Electrical and Computer Engineering Capstone

Version 1.1

6/10/2015

Erebus Labs

Open Sensor Platform

Submitted By:

Golriz Sedaghat

Steve Peirce

Colten Nye

Supervisor:

Dr. Lisa Zurk

Sponsors:

Dr. Mike Borowczak

Dr. Andrea Burrows

System Overview

openSense is a data logging device designed for K-12 students to collect data from their environment using either the custom sensor boards or those of their own design. The platform supports ten I2C and three analog sensors. The 32-bit ARM based microcontroller, STM32F205 by ST Microelectronics, communicates with the sensor boards, reads the collected data from them on user specified intervals, and logs the collected data to a micro-SD card. The open hardware design of the main board allows the more advanced user to design a custom version of the board, as well as upload their own custom binaries via the single wire debug header. The main processor device family has multiple additional abilities allowing for future expandability including a Digital Camera Interface, (DCMI), Ethernet, PWM motor control, on-board encryption and Random Number Generation (RNG).

Getting Started

Requirements

The minimum requirements for using this system are:

- Main board
- microSD card
- At least one sensor
- Jumper wires (if necessary)
- Power supply
- USB or Battery
- Micro-USB cable
- Computer with:
- USB port

- "openSense Configuration Manager" Chrome App installed
- FTDI serial-USB bridge drivers installed
- microSD card reader
- Spreadsheet application (optional)

Configure Sensors

Using the openSense configuration manager (referring to the GUI user manual, if necessary), describe the sensors that will be connected to the system.

Program the device

Once you have specified the sensors, prepare the device to receive the programming data by pressing the "Program" button (figure 1). Once the device is ready to receive programming data, use the User Interface to transmit the data (referring to the GUI user manual if necessary). If the operation was successful, you may close the GUI. Once the device is programmed, you may safely disconnect it from the host computer.

Collect Data

Now that the device is programmed it will begin logging the collected data to the SD card at the specified intervals, (figure 1). The device should always be powered on with the SD Card present. If it is not, simply insert it, reset the device and reprogram it. Ensure all sensors are connected to the appropriate headers as you specified in the GUI. Ensure that the power supply will last at least as long as the intended collection duration. When you are done collecting data, safely power down the device and remove the SD Card.

Retrieve and analyze data

You may view the collected data by inserting the SD card into a host computer and locating the file labeled "data.csv." This file is in a Comma Separated Value (CSV) format, which is easily viewable in spreadsheet applications and text editors at which point the data can be plotted, manipulated and analyzed at the user's discretion. Libreoffice, (Linux/Mac OSX) and ApacheOffice, (Windows) are both free software downloads and very powerful alternatives to the pricey MS Office suite, the extracted data was tested on both platforms without issue.

Hardware Description

Main board operates at 3.3V using either the USB cable or battery. Any input voltage in the range of 3.3 – 6V would be acceptable. To power on the board, simply shift the "Power Switch", (as labeled in figure 1), to the ON position (away from the status LED's, toward the power supply header pins).

If you are going to use an analog sensor (e.g. VOC sensor, photocell sensor, etc.), connect it to one of the headers labeled as "ADC port" in figure 1. The pin order of each ADC port from left to right is as following: 3.3V, Data and GND. In the case of using a digital I2C sensor (e.g. accelerometer, temperature sensor, etc.) connect the sensor to the "I2C bus" header with pin order of GND, 3.3V, SDA, SCL from top to bottom in figure 1.

Pins PA6, PA8, PA10, PC0, PB13 and PB14 of the microcontroller are connected to the GPIO header for the purpose of debugging/customization in future designs.

