# Thunderbolt Monitor By M1DST

## Overview

Thunderbolt Monitor is a stand-alone microprocessor-controlled LCD specifically for Trimble’s Thunderbolt disciplined clock, providing a comprehensive indication of the Thunderbolt's status, modes, and alarm conditions.

Data packets appearing on Thunderbolt's serial port is HEX data in Trimble Standard Interface Protocol (TSIP), not NMEA sentences, so this display will not work with NMEA GPS units, such as Trimble's Jupiter or any other GPS.

Ideal for amateur radio applications, Thunderbolt Display shows Time Of Day (UTC) to assist with logging contacts, and also calculates Maidenhead Grid Locator Square from the current latitude and longitude.

## Features

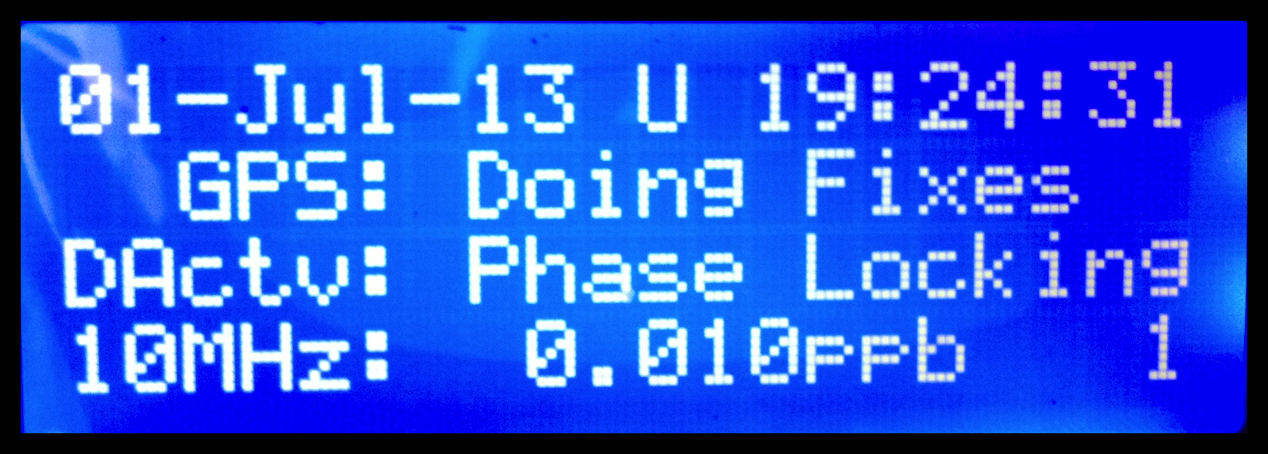
* 4 line x 20 character backlit LCD
* 5 information pages
* Alarm outputs (LEDs)
* NTP Server
* 3 Programmable buttons to send commands
* Open source hardware and software
* Shield for use with Netduino or Arduino
* Required 7.5 – 12V input (or USB powered)

## Display Pages

* Page 1: Status
* Page 2: Mode/Survey
* Page 3: Location
* Page 4: Alarms
* Page 5: Network configuration
* Splash: Shown during boot time
* Error: Shown when no data is being received.

## Sample Screen

This is an example of a screen from my Thunderbolt Monitor.



## Parts List

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Type | Part | Value | Comment | Packed | Installed |
| PCB | PCB | - |  |  |  |
| Capacitor | C1, C2, C3, C4 | 1uF |  |  |  |
| LED | LED1 | Green | Power |  |  |
| LED | LED2 | Blue | RS232 |  |  |
| LED | LED3 | Orange | Minor |  |  |
| LED | LED4 | Red | Major |  |  |
| Transistor | Q1 | 2N2222 | NPN |  |  |
| Resistor | R1 | 2K |  |  |  |
| Resistor | R2, R9, R10, R11 | 330R |  |  |  |
| Resistor | R3 | 620R |  |  |  |
| Resistor | R4, R6 | 1K |  |  |  |
| Resistor | R5 | 3.3K |  |  |  |
| Resistor | R7 | 4.7K |  |  |  |
| Switch | S1, S2, S3, S4, S5, S6 | - |  |  |  |
| Header | JP1 | 16 pins | LCD |  |  |
| Header | JP2 | 3 pins | RS232 |  |  |
| Header | JP3 | 5 Pins | External LEDs |  |  |
| Header | JP4 | 7 Pins | External buttons |  |  |
| Header | - | 16 Pins | Solder to LCD |  |  |
| IC | U1 | MAX232CPE | Use provided socket |  |  |
| Cable | - | 16 core ribbon cable | Use with LCD |  |  |
| LCD | LCD | 4x20 LCD |  |  |  |
| Header | - | Shield Headers | 4 to install. |  |  |
| Pot | - | 10K Pot | LCD Contrast |  |  |

Notes:

You don’t have to install JP3 or JP4 unless you are going to use external LEDs and/or buttons. If you are installing external LEDs then you shouldn’t install the on-board ones. You may install on-board and external buttons if you wish.

