California State Polytechnic University Pomona

Department of Electrical and Computer Engineering

ECE 3301

Section 02

Introduction to Microcontrollers

Final Term Project

Cardiac Monitoring System/ Heartbeat Monitor

Prepared by:

Gerard Aingello Cruz, Damian Flores-Menjivar, Jackie Martinez, and Chanel Williams

Presented to:

Professor Tamer Omar

11 May 2023

ABSTRACT

This paper presents the design and implementation of a heart pulse monitoring system using the heart pulse sensor with Arduino. The system is designed to measure the heart rate of an individual in real-time and display the readings and waveform on an LCD screen. The experimental methods used involved setting up the system, testing its functionality, and doing multiple trials to collect data from an individual. The results obtained showed that the system was able to accurately measure the heart rate of participants, and the data collected was reliable. The challenges faced during the experiment, such as the lack of soldering experience and difficulty in obtaining accurate heart rate measurements, were discussed, and solutions were proposed. Overall, the heart pulse monitoring system designed in this paper provides a reliable and cost-effective solution for monitoring heart rate in real time.

Table of Contents

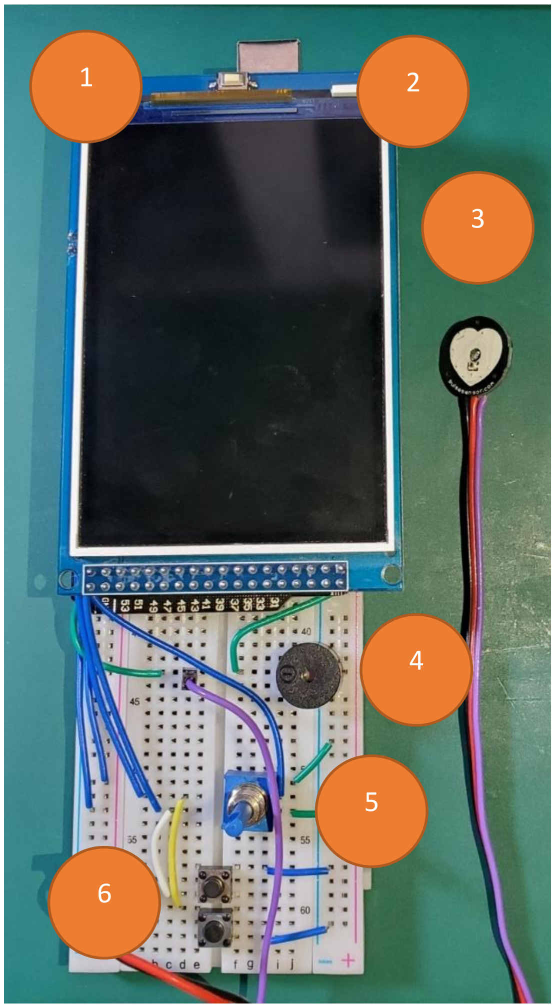
Title Page…………………………………………………………………………….1
Abstract………………………………………………………………………………2
Table of Contents…………………………………………………………………….3
Introduction……………………………………………………………………..….4
Experimental Methodology…………………………………………………………6
Experimental Results………………………………………………………………..7-12
Summary of Challenges and Resolutions……………………………………..………12-13
Conclusions…………………………………………………………………….....14
References………………………………………………………………………..15

INTRODUCTION

Students showcase understanding of the course through applying a microcontroller with a real-life application. With that, the team both designed and implemented a heart pulse monitoring system using an Arduino microcontroller, LCD screen and a heart pulse sensor. Our project aims to demonstrate the potential of Arduino technology in developing practical medical devices that can improve patient care and support medical professionals' work. The cardiac monitoring system will be able to track and record the heartbeat rate over time. The system will be able to count the number of heart beats per minute, once the user places their finger on the heartbeat pulse sensor.

HARDWARE DESCRIPTION, DEMONSTRATION, SCHEMATIC, and PSEUDOCODE

Demonstration Video Link



1. **Microcontroller Development Board:** Arduino Mega 2560, utilizing the Microchip ATmega2560 with 54 digital pins, 16 analog inputs, 4 UART serial ports, PWM, USB, and a 16MHz crystal oscillator.

2. **LCD Display:** 480x320 36-Pin LCD display utilizing the ILI9486 display controller, communicating with the Arduino using a 16-bit parallel interface bus. This also has a SD card that is used to store results.
3. **Heart Pulse Sensor:** an optical heart-rate sensor utilizing a light photo sensor, an LED, and an op-amp. The LED shines light on the skin, the light photo sensor senses how much light has been reflected, and then the op-amp amplifies the signal. The sensor can be put on finger or earlobe to sense heart rate.
4. **Buzzer:** The buzzer is connected to a digital pin on the Arduino set to PWM output to buzz at 820Hz whenever a heart beat is detected.
5. **Potentiometer:** Potentiometer output is connected to an analog input on the Arduino to control the UI and text size on the display, may the user require a larger or smaller text size.
6. **Push Buttons:** The first button is used to change the mode of the device. The second button, in normal operation, is used to change the orientation of the screen, in recording and results mode, it is used to initiate the 30 second recording that will be saved on the SD card.

