MUSIC

IN

MOTION

Eric Hoofnagle Lane Starratt Zoilo Boehme

Music in Motion (MIM) Mission Statement

Music in motion is a concept for a running device which allows you to co-create music while you workout and provide the feedback that will track your progress and help you perform. Essentially, it is a concatenation of a fitness monitor, biofeedback device, and a music player. This device will be practical—various metrics will be sampled, some of which are biological, in order to provide a much more complete description of the user's fitness session than anything that exists on the market today. This device will motivate the user—the collected data is interpreted by a number of algorithms which will quantify the user's session, track progress over time, and congratulate the user for breaking previous records in real time. This device will be fun—simply by running, it will let the user truly listen to his or her body for the first time and it will do so in the form of music. This supplementary feature adds an element of novelty to the device which is both interactive and entertaining. Thus, the biofeedback is presented via two entirely distinct channels: biological data and music.

This concept came about for several reasons. Primarily, the three of us gravitated towards biofeedback and music. In spite of the chances, we created a concept in which we could incorporate them together. It evolved from a brainstorming exercise whereby a proposal for a running devise was agreed upon. The barrage of interesting features and functionality followed, namely biofeedback and music. The underlying pattern within the project thus far has been this: to quote Steve Jobs "We don't want to simply do it better, we want to do it differently."

The objective is to provide detailed and useful feedback to the user in an entertaining manner. This device will use various metrics, such as heart rate, running distance, running speed, air temperature, and Galvanic Skin Response. Galvanic Skin Response is a commonly used element in biofeedback whereby the subject's skin conductivity is measured. The sympathetic nervous system responds to physiological arousal, in part, by changing the conductivity of the skin—sweating. This data will be imperative for calculating milestones to keep the user both engaged and motivated. We will incorporate the following milestones: top speed, farthest distance, step-counter, and best race (mile, 5k, 10k, half-marathon, etc.). In addition, the device will analyze the runner's cardiovascular response to the fitness session so as to quantify their current physiological condition and progress. This will be executed using the heart rate, distance, speed, and galvanic skin response data which will be processed by an algorithm of our design. The device will respond to the biometrics by triggering one of many possible audio samples which congratulate the user when a previous record is broken.

'Music in Motion' relies heavily on biological signals to present the user with feedback through a variety of different forms. For the sake of being effective, we will use a variety of facets with which to obtain biological data, instead of relying on a couple rudimentary metrics. This gives our device an enormous advantage in representing a more complete model of the user's fitness and progress in both breadth and depth. Galvanic Skin Response coupled with heart rate, speed, distance, and existing scientific literature on fitness will prove to be great resources with which to design our algorithms. Ultimately, the main feature for the practical side to the biofeedback will be providing specific analysis of the user's cardiovascular response and mapping progress throughout their health journey.

The other dimension to the biofeedback will be purely for entertainment and novelty. Using a combination of signal processing and musicianship, we will turn the user's biometric data into music. Essentially, we will use several matrices of audio samples which will be triggered by MIDI protocol and controlled via the microprocessor. The concept is as followed: each column in the matrix corresponds to a type of musical phrasing, such as bass, percussion, etc. Each row in the matrix corresponds to different melodies such that there are 10 possible bass melodies, 10 possible percussion rhythms, and 10 lead melodies for example. The music will be implemented such that every audio sample is compatible with every other audio sample in any category; therefore any song generated is musically coherent. These samples will be triggered by MIDI, which quantizes the triggers. The tempo of the song will be directly controlled by the speed of the runner. This will be achieved by detecting the impulses received by the accelerometer, filtering the data, processing it, and continuously feeding back the frequency of the signal to the microprocessor. We will design a linear control system to perform this subsystem. The complexity of the music (the number of accompaniments) will be determined by the characteristics of the run such as distance and speed. The appearance of this system will be that the music coming from the user's headphones will be completely synchronized with and controlled by their running. This will be manifest in the melody, rhythm, and tempo of the song. Moreover, because of the chosen matrix algorithm, there will be such a huge number of possible combinations of songs that each session for each user will be unique.

This device will be used in conjunction with a band that straps around the runners arm. It is apparent that with many different sensors, there is a tradeoff between accuracy and comfort for the user—nobody wants to run with electrodes everywhere. We have a solution. The accelerometers will be integrated within the device because the pulse can be detected from any part of the body. The GSR sensor will be adapted onto the surface of the device which will be strapped to the runners arm. The heart rate monitor will be integrated with existing headphones as an attachment or simply built onto the surface next to the GSR sensor. Achieving these goals will also help facilitate the device's small size. We intend to create a compact package with minimal wires, weight, and dimensions. Also, the product will be made as simple as possible to use, meaning very few buttons and no display on the device itself. Simplicity is the key for a product in which the user needs not to be distracted by options—it must seamlessly integrate with the runner.