I recommend you tick off the components as you install them.

## RS232 Plug

The Thunderbolt uses a DB9 to connect to the monitor via JP2. You will need to connect to pins 2, 3 and 5 of the DB9. The wiring is clearly labelled on the PCB.

Pin 2 – TX data from the Thunderbolt

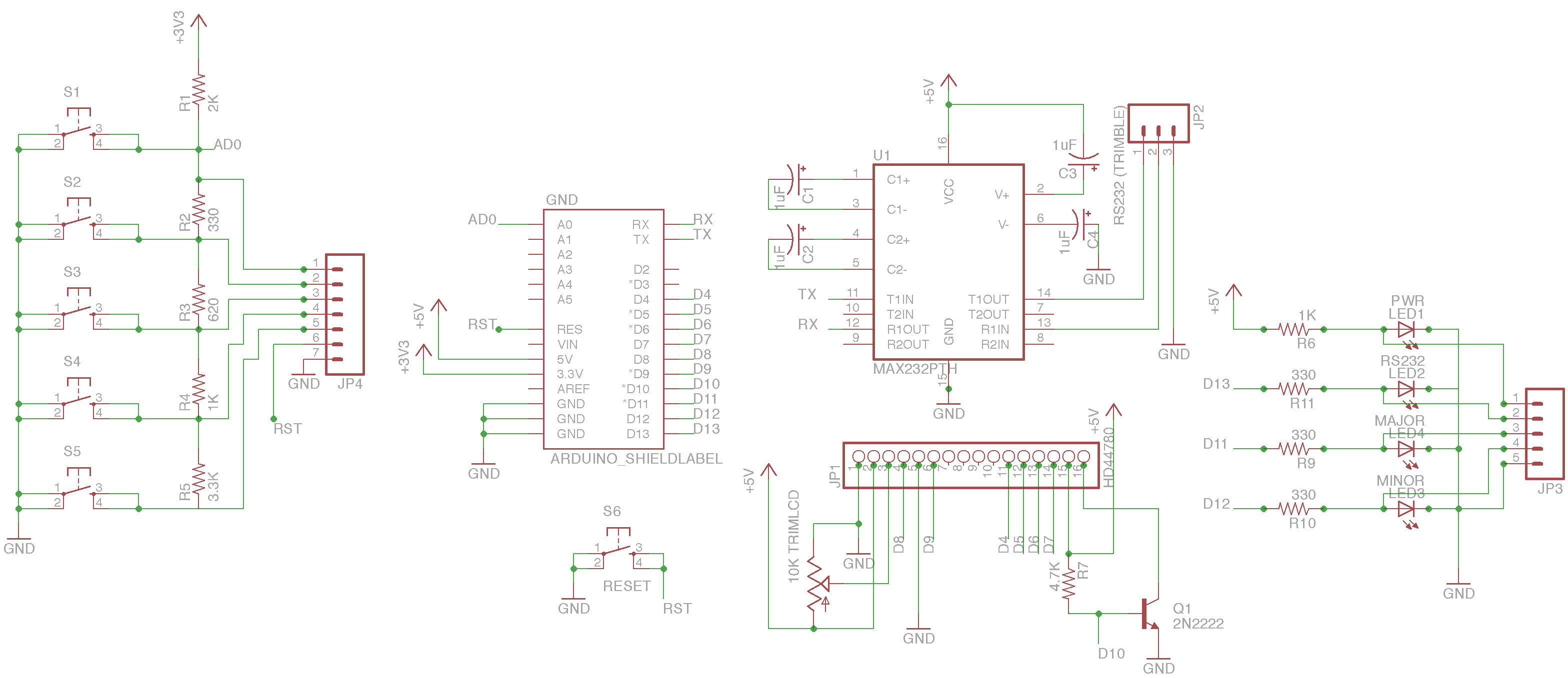
Pin 3 – RX data to the Thunderbolt

Pin 5 – GND

The Thunderbolt Monitor can be wired in parallel with another DB9 that connects to a PC/laptop.