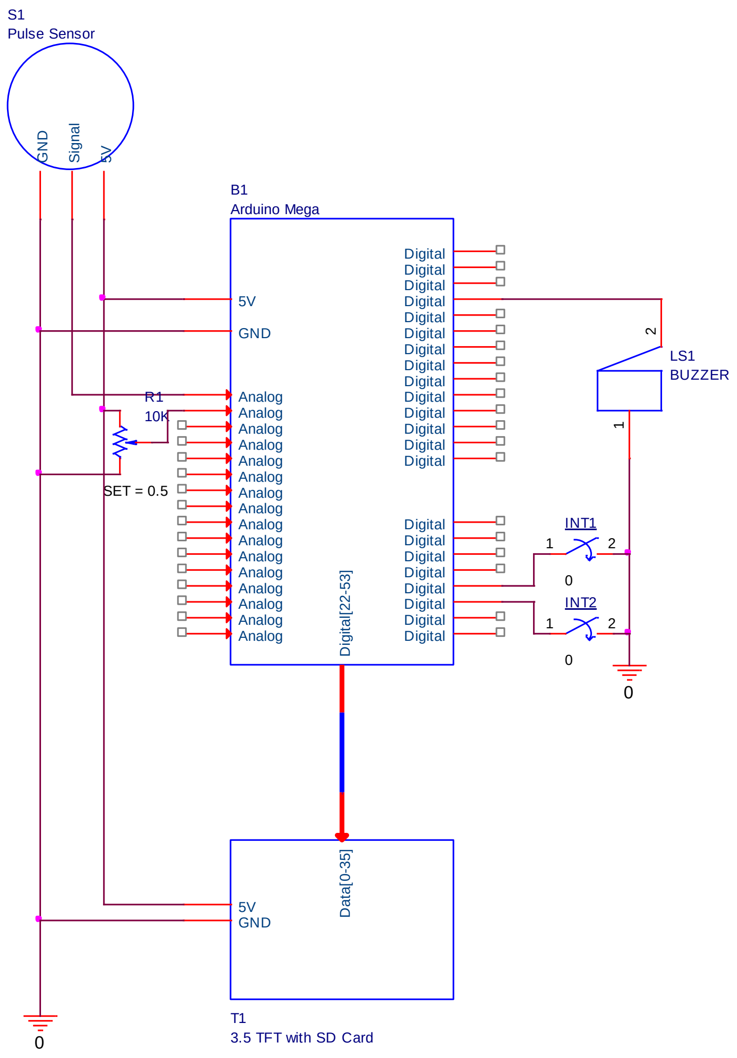


Figure 1- Cardiac Monitoring System Schematic

PSEUDOCODE

```
Setup
Include libraries for pulse sensor, LCD display, heart rate sensor, SD card
Read previous readings from SD card into array, set mode to main mode
Setup screen, sensor, ADC, interrupts
Loop
Check mode
If main
    Get sample from heart rate sensor, and draw on LCD, moving cursor to left, and
    clear lines ahead of cursor
    Check for peak of heartbeat, if found, sound buzzer, and update BPM, and update
    color if necessary
    Check ADC of potentiometer, and change text size if value has changed
If alternate mode
    Display previous results
    If record mode is set, record and display BPM and heart rate signal on LCD, like
    in main mode
    After 30 seconds, show final results, wait for two seconds, write last result to SD
    card and discard the oldest result, and go back to previous results screen
Interrupt 1
    Toggle mode if not taking measurements
Interrupt 2
    Change rotation of screen if in main mode or in recording mode
    If displaying previous results, reset 30s timer and switch to recording mode
```

EXPERIMENTAL SETUP

For the first test, the reference BPM test, a reference heartbeat waveform was created and is meant to simulate the output of the heart rate sensor at a specified, fixed BPM. The waveform is fed through a summing amplifier in order to add a DC offset to the waveform signal, then is passed through another amplifier in order to amplify the signal to have a peak of around 5V to pass the threshold set in the code. The waveform is then inputted to the same port as the heartrate sensor starting at 5 seconds after the system has been reset. This 5 second delay is to simulate the delay that occurs when the system is on, but has not been used for a few moments. Otherwise, when the signal is already being inputted to the system before the system is on, the system accurately measures the heart rate at startup.

