Final Project

T.P.S Report Pt. 4

COVER SHEET

Tad Miller
Danny Nsouli
Systems Programming

Implementation - Heartbeat Monitor (PART 4)

C Code

The C code handles the creation of the database and the new commands that need to be implemented. What we have done is integrated a SQLite database. This database takes in all sensor readings based on the time they were recorded. The sensor readings that are being stored in the database are those of the environment sensor and the heartbeat monitor. The schema for our database is as shown:

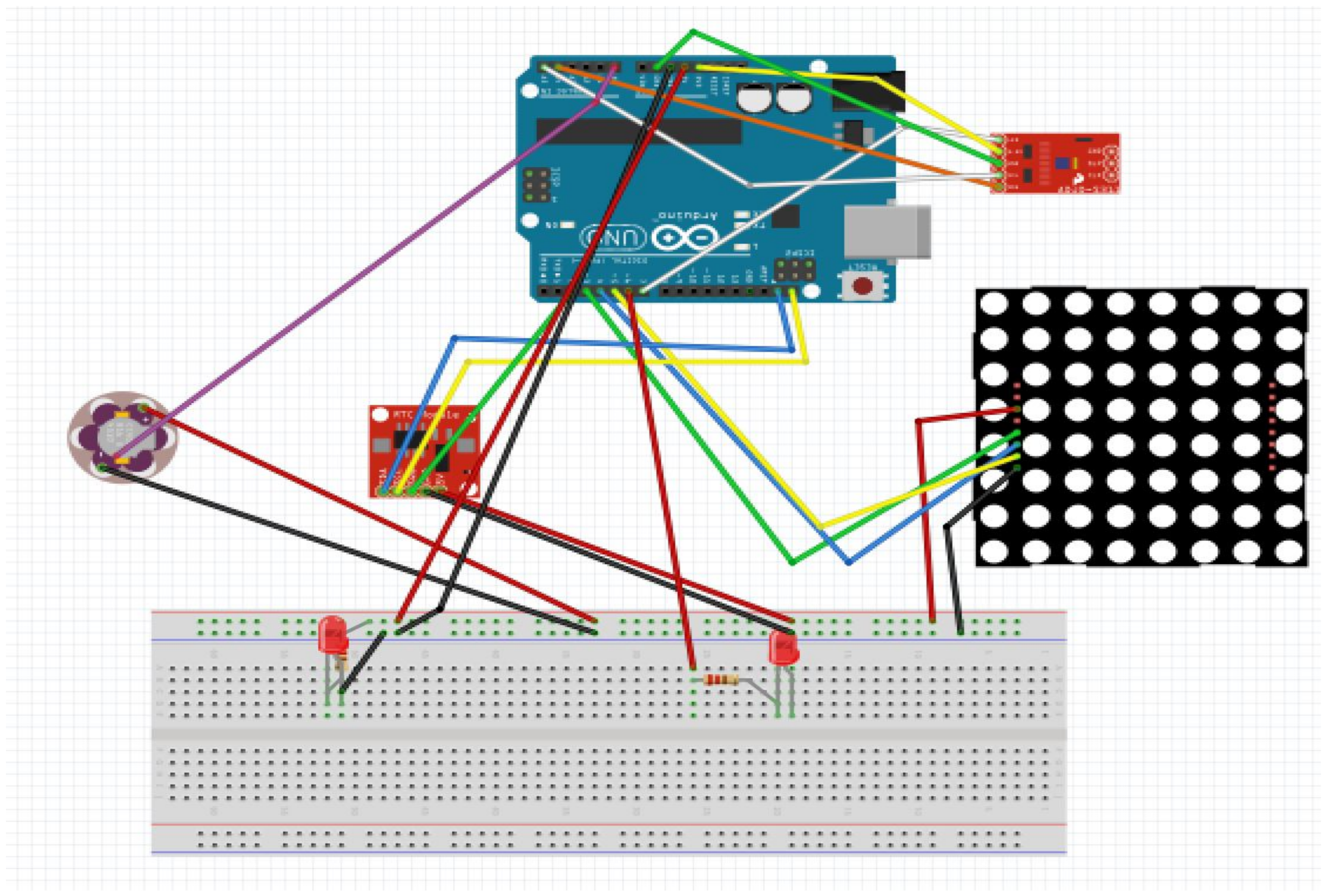
SensorData(<u>pk: integer</u>, time: integer, rate: int, env: text)

Pk is a random variable used as a primary key because some readings may be recorded at the same time or have the same env or rate values. Therefore, pk is just a value that increments in order to clearly distinguish our entries. The goal is try to figure out if there is relation between the two or not. Each is taken in as pairs in order to start a calculation for the linear regression. The command “regression X” will do this calculation for us with X being the time block requested to be analyzed. By default if there is no X specified the current time block will be used. As a result, the command will output the regression and the appropriate coefficient of determination. These pairs of variables are taken in accordance with the time they were read by the sensors. The command “date” takes in the value from the real time clock to use as output. Stat X works like regression X, however, it prints out the reading count, mean, median, mode, and standard deviation.

Arduino Code

The Arduino code is in charge of relaying data to the C file. This means that for the database the original heart rate and environmental sensor readings will be sent to the C file in order to insert into the database. The “date” command will be sent from the real time clock through the Arduino and finally into the C file. The other commands, “stat” and “regression” are only done on the C side and have no use of the Arduino other than using its sensor data.