The sophistication and elegance of the user interface will be in the Android application. When the user arrives home from their run, they can directly download their session from the device and review it on this application. This application will represent the data graphically and numerically. The data needs to be interpreted for the user and be as simple as possible while still presenting all of the crucial information. The trends representing all sessions will quantify the user's progress in an aesthetic manner. This will encourage the runner to use the product more often and thus provide incentive to run frequently in order to measure their progress.

Goals and Objectives

Objective:

To design, prototype, test, and build a running meter (the MIM system) that synchronizes the rhythm of the runners stride to a soundtrack tempo.

Features:

- Matches beat of music to pace of runners stride
- Monitors runners data and increases complexity of music and run becomes more intense
- Records personal bests and plays musical alert when runner reaches a new milestone
- Double and Half time modes
    - In the case of rapid acceleration or deceleration the music will play in double time or half time
- Small in size, lightweight, durable, and water resistant.
- Micro USB
    - Data transfer and charging from computer
    - Data transfer only from Android application

Design Goals:

- Design Stages Outline
  - SENIOR DESIGN 1
    - STAGE 1 – Identify necessary components. Preliminary budgets and design specifications.
    - STAGE 2 – Prototype design. Budget/Parts revision.
    - STAGE 3 – Initial product design. Prototype re-design. Finalize parts list.
  - SENIOR DESIGN 2
    - STAGE 4 – Prototype build and test. Final product design based upon prototype testing.
    - STAGE 5 – Build.
    - STAGE 6 – Test, Re-design, Test, Re-design.
- Sensors
  - STAGE 1 – Initial design of a system of 2-4 sensors with appropriate filter systems that feeds footfall impact timing data and runners biometric data to the processor in real time. Determine the power requirements of the sensor array. Preliminary parts list.
  - STAGE 2 – Prototype design sensor array with filters for implementation on a breadboard. Finalization of output to converters specification. First sensor array with filter design and parts list revision.
  - STAGE 3 – Final prototype design and parts list. Second array and filter design revision.
  - STAGE 4 – Prototype and test sensor array with A/D conversion on breadboard. Finalize design specifications.
  - STAGE 5 – Implement sensor array and filters on PCB.
  - STAGE 6 – Test final product.
- Processor
  - STAGE 1 – Identify processor type speed and language. Determine memory size and volatility and if external memory will be required. Determine power requirements
  - STAGE 2 – Design running data storage system. Determine A/D and D/A requirements, formats, and transmission protocols.
  - STAGE 3 – External Interface design. A/D D/A conversion design. Initial data I/O programming.
  - STAGE 4 – A/D D/A conversion on breadboard. Data I/O between sensor array and Audio Engine.
  - STAGE 5 – Implement processor and memory on PCB.
  - STAGE 6 – Test final product.
- Audio Engine/Control System
  - STAGE 1 – Identify memory storage needs for MIDI soundtrack data, Audio sample data, and control program data. Initial audio amp and output design. Initial tempo and soundtrack control system designs. Initial parts list.
  - STAGE 2 – Soundtrack memory and audio storage designs. MIDI implementation.
  - STAGE 3 – Final tempo and soundtrack control system design. Final audio amplifier design and parts list.
  - STAGE 4 – Prototype and test audio engine and control system on breadboard. Finalize design.
  - STAGE 5 – Implement design on PCB
  - STAGE 6 – Test final product.
- Additional Software Run Variables (Android/PC)
  - STAGE 1 – Designing functionality and finalizing platform language.

- STAGE 2 – Structuring application code for PC and Android. Testing and Debugging. Determining memory hierarchy. Determining I/O protocols and connectivity.
- STAGE 3 – Finalize design of software and run software simulations.
- STAGE 4 – Prototype testing alongside with prototypes of sensor array and audio control.
- STAGE 5 – Implementing final design
- STAGE 6 – Testing final product and streamlining interface for easy user control.

Specifications and Requirements

Size:

Height	**Width**	**Depth**	**Weight**
5 in.	2.5 in.	0.5 in.	5 oz.

