Department of Electronics and Telecommunication Engineering

University of Moratuwa

Muscle Cramp Detection System

Ananthakumar.T 220029T

Prathishanth.A 220480P

Dayananthan.T 220096T

Contents

Chapter 1

Introduction

Muscle cramps are sudden, involuntary contractions or spasms in one or more muscles. They can cause significant discomfort and can last from a few seconds to several minutes. Muscle cramps are a common issue, particularly during physical activity or at night. While they often resolve on their own, recurrent or severe cramps may indicate underlying health issues or deficiencies, such as dehydration, electrolyte imbalances, or nerve dysfunction.

The significance of addressing muscle cramps lies in their impact on quality of life and performance, especially for athletes or individuals engaged in physical labor. Effective solutions can prevent discomfort, improve productivity, and promote better health outcomes.

This project aims to analyze the causes and triggers of muscle cramps and propose strategies or tools to predict, mitigate, or prevent their occurrence. By understanding the mechanisms behind these involuntary actions, the project seeks to provide insights that can lead to practical applications in sports science, medicine, and ergonomics. Ultimately, the project strives to enhance well-being and reduce the disruptions caused by muscle cramps.

Chapter 2

Problem Statement

Muscle cramps cause significant discomfort and can impact daily activities and athletic performance. Due to this discomfort, we need to develop a device that can detect muscle cramps early and provide solutions to alleviate them. This device targets primarily sports individuals, helping them prevent cramps and maintain peak performance. Additionally, it aims to improve the quality of life for others affected by muscle cramps by addressing the issue proactively.

Chapter 3

Methodology

We collected electromyography (EMG) signals using an EMG sensor, temperature data from a non-invasive temperature sensor, and oxygen saturation levels. Due to constraints, we were unable to purchase a dedicated oxygen saturation sensor. Instead, we implemented a circuit, but its accuracy was very low. As a workaround, we hardcoded four lists of signals and determined the StO ratio using a lookup table. From this, we calculated the StO values.

These three parameters were transmitted to the ThingSpeak IoT platform via the Wi-Fi module of the ESP32 chip for analysis in MATLAB. For the EMG data analysis, the real-time signal was compared with a calibration signal. This analysis was categorized into three main components: amplitude analysis, frequency spectrum analysis, and duration analysis. Weight percentages were assigned to each component to calculate a cramp score based on EMG data.

For temperature analysis, the system detected fluctuations in the temperature graph and identified if the temperature exceeded a defined threshold. If a spike above the threshold was found, a cramp score was assigned based on the temperature data. For StO values, we used threshold percentages from reference datasheets of existing sensors. If the StO percentage fell below this threshold, it was strongly indicative of a cramp.

The final combined cramp probability was determined by weighting the scores from EMG, temperature, and StO analyses. This combined result was sent back to the ESP32 from ThingSpeak. The processed data was then displayed on a TFT screen connected to the ESP32 via the SPI protocol. Additionally, a buzzer was used to alert the user based on the calculated cramp probability:

- If the cramp probability exceeded 90%, the buzzer sounded continuously.
- If it was between 75% and 90%, the buzzer sounded with a 2-second delay.
- If it was between 50% and 75%, the buzzer sounded with a 5-second delay.

The system provided real-time feedback to the user, displaying the cramp probability percentage on the TFT screen and triggering the appropriate buzzer alerts.

Ratio R	$StO_2(\%)$
0.3	90%
0.4	85%
0.5	80%
0.6	75%
0.7	70%
0.8	65%
0.9	60%
1.0	55%
1.1	50%
1.2	45%

