Jacob Bauer
10/9/2019

EnviroLogger:

An easy to use Data Logger with an easy to use GUI

Abstract

The goal for this project is to make a data logger that can transmit the data acquired by the sensors attached to a server. Then the data can be queried from the server and displayed on a console based GUI. The data logger is designed to be used for educational and environmental purposes.

Summary

This project has to parts: the hardware and the software. The software will be a desktop application meaning it has to be installed. It will have a GUI that will be in charge of displaying the data. It will also have a settings tab that will open a command line in order to connect to the data logging device to the host machine and configure other settings. The user will have the option to store the past data on the host machine or use a server. When first released the user will have to program the specifics using embedded C/C++. However, to make this easier I will make a library consisting of the files needed to drive the General Purpose Input/Output. The library will make it so the user only has to add a few lines to the main file in order to add devices and read the data. I will also include tutorials on git hub with videos to allow the user to program the device and recommended circuit techniques for connecting the data logger. The data logger will have 3 interfaces I2C(I squared C), ADC(Analog to digital conversion), SPI(Serial peripheral interface). Their are 6 analog to digital converters, and one SPI and I2C interface as shown in figure 3 below.

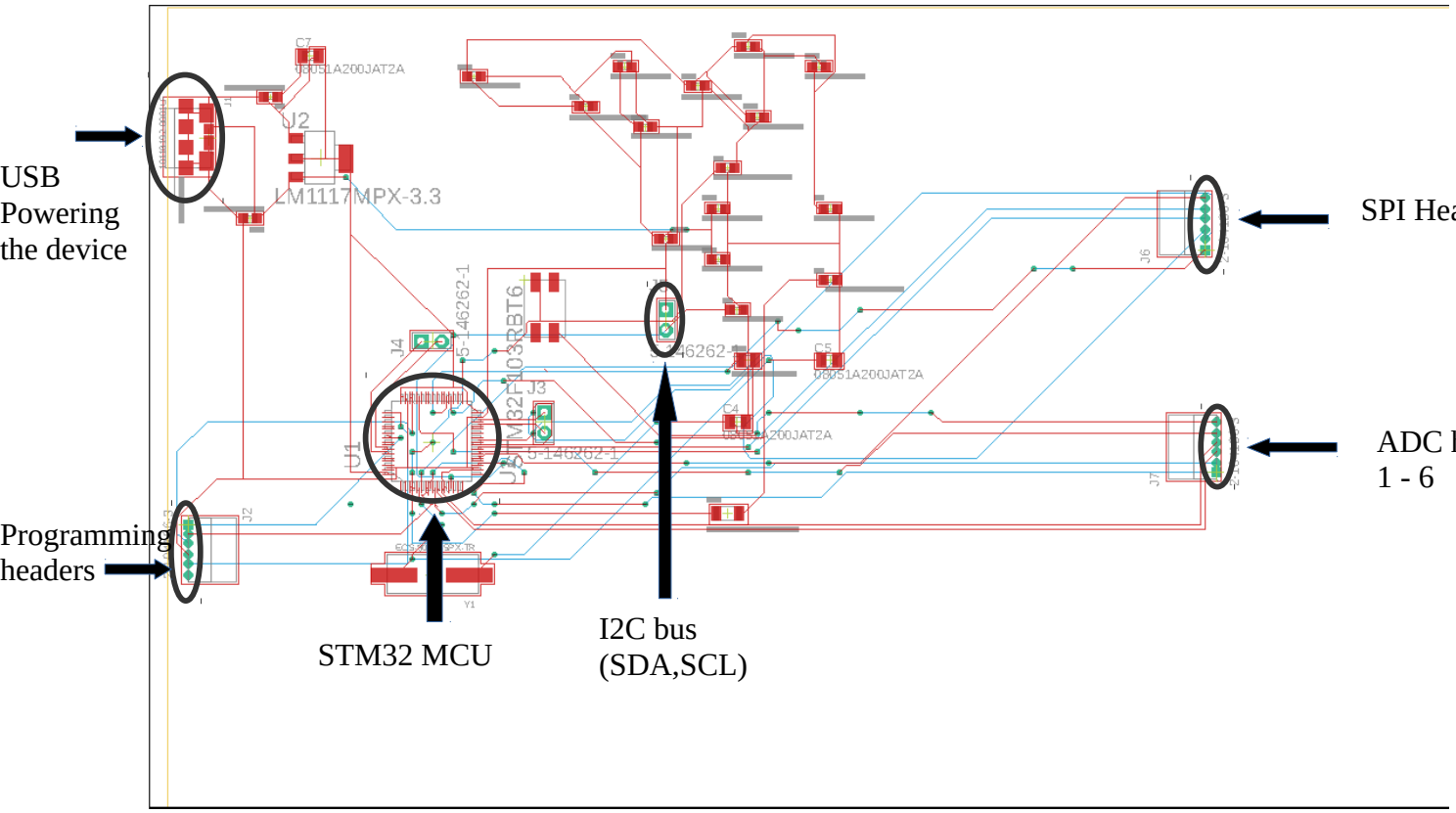

Figure 3: Data Logger Schematic done in Autodesk Eagle

The SPI will be able to support up to 3 devices, the ADC will be able to support 6 devices and the I2C buss can support practically an unlimited amount of devices.

Open and Resolved Technical Decisions

- Platform – STM32F103RB – I already have experience with this microccontroller. I considered using a higher level development board like Arduino or Raspberry PI. Those boards while great for enthusiasts and beginners tare not ideal for making a product because they are to expensive for an industrial grade product.
- Programming Languages – to keep it simple I will use C/C++ for all of the programming. The MCU will be programmed using the arm GCC/G++ compiler. The GUI will be programmed using GNU C++, and the SQL++ API for queering a database and QT API for GUI development.
- IDE – Using eclipse is probably the best IDE for this project because it has GNU MCU an easy plugin that supports Arm development. It is also the IDE I am most familiar with.
