Review the assignment due date

final-project-skeleton

- Team Number:
- Team Name:

- Team Members:
- GitHub Repository URL:
- GitHub Pages Website URL: [for final submission]

Final Project Proposal

1. Abstract

In a few sentences, describe your final project.

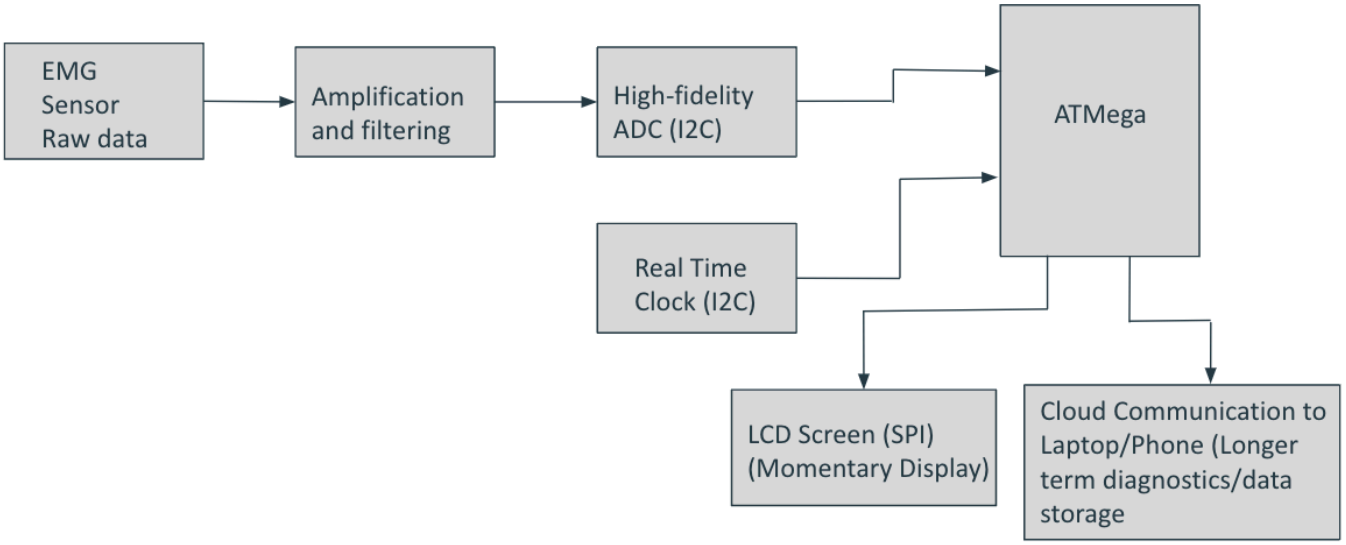
The goal of this final project is to detect and track bruxism (involuntary grinding of teeth). The device is worn as a headband when you're asleep, that uses EMG sensors to detect when your jaw muscles contract. It displays the results on an LCD screen that graphs how much time you spent grinding. There could also be a version with day time use, where it simply makes a sound whenever someone is grinding. This will alert them so they consciously stop grinding.

2. Motivation

Bruxism can lead to serious complications for a person's dental and oral health. It can cause jaw/tooth pain, wear your teeth down, and contribute to TMJ disorders. Current methods to diagnose nightly bruxism are cumbersome because it can involve having to get a sleep study. Simply wearing the device for a couple of nights offers a simpler solution for doctors to diagnose this issue.

3. System Block Diagram

Show your high level design, as done in WS1 and WS2. What are the critical components in your system? How do they communicate (I2C?, interrupts, ADC, etc.)? What power regulation do you need?



4. Design Sketches

What will your project look like? Do you have any critical design features? Will you need any special manufacturing techniques to achieve your vision, like power tools, laser cutting, or 3D printing?

