Team A5 - *Hit it!* - Use Case

Hit It! is a drum-based rhythm game that aims to serve as a middle-ground between other rhythm gaming alternatives.

The Rhythm gaming market:

- Few rhythm games involve hardware
- Limited demographic appeal

Hit it! solves these issues:

- Small and portable hardware
- User inputted songs

ECE Areas:

- Hardware Systems
- Software Systems
- Signals and Systems

Use Case Requirements

Portable

Should be small enough to fit within an average backpack (<15 Liters)

Plays User's Music

Hit It! should be capable of generating beatmaps from user provided songs

Accurate Beatmapping

Generated beatmaps should align well with (>80% accurate) external beatmap software or our hand-calculated beatmaps

Latency

Latency between player input and game reaction should be < 70 ms

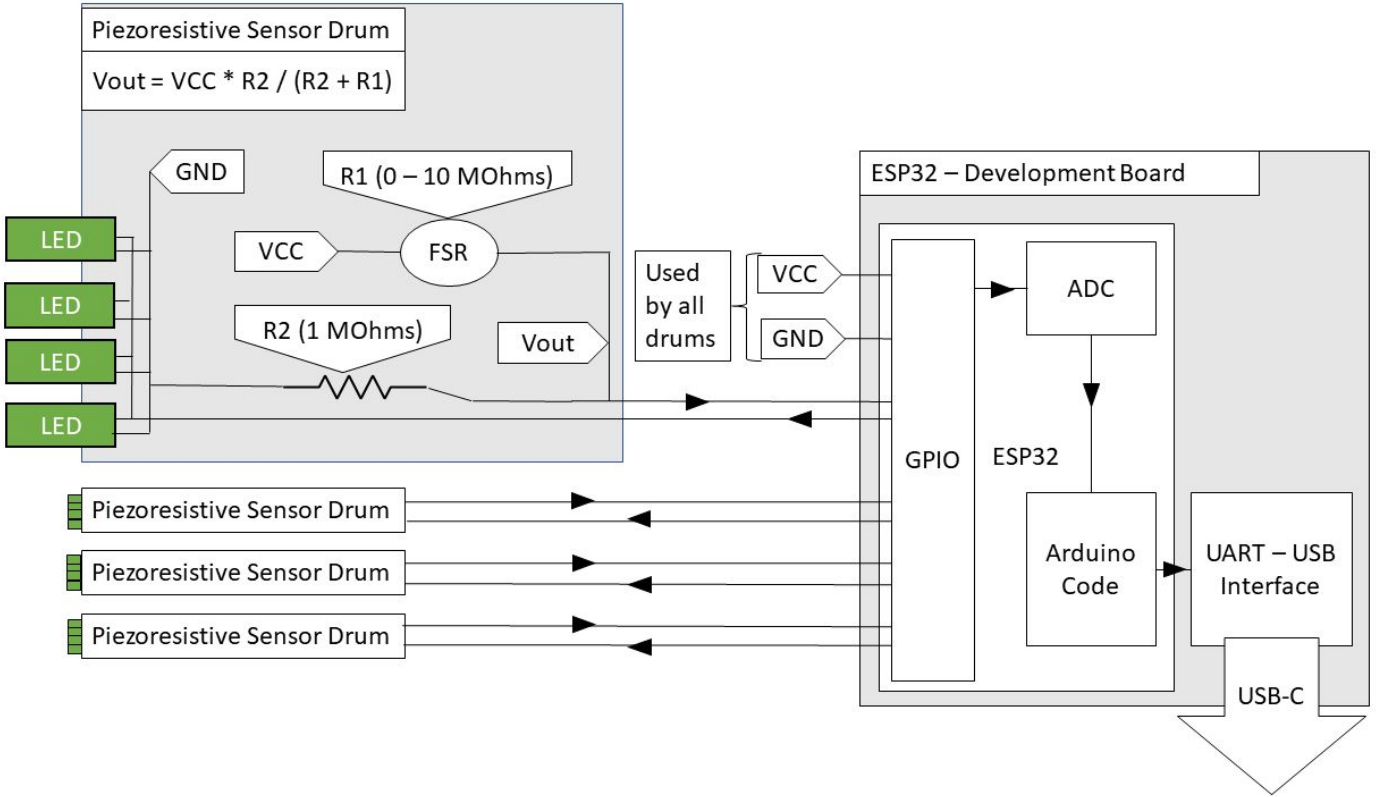
Ease of Use

Setup should be quick and easy (<1 min)