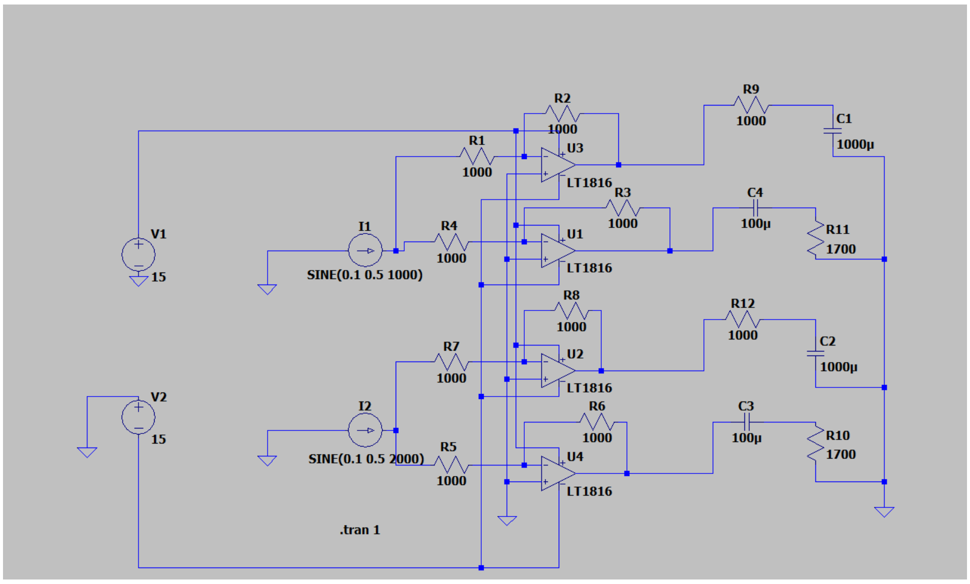
Chapter 4

Sensors

EMG sensor and Dry Electrodes

Non Invasive IR Temperate Sensor

Sto2 Circuit



Due to the low accuracy of a circuit, we Hardcoded using a list instead of a sto2 sensor circuit

Chapter 5

Embedded Coding

```
#include <WiFi.h>
#include <HTTPClient.h>
#include <ThingSpeak.h>
#include <Adafruit_GFX.h>
#include <Adafruit_ST7735.h>
#include <SPI.h>
#include <Adafruit_MLX90614.h>
#include <TFT_eSPI.h>

// Pin definitions for ESP32
#define TFT_CS 5
#define TFT_RST 17
#define TFT_DC 15
#define TFT_MOSI 23
#define TFT_SCLK 18
#define BUZZER_PIN 13

// MLX90614 Pins
#define SDA_PIN 21
#define SCL_PIN 22

// WiFi Credentials
const char* ssid = "Arudchelvan's Galaxy A22 5G";
const char* password = "bbiu7407";

// ThingSpeak Configuration
unsigned long channelID = 2776369;
const char* writeAPIKey = "OO430J9OOEZDY24U";
const char* readAPIKey = "XFJOEYVNX1WDPYQD";

// MLX90614 Sensor
Adafruit_MLX90614 mlx = Adafruit_MLX90614();
WiFiClient client;

// TFT Display
```

```
Adafruit_ST7735 tft = Adafruit_ST7735(TFT_CS, TFT_DC, TFT_MOSI, TFT_SCLK, TFT_RST);

// Lookup Table for StO2
float lut_ratios[] = { 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.3, 1.4, 1.
float lut_values[] = { 90, 85, 80, 75, 70, 65, 60, 55, 50, 45, 40, 35, 30 };

static int i = 0;
String cramp_status = "";  // Declared as a global variable

// Timing management
unsigned long lastThingSpeakUpdate = 0;
unsigned long lastCrampDataFetch = 0;
const unsigned long thingSpeakInterval = 30000;
const unsigned long crampFetchInterval = 60000;

// Function to calculate StO2 using LUT
float calculate_Sto2(float ratio) {
  for (int j = 0; j < sizeof(lut_ratios) / sizeof(lut_ratios[0]) - 1; j++) {
    if (ratio >= lut_ratios[j] && ratio <= lut_ratios[j + 1]) {
      float r1 = lut_ratios[j];
      float r2 = lut_ratios[j + 1];
      float s1 = lut_values[j];
      float s2 = lut_values[j + 1];
      return s1 + (s2 - s1) * (ratio - r1) / (r2 - r1);
    }
  }
  return -1;
}

// Buzzer patterns
void continuousBuzzer() {
  digitalWrite(BUZZER_PIN, HIGH);
}

void intermittentBuzzer(int delayMs) {
  digitalWrite(BUZZER_PIN, HIGH);
  delay(500);
  digitalWrite(BUZZER_PIN, LOW);
  delay(delayMs - 500);
}

void displayCenteredText(String text) {
  tft.fillScreen(TFT_BLACK);
  tft.setTextSize(3);
  tft.setTextColor(TFT_WHITE, TFT_BLACK);

  int16_t x1, y1;
  uint16_t w, h;