The monitor only accepts data at the Thunderbolt's default serial port parameters. (9600 baud, 8 data bits, no parity, 1 stop bit)

## Schematic



## Buttons

There are 6 buttons in use for the Thunderbolt Monitor. The buttons operate as follows:

|  |  |
| --- | --- |
| **UP (S2)** | Navigates forwards through the Monitor’s screens |
| **DOWN (S3)** | Navigates forwards through the Monitor’s screens |
| **BTN1 (S4)** | Switches the receiver into Full 3D Position mode |
| **BTN2 (S1)** | Switches the receiver into Over Determined Clock mode. |
| **BTN3 (S5)** | Starts a Site Survey. |
| **RESET (S7)** | Forces the Netduino to reboot. You may need this is your unit locks up. |

Over Determined Clock mode is the optimum mode for Time-only fixes and is important for disciplining the internal 10MHz oscillator to the best accuracy. It is does not report position whilst in this mode.

You would need to switch to Full 3D Position mode if you wish the Latitude and Longitude to update if you were to be mobile.

## Deploying to the NETMF board.

This project has was originally designed for the Netduino Plus 2. The Netduino is an open source electronics platform using the .NET Micro Framework. Since then I have created a version without the NTP functionality for the original Netduino, Netduino 2, and GHI FEZ Lemur. If you have a Netduino 3 then you should also be fine.

The project requires your board to be at version 4.3 of the firmware.

To deploy or modify the Monitor you’ll need some free tools.

**WARNING: You should NOT use Visual Studio 2015. It will not work. Use Visual Studio 2013 instead. You can get the free version from** [**http://download.microsoft.com/download/2/5/5/255DCCB6-F364-4ED8-9758-EF0734CA86B8/wdexpress\_full.exe**](http://download.microsoft.com/download/2/5/5/255DCCB6-F364-4ED8-9758-EF0734CA86B8/wdexpress_full.exe)

If you are installing on a Netduino, follow the instructions at <http://developer.wildernesslabs.co/Netduino/About/Downloads> to ensure you have all the prerequisites.

If you are installing on a GHI FEZLemur, follow the instructions at <https://www.ghielectronics.com/support/netmf> to ensure you have all the prerequisites.

Open TrimbleMonitor.sln using Visual Studio.

There are currently three projects in the solution.

* TrimbleMonitor (Fez Lemur) – Basic monitor for the GHI FEZ Lemur (No NTP)
* TrimbleMonitor (Netduino) – Basic monitor for any Netduino 1,2,3 (No NTP)
* TrimbleMonitor (Netduno Plus) – Monitor with NTP support for a Netduino 2+/3 Ethernet/Wifi

You should right click the project which fits your hardware and click “Set as StartUp project” within the solution explorer.

By default, Visual Studio runs projects in an emulator. This allows software developers to create and test programming logic for a new hardware product before the actual hardware is built. We won’t use the emulator for our purposes, so we’ll let Visual Studio know that we have physical hardware it should use instead.

Click on the Project menu and select your project’s properties (generally, the last item in the Projects menu). When the project properties appear, click on the “.NET Micro Framework” category on the left side.

Now we will change our deployment target from the Emulator to the Netduino/Lemur. Change the Transport from “Emulator” to “USB” and then make sure that the Device selection box shows your device. If it doesn’t, unplug and re-attach it.

Now, we’ll run the project. When we run the project, the code is deployed to the board and then automatically started.

To run your project, press the “Start Debugging” button in the toolbar at the top of the screen. It looks like the Play button on a music player. You can also press F5 instead.

Visual Studio will now deploy the Trimble Monitor to the hardware. In a few seconds, you’ll see the LCD fire into life and the blue LED blinking on and off every time it receives data from the Thunderbolt.

When you ran the program, it was written to the microcontroller so all you have to do to run the program again is plug it in via a micro USB cable or with a power adapter (using the power barrel jack).

You should see the splash screen appear, the LEDs flash in a pattern and the then the main program start to show you the status of your Thunderbolt.

Congratulations, you now have a working Thunderbolt Monitor.

## Photos

I am compiling photos from fellow builders which I plan to display on my website. If you would like to see your project shown on my site then please email me your photos and I will display them at <http://www.m1dst.co.uk>. My email address is [james@m1dst.co.uk](mailto:james@m1dst.co.uk).

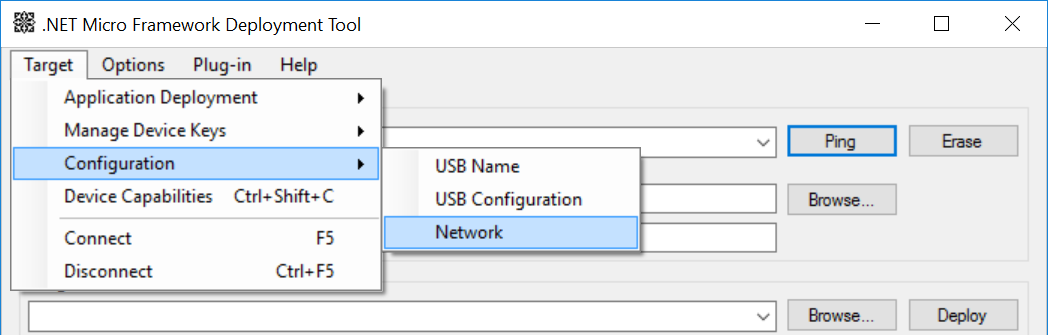
I hope you get as much enjoyment from this project as I did putting it together.