The printed circuit board will encompass all of the size while surrounded by a sturdy plastic shell. Here are all the specs that will fit onto the PCB:

- Accelerometer
- GSR
- VHR
- Ambient temperature sensor
- Microprocessor
- Memory storage
- USB Port
- Filters, A/D converter, and D/A converter

Uses accelerometer to determine when the impact of the footfalls occurs. The control section uses runner's previous velocity data to predict when the next footfall happens and places the musical beat at the time of the prediction then makes adjustments based on the difference between the actual footfall and predicted one.

Uses Galvanic Skin Response (GSR), Variable Heart Rate (VHR), and other sensors to determine the intensity of the run and control soundtrack playback accordingly.

Memory matrix of MIDI song data and audio sample files.

1 Measure MIDI Data	Loops per instrument	Instruments	Total Measures	MIDI soundtrack storage
0.5 kB – 1 kB	25	8	200	100 kB – 200 kB

Audio Sound Bank:

1 Audio Sample	Total Samples	Total Audio Memory Storage
100 kB	≈ 63	6.3 MB

Total Audio Engine Memory Storage: **6.5 MB**

Running data is stored in the memory with variables. These variables influence the beat, flow, and intensity of the music playback.

- Runtime
  - Longest recorded time
  - Average time of runs
  - Shortest recorded time
- Heart Rate (Beats Per Minute), GSR, Temperature, and Tempo
  - Highest
  - Average
  - Lowest

Software Specs (Additional Run Variables):

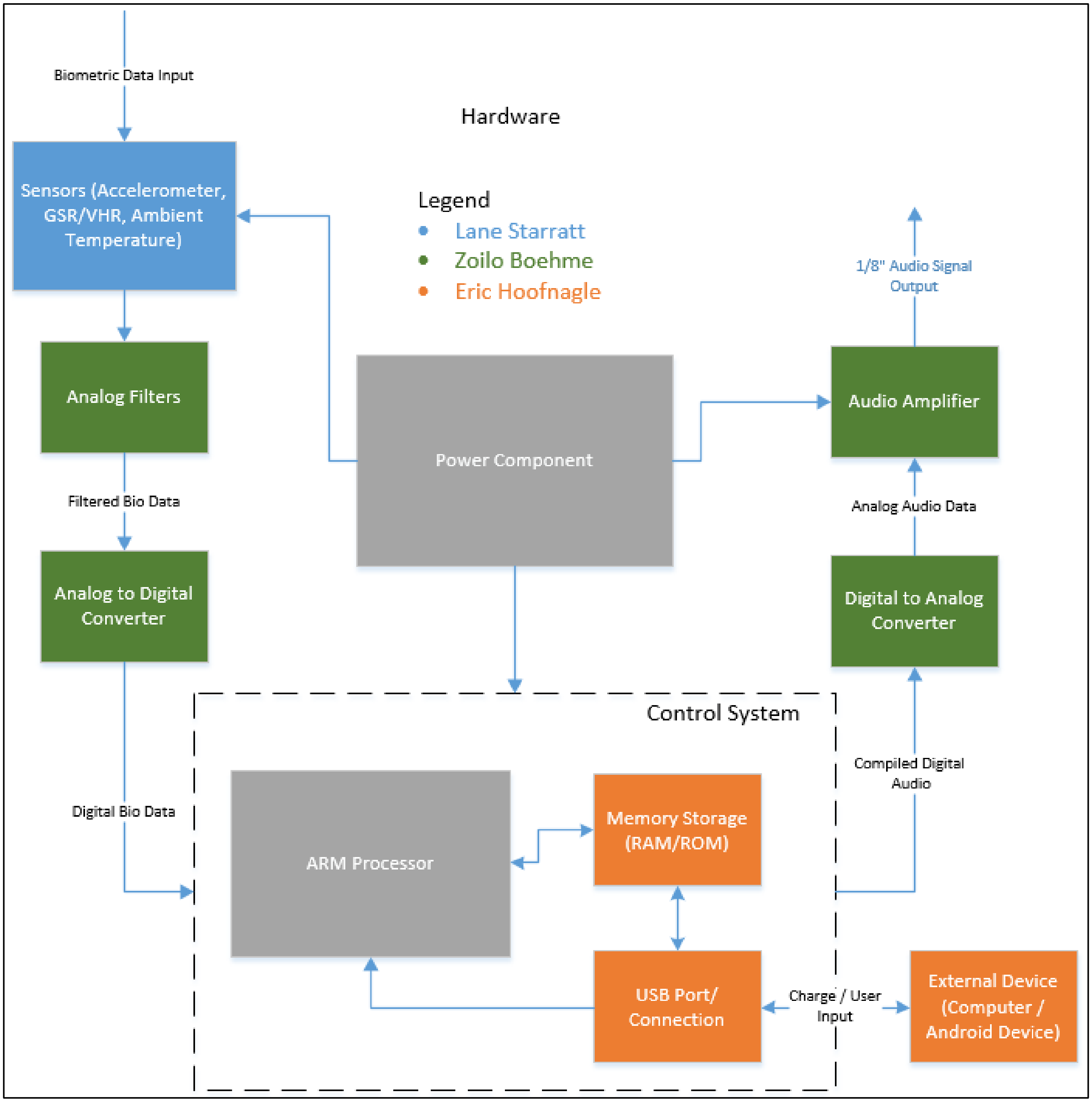
From the android application/computer program, the user stores extra variables to contribute to the music tempo, intensity and length of music. These variables include:

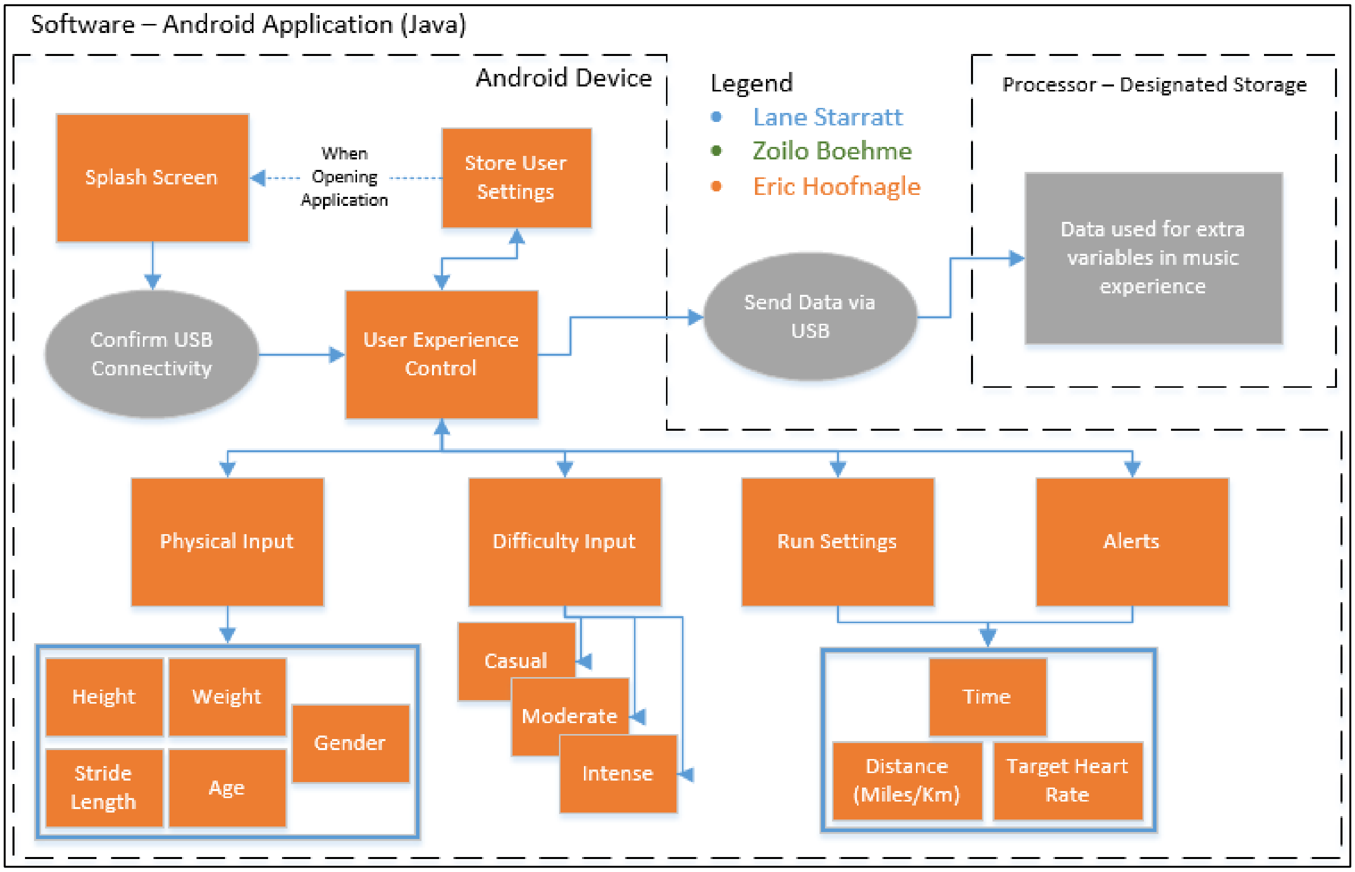
- Physical characteristics
  - Height
  - Weight
  - Age
  - Gender
  - Stride Length
- Difficulty
  - Casual
  - Moderate
  - Intense
- Type of run
  - Time
  - Distance
  - Heart Rate
- Alerts – When the runner should receive notifications (e.g. each mile time or certain heart rate)
  - Time
  - Distance
  - Heart Rate