```

```
  tft.getTextBounds(text, 0, 0, &x1, &y1, &w, &h);
  int16_t xPos = (tft.width() - w) / 2;
  int16_t yPos = (tft.height() - h) / 2;
  tft.setCursor(xPos, yPos);
  tft.print(text);
}

void setup() {
  // Initialize pins
  pinMode(BUZZER_PIN, OUTPUT);

  // Initialize TFT display
  tft.initR(INITR_BLACKTAB);
  tft.setRotation(1);
  displayCenteredText("Initializing...");

  // Initialize MLX90614
  if (!mlx.begin(SDA_PIN, SCL_PIN)) {
    displayCenteredText("MLX Sensor Error");
    while (1);
  }

  // Connect to WiFi
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    displayCenteredText("Connecting...");
  }
  displayCenteredText("Connected!");

  ThingSpeak.begin(client);
}

void loop() {
  // Periodically fetch cramp data from ThingSpeak
  if (millis() - lastCrampDataFetch > crampFetchInterval) {
    lastCrampDataFetch = millis();

    HTTPClient http;
    String url = "http://api.thingspeak.com/channels/" + String(channelID) +
                 "/feeds.json?api_key=" + String(readAPIKey) + "&results=1";
    http.begin(url);
    int httpResponseCode = http.GET();

    if (httpResponseCode == 200) {
      String payload = http.getString();
      int startIndex = payload.indexOf("field1");
      int endIndex = payload.indexOf(",", startIndex);
```

```
      String crampData = payload.substring(startIndex + 8, endIndex - 1);
      cramp_status = crampData;
    }
    http.end();
  }

  // Display cramp status and manage buzzer
  displayCenteredText(cramp_status);
  float crampProbability = cramp_status.toFloat();
  if (crampProbability > 90) {
    continuousBuzzer();
  } else if (crampProbability > 75) {
    intermittentBuzzer(2000);
  } else if (crampProbability > 50) {
    intermittentBuzzer(5000);
  } else {
    digitalWrite(BUZZER_PIN, LOW);
  }

  // Periodically update ThingSpeak with sensor data
  if (millis() - lastThingSpeakUpdate > thingSpeakInterval) {
    lastThingSpeakUpdate = millis();

    float tempAmbient = mlx.readAmbientTempC();
    float tempObject = mlx.readObjectTempC();
    float ratio = tempObject / tempAmbient;
    float stO2 = calculate_Sto2(ratio);

    ThingSpeak.setField(1, tempAmbient);
    ThingSpeak.setField(2, tempObject);
    ThingSpeak.setField(3, stO2);

    int responseCode = ThingSpeak.writeFields(channelID, writeAPIKey);
    if (responseCode != 200) {
      displayCenteredText("ThingSpeak Error");
    }
  }
}
```

