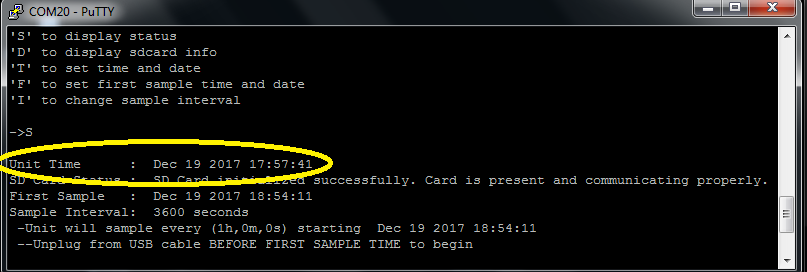
1. **Set the Unit Time and Date**

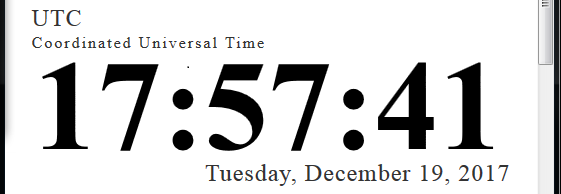
* Press ‘T’ and synchronize time with a known clock





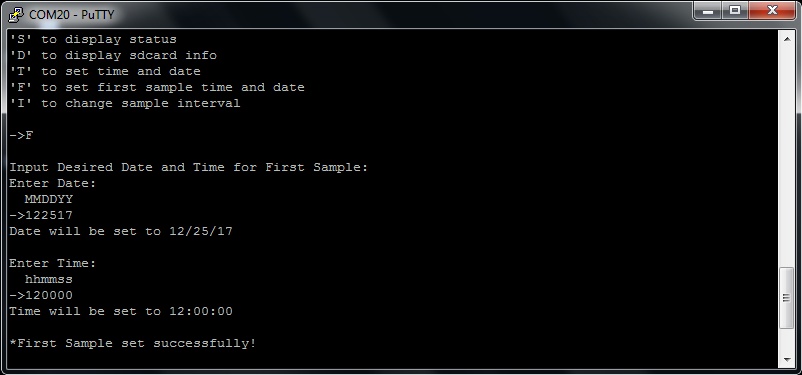
* Press ‘S’ to verify time has been set properly and clock is running