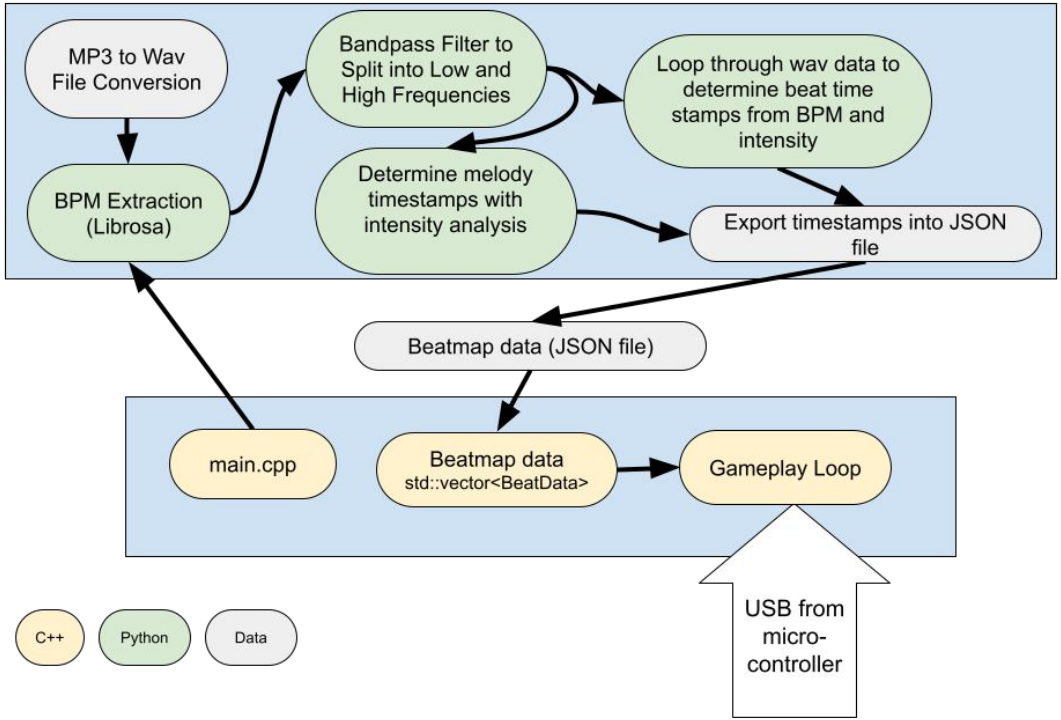
Input Recognition

The Drums used for player input should recognize the overwhelming majority (>99%) of hits by the player

