<u>CS 3410 Final Project Part 2 Write Up</u> Timothy Lytkine, Lian Showl, Simon Patel

The real time clock is configured using the I2C communication protocol . The time initially set using the function setDS2321time. The parameters of the function are obtained upon compile time from the __DATE__ and __TIME__ predefined macros in C. After this, the time is read from the real time clock using the readDS3231time function. The inputs to the function: second, minute, hour, dayOfWeek, month and year are all populated with values from the real time clock which starts counting from the time it is set to upon compile time. The time is read every time every time a command is sent from the C code (using the send_cmd function) to the Arduino and then the time is sent back to the C code in order to determine the index of the array that stores the histogram. The messages that are sent from the Arduino which include BPM, hour, minute and second are sent using Serial.write() byte by byte. There is a main loop that takes user input within the C code that allows the corresponding commands to be sent. Based upon the command requested, the values are then read into the buffer and then each byte stored in a corresponding index is set equal to a variable which is cast to a char or an int as necessary. ASCII conversion must be performed since the values sent are char's. There are 96 possible time intervals in the array and 5 possible BPM values. Hence, there are 480 possible values being stored. The hour value that is sent from the Arduino is multiplied by 4 to determine the firstIndex. If the minute value is between 0 and 15 then the first index is equal to the hour value multiplied by 4. If the minute value is between 15 and 30 then one is added to this value, if the minute value is between 30 and 45 then two is added to this value and if the minute value is between 45 and 60 then three is added to this value. There are 24 hours in a day and 4 time intervals for each hour, 24 * 4 = 96 which is why the first index can range between 0 and 96. For the second index, the BPM value sent from the Arduino that is read from the heart rate sensor is used. If the BPM is between 0 to 40 then the second index is 0, if it is between 41 and 81 then the second index is 1, if it is between 81 and 120 then the second index is 2, if it is between 121 and 160 then the second index is 3 and if it is above 160 then the second index is 4. This comprises 5 possible values for the second index: 0, 1, 2, 3 and 4. These 480 possible values are mmaped by the mapWrite, mapSync and mapRead functions. mapWrite writes the 480 values into memory by using a loop with the 2D array that stores the histogram and mapSync writes these values into the filepath. mapRead is the function that is called initially to get previous histogram values from memory. There is a function that prints the histogram on demand with the ranges of possible values based upon the time interval indicated in the file path that the histogram is stored. Outlier readings are being detected when the array that stores the histogram has its values incremented. If the current BPM which is received from the Arduino is between 0 and 40, a messages is flashed on the screen of the Arduino that says "Warning: BPM is low." In addition, if the current BPM is greater than 160, a message is flashed on the screen of the Arduino that says "Warning: BPM is high." These warnings are flashed on the LCD by the C code sending a low or high command based upon the BPM values which is then sent to the Arduino to flash the corresponding warning.