

Light Suit

Eric Peach
December 2017

Project Status

✓ Project Complete!

Objective

You want to have a suit you can wear that will glow in the dark and look cool. You will create a vest that is rigged with lights so that they can flash and pulse with the music.

- The lights have multiple modes:
 - On
 - Off
 - Fade
 - Beat with Music

Timeline

The design is being developed on a schedule so that it is ready in time for a particular December event.

Date (2017)	Event	
27 Nov	Design Started	✓
28 Nov	Measurements Made	✓
29 Nov	Circuit Designed	✓
29 Nov	Electrical Needs Finalized	✓
30 Nov	Electrical Design Complete/Parts Ordered	✓
1 Dec	Software Design Complete	✓
3 Dec	Software Written	⚠ Al mo st
5 Dec	Suit Jacket Purchased	✓
7 Dec	Test Circuit Begins Assembly	✓
9 Dec	Test Circuit Complete	✓
10 Dec	Shirt Assembled	✓
13 Dec	Shirt Tested	✓
15 Dec	Day of First Event	✓

Table of Contents

- Objective
- Timeline
- Initial Brainstorming
 - Choice of Lights
 - Light Placement
- High-Level Design
 - Functional Block Diagram
 - Subsystem Description
 - Interface Description
 - Electrical Layout
 - Preliminary Part List
- Requirements
- Risks
- Components
 - Circuit Diagram
 - Interface Boundaries
 - Detailed Part List
- Implementation Plan
 - Unit Testing
 - Integration Tests
- Assembly Details
 - Shirt Layout
 - Light Installation
 - Circuit Assembly
 - Suggestions
- Software Details
 - High-Level Design
 - Filtering Algorithm
- Costs
- Recommendations and Ideas

Initial Brainstorming

Choice of Lights

There was a debate of whether I should use the row LED lights, or try to cover the suit in individual 2-legged LEDs.

Decision

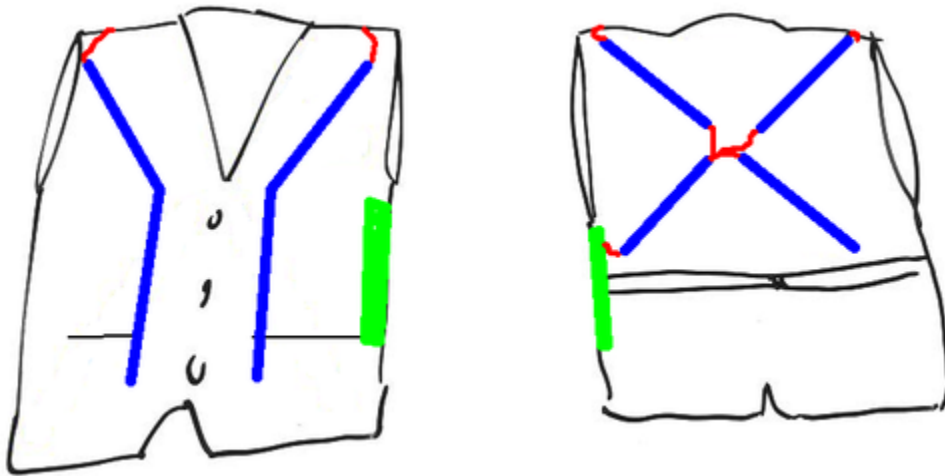
Going to go with the LED Tape design.

It's easier to attach to clothing and less likely to break, as the TRON costume has proven. While the individual LED Design does look better, it would be harder to do, more fragile when completed, and may require a lot of sewing and wiring that could be avoided.

I can cover up the metallic appearance between the lamps by covering them over with selectively chosen construction paper or fabric.