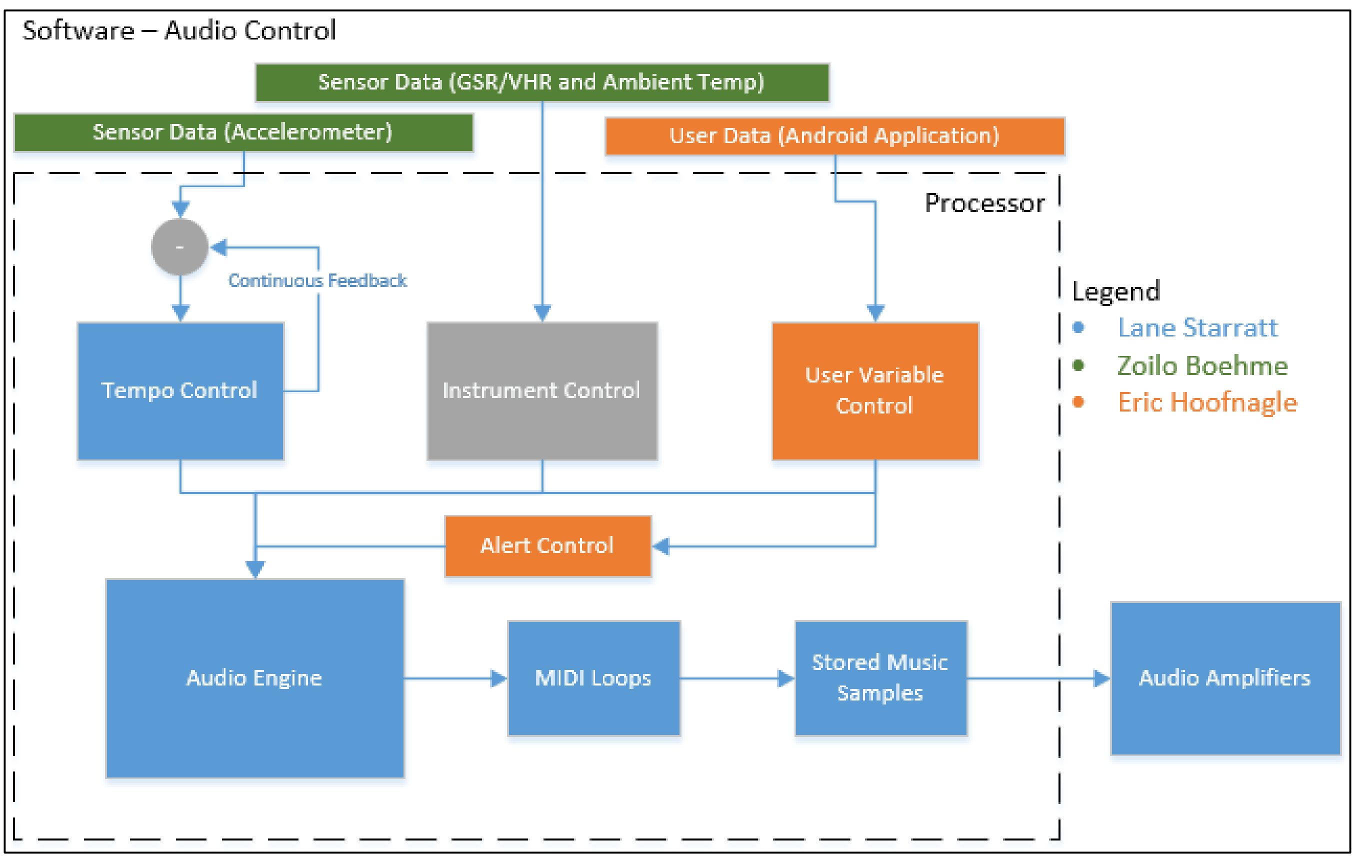
The necessary Android API is 2.2 or higher. There only needs to be roughly 5-10 MB of free space in order to download and use the app. No additional memory is added to the phone from the app because the user can only save up to 3 separate run settings.

