ECE 445

Fall 2023

Senior Design Final Report

Bed Sensor Alarm

Team 35

Colby King

Syed Ahmed

Ebaad Siddique

December 6, 2023

TA: Abhisheka Mathur Sekar

Abstract

In this project we designed, built, and tested a bed sensor alarm. This project had two parts to it: the weight sensor and alarm system. The weight sensors are wirelessly connected to the alarm to sense when the user is on and off the bed. In this report, we will be describing the work and contributions of team 35 where we worked on creating a bed sensor alarm to stop users from turning the alarm off and staying in bed. The goal of this report is to document what was done for this project, future plans, and obstacles we went through to finish building this product.

Table of Contents

Introduction

Project Overview

Most people wake up to an alarm in the morning. It is the easiest way to get the day started at the time you want. The problem is that it is too easy to see the alarm and go back to bed. With the snooze option or just general sleepiness, some people find themselves just going back to sleep regardless of the alarm system. There needs to be an alarm that can determine if someone is still in bed and not getting up at the correct time. To stop the user from turning the alarm off and staying in bed, there should be a way to tell if the user has gotten out of bed and stayed out of bed.

The solution to that problem is weight sensors. Using weight sensors this product can detect when weight has been removed from the bed and check to make sure the weight has stayed off the bed. The system will have an alarm function that the user can set and the weight sensing will be the main way to turn off the alarm. The sensor will fit between the bed frame and the mattress and the user can lay on the bed normally without any problems. This device will be able to sense the total weight of the bed and use that to make sure the user has not gotten back into bed. When the alarm goes off the user will have to press a button and the alarm will go off only if the weight on the bed has decreased by a set amount. Once the button has been pressed the user has five minutes to not get back into bed. If the bed senses another large weight increase within that time, the alarm will turn back on and they have to repeat the process. This will all occur without disrupting the stability of the bed.

High Level Project Functionality

Throughout our Bed Sensor Alarm project we had three high level requirements that would tell us when this project is fully functional. The first requirement is that the system should have three identifiable states. The first state being regular standby mode which is just the normal clock operation. Second is the alarm mode which is when the alarm sounds for the user to wake up and the snooze button needs to be pressed for it to stop. The last state is timer mode. In timer mode the snooze button has been pressed and the timer starts. The timer is set to five minutes. Within those 5 minutes the alarm is off but if the user returns to the bed the alarm will turn back on and the 5 minute timer will reset once the user gets off the bed again. The second high level requirement is that the sensors should be able to wirelessly share the weight information with the central alarm and be able to sense when more than 50 lb of weight has been removed from the bed.
The third requirement is after the snooze button has been pressed, the system will return to alarm mode if the 50 lb of weight has been added back onto the bed before the timer goes off.