- Debugging – OpenOCD is an open source debugger that can be configured with eclipse to download and debug the software on the Microcontroller.
- Other software – EnviroLogger will also be developed with STM32CubeMX software. This software allows you to choose the output mode of each pin of the MCU and it generates the C code with the compiler the user supplies to it. This will be the base code I will expand on in-order to make it more specific toward my device. EnviroLogger will also be developed in auto desk eagle because it is free for students and I am familiar with the syntax.
- Unresolved decisions – I have some unresolved Technical decisions like what device I will use to transmit the data. I also do not know what device I will use to transmit the data however I am leaning toward using Data because it can be accessed by just about anywhere. Also I don’t know what server I am going to use but I am thinking amazon AWS.

Out of Scope Future Enhancements

- Casing – I would like to add 3d printed casing made out of PLA plastic to protect the PCB in harsh weather environments.
- Predict future data - I would like the GUI to have a mode where it can actually predict future data by using the past data and some form of machine learning algorithm
- Code generator – Since I already have the source files from CubeMX I would like to have the ability to add something to the GUI that could actually generate the code needed
- Custom Bootloader – I would like to create a bootloader that would allow the Data logger to be USB Bootable. This will greatly simplify the hardware design.
- RTOS – I would like to make my own real time operating system or use FreeRTOS. This can help by prioritizing tasks since all the interfaces don't collect data at the same speed. This will help get the most accurate data possible.

Personal Significance

I am interested in completing this problem for several reasons. The first is I care deeply about environmental issues and I am a big supporter for creating things that can help the environment or in this case environmental scientists. I believe that I should be making things to help society not harm it. Also I have always been interested in Embedded Systems. I like learning about the intersection

between hardware and software. However when I went online to try to learn about embedded systems there is a lack of content about things other than Arduino and Pi which have little to no real world applications in industry. So this device could also teach students how to use embedded C and Arm programming. In fact with STM32CubeMX they can reconfigure the MCU to act as a development board similar to Arduino.