Light Placement

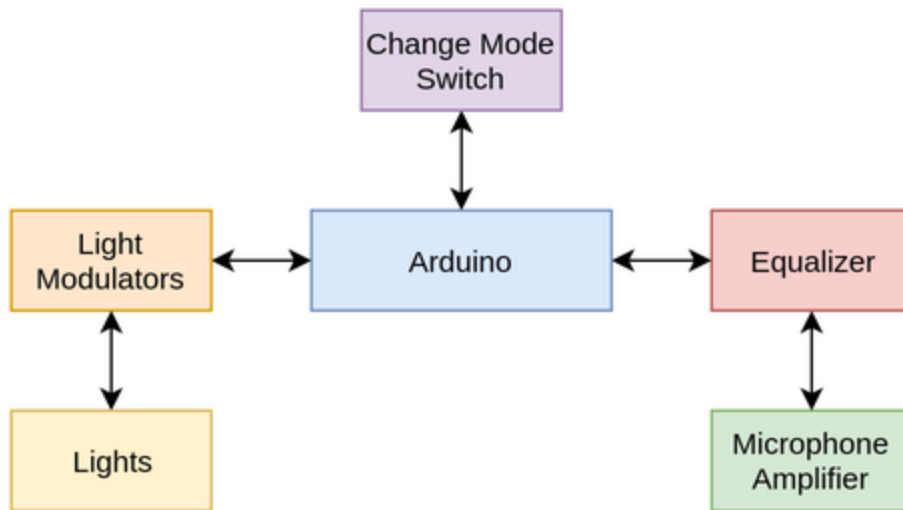
I'll need to stick the control box onto the vest somewhere. Initially I thought under the armpit would be a good spot, but it turns out that the lower back is more comfortable to wear and easier to conceal.



High-Level Design

To get a good idea of the design, the subsystems are thought up, as well as details on how they'll be connected to each other.

Functional Block Diagram



Subsystem Description

Module	Attention
Lights	Powered by 12V. Light is engaged by pulling one of the R,G,B channels to ground. http://a.co/fza6gsL
Modulators	Probably just a FET transistor, which are robust and easy to connect.
Arduino	Powered by 12V Plug. Provides common ground and 5V to rest of system. https://www.adafruit.com/product/72
Change Mode Switch	Just 2 wires, that connects a sense pin to either 5V or Gnd.
Equalizer	Bit of a weird configuration requiring a few capacitors and resistors here and there. Powered by 5V. Remember to put in a ripple cap on the VDD.
Microphone Amplifier	3V and Ground hookup. May require a voltage regulator if Arduino only provides 5V. https://www.adafruit.com/product/1063

Interface Description

Interface	Description																
Amplifier Equalizer	The waveform should be AC coupled, so stick about a 0.1uF capacitor in between the Amp output pin and the input pin.																
Equalizer Arduino	<div>The Arduino signals the equaliser chip through 2 pins and reads in an analogue signal through a third.</div> <table><tr><th>Pin</th><th>Purpose</th><th>Pin on Arduino</th><th>Pin on Equalizer</th></tr><tr><td>OUT</td><td>The filtered DC value on the MUX. 700Ohm Impedance</td><td>Any Analog INPUT</td><td>3</td></tr><tr><td>STROBE</td><td>A pulse delivered from Arduino to chip to change channel.</td><td>Any Digital OUTPUT</td><td>4</td></tr><tr><td>RESET</td><td>A pulse delivered from Arduino to clear settings. Equalizer goes live when RESET is low.</td><td>Any Digtal OUTPUT</td><td>7</td></tr></table>	Pin	Purpose	Pin on Arduino	Pin on Equalizer	OUT	The filtered DC value on the MUX. 700Ohm Impedance	Any Analog INPUT	3	STROBE	A pulse delivered from Arduino to chip to change channel.	Any Digital OUTPUT	4	RESET	A pulse delivered from Arduino to clear settings. Equalizer goes live when RESET is low.	Any Digtal OUTPUT	7
Pin	Purpose	Pin on Arduino	Pin on Equalizer														
OUT	The filtered DC value on the MUX. 700Ohm Impedance	Any Analog INPUT	3														
STROBE	A pulse delivered from Arduino to chip to change channel.	Any Digital OUTPUT	4														
RESET	A pulse delivered from Arduino to clear settings. Equalizer goes live when RESET is low.	Any Digtal OUTPUT	7														
Chg Mode Arduino	<div>When pressed, the switch sets 5V directly to a DIGITAL INPUT pin on the Arduino.</div> <div>When not pressed, the pin is pulled down by a 10k resistor to GND.</div>																