The test is ran as follows: Reset the Arduino by pressing the reset button, either on the display, or on the board itself. Wait 5 seconds, then start the input reference waveform, and record the BPM each time a beat is detected (a beep on the buzzer) for 25 beats. The data from this test is given in the next section. (Green signifies the value and at what beat where the recorded BPM has settled)

EXPERIMENTAL RESULTS

	Reference Waveform BPM								
Beat #	40	45	50	55	60	65	70	75	80
1	32	35	37	40	44	48	52	55	59
2	32	35	37	41	46	49	55	57	61
3	33	36	38	42	49	50	58	60	65
4	34	37	39	43	51	51	61	65	69
5	34	38	41	45	53	53	63	67	71
6	35	39	42	46	55	55	65	69	73
7	36	40	43	48	57	56	67	71	76
8	36	41	45	49	59	58	69	74	79
9	37	42	46	51	61	60	71	76	81
10	38	43	48	52	63	62	74	79	84
11	39	44	49	54	66	64	75	82	88
12	40	45	50	54	65	65	72	76	88
13	40	45	50	55	61	65	71	75	84
14	40	45	50	55	60	65	70	75	80
15	40	45	50	55	60	65	70	75	80
16	40	45	50	55	60	65	70	75	80
17	40	45	50	55	60	65	70	75	80
18	40	45	50	55	60	65	70	75	80
19	40	45	50	55	60	65	70	75	80
20	40	45	50	55	60	65	70	75	80
21	40	45	50	55	60	65	70	75	80
22	40	45	50	55	60	65	70	75	80
23	40	45	50	55	60	65	70	75	80
24	40	45	50	55	60	65	70	75	80
25	40	45	50	55	60	65	70	75	80

Beat #	85	90	95	100	110	120	130	140	150
1	63	68	71	75	85	91	160	150	160
2	65	70	73	77	87	94	166	144	154
3	68	74	77	79	91	96	163	143	153
4	73	79	82	81	98	99	160	142	157

5	76	81	85	83	100	99	151	147	163
6	78	84	88	86	103	102	148	152	161
7	81	86	90	88	106	105	145	151	160
8	84	89	94	91	109	108	142	150	159
9	86	92	97	94	112	111	140	149	158
10	90	95	101	97	116	115	137	148	164
11	93	99	104	100	120	118	135	153	164
12	93	99	101	100	121	119	129	147	158
13	91	95	96	100	117	119	128	142	152
14	86	91	96	100	111	119	132	142	153
15	86	91	96	100	112	121	132	142	153
16	86	91	96	100	112	122	132	142	153
17	86	91	96	100	112	122	132	142	153
18	86	91	96	100	112	122	132	142	153
19	86	91	96	100	112	122	132	142	153
20	86	90	96	100	112	122	132	142	153
21	86	90	96	100	112	122	132	142	153
22	86	91	96	100	112	122	132	142	153
23	85	90	96	100	111	122	132	142	153
24	85	91	96	100	111	122	132	142	153
25	85	91	96	100	111	122	132	142	153

Table 1 – Measured BPM vs Number of Beats

The data is thus graphed below in Figure 2, where the results signify that it takes at most 15 beats before the measured beat per minute stabilizes.