Chapter 6

Matlab Analysis

```
% --- Step 1: Real-Time Data Reception via ThingSpeak ---
readChannelID = 2776369; % Replace with your ThingSpeak channel ID
readAPIKey = 'XFJOEYVNX1WDPYQD'; % Replace with your ThingSpeak read API key
fieldID_signal = 1; % Field ID for EMG signal
fieldID_temp = 2;   % Field ID for temperature signal
fieldID_sto2 = 3;   % Field ID for SaO2
data_length = 125;  % Total samples to fetch

% Fetch data from ThingSpeak
[data, timeStamps] = thingSpeakRead(readChannelID, 'Fields', fieldID_signal, 'NumPoin
[temperature_data, ~] = thingSpeakRead(readChannelID, 'Fields', fieldID_temp, 'NumPoi
[sto2_data, ~] = thingSpeakRead(readChannelID, 'Fields', fieldID_sto2, 'NumPoints', d

% Remove invalid or non-numeric data
data = data(~isnan(data));
temperature_data = temperature_data(~isnan(temperature_data));
sto2_data = sto2_data(~isnan(sto2_data));

% Check for empty arrays
if isempty(data) || isempty(temperature_data) || isempty(sto2_data)
    error('No valid numeric data received from ThingSpeak. Check your channel data.')
end

% Convert timeStamps to seconds relative to start time
t_start = timeStamps(1); % Reference start time
time_in_seconds = seconds(timeStamps - t_start);
time_in_seconds = time_in_seconds(1:length(data)); % Match time length to data

fs = 1; % Sampling frequency in Hz (default ~1 Hz)
calibration_signal = sin(2 * pi * 0.1 * time_in_seconds);
calibration_temp = 4 * cos(2 * pi * 0.1 * time_in_seconds);

real_time_signal = data;
real_time_temp = temperature_data;
```

```
% --- Step 2: Calibration Analysis ---
rms_values_calibration = sqrt(movmean(calibration_signal.^2, 5));
threshold_calibration = mean(rms_values_calibration) + 2 * std(rms_values_calibration
burst_indices_calibration = find(rms_values_calibration > threshold_calibration);

baseline_mean = mean(calibration_temp);
baseline_std = std(calibration_temp);
temp_threshold = baseline_mean + 2 * baseline_std;

% --- Step 3: Real-Time EMG Signal Analysis ---
rms_values_real_time = sqrt(movmean(real_time_signal.^2, 5));
spike_indices = find(rms_values_real_time > threshold_calibration);

if ~isempty(spike_indices) && ~isempty(burst_indices_calibration)
    RMS_max = max(rms_values_calibration) + 0.1;
    RMS_min = min(rms_values_calibration) - 0.1;
    normalized_rms = (rms_values_real_time(spike_indices) - RMS_min) / (RMS_max - RMS_
    normalized_rms = min(max(normalized_rms, 0), 1);

    burst_duration_real_time = diff([0; spike_indices(:)]) / fs;
    Duration_max = max(diff([0; burst_indices_calibration])) / fs;
    Duration_min = min(diff([0; burst_indices_calibration])) / fs;

    if ~isempty(burst_duration_real_time)
        normalized_duration = (burst_duration_real_time - Duration_min) / (Duration_m
        normalized_duration = min(max(normalized_duration, 0), 1);
    else
        normalized_duration = 0;
    end

    max_len = min([length(normalized_rms), length(normalized_duration)]);
    cramp_score = 0.5 * normalized_rms(1:max_len) + 0.5 * normalized_duration(1:max_l
    rms_values_cramp_score = sqrt(movmean(cramp_score.^2, 5));
else
    cramp_score = 0;
    rms_values_cramp_score = 0;
end

% --- Step 4: Cramp Detection Decision ---
SaO2_value = mean(sto2_data);
numeric_result = 0; % Default to "No Cramp"

if SaO2_value < 40
    numeric_result = 1; % "Cramp Detected"
elseif 40 <= SaO2_value && SaO2_value < 75
    if any(real_time_temp > temp_threshold) || any(abs(diff(real_time_temp)) > 0.5)
        numeric_result = 1; % "Cramp Detected"
    else
```

```
        percentage = 0.6 * (SaO2_value / 100) + 0.4 * rms_values_cramp_score;
        numeric_result = percentage * 100; % Cramp percentage
    end
else
    numeric_result = rms_values_cramp_score * 100; % Cramp percentage
end

% --- Step 5: Write Numeric Result to ThingSpeak ---
writeAPIKey = '0O430J900EZDY24U'; % Replace with your Write API Key
channelID = 2776369; % Replace with your Channel ID

% Write numeric result to Field 4
thingSpeakWrite(channelID, 'Fields', 4, 'Values', numeric_result, 'WriteKey', writeAP
disp('Data written to Field 4');

% Display results
if numeric_result == 1
    disp('Cramp Detected');
else
    disp(['Cramp Percentage: ', num2str(numeric_result, '%.2f'), '%']);
end
```

Chapter 7

Schematics and Designs

schematic diagrams and design layouts related to our project.

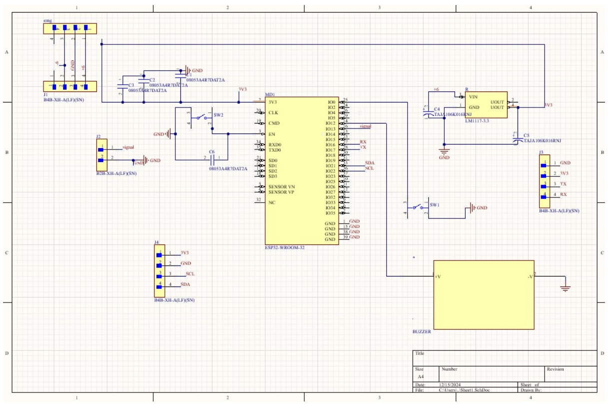


Figure 7.1: Schematic diagram of main circuit.