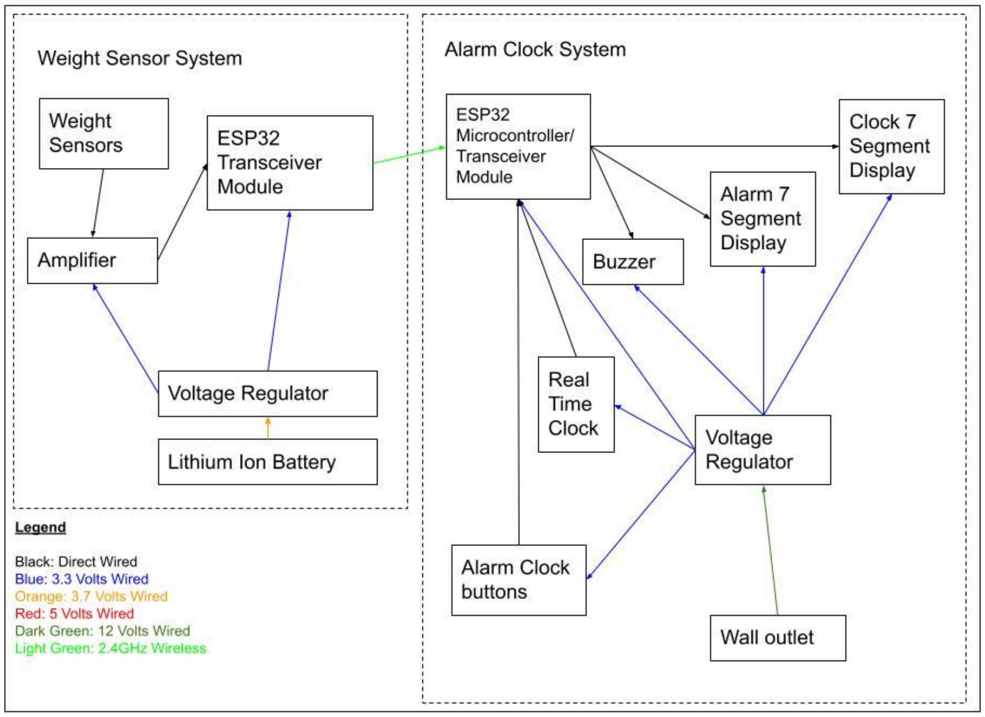

Figure 1.1: Original Top Level Diagram

As we were building and testing our design we had a major issue occur that required us to change our design. Many of the components we had ordered did not arrive and one of them arrived the week of the final demo. The components that did not arrive were the batteries, the battery holder, the jack power adapter, and our LED displays. To account for this we ordered some components on amazon last minute to complete our testing and building. This also led to us being unable to use our PCB as most of the components we used would not fit into the old PCB design. We did not have enough time to update the PCB and get it reordered.

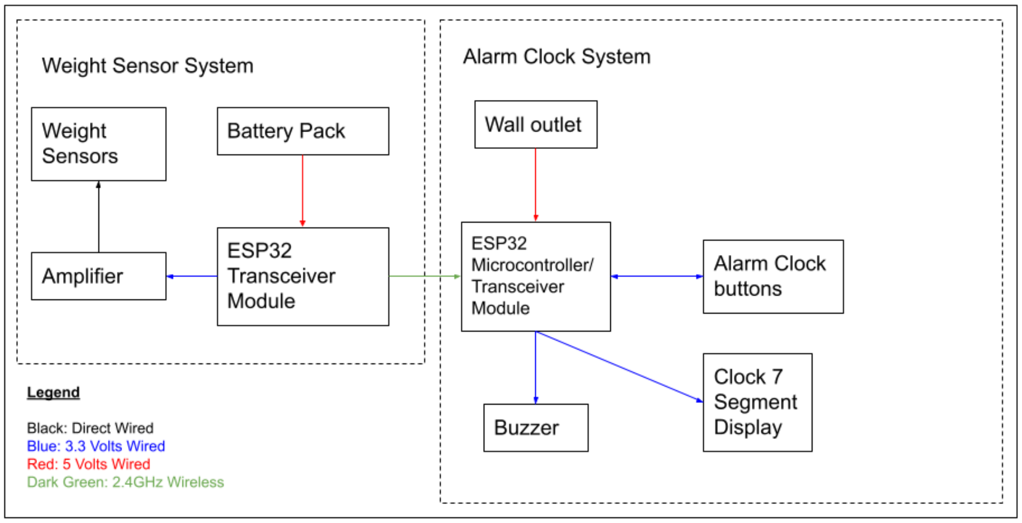

Figure 1.2: Updated Top Level Diagram

Description of Weight Sensing System:

Old Power Subsystem: This entire system will be powered by two 18650 lithium-ion batteries that operate at 3.7 volts. Using a voltage regulator there will be a voltage step down to 3.3 Volts. It will power the ESP32 wireless data chip and the weight sensors. The wireless data chip will send the live weight information to the alarm system through Bluetooth Low Energy (BLE).

New Power Subsystem: The system will be powered by a battery pack with a 5V output. Using a USB to Micro USB adapter, the battery pack will power the ESP32 at 5V. The ESP32 chip has a 3.3V output that can be programmed for its pins. The weight sensors require a 3.3V input that the ESP32 will provide through the amplifier that is connected to the weight sensors.

Weight sensor subsystem: The weight sensors will be multiple half-bridge load cells connected to an amplifier. The amplifier increases the voltage information that the load cells produce to get a reading from the weight sensors. This will allow us to accurately get weight data from the sensors and pass the data on to the alarm subsystem wirelessly. These weight sensors will be placed between two mattress sized boards in the corners to allow for maximum distribution of weight. There will also be an additional ½ inch thick board placed under each sensor to allow for enough space between the boards for circuitry and the battery. These sensors will also have 3D printed brackets to stabilize them. This will ensure all the weight is accurately on the load cells and to eliminate any inconsistent data from the sensors shifting.

Control subsystem: This control subsystem will take the total weight of the sensors and pass the bit information to the alarm clock system for the overall logic. We will use two ESP32 Bluetooth Chips to communicate between the two systems wirelessly. The signal will be sent using the standard BLE at 2.4GHz. This chip is only sending data and not receiving any so any low powered bluetooth chip would have worked for this component. The BLE runs at anywhere from 1% to 50% the power consumption used for normal bluetooth transmission. Since this system will be under a bed, battery usage is a big concern for long term functionality.

<u>Description of Alarm System:</u>

<u>Old Power subsystem:</u> This entire system will be powered by a 12V wall outlet with a voltage regulator that can step down the voltage to 3.3V. This will directly power every chip on the alarm clock system. Every chip on this system has a voltage range that allows for a 3.3V input.