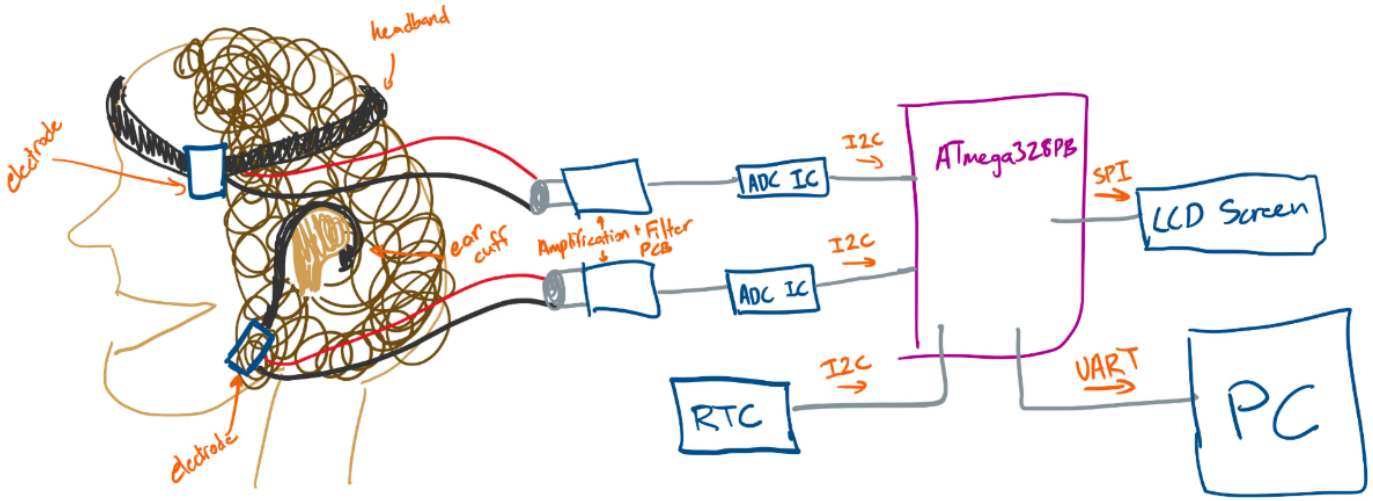
Design considerations:

1. Placement of sensors and electronics
2. Sensitivity of sensor that measures contraction
3.

Manufacturing

1. There won't be any special manufacturing techniques other than 3D printing and/or laser cutting an enclosure for the electronics.

Design Sketch



5. Software Requirements Specification (SRS)

Formulate key software requirements here. Think deeply on the design: What must your device do? How will you measure this during validation testing? Create 4 to 8 critical system requirements.

These must be testable! See the Final Project Manual Appendix for details. Refer to the table below; replace these examples with your own.

5.1 Definitions, Abbreviations

Here, you will define any special terms, acronyms, or abbreviations you plan to use for hardware

EMG: Electromyography

LCD: Liquid crystal display

Temporalis muscle: Jaw muscle that can be felt contracting on one's temple

5.2 Functionality

ID	Description
SRS-01	The EMG sensor will detect changes in muscle activation of the temporalis muscle
SRS-02	The ADC will convert the analog signals from the EMG into digital values

ID	Description
SRS-03	The LCD display will show the intensity of grinding over time for the night (using the real time clock to track time)
SRS-04	During the day time a buzzer sounds when the clenching is above a certain threshold
	LCD will comunicate over SPI
	RTC will communicate over I2c
	ESP32 will communicate to atmega over uart, and it will communicate with the blynk iot app

6. Hardware Requirements Specification (SRS)

Formulate key hardware requirements here. Think deeply on the design: What must your device do? How will you measure this during validation testing? Create 4 to 8 critical system requirements.

These must be testable! See the Final Project Manual Appendix for details. Refer to the table below; replace these examples with your own.

6.1 Definitions, Abbreviations

Here, you will define any special terms, acronyms, or abbreviations you plan to use for hardware

ADC (analog to digital conversion)

6.2 Functionality

ID	Description
HRS-01	The EMG sensors should be substantially sensitive to detect small muscle contractions (quantify)
HRS-02	The ADC module should support sufficient resolution for the signal (quantify)

ID	Description
HRS-03	The headband should be somewhat comfortable
HRS-04	Have a real time clock that can track over long periods time
	Power
	Filter (specify what frequencies it's cutting off)
	ESP32 with logic shifter

7. Bill of Materials (BOM)

What major components do you need and why? Try to be as specific as possible. Your Hardware & Software Requirements Specifications should inform your component choices.

In addition to this written response, copy the Final Project BOM Google Sheet and fill it out with your critical components (think: processors, sensors, actuators). Include the link to your BOM in this section.

https://docs.google.com/spreadsheets/d/1-zWV1aL9oH2EXkD6tylsm7H7TjFfnVomCbpaZuMzBcA/edit?usp=sharing

8. Final Demo Goals

How will you demonstrate your device on demo day? Will it be strapped to a person, mounted on a bicycle, require outdoor space? Think of any physical, temporal, and other constraints that could affect your planning.

For demo day we will test the device on one of our team members, and show the LCD updating in real time as well as the buzzer.

9. Sprint Planning

You've got limited time to get this project done! How will you plan your sprint milestones? How will you distribute the work within your team? Review the schedule in

the final project manual for exact dates.

Milestone	Functionality Achieved	Distribution of Work
Sprint #1	Reading data from muscle contraction	1 team member sets up hardware, other 2 code it
Sprint #2	LCD graph of clenching time, buzzer that alerts to clenching	all 3 team members code and debug
MVP Demo	Integration of previous sprints	3 team members optimize embedded system
Final Demo	All parts working along with sleeker mechanical design	All members work on mechanical design and final

This is the end of the Project Proposal section. The remaining sections will be filled out based on the milestone schedule.