James, M1DST

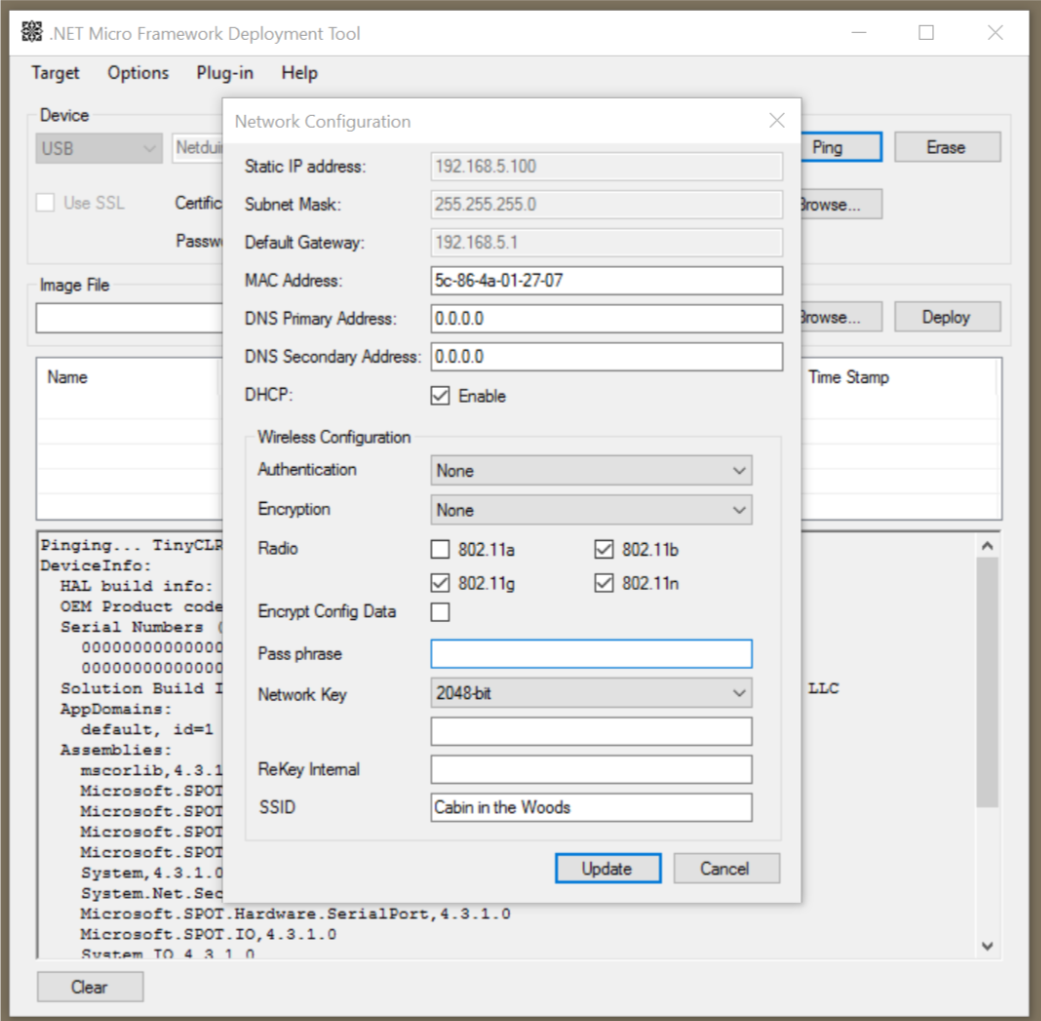
## Networking

Many network settings, such as which WiFi network to connect to, or whether or not to use DHCP (or a static IP), are configured at deploy time via MFDeploy on Windows (Mac network configuration support is in progress).

MFDeploy is included in the .NET MicroFramework installation. To configure network settings on Netduino, launch MFDeploy from the start menu and then access the settings via the Target > Configuration > Network menu:



From there, Network settings are configured in the popup. For example, the following settings will configure a WiFi enabled Netduino 3 to connect to the open wireless network “Cabin in the Woods” using 802.11 b, g, or n radio and use DHCP to obtain an IP Address



Note that if you get the error unable to erase configuration sector memory, when trying to save, you may have to close the network settings dialog and run Plug-in > Reboot Stop from the menu first.