Low Cost Sleep Tracking with Arduino

Rachel Bai

Purpose

The purpose of this project was to create a system that tracks the quality of sleep to decide on an appropriate wake-up time and act as advance notice for various sleep disorders.

Question

Can the quality of sleep and various sleep disorders be tracked using wrist actigraphy?

Background Research

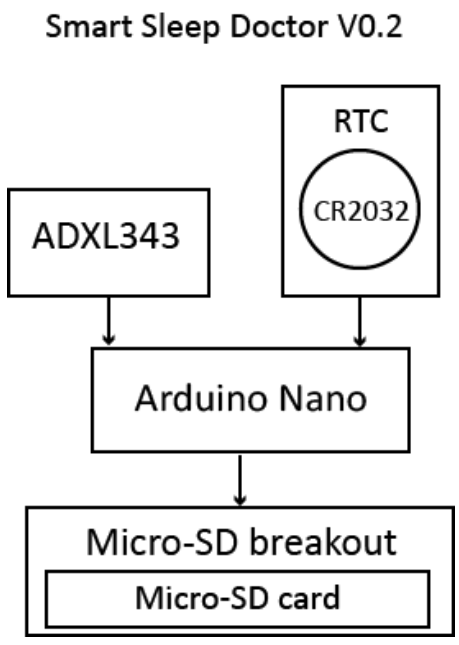
The standard model of sleep,

Hypothesis

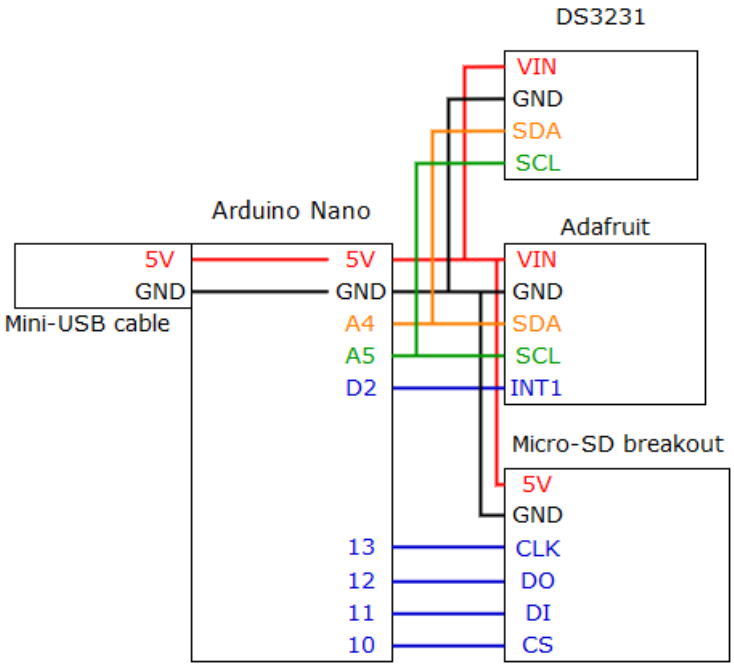
Main Hypothesis: Tracking the quality and health of sleep, as well as determining the optimal time for awakening, can be accomplished using wrist actigraphy.

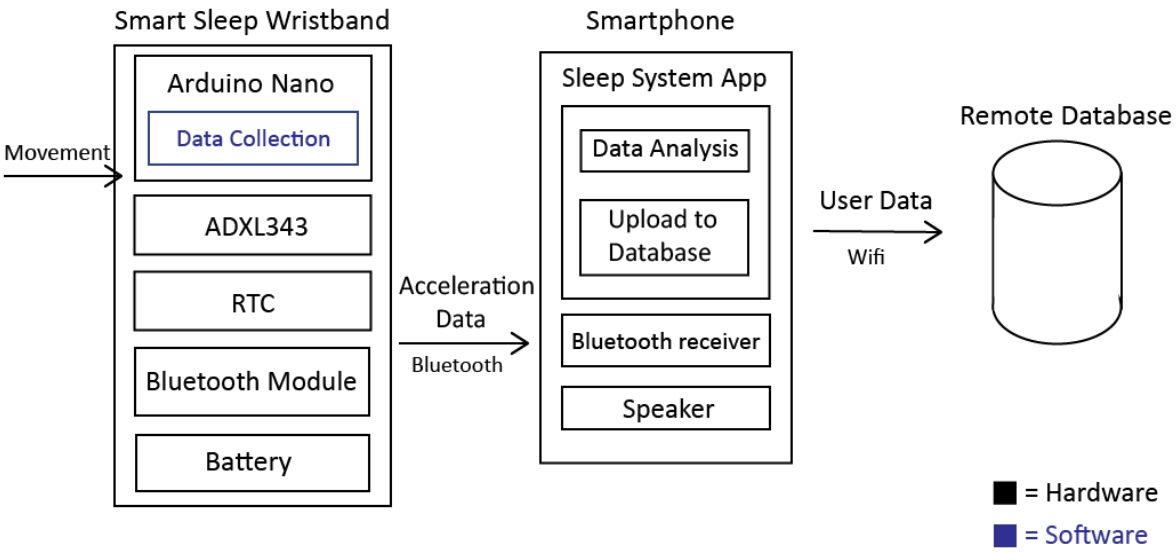
Sub-Hypothesis: Using wrist actigraphy, movement during sleep can be detected and tracked.