<u>New Power subsystem:</u> The system will be powered by an AC wall charger with a 5V output. Using a USB to Micro USB adapter, the wall plug will power the ESP32 at 5V. The ESP32 chip has a 3.3V output that can be programmed for its pins. The buzzer has an input range of 1.5V to 12V while the LED display has an input voltage of 3.3V to 5V. The 3.3V output from the ESP32 is enough to power every component in this system.

<u>Control subsystem:</u> We will use an ESP32 to wirelessly accept the weight information from the weight sensor system. Using the Arduino IDE software to download code onto the chip we will check the weight before the alarm goes off and compare it to the current weight on the bed to ensure the user has gotten off the bed. When that happens and the snooze button is pressed, the alarm will turn off and a five-minute timer will start. Once no weight has been added back onto the bed within five minutes the alarm will stay off. If the User gets back into bed the alarm will turn back on. Once the user gets off again the alarm will turn off and the timer restarts.

<u>User subsystem:</u> This subsystem will consist of an alarm clock containing 5 buttons, a buzzer, and one LED seven-segment display. The four control buttons will allow the user to set the current time or the alarm time while the fifth button will be used as a snooze button for the alarm. The four control buttons consist of a clock button, an alarm button, an hour increment button, and a minute increment button. If the alarm or clock button is pressed, the screen will flash for 5 seconds showing the time they are attempting to change. While the time is flashing, the user can press the hour or minute button to increment the time they are changing. After any changes the 5 seconds restarts and if no buttons are pressed during that time, the clock returns to normal operation.

When the alarm starts, the snooze button can only be pressed after the user has gotten off the bed. The LED display will accurately update for all button inputs.

Project Design

Weight Sensor System

The weight sensor system is the simpler of the two systems and was so simple we decided that we didn't need a PCB for this system. For the weight sensors specifically, we chose half bridge load cells. These weight sensors were very powerful and we picked them primarily for their size. They are only 8mm thich which meant they would be small and easily fit between the two boards.

We were going to use a RN-41 chip for the bluetooth on this system but after discussing, the better solution was to use the same chips for both sides of the project for compatibility. This verified our choice of using the ESP32 chip.

The most important aspect of this system is the power. We originally had 3.7V lithium ion batteries and 3.3V output voltage regulator. After dealing with the delivery issues, we had to settle for another option. Colby had a 5V portable charger that we were able to reuse for a power source since the ESP32 can be powered with a 5V source. This battery was smaller in capacity but gave us the same function the batteries had but with less pieces. We had worries about amperage but the ESP32 had a max current of 500ma while the battery could output up to 3 amps. With the needed value being so much lower than the battery output we knew there would be little to no concerns regarding power.

While building we also decided to 3D print brackets for the load cell to keep the sensors stable while it reads data. In the testing phases there were some notable differences in reliability of the load cells so the brackets were to help keep everything stable and decrease shifting from the individual sensors.

Below is a picture of the final product with the weight sensor system being shown on the left side. The blue brackets are visible in the corners with the load cells attached. In the center the ESP32 chip and battery cell can be seen providing power to the system.

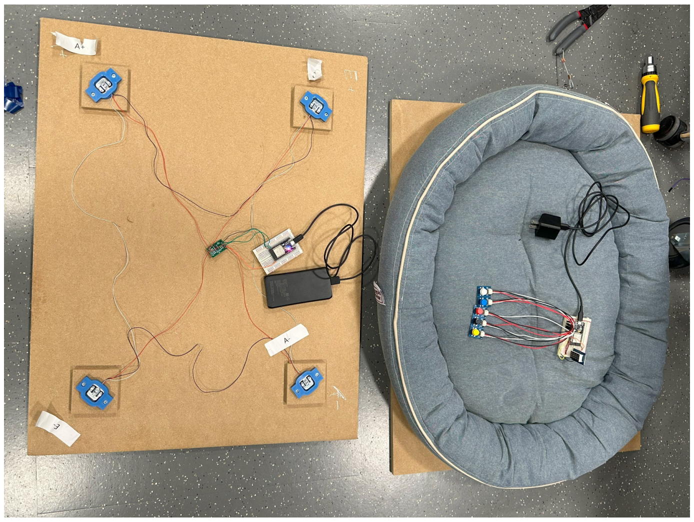

Figure 2.1: Final Build

Alarm Clock System

We knew going into this project that the alarm side would require a lot more thought and effort since it is the main operational system.

