Laboratory 8: Custom Firmware for xDot
Instructor: Young H. Cho

1. Use the USB port on xDot to connect to the PC. The status light will turn on, indicating it has power.

2. Go to https://developer.arm.com/downloads/-/gnu-rm/10-3-2021-10

   Download and Install GNU Arm Embedded Toolchain

3. Go to https://os.mbed.com/studio/

   Download and Install Mbed Studio

4. Run Mbed Studio, then select "MultiTech xDot (STM32L151)" under Target

5. Start loading a new program by clicking on "+New program"

6. Under Example program, select "mbed-os-example-blinky" then click on Add Program

7. Once the example code is downloaded, examine the code "main.cpp"

   The files include libraries required to drive and run the xDot. The lib mbed-os provides abstract functions to communicate with xDot. The file, main.cpp, contains a hello world example code, where it blinks LED1 with a 500ms delay and reports the stats every 10 seconds.

8. Click on a little hammer to compile the program. For debugging, set the compiler on the DEBUG mode.

9. Once it is completed the compilation, search for "Mbed Programs" folder and verify that the subfolder "mbed-os-example-blinky/BUILD" contains a newly compiled program "mbed-os-example-blinky.bin"

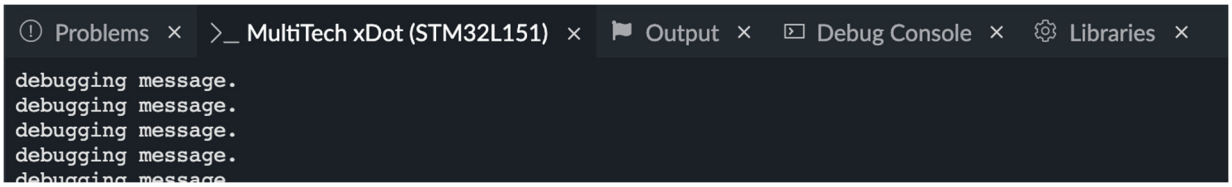
10. Copy and Paste/Drag the "mbed-os-example-blinky.bin" into a small storage folder for xDot. If you Cannot copy/paste the .bin file output from Mbed studio into the xDot, you can program the device by just clicking the run button in the Mbed Studio.

11. Once the transmission to xDot is complete (make sure the transmission did not fail) hit the reset button on the xDot.

12. For Windows PC, please install the driver from https://os.mbed.com/handbook/Windows-serial-configuration, and reboot the PC.

13. Stats are sent to the debug port
    Download a terminal program like Putty, Teraterm, or minicom on Windows or use "cu" on Linux to connect to xDot's debug port through UART
    https://www.cyberciti.biz/hardware/5-linux-unix-commands-for-connecting-to-the-serial-console/

    For MacOS, use Mbed Studio for checking the debug messages. (Under the xDot label) as following figure.



Specify the baud rate to 9600, Data bits to 8, Parity to N, Stop bits to 1, and Flow Control to OFF