Block Diagram



Schematic Diagram





BOM

Name	Mass	Size	Product number	Price	Power Consumption
Arduino Nano	7g	18 x 45 mm	A000005	$22	19 mA
Triple Axis Accelerometer breakout	1.7g	25.5 x 17.8 x 2.9 mm	ADXL343	$14.95	
Micro SD Storage Board	9.1g	25.4 x 20.32 x 2.54 mm	ADA254	$11.89	link
Micro SD card	9.1g	1 x 15 x 10.9 mm	SDSQUAR-016G-GN6MA	$5.45	link
Jumper Wires	136g	99 x 17.8 x 99 mm	B07GD2BWPY	$5.79	link
RTC module	4.5g	43 x 22 x 15 mm	B01IXXACD0	$6.50	link
Coin cell battery	6.8g	30 mm diameter x 3.2 mm	B0002RID4G	$4.74	link

Total Cost: $71.32

Accelerometer Comparison

We picked the ADXL343 for its range of sensitivity, low power consumption, and the built-in libraries and documentation that came with it. We also tested the MMA8451, but found that the ADXL343 had more useful features with a lower price tag.

Features	**ADXL343**	**ADXL335**	**MMA8451**
URL	**https://www.adafruit.com/product/4097**	**https://www.adafruit.com/product/163**	**https://www.adafruit.com/product/2019**
Dimensions	25.5mm x 17.8mm x 2.9mm	19mm x 19mm x 3.14mm	21mm x 18mm x 2mm
Sensitivity range	2-16g	3g	2-8g
Digital/Analog	digital	analog	digital
# of pins	9	7	8
Voltage	2.0-3.6 V	5V	3V or 5V
Current	23 µA	350 µA	6 µA to 165 µA
Weight	1.7g	1.27g	1.3g
Size of IC	3 mm × 5 mm × 1 mm	4 mm × 4 mm × 1.45 mm	3 mm by 3 mm by 1 mm
Additional soldering	headers	headers	headers
Interface	I2C or SPI	Analog	I2C
Integration Effort	Arduino with library	calibration and analog conversion	Arduino with library
Additional Software	library		library
Tutorial/Example	yes	calibration	yes
Interrupt pins	2	2	2
Cost	$5.95	$14.95	$7.95

Software flowchart

Software design and implementation

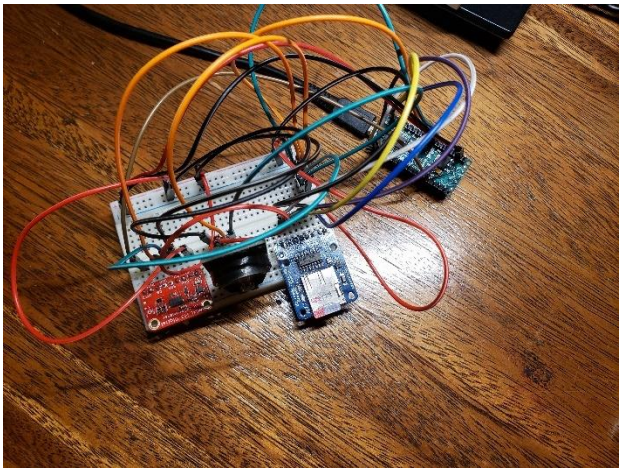
Final Implementation

We decided to use the Arduino Nano with the Arduino IDE for low-cost and fast prototyping. The Adafruit ADXL343 breakout board we used also came with a built-in activity interrupt, so that we were able to decrease the power consumption of the final build.