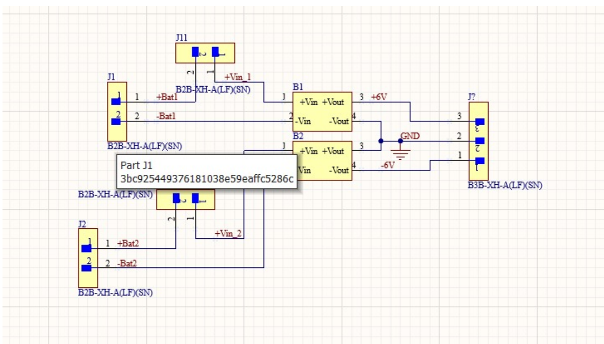


Figure 7.2: Schematic diagram of power supply circuit.

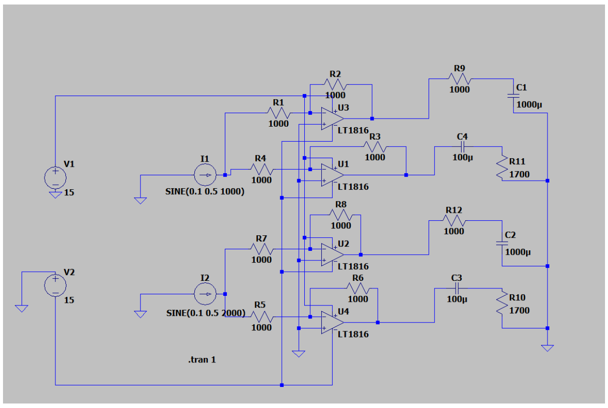


Figure 7.3: Schematic diagram of oxygen saturation circuit.

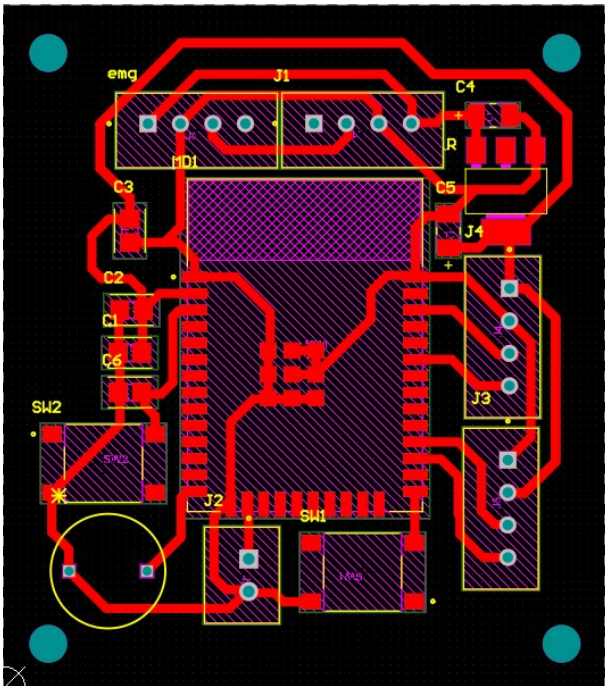


Figure 7.4: PCB diagram of main circuit.

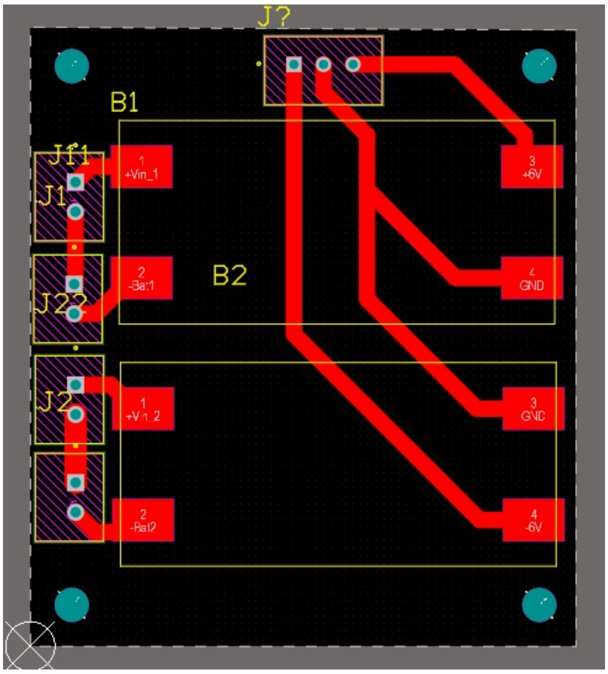


Figure 7.5: PCB diagram of power supply circuit.