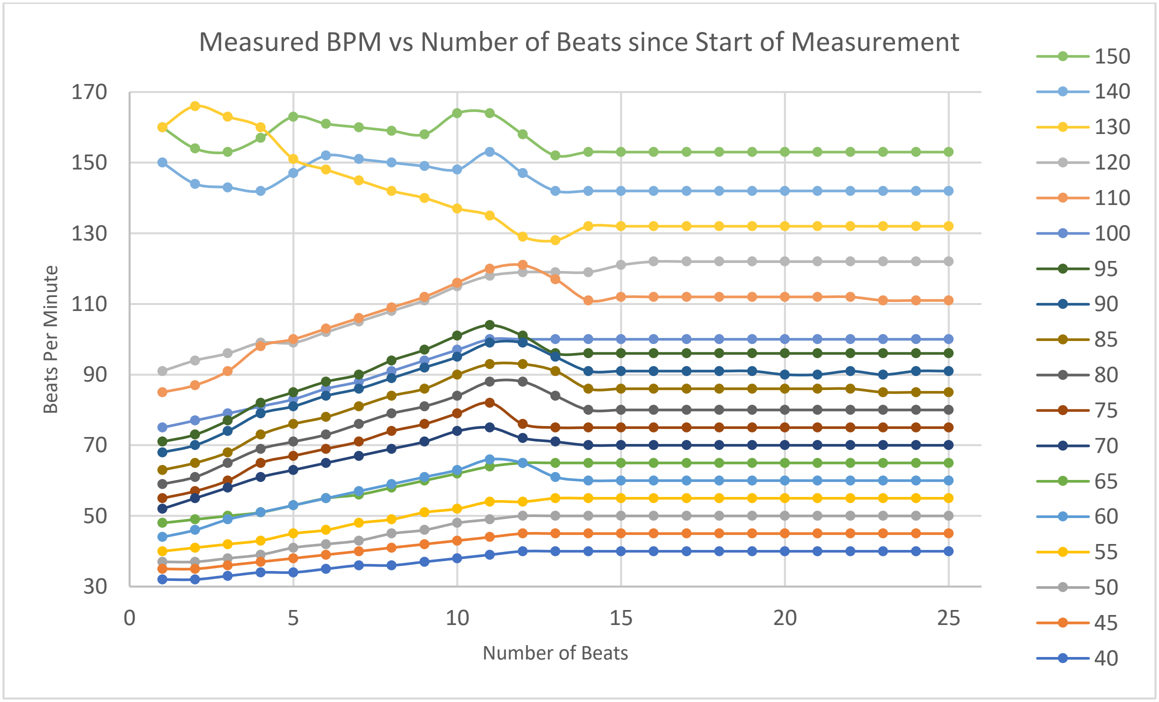


Figure 2- Measured BPM vs Number of Beats since Start of Measurement

In Figure 3, the results display the percentage error of the measured beats per minute ranging from 0 to 25 beats after starting measurements. Since it takes about 15 beats to obtain a stabilized heartbeat measure, the percentage error decreases after.

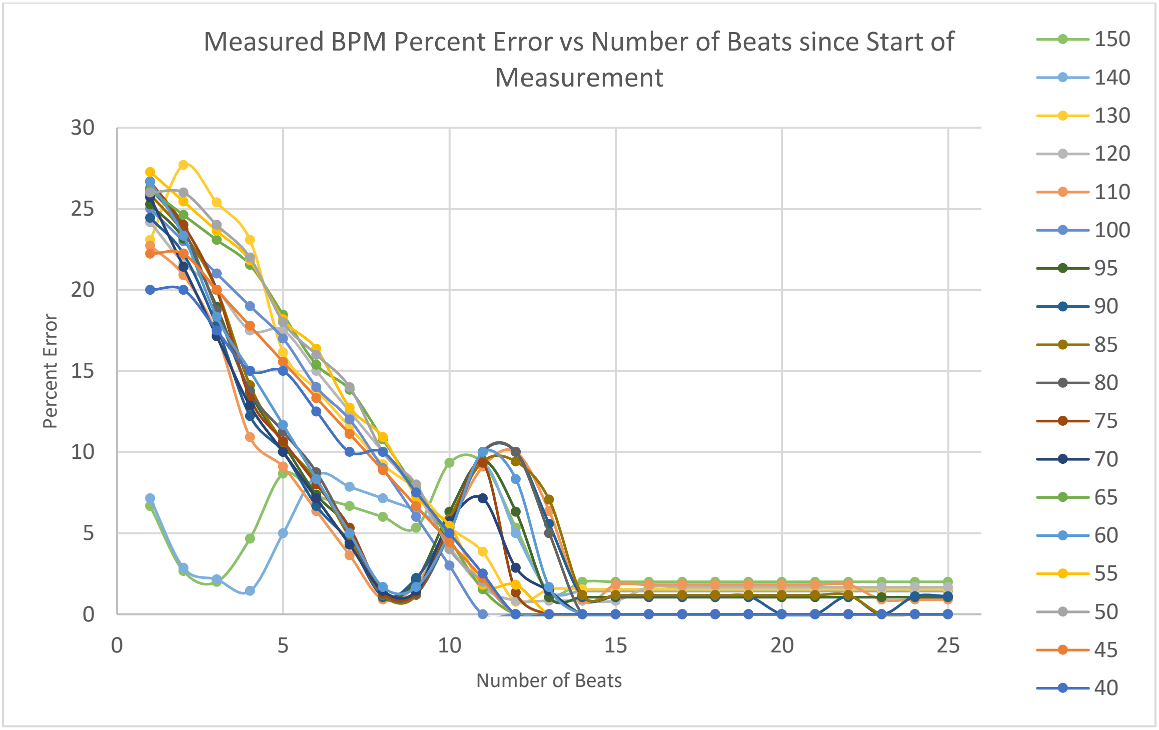


Figure 3- Measured BPM % Error vs Number of Beats since Start of Measurement

In Figure 4, the results display the percentage error from 40 BPM to 55 BPM of the measured beats per minute ranging from 0 beats to 25 beats after starting measurements. As stated earlier, percent error decreases after at most 15 beats.