If the user downloads the application on their computer, they can detect MIM through USB so they can transfer data from the application.

Block Diagrams




Software – Android Application (Java)
Android Device
Legend
Lane Starratt
Zoilo Boehme
Eric Hoofnagle
Processor – Designated Storage
Splash Screen
When Opening Application
Store User Settings
Confirm USB Connectivity
User Experience Control
Send Data via USB
Data used for extra variables in music experience
Physical Input
Difficulty Input
Run Settings
Alerts
Height
Weight
Gender
Stride Length
Age
Casual
Moderate
Intense
Time
Distance (Miles/Km)
Target Heart Rate


Software – Audio Control
Sensor Data (GSR/VHR and Ambient Temp)
Sensor Data (Accelerometer)
User Data (Android Application)
Processor
-
Continuous Feedback
Tempo Control
Instrument Control
User Variable Control
Legend
Lane Starratt
Zoilo Boehme
Eric Hoofnagle
Alert Control
Audio Engine
MIDI Loops
Stored Music Samples
Audio Amplifiers

Music In Motion Budget Spreadsheet

Part Descriptions	# of units Min Cost	# of units Max Cost	unit subtotal Min Units	unit subtotal Max Units	Total Min Subtotal	Total Max Subtotal
Embedded Processor	$ 12	$ 20	2	4	$ 24	$ 80
Accelerometer	$ 10	$ 15	2	4	$ 20	$ 60
GSR Sensor	$ 5	$ 10	2	4	$ 10	$ 40
VHR Sensor	$ 5	$ 20	2	4	$ 10	$ 80
Temperature Sensor	$ 10	$ 15	2	4	$ 20	$ 60
PCB	$ 50	$ 100	2	2	$ 100	$ 200
Power System	$ 75	$ 125	1	2	$ 75	$ 250
Audio Amplifiers	$ 10	$ 20	2	3	$ 20	$ 60
DA AD converters	$ 10	$ 20	4	8	$ 40	$ 160
Miscellaneous - Tools, Connectors, Prototype parts, Installation, Spare Parts	-	-	-	-	$ 175	$ 250
TOTAL	-	-	-	-	$ 494	$ 1,240

Music In Motion Milestone Chart

<table>
<tr><th colspan="16">Weeks</th></tr>
<tr><th rowspan="2">Semesters</th><th>Senior Design I</th><td>27-Jan</td><td>3-Feb</td><td>10-Feb</td><td>17-Feb</td><td>24-Feb</td><td>3-Mar</td><td>10-Mar</td><td>17-Mar</td><td>24-Mar</td><td>31-Mar</td><td>7-Apr</td><td>14-Apr</td><td>21-Apr</td><td>28-Apr</td></tr>
<tr><th></th><td colspan="4"><u>Stage I</u>
Initial Documentation/Project Brainstorming
Preliminary Designs / Initial Parts List
Sponsorship
Initial Budget</td><td colspan="6"><u>Stage II</u>
Prototype Designs
Budget and Parts List Revisions
Software Initialization</td><td colspan="4"><u>Stage III</u>
Gather Initial Parts
Finalizing Project Document
Initial Prototype Build</td></tr>
<tr><th></th><th>Senior Design II</th><td>12-May</td><td>17-May</td><td>22-May</td><td>27-May</td><td>1-Jun</td><td>6-Jun</td><td>11-Jun</td><td>16-Jun</td><td>21-Jun</td><td>26-Jun</td><td>1-Jul</td><td>6-Jul</td><td>11-Jul</td><td>16-Jul</td></tr>
<tr><th></th><th></th><td colspan="4"><u>Stage IV</u>
Complete Software Requirements
Finish Prototype Build
Final Design based on Prototype Testing
Compile Notes/Information for Presenting</td><td colspan="5"><u>Stage V</u>
Construct Device from Final Design
Synchronize Software/Hardware
Test Synchronization</td><td colspan="5"><u>Stage VI</u>
Final Testing of Real-World Results
Compile Documentation for Presentation/Demonstration</td></tr>
</table>