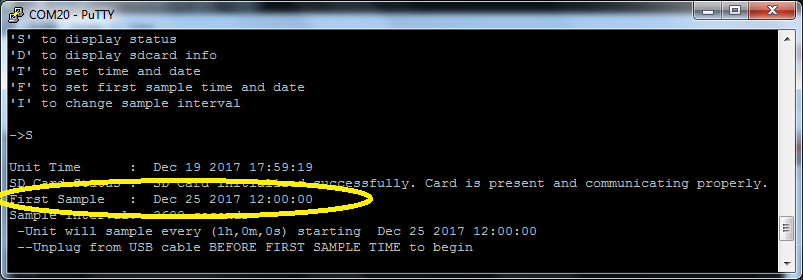


1. **Set the First Sample Time and Date**

* Press ‘F’ and enter date and time of first sample

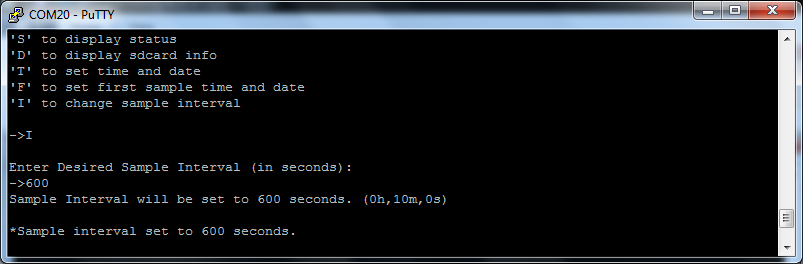


* Press ‘S’ and verify first sample has been set properly

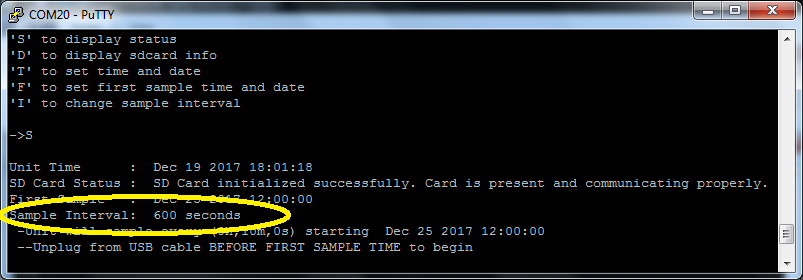
**

1. **Set the Sample Interval**

* Press ‘I’ and set the sample interval (in seconds)
  + Refer to Section 6.5 for estimates of sample intervals vs. battery life

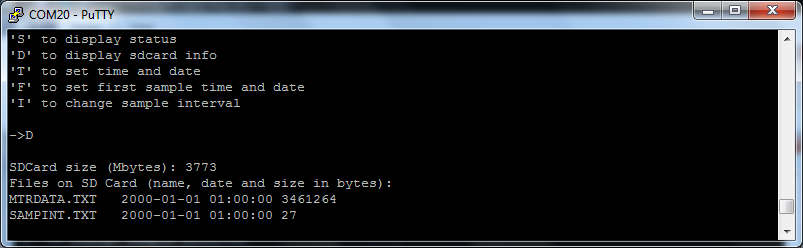


* Press ‘S’ and verify sample interval has been set properly

**

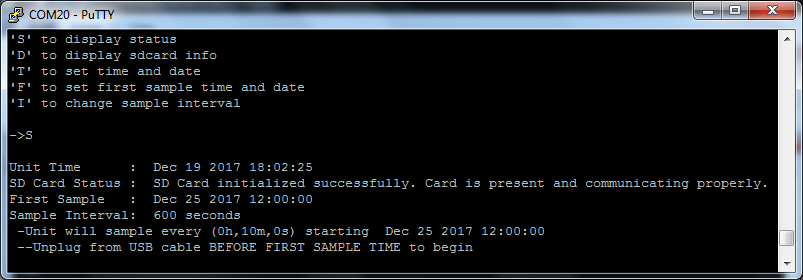
1. **Check the SD Card**

* Press ‘D’ and verify that files ‘MTRDATA.TXT’ and ‘SAMPINT.TXT’ are present



1. **Verify all Settings**

* Press ‘S’ and verify all settings

**

1. **Unplug Instrument from USB**

* Unplug the instrument from the USB Cable before the first sample time to begin
  + Instrument LEDs will light up for approximately 1 second during each sample. This can be used as additional verification that the instrument is operating properly.

## Battery Estimates and Sample Intervals

The tables below show estimated battery life for various sample intervals. They are shown for both lithium and alkaline batteries, and assume the worst case scenario by de-rating for a temperature of -2°C for the entire deployment. *This table should be verified and update after instruments have been in use for a few seasons.*





## A note on Self-Heating Effects

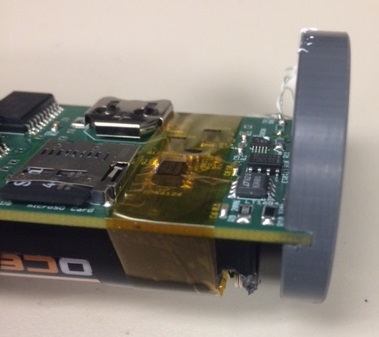
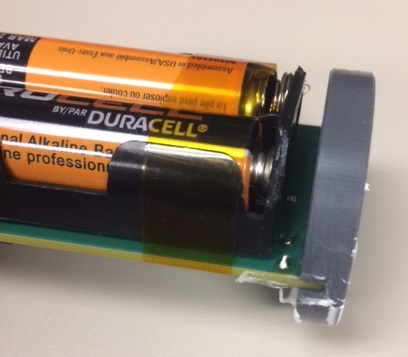
**In order to eliminate the possibility that self-heating of the thermistor causes errors in calibration or deployment data, the sample interval should always be at least 30 seconds, especially during calibrations.**

The miniscule amount of current running through the thermistor (<4A) can still cause small errors on the order of 0.01°C - 0.02°C if the sample interval is faster than 30 seconds. Since the mass of the MTR gives the instrument a thermal response time of several minutes, limiting the sample interval to 30 seconds should not affect data quality in any practical applications.