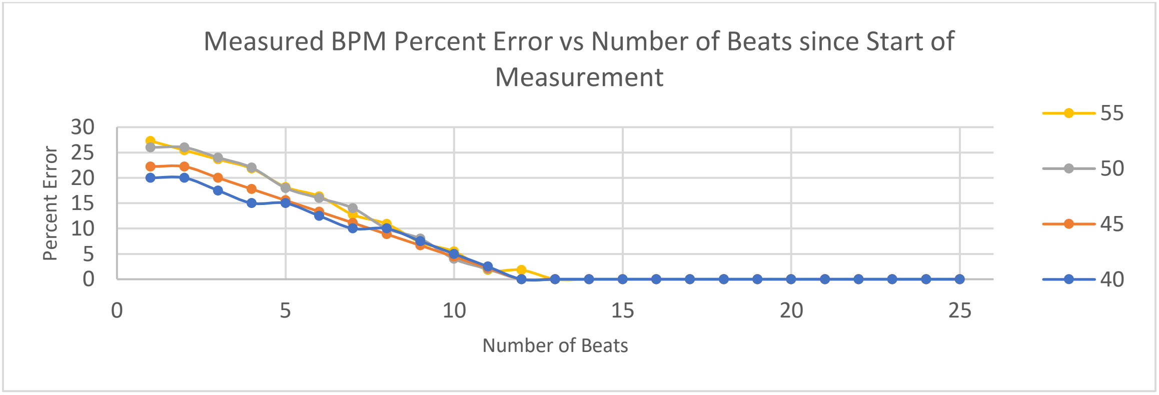


Figure 4- Measured BPM % Error vs Number of Beats since Start of Measurement

In Figure 5, the results highlight the percentage error from 60 BPM to 75 BPM of the measured beats per minute ranging from 0 beats to 25 beats after starting measurements. The percent difference decreases after at most 15 beats have occurred.

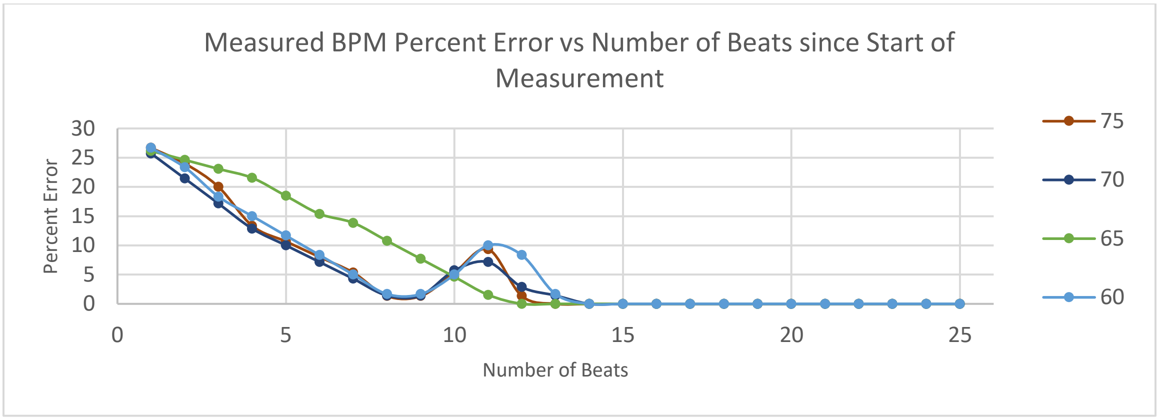


Figure 5- Measured BPM % Error vs Number of Beats since Start of Measurement

In Figure 6, the graph highlights the percentage error from 80 BPM to 95 BPM of the measured beats per minute ranging from 0 beats to 25 beats after starting measurements. The percent error decreases after at 15 beats at most have occurred.

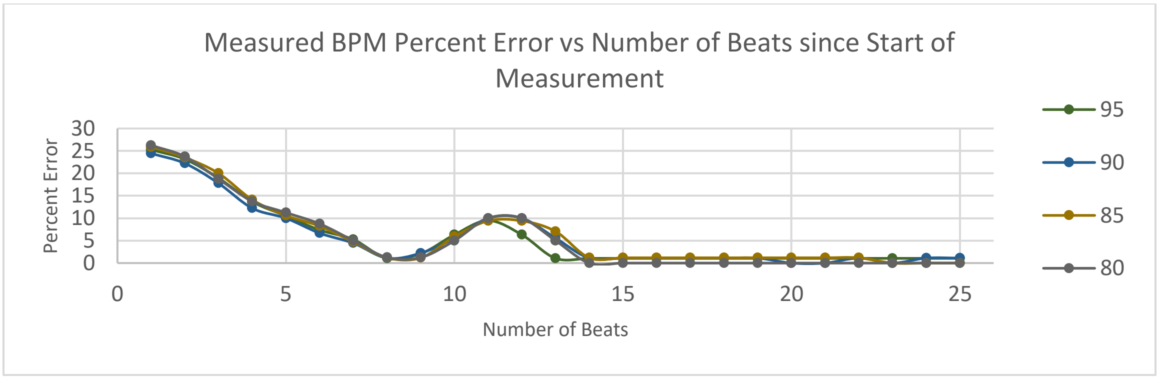


Figure 6- Measured BPM % Error vs Number of Beats since Start of Measurement

In Figure 7, the graph display the percentage error from 100 BPM to 150 BPM of the measured beats per minute ranging from 0 beats to 25 beats after starting measurements. It takes at most 15 beats for the measurement to stabilize.