We connected the Arduino Nano, SD module, RTC, and ADXL343 based on the Schematic Diagram using a breadboard and jumper wires.

We collected data from the ADXL343 in the form of activity interrupts and printed them to the micro-SD card, along with a timestamp for each.

```
//print the acceleration data from the moment activity detected
    data.print(event.acceleration.x); data.print(" ");
    data.print(event.acceleration.y); data.print(" ");
    data.println(event.acceleration.z);
//print the number of times activity detected
    Serial.println("Activity detected!");
    Serial.print("\tActivity Count: "); Serial.println(g_int_stats.activity, DEC);
```



- Research on the standard model of sleep
- Research report placed on website
- Finished BOM and parts collected
- Acceleration on each axis printed out and stored on SD card
- RTC module added for timestamps
- Activity interrupt on ADXL343 used to notify Arduino Nano

I first did research on the standard model of sleep and collected sources about sleep and the different stages of sleep. To organize this information, I wrote a research report and put it in my website. After deciding on a schedule and two accelerometers, I added all the parts needed into the BOM and bought them online. After soldering the headers and testing the accelerometers and SD module individually, I put them together and combined the code so that the acceleration from each axis was printed in the serial monitor and micro-sd card with a timestamp. For a more efficient way to track activity, I decided to use the activity interrupt on the ADXL343 to notify the Arduino Nano. At first, the interrupt never triggered. After debugging with the example sketches and setting registers based on the datasheet, the interrupt started triggering. I adjusted the activity threshold and integrated the other components into the final sketch.

Testing

We first tested the ADXL343 alone by outputting the data to the serial monitor and checking if it was calibrated properly.

Evaluation

Gantt Chart

Timeline	6/4	6/8	6/11	6/15	6/16	6/18	6/22	6/25	6/29	7/2
Milestones			Website v1.0		Block Diagram v1.0		Code for prototype	Prototype v0.1		Prototype v0.2
			Research Paper v0.5		Schematic v1.0		All parts bought			
					BOM v1.0					
1. Learn:										
Research sleep literature										
Research existing technologies										
Develop project website										
2. Design:										
Use case										
Block Diagram										
BOM (parts list)										
Circuit Schematic										
3. Build:										
Buy parts										
Prototype with accelerometer										
Prototype with SD and RTC										
4. Test:										
Test for activity sensing										
Evaluate										
5. Document:										
website										
Research report										

Results

Upward and downward motion on z-axis

X- axis	Y-axis	Z-axis	
-0.24	0.51	9.38	
-1.45	0.2	11.04	
-0.43	-2.31	4.28	Activity detected
-0.2	0.24	9.34	

Rotation from z-axis to x-axis

X-axis	Y-axis	Z-axis	
-0.24	0.12	9.38	
0.9	-0.16	10.63	
5.96	0	6.47	Activity detected
9.07	0.27	2.71	
9.85	-0.08	0.9	

Lines of Code	130
Frameworks	1.8.9
Libraries	Wire, RTClib, SD, Adafruit_Sensor, Adafruit_ADXL343
Calibration	ADXL343 activity threshold set to 20

The activity tracking feature on the ADXL343 detects activity when the change in acceleration on any axis surpasses a certain threshold. Using a threshold is most effective in measuring larger movement, and smaller movements are often missed by the sensor. However, movements during sleep are usually small. This problem was fixed by changing the sensitivity. Because the sensor is mounted on the wrist, the two main types of movement are linear motion and rotation. The ADXL343 is able to accurately detect these types of movement. Using this system, movement during sleep can be tracked accurately.

I learned how to solder through-hole pins and the process of checking each pin to make sure the connection is reliable and there is no connection between VIN and GND, which would cause a short circuit.

We had trouble using the activity interrupt on the ADXL343, where it seemed like the Arduino was not accepting any input. When checking if the interrupt was firing by measuring the voltage of the interrupt line, there was a voltage difference, which meant the interrupt was firing, but the Arduino was not detecting it. I decided to look at the ADXL343 library and datasheet, and I saw that the library set up the single tap registers in the begin function, so we set the activity inactivity control register to 1. However, this also didn't work. The registers had to be set in binary, so a setting of 1 did not turn everything on, it turned the last setting on. After changing the number, the activity interrupt finally started working.

In the future, we plan on integrating this part of the project into the final design by compressing the build, testing during full nights of sleep, and advancing the movement detection algorithm.

Bibliography

Full Code