## Installing Into Pressure Housing

Before the MTR-Duino is installed into the pressure housing, tape a small desiccant pack the underside of the instrument, next to the battery clip. This should ideally be done using anti-static tape (kapton tape), but standard electrical tape can also be used.

Place a small band of tape around the lower end of the battery clip to help secure the clip itself to the board. **This should be done using anti-static tape (kapton tape)** since this tape goes directly over the highly sensitive analog portion of the circuit.

**** 

Ensure that the thermistor is mounted with a thermal epoxy or thermal paste. This will help transfer heat from the pressure housing directly to the thermistor, reducing the thermal time constant of the instrument and also reducing any self-heating effects from current through the thermistor.

Once the instrument is prepped, simply slide the electronics into the case until the Velcro can be felt to engage.

## Sealing the Pressure Case

The O-ring and its seating surfaces should be cleaned and lightly lubricated with O-ring grease. The case should be filled with dry gas (and/or a desiccant pack) and the endcap screwed on until it is securely seated.



## Deployment Considerations (Dissimilar Metals)

The instrument case can be secured to anything with clamps, tape, or bailing wire. Just be very careful not to have any other metal in contact with the case, as most other metals are sacrificial to titanium. If stainless hose clamps are used, insulate them from the case with heat shrink tubing or electrical tape.

# Post-Recovery Procedures

## Opening the Case

The MTR-Duino should be rinsed thoroughly with fresh water and dried around the cap before opening. Allowing the unit to come to ambient temperature before opening will prevent condensation from forming on the board. If any water has leaked into the pressure case during deployment, or if there have been any problems with the lithium batteries, potentially dangerous pressure could be built up inside. This is not a major issue of concern as the case will be able to hold the pressure and as soon as the cap is opened enough to unseat the O-ring, the pressure will be discharged through the safety groove in the cap threads. As always, just use caution not to open the case near any sparks or flames that could ignite gasses inside.

## Post-Calibration of Clock

If the user has synchronized the MTRDuino’s clock with a known clock, then the two clocks can be compared to correct for any drift. The maximum drift in the DS3234 RTC when operating anywhere between -40°C and 85°C is 1.84 minutes/year. Even if the unit’s primary batteries have died, the time will be retained because of the backup coin battery.

Simply connect the unit as described in *Section 5 - Interface*, Press ‘S’ for Unit Statusand record the unit’s time vs. the known clock at the same moment.

# Data

## Recovering Data

Simply remove the microSD card from its slot and connect it to a computer with a microSD card reader.

Sometimes when using microSD adapters, you’ll find that not all adapters are compatible with every microSD card. In this case, your computer might read that an adapter has been inserted, but it might not be able to access the data on the card. **Do not reformat the card if this happens -- even if the computer asks you to format it -- because formatting will erase all of the data on the microSD card**. Just try another adapter or see if any updated software drivers are available for the adapter you have.

## Data Format

Data is stored on the SD Card in the following format:

MM/DD/YYYY hh:mm:ss, x1, x2, …, xn, y

Where:

*MM =* Month

*DD =* Day

*YYYY =* Year

*hh =* hour (24 hour format)

*mm =* minute

*ss =* second

*x =* ADC reading from thermistor

*n =* number of measurements per sample (normally 5)

*y=* ADC reading from reference resistor

## Converting Data to Temperature

The data stored on the SD Card is a voltage reading from the unit’s ADC (Analog-to-Digital Converter). The voltage value is fit directly to a Steinhart-Hart curve during the calibration process so that no accuracy is lost by making conversions. Temperature can be calculated using the following equation:

Where:

T = Temperature in °C

x = ADC reading

A,B, and C = Steinhart-Hart Coefficients

## Reference Resistor

Describe reference resistor and offset calculation here