We chose to use a 7 segment LED display over a screen due to the simplicity and it is a very common display for alarm clocks. There was no need to overcomplicate the design. When the original LED did not arrive, a new 7 segment display was ordered off of Amazon quickly to replace the other one. The new display required different pin outputs and a slightly different pin layout. Due to the new display being used, we opted to use just one single display instead of the original design of 2 separate displays. The original design was going to have the second display only for the alarm clock function. It would show the alarm time and become a timer when the snooze button was pressed alerting the user how much time they had before they could get back into bed. The changing components combined with the missing power adapter made us have to completely redesign this system and throw away our PCB.

Since the jack adapter never arrived and the design was rebuilt some other power changes had to be made. Once again Colby had a wall plug power brick with a 5V output so we decided to use that as the new power source. We connected it to the ESP32 using a USB to Micro USB cable. We knew we wanted this side to be wall powered so that the alarm clock would always be functioning and not have to worry about batteries. Now instead of the original wall plug and voltage regulator, the ESP32 was powering all the components. The plug and cord can be seen in the photo above with the complete design.

For the buttons we ordered a basic 5 button package with big enough buttons for normal human operation and connected them to the chip. They were the simplest part of the design and required no changes at all.

The last physical component was the audio output for the alarm. When designing the alarm originally, we opted for a PFPlayer chip that could play downloaded audio whenever the alarm went off. After discussing with the TA's and considering the cost and difficulty aspect of the design, a change was made to use a buzzer instead. This is a much simpler piece of equipment and also operates in the 3.3V output range that the ESP32 can produce. Adding it to the design was as simple as connecting two wires and a few lines of code to set the frequency.

Bluetooth Operation

When looking at the bluetooth design and options, we chose BLE which has a much smaller power output compared to traditional bluetooth. As discussed before, we chose to use the same chips for both systems of our project. This kept everything more consistent and there was no need to learn about multiple types of chips.

PCB Design

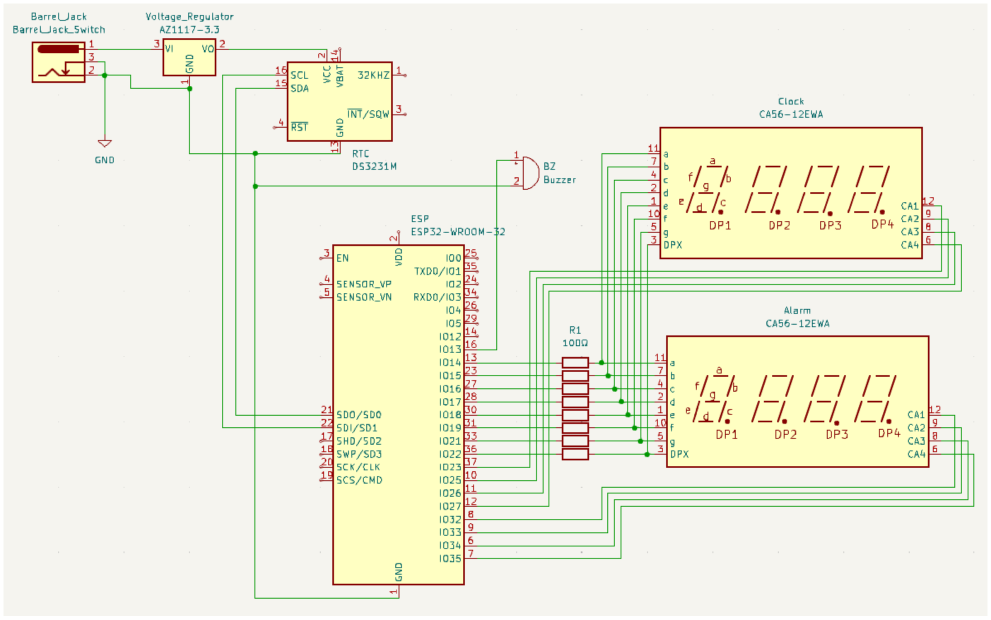

Figure 2.2: PCB Schematic

Above is the original PCB design before the delivery issues. In this image the barrel jack and LED displays never arrived. Due to this, we could not power the PCB or get an output on the display. With no PCB we were unable to use the ESP32 chips we had purchased. Luckily, we were using 2 ESP32 dev boards for testing that we were able to incorporate into the final design. Our original goal was not to use the dev boards on the final product but we were left with no choice given the lack of time needed to reorder a new PCB. Below is our footprint