ArduinoModulators	<p>When it has been verified that 5V on the gate of the transistor turns on the light, the brightness of the light can be modulated with PWM.</p> <p>Arduino's ANALOG OUT pins (0 to 5 on the boarduino, IIRC) will connect to the gate of the transistors.</p>
ModulatorsLights	<p>I think you just need to pull each of the channels to GND using the third leg on a transistor. Verify that 5V IN does the job and doesn't exceed any power/current requirements.</p> <p>If adjustment is needed, you can put a resistor between the third leg and the Arduino.</p>

Do not exceed any power or current limitations imposed by the Transistor (IRL2703), the Lights, or the Battery.

Electrical Layout

It looks like there's not gonna be a nice way of wearing the control module once it's all put together. I expect I'll need the following pattern:

- Microphone Amplifier worn on lapel with clip.
 - Combine with the change-mode switch, and maybe an "on" LED.
- Control module placed in custom box and stored in back.
- Run power mains to battery in pocket.


Preliminary Part List

This is a preliminary list for the historical record. A more complete list is further down!

- Accessories
 - Thin Solder with rosin
 - Prototype boards, spacers
 - IC Sockets
- Components
 - 12V, 3-channel lights
 - Capacitors
 - Resistors
 - Status LED
 - Microphone amplifier
 - Equalizer Chip: <https://www.digikey.ca/short/q32hzq>
 - Boarduino or Arduino Mini with 12V supply.
- Wires
 - 2 Channel Wire
 - 4 Channel Wire
- Power
 - New 12 V Battery
- Connectors
 - 4 Channel wire connectors
 - LED Strip connectors
 - Mode Change Switch
 - 12V Barrel Jacks
 - XT60 Battery Connectors

Requirements

	Requirement	Test Method	Verified
1	<p>The light suit should be self-contained; it should live inside the vest where it is implemented.</p> <ul style="list-style-type: none"> • The battery may be stored externally. 	Put it on.	
2	The battery should be able to power the light suit at full power continuously for one hour.	Activate and monitor for 10 minute intervals.	












R1	200k
R2	1k
R3	10k
R4, R5, R6	0  Not needed with these lights.
C1	0.1F
C2	0.1F
C3	0.1F
C4	33pF
LED1	Small Red
T1, T2, T3	IRL2703
SW1	Button Switch

Interface Boundaries

Boundary	Technique
Mic to Main Boards	4 Channel Wire with a 4-Pin MTA-100 connector.
12V Into Arduino	2.1x5.5mm Barrel Plug
Main Boards to Light Array	4 Channel Wire with a 4-Pin MTA-100 connector.
12V into main circuit	2.1x5.5mm Barrel Plug + Jack combo
Battery onto 12V	XT60 + 2-channel wire

Detailed Part List

Note that this list only includes things circuit components and interface boundaries (plugs, jacks, etc). The actual *circuit board*, as well as any solder, jumper wire or accessories are not included on this list but are discussed further down.