Solution Approach - Hardware



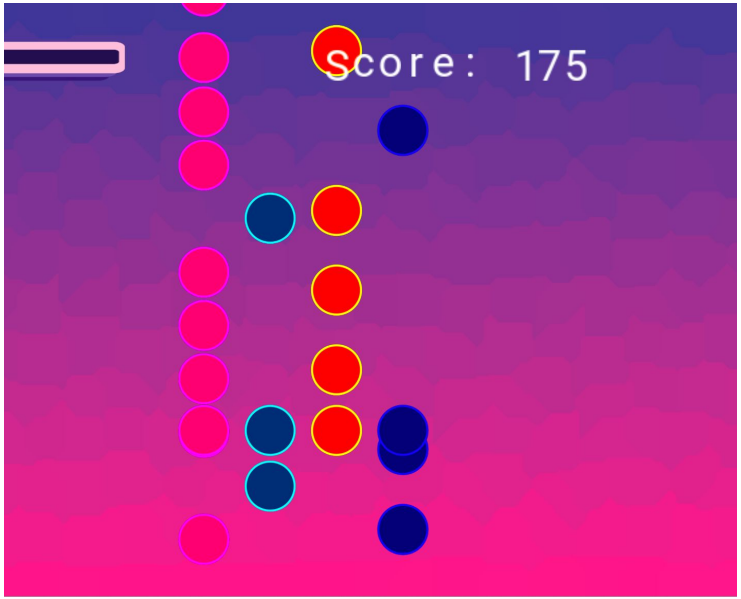
Solution Approach - Software



Complete Solution (Final Demo)

Playtesting - Qualitative

- Conducted 2 playtesting sessions during the week of April 18
  - Played game with mallets or hands
  - Connected wires to drum module
- Gained lots of qualitative feedback
  - Need more visual feedback of drum hits
  - Bug with double drum hits
  - Music not perfectly synchronized with beatmap
  - Sometimes there are erroneous notes
  - Game is too hard



Playtesting - Quantitative

Ease of Setup

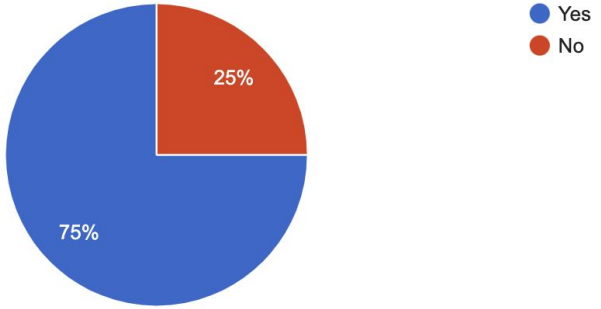
Results (seconds):

1:59	26.50	36.08	32.00	44.92
25.20	38.30	38.40	37.40	27.94

Avg: 42.57 seconds

Note: The first result was likely due to the subject being overly cautious as to not damage the drums, as well as the connectors being new/stiff.

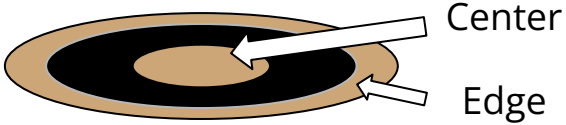
At any point during gameplay, did you ever feel like you hit the drum but the game did not react properly?



Hardware Testing

Drum Recognition Accuracy

Type of Hit	D1	D2	D3	D4
Manual Hit - Centered	250/250	250/250	250/250	250/250
Gravity - Centered (3 inches)	250/250	250/250	250/250	250/250
Manual Hit - Edge	242/250	235/250	236/250	247/250

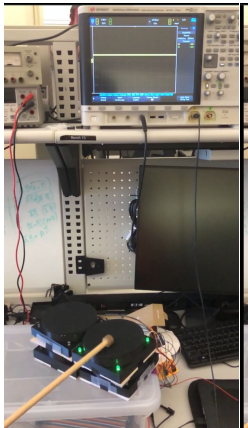


Hardware Latency

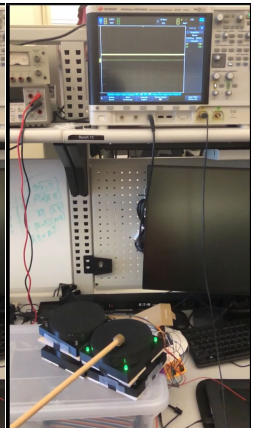
Start (s)	0.85	0.75	1.03	0.86
End(s)	0.89	0.79	1.06	0.90