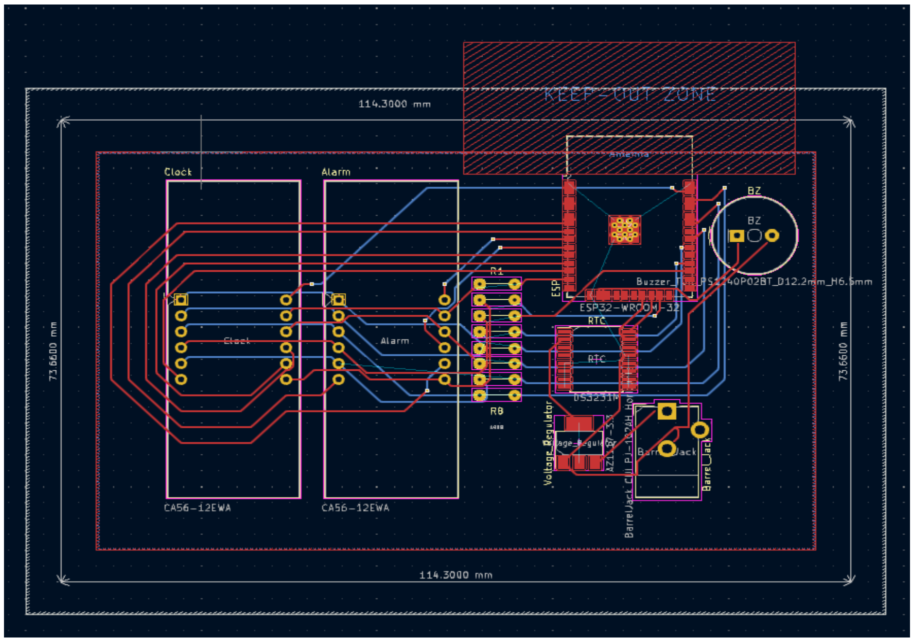

Figure 2.3: PCB Footprint

Requirements and Verifications

The weight sensor and alarm clock systems both underwent rigorous verification tests to ensure that our technical requirements were met for both systems.

Weight Sensor System Requirements

Power	
Requirement	Verification
The voltage regulator must be able to supply 3.3 V to the HX711 load cell amplifier and ESP32 from a 3.7 V battery supply.	Utilize a voltmeter with a DC power supply to confirm voltage to make sure it meets the requirements of the sensor.
Weight Sensors	
Weight sensors must accurately detect the weight placed on the platform.	Place known weight on the platform and compare weight sensor output values.
The arrangement of load cells on the board must provide accurate weight readings regardless of weight distribution.	Place known weights at different locations on the platform and observe change in response to redistribution of weight.
The weight sensor subsystem must successfully detect changing loads in real time.	Add and/or remove weight during operation and observe weight data over time.

Weight Sensor Verification Tests

Three tests were employed to demonstrate different aspects of the weight system's reliability and capability.

Test #1: Consistency & Accuracy

Test subjects were asked to step on and step off the weight platform three times. The test subjects were also asked to stand on a commercial bathroom scale in order to record their actual weight. This test aimed to demonstrate how consistent and accurate the readings of our weight platform were.

Test Subject	Platform Test #1 (lbs.)	Platform Test #2 (lbs.)	Platform Test #3 (lbs.)	Bathroom Scale (lbs.)	Test Standard Deviation	Percent Difference (Test Average/Scale)
Ebaad	180	178	181	203	1.527525	88.5%
Syed	188	189	187	192	1	97.92%
Colby	156	154	154	156	1.154701	99.15%
Kylie	107	107	107	106	0	100.94%

Table 3.1: Consistency & Accuracy Test Results

Test #1 results demonstrate the high accuracy and consistency of the weight sensor system. The standard deviation indicates that readings vary only slightly more than 1.5 lbs on average for the same user, showing great consistency. Furthermore, readings from the weight sensor deviated from the bathroom scale results by less than 12% for all users. This test does suggest that the system suffers in both consistency and accuracy at higher loads, as suggested by the discrepancy in standard deviations and percent differences between subjects with low and high body weights.

Test #2: Balance

Test subjects were asked to place their weight in various configurations in order to demonstrate how the weight sensor's readings may be affected by weight distribution. Figure 3.1 shows how the board was marked for this test.