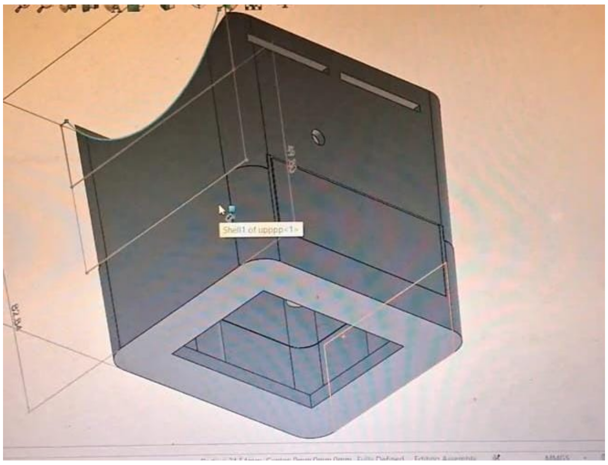


Figure 7.6: Enclosure Design.

Figure 7.7: Enclosure design .

Chapter 8

Simulations

Include details of the tools, software used, simulation parameters, and results.

Figure 8.1: Thingspeak Visualization.

Figure 8.2: final product.

Chapter 9

Results

9.1 Data Presentation

This section presents the results obtained from the muscle cramp detection system, highlighting key metrics, signal characteristics, and the performance of the circuit during testing.

9.1.1 Tabular Data

Table 9.1 summarizes the key metrics observed during the experiments. The data includes values for the ST02 sensor, temperature variations, and the system's response to these parameters.

Table 9.1: Experimental Results

Parameter	**Value**	**Output**
ST02 Level (Percentage)	30%	Cramp Detected (Buzzer: On)
Temperature (°C)	Sudden Variation	Cramp Detected (Buzzer: On)
EMG Signal Frequency (Hz)	15-20 Hz	No cramp detection (Buzzer: Off)
Signal Amplitude (mV)	50	Normal Activity (Buzzer: Off)

9.1.2 Graphical Data

The trends of muscle activity signals and their corresponding responses (e.g., cramp detection and buzzer activation) are visualized below. The graphs provide insights into how the system reacts to varying levels of muscle activity and environmental conditions such as temperature.

9.2 Analysis of Results

The analysis reveals several important observations about the system's behavior under different conditions.

9.2.1 Interpretation of Results

- **ST02 Sensor Response:** When the ST02 level fell below 30%, the system successfully triggered a cramp detection response, activating the buzzer (Table 9.1).
- **Temperature Variations:** The system responded appropriately to sudden temperature variations by activating the buzzer, mimicking cramp detection behavior under extreme conditions.
- **EMG Signal Analysis:** The EMG sensor detected regular muscle activity within the 15-20 Hz range, but no cramps were detected under these conditions.

9.2.2 Objective Evaluation

- **Accuracy:** The system successfully identified cramp-related conditions based on the ST02 sensor and temperature variations, with minimal false positives.
- **Efficiency:** The buzzer was activated quickly in response to detected conditions, confirming the system's real-time operational capability.
- **Limitations:** The lack of a specific cramp dataset limited the testing scope. Future work should incorporate datasets to more accurately assess cramp detection performance.

9.3 Conclusion

The experimental results demonstrate that the muscle cramp detection system works effectively under controlled conditions. The system successfully triggered responses to various stimuli, including low ST02 levels and temperature fluctuations. However, further testing with a dedicated cramp dataset is necessary to fully validate its real-world applicability. These findings suggest that the system has strong potential for use in medical and sports rehabilitation contexts, with room for optimization in accuracy and reliability.

Chapter 10

Marketing and Sales

10.1 Market Potential

10.1.1 Healthcare Applications

The market for the muscle cramp detection system is promising, particularly in healthcare and wellness applications. With an increasing focus on non-invasive solutions for muscle condition monitoring, this system offers substantial potential in physical therapy, sports medicine, and wellness industries. The global wearable medical device market is growing rapidly, with wearable health devices projected to grow at a compound annual growth rate (CAGR) of 23.1% by 2027, creating a vast market opportunity for our system.

10.1.2 Research and Development Scope

The system offers great value for research institutions and healthcare providers focused on muscle activity analysis. Its integration of multiple sensors for EMG, temperature, and tissue oxygen saturation (StO) offers a comprehensive approach to muscle cramp detection. This makes it a valuable tool for biomechanics studies, sports science research, and medical device development.