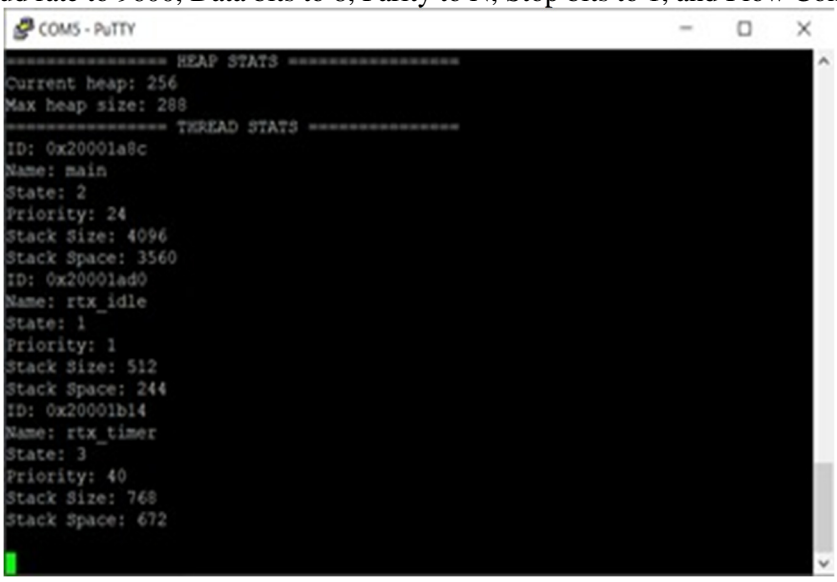


Figure 1: Monitoring debug port using Putty on Windows

14. Review the following web page for building thermistor
    https://learn.adafruit.com/thermistor/using-a-thermistor

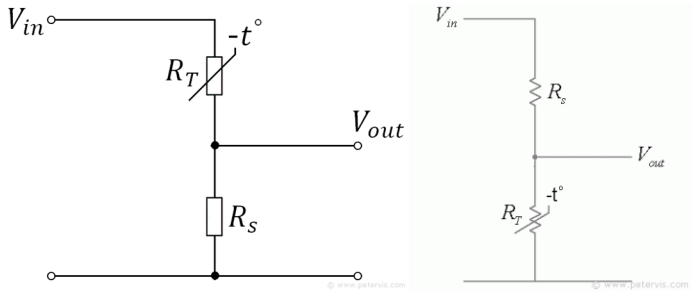


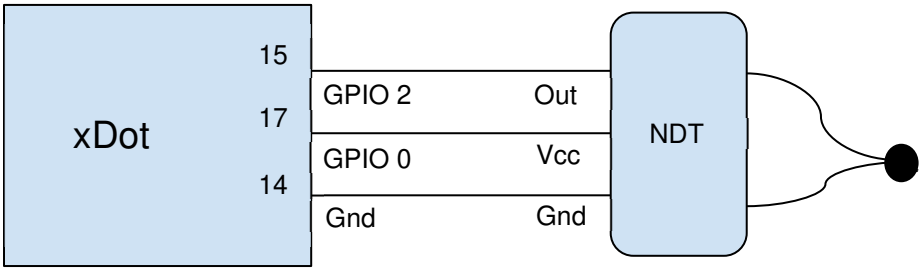
Figure 2: Examples of NTC thermistor circuits

You may purchase parts from Amazon (fast delivery) or other electronic component stores to build a thermistor for xDot. You should consider purchasing other components for your sensor capabilities for the step 12 of this lab as well as your final project.

Keep in mind that the upper voltage limit should be 3.3V for Vout.

15. Use the example code below to capture data from GPIO2(PB_0) ADC and GPIO 0 as source for the sensor. The parameter values should be adjusted based on your specific configuration for the thermistor. Refer to the following document for details of gpio or pin-diagrams of the xDot:
    https://www.multitech.net/developer/products/multiconnect-dot-series/multiconnect-xdot/

The connections are shown below:

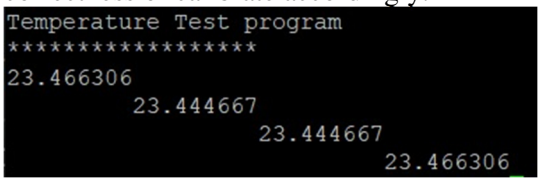



```
// Example Code
#include "mbed.h"
#define BETA 3975
AnalogIn sensor(PB_0);
DigitalOut vcc(GPIO0);
double Thermistor(int RawADC) {
  double Temp;
  Temp= (float) 10000.0 * ((65536.0 / RawADC) - 1.0);
  Temp = (1/((log(Temp/10000.0)/BETA) + (1.0/298.15)));
  Temp = Temp - 273.15;            // Convert Kelvin to Celcius
   //Temp = (Temp * 9.0)/ 5.0 + 32.0; // Convert C to F
   return Temp;
}

int main() {
    printf("\r\nTemperature Test program");
    printf("\r\n********\r\n");
    vcc = 1;
    unsigned int val;
    while (1) {
        val=sensor.read_u16();
        double temp =  Thermistor(val);
        printf("%f\n",temp);
        wait(5);
    }
}
```

The function Thermistor() converts the analog value to a readable temperature value scaled on the celsius scale using information from the NTC sensor documentation.
For additional information on mBed OS review the following link:
https://os.mbed.com/docs/mbed-os/v5.14/introduction/index.html

16. Compile the code and download it to the xDot.
    Open a serial connection to debug port with a baud of 115200 and check for the sensor value correctness or calibrate accordingly.



17. Dig in and experiment with one additional sensor of your choice, sampling rates, and GPIO capabilities. One easy and interest experiment might be to get a 3.5mm audio/mic connector with wires to feed in your phone audio output to xDot analog input and see if you can capture or detect sound.
    https://www.amazon.com/Qaoquda-Balanced-Terminal-Adapter-Connector/dp/B07RXS3ZYN