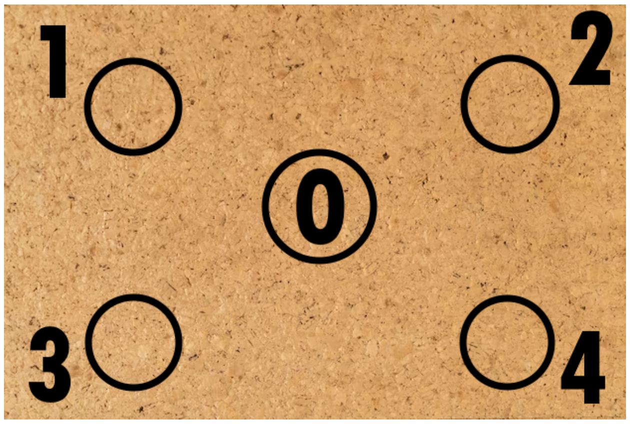

Figure 3.1: Balance Test Markings

Test Subjects	1&2 (lbs.)	3&4 (lbs.)	1&3 (lbs.)	2&4 (lbs.)	2&3 (lbs.)	1&4 (lbs.)	1&2&3 &4 (lbs.)	0 (lbs.)	Test Standard Deviation	Test Standard Deviation Without 0
Colby	151	155	155	143	157	155	149	128	9.65752	4.87950036
Ebaad	160	192	178	161	173	188	192	136	19.4642	13.7199889
Syed	178	188	185	173	187	189	186	148	13.7710	5.9361684
Kylie	105	104	107	107	107	107	106	98	3.09088	1.21498579

Table 3.2: Balance Test Results

Test #2 results demonstrate that the weight senor platform is able to measure the weight placed on it with reasonably accurate results regardless of weight distribution. The greatest standard deviation value recorded, slightly less than 20 lbs, remains well below the weight difference threshold required for the snooze button to become enabled (50 lbs). Furthermore, when test 0 is removed from the calculations, the largest standard deviation recorded drops to slightly less than 14 lbs. Test 0 had subjects place all their body weight in the center of the board, causing the platform board to bend significantly in the middle. Due to this, and the fact that this sort of weight distribution is extremely unlikely for users laying down in bed, this test can very reasonably be removed from the standard deviation calculations.

Test #3: Real Time

The test subject (Colby King) was placed on the board and backpacks of different weights were handed to him. This test aimed to demonstrate the board's ability to accurately detect changes in weight in real time during operation.

Test Subject	Step on	Grab 8 lbs bag	Grab 17 lbs bag	Remove 17 lb bag	Remove 8 bag
Colby	147	154	165	153	146
	Δ147	Δ7	Δ11	Δ-12	Δ-7

Table 3.2: Real Time Test Results

Test #3 results demonstrate the ability of the weight sensor system to adjust to real time changes in load. Once again, we do see that accuracy suffered when Colby grabbed the heavier bag as opposed to the larger bag.

Alarm Clock System Requirements

Power	
Requirement	Verification
The voltage regulator must be able to supply 3.3 V to the ESP32, RTC, buttons, and 7 segment display(s) from a 12V wall outlet supply.	Utilize a voltmeter with a DC power supply to confirm voltage to make sure it meets the requirements of the sensor.
User I/O	
The ESP32 must DISABLE the snooze button UNTIL it detects a >50 lbs difference between the weight on the platform before and after the alarm goes off.	Add and/or remove weight during operation and test the functionality of the snooze button throughout.
The alarm must reactivate if it detects a >50 lbs weight increase in the platform after the snooze button has been pressed within a set timer.	Add weight to the platform after the snooze button has been pressed.
The 7 segment LED(s) must display the alarm clock logic correctly.	Visually inspect LED(s) and check for errors.
The buttons must control the clock, alarm, and snooze features accordingly	Press buttons in random sequences to check for unexpected behavior
Bluetooth	
Alarm clock ESP32 is capable of receiving data from the weight sensor system ESP32.	Pair the two ESP32 chips and run a program on one chip to be sent to the other chip. Using the serial monitor, read the data being sent and the date being received and compare the two values. Once operational, set the value being sent as the weight information and confirm live data is accurate.

Alarm Clock Verification Tests

There were two specific components included in the alarm clock verification tests. The bluetooth and the operational alarm clock.

Bluetooth

For the bluetooth tests there were no quantitative results to record other than successful operation. Once the chips were paired very simple programs were tested to see the results.

Chip A had the LED display connected and running while Chip B had the test program. Chip A was running a code that would take the integer value received from Chip B and display it on the LED. The first program was a simple incremental counter going from 1 to 9999. This test worked perfectly so we did a couple more including using a random number generator and connecting the weight sensors and getting real time feedback across the chips.

Alarm Clock

For this test we just had to prove that the written code for the alarm clock had no bugs or glitches that would impede functional use of the alarm clock. For testing we had each member use the alarm clock normally to get familiar with the buttons then try to intentionally break the alarm by doing uncommon button presses or pushing the limits of certain operations.

These two tests proved to be effective by exposing 2 major flaws in the alarm clock. First the set alarm function did not account for looping so someone could set a time that didn't exist such as 3:65 AM. This alarm time would never be reached and the alarm would be turned off. The second problem was if you change the alarm time or clock time it would flash the screen for 5 seconds then go back to normal operation. This would create a 5 second lag in the data being received by the bluetooth chips and break the complete build alarm clock. Using these tests we were able to fix the problems and create a fully functioning alarm clock that can read the weight in real time.