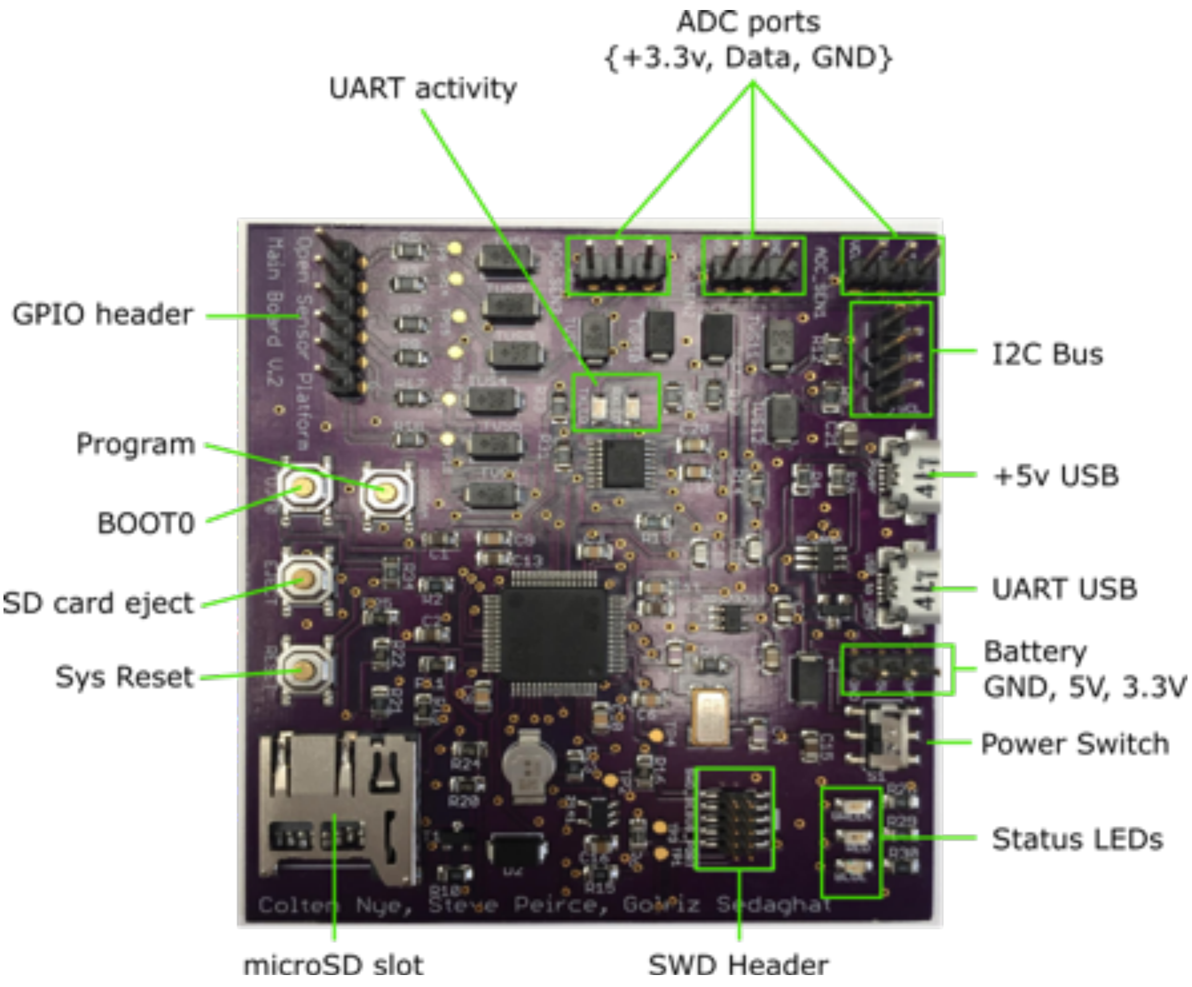


Figure 1 – Main board

Figure 2 indicates Volatile Organic Compound (VOC) sensor board which employs TGS2602 sensor model operating at 5V. The board can be powered up directly through the main board by connecting the 5V pin of the sensor to the middle pin of the battery header on the main board. The potentiometer on the VOC sensor board controls the applied voltage to the data pin which sends analog input to the board.

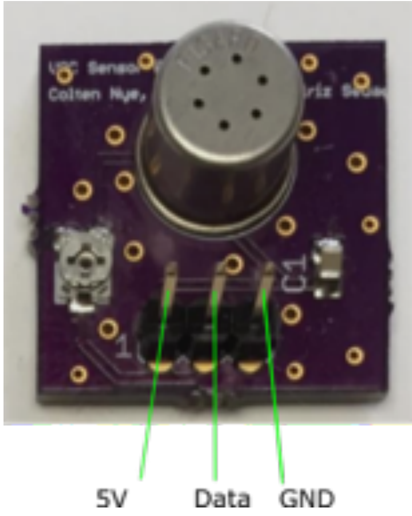
Figure 2 – VOC Sensor board

Figure 3 shows the accelerometer board which communicates over I2C with the main board. The employed accelerometer is ADXL345. The three-pole switch on the board provides two alternate I2C addresses. When the switch is in the ON position (as labeled in figure 3) the I2C address for the device is 0x1D; on the other hand, when the switch is in the OFF position (as labeled in figure 3) the alternate I2C address is 0x53. This would allow employing two sensors on the same bus.

Advanced:

Create sensors

Design the board



Create Sensor Description File (SDF)

A future design goal is to have a specific file format allowing for sensor abstractions to be described simply. These would be added to the GUI by placing them in a specific directory on the end-user's PC, allowing future sensors an easy way to be supported by the project without binary modification or re-flashing the firmware on the board itself.

Modify Firmware

The existing firmware was developed leveraging the Keil ARM-MDK toolchain. Natively supported by ARM International Holdings, the platform is robust and available for free download as an evaluation version. The primary limitation is a maximum 32KB binary size, (including code and data). The SD card system utilizes the on-board SDIO abilities of the ST Microelectronics chip used in the design.

An exported version of the Keil project is available on the project GitHub.

The FatFS filesystem was initially developed and licensed by Microsoft and remains highly prevalent throughout many of today's everyday devices including, (but not limited to) digital cameras, computers, USB sticks, etc. As this project is an open-source effort a specific variant was ported to our platform. This version was written by a very talented programmer by the named of Chan and is referred to by the community as "Chan FatFS,"it requires no licensing fee, (unlike Microsoft's version) and remains completely compatible. Though cryptic, the code is very powerful and robust once the underlying, platform-specific details are handled.

Modify Hardware

All hardware design files are available for download within the Design sub-directory of the project GitHub.