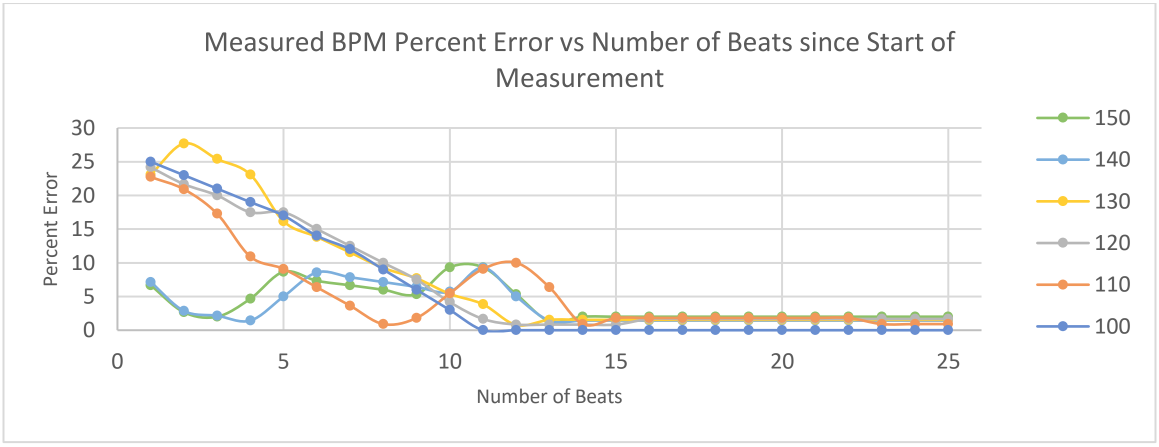


Figure 7- Measured BPM % Error vs Number of Beats since Start of Measurement

Figure 8 represents the percent error of the final settled BPM measurement vs the reference BPM waveform, i.e., the error between the measured BPM vs the actual BPM. At BPMs 40 to 85, there is no error at all, but as the BPM gets faster, the error increases, but only to a maximum of 2%.

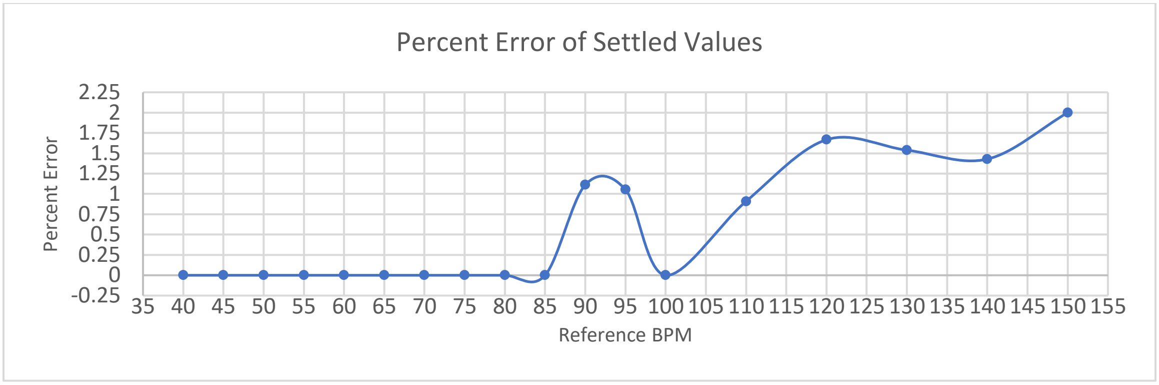


Figure 8- Percent Error of Settled Values

Figure 9 displays two measuring devices, the Arduino Mega project versus a heart watch monitor. Both devices measure the user's heart pulse for about one hundred seconds at the same time while the user is resting. Aside from some large outliers, both devices measure the beats per minute relatively close to one another.

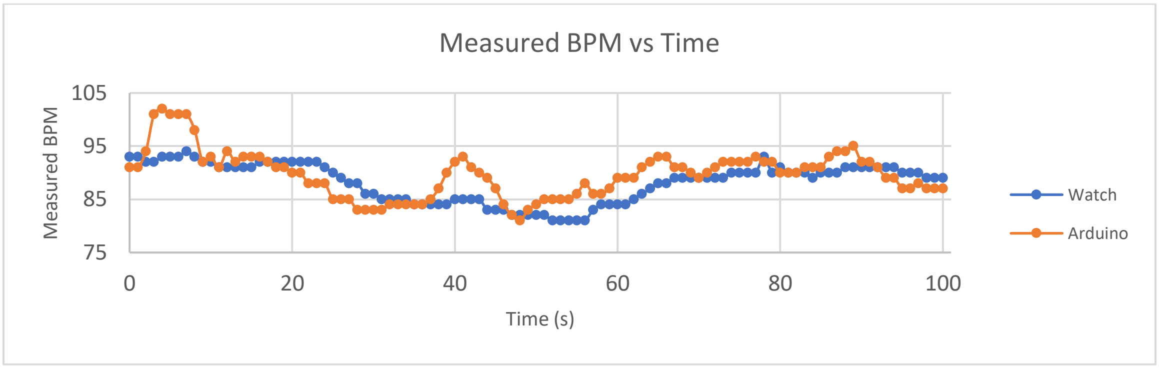


Figure 9- Measured BPM vs Time

Figure 10 represents the percentage error between the differences between beats per minute in the Arduino data and watch data shown in Figure 10.