Conclusion

Accomplishments

Our project exceeded expectations in terms of accuracy with the sensors and working with the alarm. The load cells were capable of sensing on the board at all five positions when we were calibrating. When collecting data the weight load cells were able to show weight in pounds with a 10% margin of error which is really close. We were also able to pass the weight data to the alarm system wirelessly. When configuring the alarm clock system, we got it to work with all buttons, the main ones being snooze and the alarm set button.

Uncertainties/Challenges

Some obstacles that we faced were with the PCB design. Many of our parts didn't arrive or came late. For example, the 7 segment hex displays, the barrel jack, and our lithium ion batteries. So we had to improvise and breadboard mostly everything. We ordered some parts that could make maybe half the PCB work, but while testing it wasn't functioning with the sensors, as the PCB was for the alarm side. The issue was most likely with the soldering and the programming of the hex display which led to this issue. However, we were still able to deliver a successful project with all of our high level requirements being satisfied.

Future Work/Alternatives:

Overall, we learned a lot from creating a project from the ground up. We learned about the different ways to implement an alarm while figuring how to integrate a designed PCB to our bed sensor alarm. When creating the alarm, we researched the different types of bluetooth operations and the limitations associated with each. This essentially helped up debug code for both the weight sensors and alarm because we thought of optimal ways of debugging it with the physical hardware components. We had many obstacles when creating this project but collecting data was the main factor that helped us identify the root of the problem and fix it.

Furthermore, some recommendations for future work could be integrating our alarm with smart home ecosystems such as Google Home and Amazon Alex since these two also have alarm notification systems. We could also create a user friendly mobile app for monitoring and configuring the bed sensor alarm. This would mostly be useful for patients since tracking their health is crucial. We could also integrate this with current health monitoring apps. The component we initially wanted to attach in our project was the DF Player chip for programmable audio on the alarm system but we couldn't get it to work so we went with a buzzer. Lastly, creating an algorithm for bed movement detection could be useful when taking in sensitivity and specificity of sensors.

Ethics and Safety: (mostly relevant to the software part of this project)

We will adhere to follow the ethical and safety guidelines from the IEEE Code of Ethics –

1.) **To be honest and realistic in stating claims or estimates based on available data; [2]** – We will always do detailed research before we perform any sort of design and will make sure we know everything about our components and the steps to carry out the work.

2.) **To avoid real or perceived conflicts of interest whenever possible, and to disclose them to affected parties when they do exist; [2]** – We will always be clear about the design concepts and the functionalities in the final report so We can make sure there will be no conflicts. We will also discuss my ideas with the TA or any doubts we have about certain components, so we can create and decide on a final design and so there are no team conflicts as well.

3.) **To reject bribery in all its forms; [2]** – We will not be involved in any sort of malpractice.

4.) **To improve the understanding of technology; [2]** – We will make sure that my part of the final report will be detailed so that there won't be any doubts or confusion about the functionality of our device. We will make sure to write everything in a way so that even common people who are trying to build this can understand it. The report will also include any flaws or limitations of the device so the person who is using the product is aware.

5.) **To maintain and improve our technical competence and to undertake technological tasks for others only if qualified by training or experience, or after full disclosure of pertinent limitations; [2]** – We will help others with my experience and expertise or understanding of concepts that I only know. We will also ask permission from peers, TAs, and professors before trying to help with something or someone.

6.) **To seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, and to credit properly the contributions of others; [2]**- We will always take honest criticism from TAs, professors, and peers in a positive manner so we are able to improve the project and the design portion of it. We will also offer honest criticism to members of the team as well as other teams so we can all help improve or correct the certain work or data being completed. We will also credit outside references used in our report.

7.) **To treat fairly all persons and to not engage in acts of discrimination based on race, religion, gender, disability, age, national origin, sexual orientation, gender identity, or gender expression; [2]**- We will accept people from all different race, religion, gender, disability, age, national origin, sexual orientation, gender identity or gender expression and will not discriminate anyone in the class or anywhere else.

8.) **To avoid injuring others, their property, reputation, or employment by false or malicious action; [2]**- We will not be involved in activities that will injure or hurt others and their property.

9.) **To assist colleagues and co-workers in their professional development and to support them in following this code of ethics; [2]**- We will work together with my team members to complete our goals. We will do this by keeping the IEEE code of ethics in my mind while performing my tasks alone or with them.