Name	Value	Quantity	Already Have Lying Around	Datasheet	DigiKey Part No
Resistor	200k	1			CF14JT200KCT-ND
Resistor	1k	2			
Capacitor	0.1uF	3			BC1160CT-ND
Capacitor	33pF	1			BC1007CT-ND
LED	(Red)	1			
FET	IRL2703	3		IRL2703	IRL2703PBF-ND
Switch	Button Switch	1			
MTA-100	Male	2			A1922-ND
MTA-100	Female	2			A31082-ND
Barrel Plug	Male	2			EP501A-ND
Barrel Jack	Female	1			EJ501A-ND

IC Socket	8 Pin	1	⚠		AE9986-ND
XT60	Male	1	✓		
XT60	Female	1	✓		
3V Voltage Regulator	3.0V Regulator	1	✓	MCP1827S 30EAB	MCP1827S-3002E/AB-ND
Microcontroller	Boarduino	1	✓	Boarduino	1528-1113-ND
Equalizer	MSGEQ7	1	⚠	MSGEQ7	1568-1335-ND
Battery, Rechargeable, 12V	2300 mAh	1	⚠		N703-F025-ND

Implementation Plan

Unit Testing

Item	Procedure	Done
Audio Amplifier	Hook up to Digilent Analog Discovery Module (ADM) and test that you're getting a reasonable signal out of it.	✓
Multiplexer/Equaliser Chip	Wire up to ADM with a function generator and analogue recorder. Add 2 buttons for Reset and Strobe. For each bucket, Reset and Strobe to the desired bucket. Sweep the frequency over the active frequency and ensure that a pulse is recorded in Analog In.	✓
Lights	They better work!	✓
Lights with Transistor	Check the lights first to see the current and power usage, and determine if I need a resistor at all, or if I can just hook up the transistor by itself. I should be able to modulate light brightness by simply modulating a voltage at a single point.	✓

Integration Tests

Item	Procedure	Done
Amplifier Plus Equaliser	Complete circuit. Wire up to ADM with Analog Recorder. Prepare the RESET and STROBE buttons again. Use LMMS to generate a tone, and ensure that it shows up on the scope when I strobe to it. Watch out! Sampling the Analog Output can change the result by 10%, so test with different strobe frequencies.	✓
Light Modulator to Arduino	You should be able to simulate the bright light behaviour using 3 LEDs instead of the complete strips.	✓
Bench Test	The lights should be connected. Verify that the system works. Include the battery and master switch.	✓
Light Installation Test	As lights are installed in the suit, the off-the-shelf controller should be used to verify the various connections on the lights. All colors should work at all points on the vest.	✓
Solder Test	After soldering system together, test again.	✓

Box Test	After mounting system in a box, test again.	✓
Wearing Test	Wear everything and turn it on. It should work.	✓

Assembly Details

Shirt Layout

- Stitched some fluorescent traces onto the vest in a construction-vest kind of layout
- Applied strips and used hot glue on remaining spots.
- The box should be placed in a pocket in the inside of the vest so that it rests against the wearer's small-of-back.

Light Installation

- Items
 - Lights: <http://a.co/3UdRtRP>
 - Connectors <http://a.co/4q1DBG0>
 - Wires: <http://a.co/7DpS3i0>
- Complications:
 - The connectors kinda suck at biting onto the wires when establishing a connection. I tried heating the wires, partially stripping the wires, and using crimping pliers. Fact is, if the little knives in the connector don't want to penetrate the insulation of the wire, they won't.
 - I got it to work just by carefully "chewing" the wire by moving the jaw of the connector up and down until it eventually could be closed without pliers.
 - Sewing is hard to do.

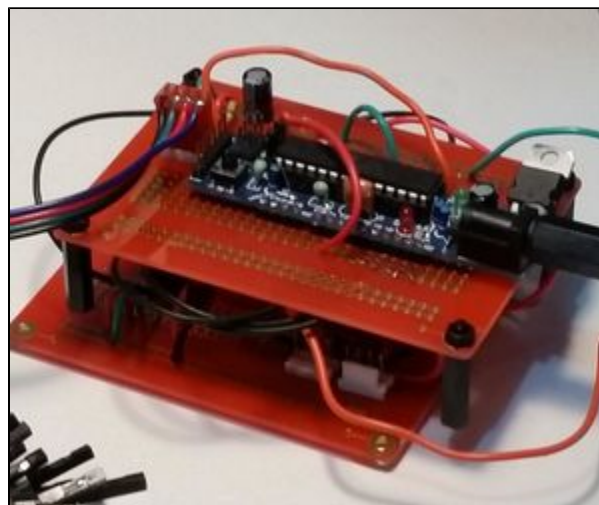
Circuit Assembly

- Assembled circuit using two boards like these: <http://a.co/9yLAEqu>
- 3D-printed a box for them to live in.