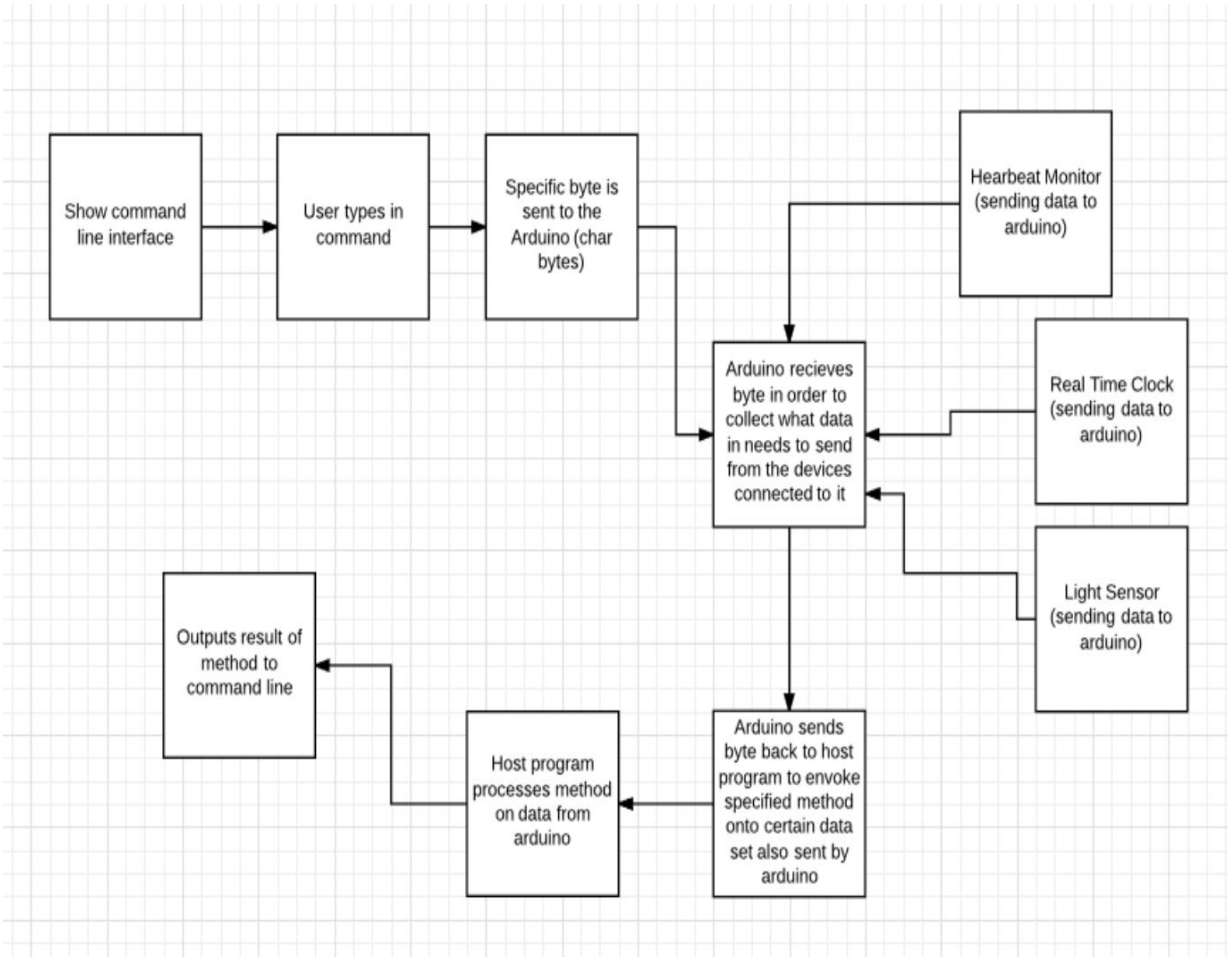
Final Fritzing Diagram



More Computation Explanations

Outlier detection is a component of our standard deviation method within the “stat X” command. This is because standard deviation, by definition, determines the extent of deviation for an entire set of data. Therefore, the result of taking the standard deviation of the data at that specified time block will allow us to consider the appearance of outliers in the data. To handle the time synchronization between the Arduino and host program, we have a method in our shell called arduino_clock_sync in ardlib.c. This method sends bytes to the Arduino in order to receive the correct time from the real time clock. When we send bytes, however, we must add 32 in order to get plausible character values due to the specifications of the ASCII table. In the ASCII table, values lower than 32 are not characters, which cannot work for this method. To build sensor reading pairs, we are grouping up heart rate and environmental sensor data taken at the same time into the database. Even if there is an issue with two tuples having the same time, we have a placeholder primary key that increments in order to distinguish each tuple according to the number of insertions of data that have occurred already.

State Machine Diagram


Show command line interface
User types in command
Specific byte is sent to the Arduino (char bytes)
Hearbeat Monitor (sending data to arduino)
Arduino recieves byte in order to collect what data in needs to send from the devices connected to it
Real Time Clock (sending data to arduino)
Light Sensor (sending data to arduino)
Outputs result of method to command line
Host program processes method on data from arduino
Arduino sends byte back to host program to envoke specified method onto certain data set also sent by arduino

Protocol

For the project we developed a transmission protocol to be used between the C program and the Arduino code. This allows us to send and receive the heartbeat data, the timestamp, and the environmental sensor data from the Arduino to our computer. Our protocol is structured in a way that bytes represent each the messages being sent to the Arduino from the host program. Once the byte gets to the arduino, depending on what the byte value is, the arduino will send the required data to the host program to complete the method for the command entered in the command line to go through (output a result).