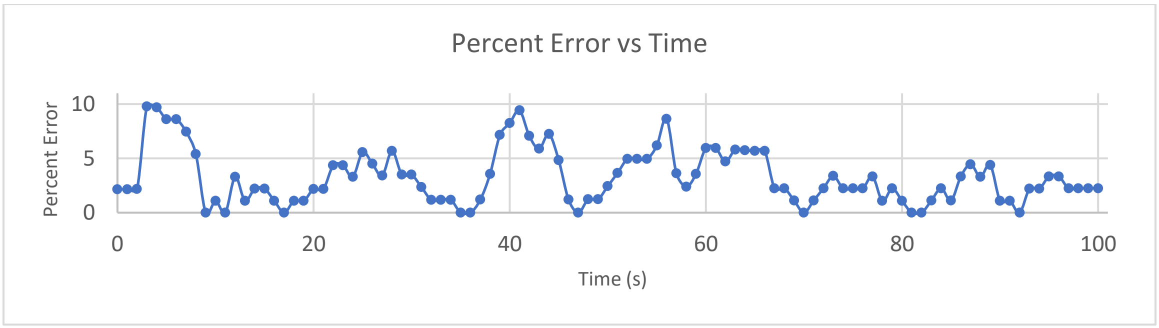


Figure 9- Percent Error vs Time

Figure 11 displays two measuring devices, the Arduino mega project versus a heart watch monitor. This data is the moving average (10 values) of the data in Figure 10 in order to smooth the outliers. From this data, the team notice that the Arduino sensor records the same values as the watch, but before the watch records them; this may be due to the watch already taking a moving average before displaying to the user.

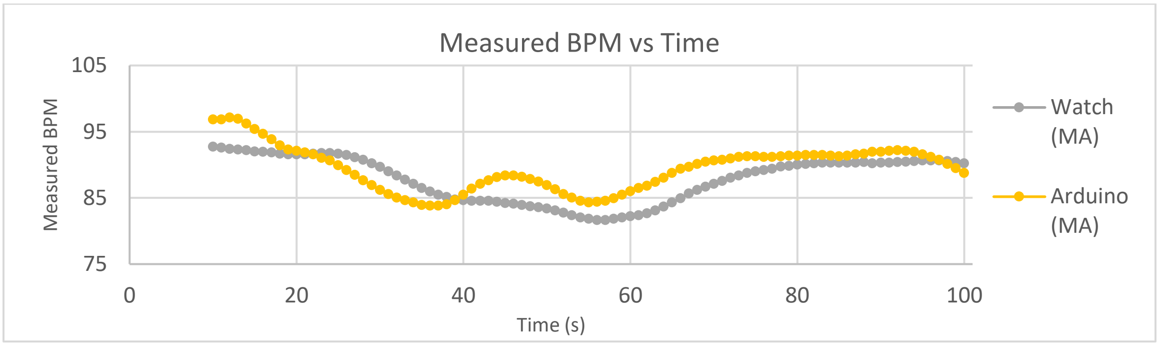


Figure 10- Measured BPM vs Time

Figure 12 represents the percentage error between the differences between beats per minute in the Arduino data and watch data shown in Figure 12.

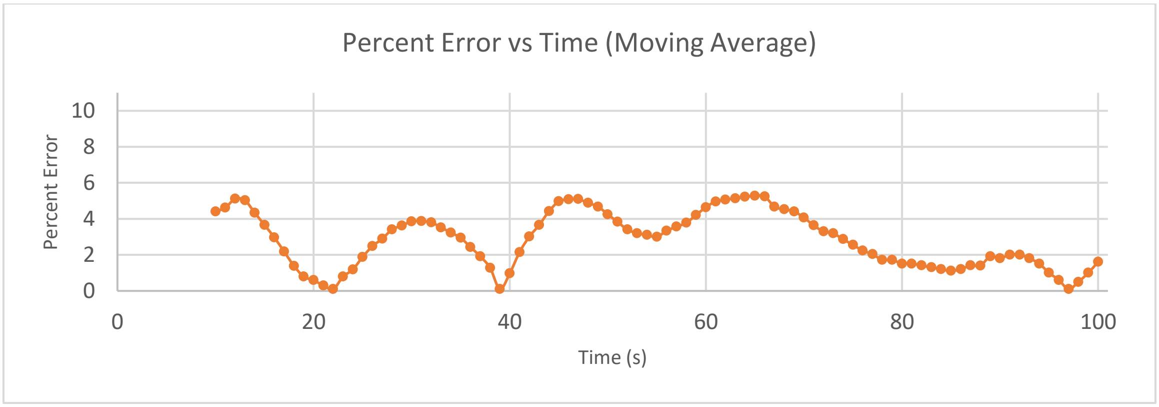


Figure 11- Percent Error vs Time

SUMMARY OF CHALLENGES AND RESOLUTIONS

Challenge #1: LCD Display

The first challenge arose during the hardware portion of the heartbeat system. At first, the decision to use a 16x2 dot matrix display fell short without soldered headers and the lack of soldering experience. To find a solution, implementing a different LCD screen brought up another dilemma where it was not updating fast enough. In an attempt to solve this issue, the team tried using different types of displays, starting with a serial display using SPI. However, the team initially misconfigured the pins resulting in the usage of software SPI. After some reconfiguration, the team was able to use hardware SPI. Despite these efforts, the display was still updating too slowly. The team then experimented with different calculations to improve the update rate.