Suggestions

- Next time, for improved compactness, use a printed board with the traces already there.
- Use 3D CAD to plan the circuit layout.
- Assemble the circuit on a breadboard first.
- Add extra header pins onto the PCB to make oscilloscope debugging easier.

Pictured below is the circuit as its undergoing assembly. The microphone connected on the top, and the lights and connectors joined on the lower level. These proto-boards were excellent for first-time prototyping, but a much more compact circuit can be achieved using a specially designed PCB which would probably not require 2 levels.



Software Details

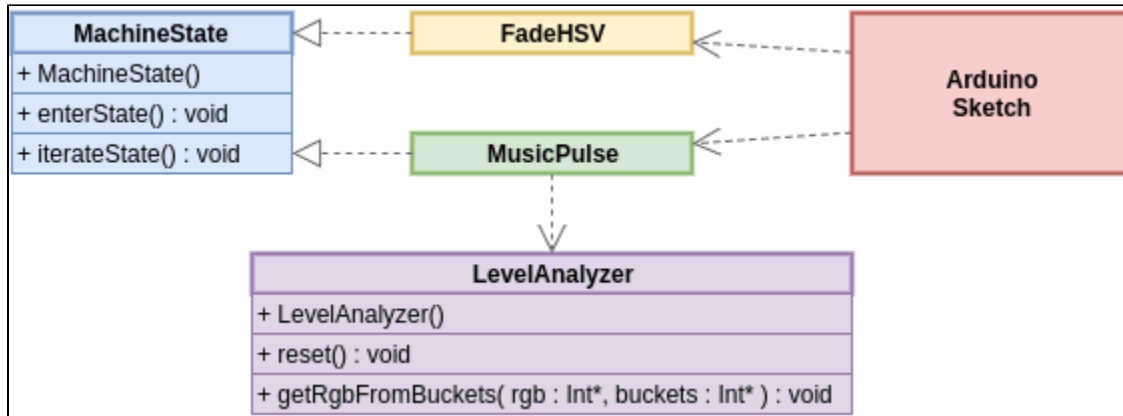
High-Level Design

The main loop of the Arduino program allows the wearer to cycle between different modes that define the suit's operation. By pushing the 'change mode' button, the Arduino program changes to a new mode and executes that mode's subroutines.

Each mode has a contract that requires that:

- The mode (MachineState) is instantiated during the execution of **setup()**.
- The mode has a method **enterState()** that gets called when the user switches to that mode.
- The mode has a method **iterateState()** that gets called during **loop()** when the user has not changed modes.

Pin assignments are maintained in a global constants file but each mode is responsible for correctly configuring the pins that it uses.



Filtering Algorithm

A filtering algorithm is required to sort the volume levels in each of the frequency bins (7) into three colours (R,G,B) in such a way that the lights seem to jump along with the music. The algorithm must result in a light display that:

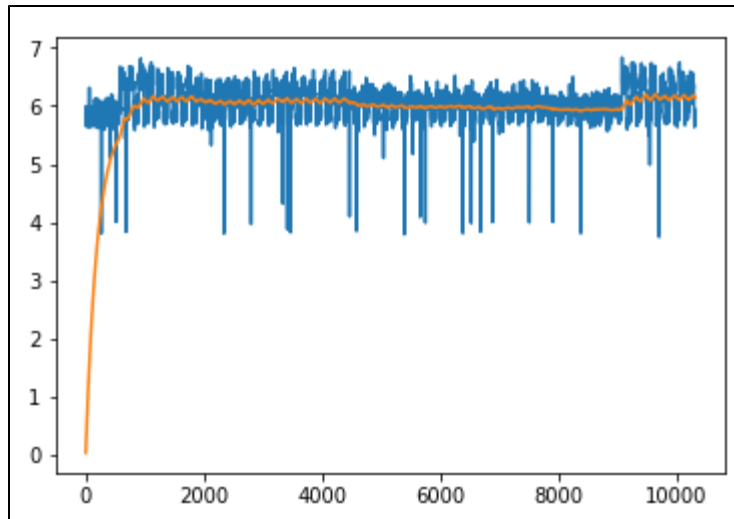
- Looks cool to watch
- Responds appropriately to a variety of types of music,
- Responds appropriately to different volumes of music, and
- Ideally, is not disrupted by non-musical sounds in the area.

It turns out this is quite difficult to achieve. A bunch of algorithms have been experimented with which:

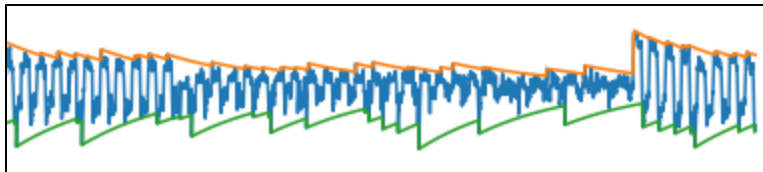
- Bias certain channels to achieve a certain duty cycle of "on-time"
- Try to measure a max/min envelope of the input signal and scale the output accordingly.
- Measure the audio levels with zero input and subtract that amount as background noise.

The algorithm that seems to work is as follows:

1. Take the natural log of the input bins.
2. Sort the input bins into red, green and blue colours. (The remainder of the algorithm focuses on a single RGB channel).
3. Measure a running average of the allocated signal, and subtract it from the allocated signal to get a zero-centred signal.
 - Pictured below, the input signal for the Red channel (in blue ink) and the running average of the Red channel (orange).
 - After the subtraction, the red channel signal is centred around zero.



4. Smooth the signal slightly.
5. Calculate a running maximum from the smoothed signal.
 - I did this using a continually decaying value that can be "bumped up" if it finds a higher value.
 - Note: In this picture, the running minimum (green) is also shown. However, it isn't used for anything in the current generation of algorithm.



6. Take any negative values on the smoothed signal and set them to zero.
7. Normalize the smoothed signal (to keep it between 0 and 1) by dividing it by the running maximum. There are two caveats:
 - a. If the maximum is less than 0.1 decibels, the resulting signal is simply zero because it's too quiet.
 - b. If the maximum is between 0.45 and 0.1 decibels, the result is linearly reduced to zero.
8. Perform a final smoothing of the normalised signal, and multiply by 255 (or 260, if you prefer brighter lights).
9. Clamp the resulting float (approximately 0 to 260) to an *integer* between 1 and 255.
10. The resulting integer can be written to the lights.

Costs

Way too much....

▼ [Click here to expand...](#)

Item	Cost
Suit Vest	\$12
Green Cloth	\$3
Dark Cloth	\$3
Reflective Tape	\$3
RGB Connectors	\$13
RGB Wire	\$16
RGB Lights	\$42
Solder	\$14
Prototyping Boards	\$16
Board Spacers	\$13
(Various Components and Connectors)	\$38

Amplifiers (2)	\$15
Battery	\$65
Battery Charger	\$35
3D Printing the Project Box	\$25
Total	~\$325

Recommendations and Ideas

- Do not get batteries from Digi-Key. You can similar ones at half cost from Amazon.
- Add a "Police Car" mode for the lights!
- Add a manual override for light brightness, which can be manually controlled.
- Fix up the filtering algorithm!
- See suggestions above for circuit construction.
- Learn how to sew.