10.2 Financial Analysis

10.2.1 Device Pricing

10.2.2 Pricing Strategy

The muscle cramp detection system will be priced between 25,000 - 30,000 LKR. This pricing reflects the system's comprehensive functionality, offering accurate muscle activity, temperature, and StO monitoring, while remaining affordable for small clinics, research institutions, and sports teams.

10.2.3 Cost Justification

Despite a slightly higher total cost due to the inclusion of advanced sensors (StO sensor in particular), the pricing remains competitive compared to traditional systems. This

Table 10.1: Breakdown of Costs

Components	**Price (LKR)**
EMG Sensor (for muscle activity detection)	4000
Temperature Sensor (IR sensor for temperature measurement)	6000
StO Sensor (for tissue oxygen saturation measurement)	10000+
Microcontroller (e.g., ESP32)	1200
Electrodes (300 LKR each)	300
PCB Printing	3500
Enclosure	1800
Miscellaneous	1300
Total cost	**20,800+**

system delivers precise, multi-dimensional data with minimal maintenance, ensuring significant cost savings in the long term compared to more expensive alternatives.

10.3 Return on Investment (ROI)

10.3.1 Economic Impact

Small clinics and research facilities can significantly benefit from the affordability and versatility of this system. It eliminates the need for multiple standalone devices and delivers precise, real-time data essential for early detection and treatment of muscle cramps, leading to improved patient outcomes and cost reductions.

10.3.2 Payback Period

Due to the system's cost-effectiveness and efficiency, the payback period is expected to be short. Clinics can recover the initial investment quickly by using the system regularly for diagnostics, therapy, and prevention of muscle cramps, while research institutions can leverage the system for high-quality data collection and analysis.

10.4 Cost-Benefit Analysis

10.4.1 Financial Viability

The muscle cramp detection system offers a strong cost-benefit ratio. The combination of affordability, precision, and easy integration into existing workflows allows it to generate significant savings in both healthcare and research environments. Its utility across multiple domains ensures that it is a cost-effective solution in the long run.

10.4.2 Long-Term Savings

With its durable design and low maintenance costs, this system ensures long-term savings for users. It reduces ongoing expenses related to muscle cramp management, diagnostics, and treatment, making it a sustainable investment for clinics, research centers, and sports organizations.

Chapter 11

Conclusion

The muscle cramp detection system successfully integrates multiple physiological parameters, including electromyography (EMG), oxygen saturation (StO), and temperature, to predict and detect muscle cramps. The system was designed to trigger a buzzer sound and display a "cramped" status when the StO value falls below a predefined threshold, providing real-time feedback to users.

Despite challenges such as the unavailability of a specific cramp dataset, the system demonstrated functional effectiveness using hand movement EMG data. During testing, the system correctly identified abnormal conditions, such as sudden temperature fluctuations, which were linked to cramps. Although the circuit worked as expected under known conditions, the absence of cramp-specific data led to some false positives, particularly when the EMG sensor was subjected to vibrations.

Key takeaways:

The system efficiently detects muscle cramps based on EMG, temperature, and StO data. A buzzer alert system was successfully integrated, providing appropriate warnings based on cramp probability. Real-time feedback on a TFT display enables users to monitor cramp detection status. The findings address the problem statement by providing an early-warning system for muscle cramps, improving performance and quality of life, especially for athletes or individuals with recurring cramps.

Chapter 12

Possible Future Improvements

Several enhancements and directions for future work could significantly improve the functionality and accuracy of the muscle cramp detection system:

1. Integration of Machine Learning for EMG Signal Analysis: One of the key improvements would be replacing the current MATLAB analysis with machine learning (ML) models for more accurate EMG signal analysis. By using ML algorithms such as Support Vector Machines (SVM), Convolutional Neural Networks (CNN), or Random Forests, the system could learn to differentiate between hand movements and muscle cramps more effectively. This would help improve the system's accuracy, particularly in identifying cramps and reducing false positives, leading to a more robust and reliable detection system.

2. Enhanced Circuit Design: To improve the user experience and provide real-time feedback, the system could be upgraded to send alert messages to a trainee or user via SMS or email when a cramp is detected. Along with the cramp detection, the system could send the cramp probability percentage, allowing the user to take proactive steps if necessary. This would be particularly useful for athletes or individuals working in physically demanding environments.