<u>Resolution #1: LCD Display</u>

After encountering several issues with the LCD display, the team came up with several resolutions. One solution was to switch to using a parallel interface display, which allowed for a much faster refresh rate. This allowed for less time sending signals to refresh the display and more time to do other essential tasks, such as ADC conversion and calculating the BPM. The faster refresh rate of the new display allowed the team to create a more efficient and accurate heartbeat monitoring system.

<u>Challenge #2: Pulse Rate Calculation</u>

Intertwined with challenge one, errors arose when dealing with pulse rate calculations. Despite successfully getting the hardware SPI to work, the team encountered another issue with incorrect BPM readings. The error proved to be difficult to correct as it was non-linear.

<u>Resolution #2: Pulse Rate Calculation</u>

As we fixed challenge #1, we realized that challenge 1 fell into challenge 2. As we continued the project, we realized that challenge 2 was because, as mentioned before, that we were using a serial display using SPI, so when we switched to a parallel interface display, it fixed our problem, and we were starting to get more accurate BPM readings.

<u>Challenge #3: Waveform Clipping</u>

During the development of the heartbeat monitoring system, the team encountered a challenge with waveform clipping. Despite not appearing on the oscilloscope, the waveform from the sensor was consistently clipping. This issue was thought to be related to the board using supply power of USB as reference, which was measured to be slightly less than 5V. However, the measured waveform on the oscilloscope had peaks up to 5V. While this phenomenon did not appear to affect accuracy, it was still a concern for the team during development.

<u>Resolution #3: Waveform Clipping</u>

As mentioned above, this waveform clipping had no effect on accuracy, which served as a minor concern during the trial and error of the designing of the project. The result of the entirety of the project had no effect on measuring heart beats per minute. Our concern revolved around the project's ability to read and write measurements when a user wishes to obtain their heart BPM.

CONCLUSION

This paper centered on replicating a medical field heart pulse monitoring system, with the aid of an Arduino. The LCD showcased real-time heat beat measurements. When testing the functionality of the system, challenges arose. Some challenges discussed included lack of soldering experience, utilizing correct heart rate formulas, alongside others. Overall, this paper and project aimed to mimic a real-time heart pulse monitoring system the proves reliable and cost-effective. Measuring heartbeat activity will aid both in the medical field and educational field, where this project serves many purposes. Some purposes include heightening students understanding about the cardiovascular system, heart rate monitoring, and principles of electronics. This device is more commonly seen and used in the medical field to detect heartbeat imbalances.

REFERENCES

[1] Z. Vlad, “Cardiac monitoring system - EKG,” Hackster.io, https://www.hackster.io/user8702584/cardiac-monitoring-system-ekg-45f156 (accessed Mar. 1, 2023).

[2] Admin, “Pulse rate (BPM) monitor using Arduino &amp; Pulse Sensor,” How To Electronics, https://how2electronics.com/pulse-rate-bpm-monitor-arduino-pulse-sensor/ (accessed Mar. 8, 2023).

[3] Bodmer and per1234, “Bodmer/TFT_HX8357: Arduino Library for HX8357 TFT display,” GitHub, https://github.com/Bodmer/TFT_HX8357 (accessed Apr. 12, 2023).

[4] WorldFamousElectronics, “WorldFamousElectronics/PulseSensorPlayground: A pulsesensor library (for Arduino) that collects our most popular projects in one place.,” GitHub, https://github.com/WorldFamousElectronics/PulseSensorPlayground (accessed Mar. 15, 2023).

[5] Adafruit, “Adafruit/ADAFRUIT_ILI9341: Library for Adafruit ILI9341 displays,” GitHub, https://github.com/adafruit/Adafruit_ILI9341 (accessed Mar. 25, 2023).

[6] Arduino-Libraries, “Arduino-libraries/SD: SD Library for Arduino,” GitHub, https://github.com/arduino-libraries/SD (accessed Apr. 20, 2023).

[7] Last Minute Engineers, “In-depth: Detect, Measure &amp; Plot Heart Rate Using Pulse Sensor &amp; Arduino,” Last Minute Engineers, https://lastminuteengineers.com/pulse-sensor-arduino-tutorial/ (accessed Mar. 20, 2023).

[8] T. A. Team, “Mega 2560 REV3,” Arduino Documentation, https://docs.arduino.cc/hardware/mega-2560 (accessed Apr. 26, 2023).