Safety Concerns

The main safety issues for our project would be assembly. We could experience issues with the soldering or getting electrocuted as we put it together. This could eventually become an issue once we put together the final product since there could be loose components or wires that could cause electrocution. The main concern being the sensor, since if the wires cause electrocution then the sensors would catch on fire. And this would cause the bed to catch on fire since the sensors are between two pieces of plywood and plywood is a source that causes flames to grow. This would essentially be a big risk for the user who is sleeping on the bed. Moreover, the alarm system would be plugged into a wall outlet, and it would have a higher wattage than our battery so to combat this we will use a voltage regulator to step down the voltage. We will make sure that any sort of exposed metal on the wall outlet is covered or put away so that it doesn't harm the user or while the product is in use. As told in the IEEE Code of Ethics, we will paramount the safety , health, and welfare of the public to strive to comply with ethical design and sustainable development practices.

Cost and Schedule of Project

Schedule

Week	**Date**	**Colby**	**Ebaad**	**Syed**
1	10/21	Initial Web Board Post		
2	10/28	Web Board Post Approval		
3	9/4	Early Project Approval		
4	9/11	Project Approval		
5	9/18	Project Proposal and Team Contract		
6	9/25	Design Document plus order parts		
7	10/2	Design Review and Individual Peer Review		
8	10/9	Programming: Communication to alarm clock from weight sensors	Assembling weight sensors to board while writing code to testing on actual bed	
9	10/16	Programming: Communication to alarm clock from weight sensors	Assembling weight sensors to board, testing on actual bed	
10	10/23	Building alarm clock using display segment	Assembling weight sensors to board, testing on actual bed	Building alarm clock using display segment
11	10/30	Testing alarm clock with sensors, checking functionality		
12	11/6	Testing, checking for issues with the alarm.	Testing, checking for issues while creating 3D printed brackets for sensor stability	
13	11/13 (Mock Demo Week)	Test and make sure everything works with the alarm, had a few issues with sensors not reading anything, fix it immediately		
14	11/20	Fall Break		
15	11/27 (Final Demo Week)	After demo made a few final touches to get an exact video to show in the presentation		
16	12/4	Final Presentation and Paper		

Parts Cost Analysis

Bed Sensor		
Part Description with Link:	**Quantity**	**Cost**
Half bridge sensors *Includes amplifier	4	8.99
ESP32 Bluetooth Microchip (School provided) RN-41	1	0.00
ESP32 Dev Board (bought ourselves) (2 pack, one used for alarm)	1	7.49
Wooden Boards (School provided) (Dimensions: 30” x 24”)	2	0.00
Small Fiber Board Pieces (School provided) (Dimensions: 3” x 3”)	4	0.00
Total		16.48

Table 5.1: Bed Sensor Parts

Alarm Clock		
Part Description with Link:	**Quantity**	**Cost ($)**
7 Segment Display (2 pack, used only one display) (bought ourselves)	1	6.99
ESP32 Microcontroller and Dev Board	1	7.49
Wall Plug	1	10.99
Buzzer (a pack, used only one)	1	6.99
PCB (designed)	1	0.00
Buttons	5	0.99
Total		33.45

Table 5.2: Alarm Clock Parts

Labor Cost

Name	**Weekly Hours**	**Hourly Pay**	**Weeks**	**Scaling Factor**	**Cost (USD)**
Colby K.	**10**	**50**	**12**	**2.5**	**15,000**
Syed A.	**10**	**50**	**12**	**2.5**	**15,000**
Ebaad S.	**10**	**50**	**12**	**2.5**	**15,000**
Total	**$45,000**				

Table 5.3: Labor Cost

The total cost for parts as seen above comes out to a total of $49.93. We will assume it took us 10 hours per week for 12 weeks. We can expect a salary of $50/hr (approximately the average salary of an electrical engineer in the US) ×2.5 (overhead factor) × 120 hours worked = $15,000 per team member. The total labor cost should account for all 3 members, therefore, $15,000 x 3 = $45,000.

Citations

1. Fliflet, Arne. “ECE 445 Parts Inventory.”
https://docs.google.com/spreadsheets/d/1pqk_fN6fMIju8Ngkg-_4KTQ6uj0hrlnuv49PCC0ijBk/edit#gid=1992434365. Accessed 27 Sept. 2023.

2. "IEEE code of ethics," 2016. [Online]. Available:
http://www.ieee.org/about/corporate/governance/p7-8.html. Accessed: Oct. 3, 2016.

3. Team, The Arduino. “Nano 33 BLE.” Arduino Documentation Datasheet, docs.arduino.cc/hardware/nano-33-ble. Accessed 27 Sept. 2023 4.

4. Tobo, and Sara Santos. “ESP32 Pinout Reference: Which GPIO Pins Should You Use?” Random Nerd Tutorials, 8 Nov. 2022, randomnerdtutorials.com/esp32-pinout-referencegpios/.