Sprint Review #1

Last week's progress

Over this week our goal was to try to get the jaw clenching sensing down and interfaced with the atmega. We made significant progress with this by testing various sensors and we were successfully able to read values from the force sensitive resistor. We set up a voltage divider with the sensor and a set resistance value and converted the analog signal to a digital one. We aren't sure exactly if this will pan out to the final product because it's not very sensitive to the muscle movements. So we also tested various EMG sensors that we ordered/had available. They seem to be tracking the muscle contractions better (since thats what they're actually built for. We're able to read analog values on the scope, but they will need a lot more processing for the final product.

https://drive.google.com/file/d/1IZ25nzlBHVpCWKgTvnpPDgVclyI4WT8e/view?usp=drive_link

Current state of project

One sensor option working, (thought we were told it wasn't complex enough).

Next week's plan

Divide and conquer. One team member works on the RTC and LCD screen, one team member figures out the signal processing, one team member figures out the app.

Sprint Review #2

Last week's progress

Over the past week our goal was to solidify the sensor readings and get the LCD screen plotting graphs. So far we have gotten the LCD screen to display a graph image and have successfully plotted dummy graph data. We got the EMG sensor to output an analog signal that can be succesfully converted to digital values. There's still some work left to do to process this data and then display it on the graph. We also got the real time clock to work over i2c. Another group member was working on the ESP32 app integration, but there were some debugging issues there.

https://drive.google.com/file/d/1-YQYbSzBp0TBx5U1PUz_vQ8te4gLAX2D/view?usp=sharing

https://drive.google.com/file/d/1UxyY6kfUbMNISJIG7tiHbU9iLg-6Rss-/view?usp=sharing

Current state of project

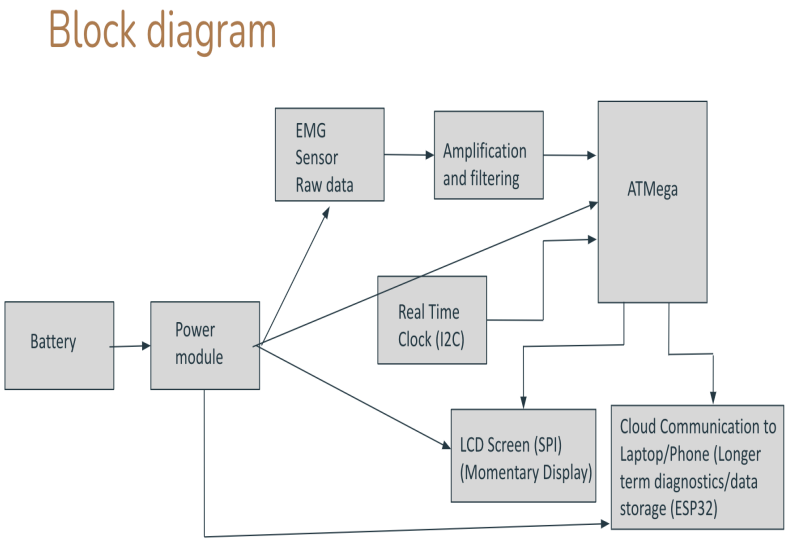
We essentially just have to integrate all the parts of the project together, and get the app working.

Next week's plan

Over this next week we will finalize each individual piece of the project and then put it together.

MVP Demo

1. Show a system block diagram & explain the hardware implementation.



2. Explain your firmware implementation, including application logic and critical drivers you've written.

   Most of the LCD implementation was taken from Lab4, and we added the logic/code that generates a graph over time. For the real time clock, we had to create our own i2c library for it. After setting the necessary register bits in the atmega for i2c, we created a few functions that set the time and get the time of the clock. Time is stored in a few registers from 00 to 07. The registers are seconds, minutes, hours, day, month year. In order to set time, we have to write the address of the clock, then the address of the register that we want to write, then the value in BCD format. Getting time is a similar process, but in reverse.

3. Demo your device.

   https://drive.google.com/file/d/1qDcoZRG4p9KLhYcpPG2Q98YLUuNXgrrb/view?usp=sharing

4. Have you achieved some or all of your Software Requirements Specification (SRS)?

   We have achieved most of our software requirements but not all of them

   - We WERE able to get the EMG sensor to detect the temporalis muscle clenching. We first made sure we got analog values by hooking up to a scope, then proceeded to implement interfacing with the atmega
   - We WERE able to convert the analog sensor values to digital values with the atmega onboard ADC. After we got the scope to read the sensor values, we implemented communication with the atmega.