Avg Difference (ms): 37.5 +- 5

0.85 s



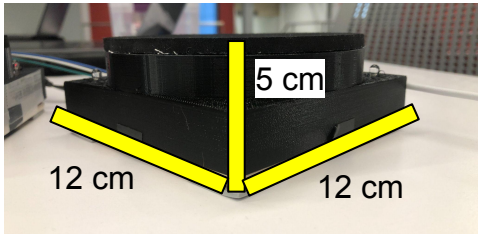
0.89 s



Compactness

Total Volume (5 modules):

= (5 * 720 cm^3) = 3.6 liters



Hardware Testing Goals and Results

Area	Goal	Result	Possible Improvements
Drum Recognition Accuracy	>99% Recognition Rate	100% for centered hits 96% for hits on edge	Increasing sensor sensitivity and further decreasing “hit” threshold to pick up more hits along the edge.
Latency	< 37 ms	~37.5 ms +- 5 ms	Rework Arduino code to not have constant delays and have less logic.
Compactness	< 15,000 cm^3	3,600 cm^3 or 3.6 Liters	No improvements necessary
Ease of Setup	< 1 minute Setup Time	Average of 42.57 seconds	Add descriptive markings on the drums and connectors to make setup more intuitive

Beat Mapping Testing and Verification

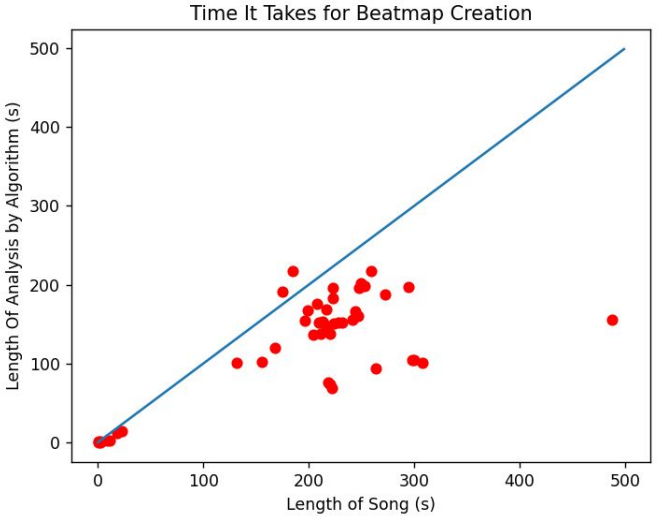
Length of analysis equal to length of audio file

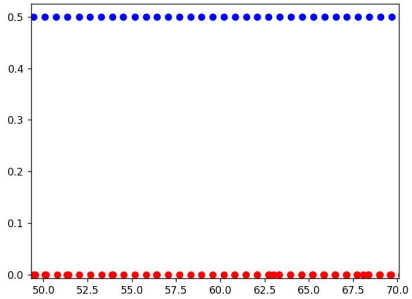
- Graph with fifty samples to the right
- Blue line represents acceptable time to complete

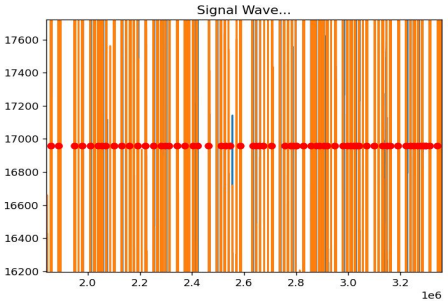
Weakness: Audio files often same length

Beat Testing Accuracy

- No false negatives found
- Librosa's algorithm detects about 400 beats for its tracker and ours detects about 600 suggesting about a 67% correctness







Trade offs

- Had to leave out extra aesthetics to leave time to playtest game
- Melody tracking less advanced due to time constraints
- Had to remove cross platform capability due to time constraints
- 3D printing
  - 3D printing modules - $36 per module
  - PVC piping and wood - ~$10 per module
- Drum Recognition Sensitivity
  - No false positives for rebounding hits vs. Higher accuracy on hit.

Schedule & Division of Labor