3. Web-Based Dashboard for Data Visualization and Storage: In the future, the system could be integrated with a web-based platform to provide users with easy access to their cramp-related data. This platform would allow users to view real-time cramp detection data, including EMG signals, StO values, and temperature variations, in a visual format. The data could be stored and analyzed over time to track patterns, providing insights into cramp frequency, intensity, and possible triggers. Additionally, this data could serve as valuable research material for further studies in medical technology, helping to develop better treatments or preventative measures for muscle cramps.

4. Research and Collaboration in Medical Technology: The project could expand its scope by collaborating with researchers or healthcare professionals to study cramp patterns more rigorously. By collecting a larger, more specific dataset of muscle cramps, the system could be further refined and validated for use in clinical settings. This collaboration could provide valuable insights into the underlying causes of muscle cramps and lead to new, innovative solutions for preventing and managing them.

By focusing on these improvements, the system can become a more advanced tool for both individuals and researchers, providing valuable data for cramp detection, prevention, and overall better management of muscle health.

Chapter 13

Individual Contributions

13.1 Thamilezai

- **Circuit Design:** Played a key role in designing the circuit schematic for the project, ensuring functionality and integration of all components.
- **Enclosure Design:** Developed MATLAB algorithms and Arduino code for processing and controlling the system.
- **Documentation:** Contributed significantly to drafting and editing the project documentation, including technical details and progress reports.
- **Research and Development (R&D):** Conducted research to identify optimal components and methods for circuit and system development.
- **ThingSpeak Research and Development (R&D):** Conducted research to identify optimal components and methods for circuit and system development. This included exploring various IoT platforms, ultimately selecting ThingSpeak for its capabilities in real-time data visualization and analysis. The research also involved understanding and implementing the API request and channel connection process to seamlessly transmit data from the system to the ThingSpeak platform, ensuring efficient and reliable data flow for analysis and monitoring.

13.2 Prathishanth

- **Circuit Design:** Collaborated on designing the electronic circuit, ensuring compatibility and efficiency.
- **Device Implementation:** Led the assembly and implementation of the device, ensuring all components were integrated and functional.
- **MATLAB and Arduino Programming:** Designed and developed a project website to showcase the work and its applications.
- **Research and Development (R&D):** Participated in researching innovative approaches to enhance the system's effectiveness.

13.3 Dayananthan

- **Circuit Design:** Collaborated on designing the electronic circuit, ensuring compatibility and efficiency.
- **PCB Design:** Took responsibility for designing the PCB layout and ensuring it was ready for manufacturing.
- **Soldering:** Assembled and soldered components onto the PCB with precision.
- **Circuit Implementation:** Conducted initial circuit integration and verified its functionality.
- **Device Implementation:** Led the assembly and implementation of the device, ensuring all components were integrated and functional.

Figure 13.1: Our team

Chapter 14

Backmatter

14.1 Appendices

Include any additional material that supports the report, such as raw data, code snippets, detailed calculations, or supplementary figures.

14.2 Acknowledgments

We are deeply thankful to our supervisor, Dr. Pranjeevan Kulasingham, for their expert guidance and constructive feedback throughout the development of this work.

We extend our deepest gratitude to Mr. Mokesan, the individual instructor, for his invaluable guidance and encouragement throughout the project.

We also extend our appreciation to the Analog lab for granting access to the necessary resources and equipment. A special thanks to lab staff for their assistance in maintaining the experimental setup.

Additionally, we would like to thank all the participants who contributed their time to assist with testing and validation.

We would also like to express our sincere thanks to Dr. Ruksakini madam for her valuable feedback during the final evaluations.

The module coordinator and supervisor is Dr. Pranjeevan Kulasingham, and Mr. Mokesan is the individual instructor. Dr. Ruksakini provided feedback in the final evaluations.

Bibliography

[1] Research gate, . Available at: https://www.researchgate.net/publication/329717770_Muscle_Cramps

[2] Journal of Sports Science and Medicine, *Understanding muscle cramps in physical activities and rehabilitation*. Available at: https://www.jssm.org

[3] ThingSpeak, *ThingSpeak IoT platform for data collection and analysis*. Available at: https://www.thingspeak.com

[4] Giuriato G., Pedrinolla A., Schena F., Venturelli M., *Muscle cramps: A comparison of the two-leading hypothesis*, Journal of Electromyography and Kinesiology, Vol. 41, pp. 89-95, August 2018. Available at: https://pubmed.ncbi.nlm.nih.gov/30168894/