- The LCD DOES show the intensity of clenching over time on a graph
- We have not yet impemented the buzzer sound that occurs during the day when a clenching event occurs. At this point it will be an addon if we have time, as our ultimate goal is implementing the night diagnoses
- The LCD DOES communicate over SPI
- The RTC DOES communicate over I2C
- We have had trouble getting the ESP32 to communicate with the atmega32, so have not accomplished this yet

5. Have you achieved some or all of your Hardware Requirements Specification (HRS)?

    - The EMG sensor WAS able to pick up the small muscle contractions of the jaw muscle. This is quantified by the range of voltage that the emg sensor would pick up. When there was no clenching, the scope read 0 V. When their was clenching there was about 3V.
    - The ADC module DOES have sufficient resolution to convert the analog values of the sensor to digital values. After hooking it up to the atmega, we were able to get the ADC conversion to scale from 0 to 1023, which was our goal.
    - We haven't gotten to the point of putting the whole product together with the headband (as all the electronics are still a jumble of wires), however currently when wearing the headband with just the sensor it's fairly comfortable.
    - The real time clock worked very well
    - We added a custom power module in order to run the device when not plugged in to the computer. It is simply a buck converter with a few external components surrouonding it (mostly capacitors and the battery mount thing). Now, when a 9V battery is connected, the whole system can be powered which is the ultimate goal for this product (since you don't want to sleep with something plugged into an outlet/computer).
    - In order to get the EMG sensor values to properly be read, we had to include an RC high pass filter that filtered out the noise. This is ultimately what made us able to read the values accurately.
    - The ESP32 and logic shifter were not included in our original HRS so this is a note that we added those componenets

6. Show off the remaining elements that will make your project whole: mechanical casework, supporting graphical user interface (GUI), web portal, etc.

    We begun the design of the mechanical casing in solidworks, but still need to make significant progess with that.

7. What is the riskiest part remaining of your project? The riskiest part will be transferring the completed circuit to the mechanical enclosure (where we will also try to switch out the wires to the flat ones that don't clutter up the circuit). This is risky because if we make any mistakes in doing this, we will have to debug the hardware for an issue, which will be stressful as we approach the final demo day deadline.
    1. How do you plan to de-risk this? We will make sure to transfer each subsystem one by one and very carefully, and then test the code for that subsystem to make sure that it works.
8. What questions or help do you need from the teaching team?

See ED discussion (the uart issues)

Final Project Report

Don't forget to make the GitHub pages public website! If you've never made a GitHub pages website before, you can follow this webpage (though, substitute your final project repository for the GitHub username one in the quickstart guide): https://docs.github.com/en/pages/quickstart

1. Video

[Insert final project video here]

- The video must demonstrate your key functionality.
- The video must be 5 minutes or less.
- Ensure your video link is accessible to the teaching team. Unlisted YouTube videos or Google Drive uploads with SEAS account access work well.
- Points will be removed if the audio quality is poor - say, if you filmed your video in a noisy electrical engineering lab.

2. Images

[Insert final project images here]

Include photos of your device from a few angles. If you have a casework, show both the exterior and interior (where the good EE bits are!).

3. Results

What were your results? Namely, what was the final solution/design to your problem?

3.1 Software Requirements Specification (SRS) Results

Based on your quantified system performance, comment on how you achieved or fell short of your expected requirements.

Did your requirements change? If so, why? Failing to meet a requirement is acceptable; understanding the reason why is critical!

Validate at least two requirements, showing how you tested and your proof of work (videos, images, logic analyzer/oscilloscope captures, etc.).

ID	Description	Validation Outcome
SRS-01	The IMU 3-axis acceleration will be measured with 16-bit depth every 100 milliseconds +/-10 milliseconds.	Confirmed, logged output from the MCU is saved to "validation" folder in GitHub repository.

3.2 Hardware Requirements Specification (HRS) Results

Based on your quantified system performance, comment on how you achieved or fell short of your expected requirements.

Did your requirements change? If so, why? Failing to meet a requirement is acceptable; understanding the reason why is critical!

Validate at least two requirements, showing how you tested and your proof of work (videos, images, logic analyzer/oscilloscope captures, etc.).

ID	Description	Validation Outcome
HRS-01	A distance sensor shall be used for obstacle detection. The sensor shall detect obstacles at a maximum distance of at least 10 cm.	Confirmed, sensed obstacles up to 15cm. Video in "validation" folder, shows tape measure and logged output to terminal.

4. Conclusion

Reflect on your project. Some questions to address:

- What did you learn from it?
- What went well?
- What accomplishments are you proud of?
- What did you learn/gain from this experience?
- Did you have to change your approach?
- What could have been done differently?
- Did you encounter obstacles that you didn't anticipate?
- What could be a next step for this project?

References

Fill in your references here as you work on your final project. Describe any libraries used here.