ECS724 Interactive Digital Media Techniques · Autumn 2014 · Project 2

Do Something Cool

Beici Liang 140253278

Overview

I made a Beat Dress by Arduino and Processing as my IDMT final project. The LEDs inserted in the dress can blink with the beats of kick drum, snare drum and hat drum detected in a dance music, which are separately shown by different colours of LEDs. There is also a little box connected with the dress. It will monitor people's heartbeat by holding the pulse sensor against the fingertip. The results of drum beats and heart rate can be displayed on the computer's screen. Thus the dress can detect both the song's beat and heartbeat, which is the reason why I call it Beat Dress.

Principle

1. Drum Beat Detection Algorithm

According to Frederic Patin's algorithm [1], the sound will be heard as a beat when the energy is largely superior to the sound's energy history, which can be realised by comparing the instant energy, using the current *Ni* samples, with the every energy of the last sound, using the previous *Nae* samples. The algorithm can be summarised by the following steps:

$$LE_i[N_i] = \text{sample}(\text{ Left, } N_i\text{ })$$
$$RE_i[N_i] = \text{sample}(\text{ Right, } N_i\text{ })$$
$$E_i[N_i] = \frac{LE_i[N_i] + RE_i[N_i]}{2}$$

- Calculate the average after sampling the left and right channels of the signal.
- Use a Fast Fourier Transform (FFT) on the average samples to move from a time domain to a frequency, and divide them into *X* sub-bands.

$$E_s[j] = \frac{X}{N_i} * \sum_{k=j*(Ni/X)}^{(j+1)(Ni/X)} E_i[k] \quad , \quad 0 < j < X - 1$$

- Calculate the average sound energy of the previous *Nae* sample.

$$E_a = \frac{N_i}{N_{ae}} * \sum_{k=0}^{(Ni/Nae)-1} E_s[k]$$

- Compare instant energy *Ei* to *C * Ea* where *C* is a constant which will determine the sensibility of the algorithm to beats. If $Ei > C * Ea$, a beat has been detected.
- Since the FFT is used to obtain a spectrum, each sub-band can be tracked individually. Thus it is possible to track sounds that occur in different parts of the frequency spectrum independently, like the kick drum, snare drum and hat drum in our case.

2. Heartbeat Sensor

The sensor pressed closely to the fingertip is used for turning the mechanical pulsing action to a electric signal. The amount of infrared light from the emitter that gets reflected back to the detector will increase when the blood pressure suddenly rises, which makes the detector passes more current so as to lead a voltage drop to enter the amplifier. LM324, a single integrated circuit which includes two operational amplifiers, is used to emphasise the peaks and filter out noise. The signal will be sent to the Arduino Uno, and then drawn by Processing.

Parts List

1. Basic Parts: Dress x 1, Arduino Uno x 1, Wire, Breadboard, PC Board, Computer with Arduino and Processing, Electric Tools

2. LED Blink with Drum Beat Parts: TLC5940 x 1, 3mm White LED x 6, 3mm Yellow LED x 5, 3mm Blue LED x 5, 2.2k Ω Resistor x 1, 10k Ω Resistor x 1, 560 Ω Resistor x 3
3. Heartbeat Sensor Parts: 220 Ω Resistor x 1, 1k Ω Resistor x 1, 1.8k Ω Resistor x 1, 8.2k Ω Resistor x 1, 39k Ω Resistor x 1, 68k Ω Resistor x 2, 470k Ω Resistor x 1, 0.1µF Ceramic Capacitor x 2, 1.0 µF Tantalum Capacitor x 2, Transistor 2N3904 x 1, LM324 Quad Op Amp x 1, Infrared LED Emitter and Detector

Wire Up

1. LED Blink with Drum Beat

Since the PWM outputs in Arduino Uno were not enough for the project, I used TLC5940 to drive 16 LEDs. The schematic is shown in Figure 1. It should be noticed that there are three 560 Ω resistors connected with yellow LEDs in parallel. It is because the forward voltage of yellow LED should be no higher than 1.8 V, unlike 3.3 V for white, blue and green LEDs. If the forward voltage is higher, the LEDs may be dim or may not even light. After checking the voltage between each output pin of TLC 5940 and source, several resistors are used for protection.

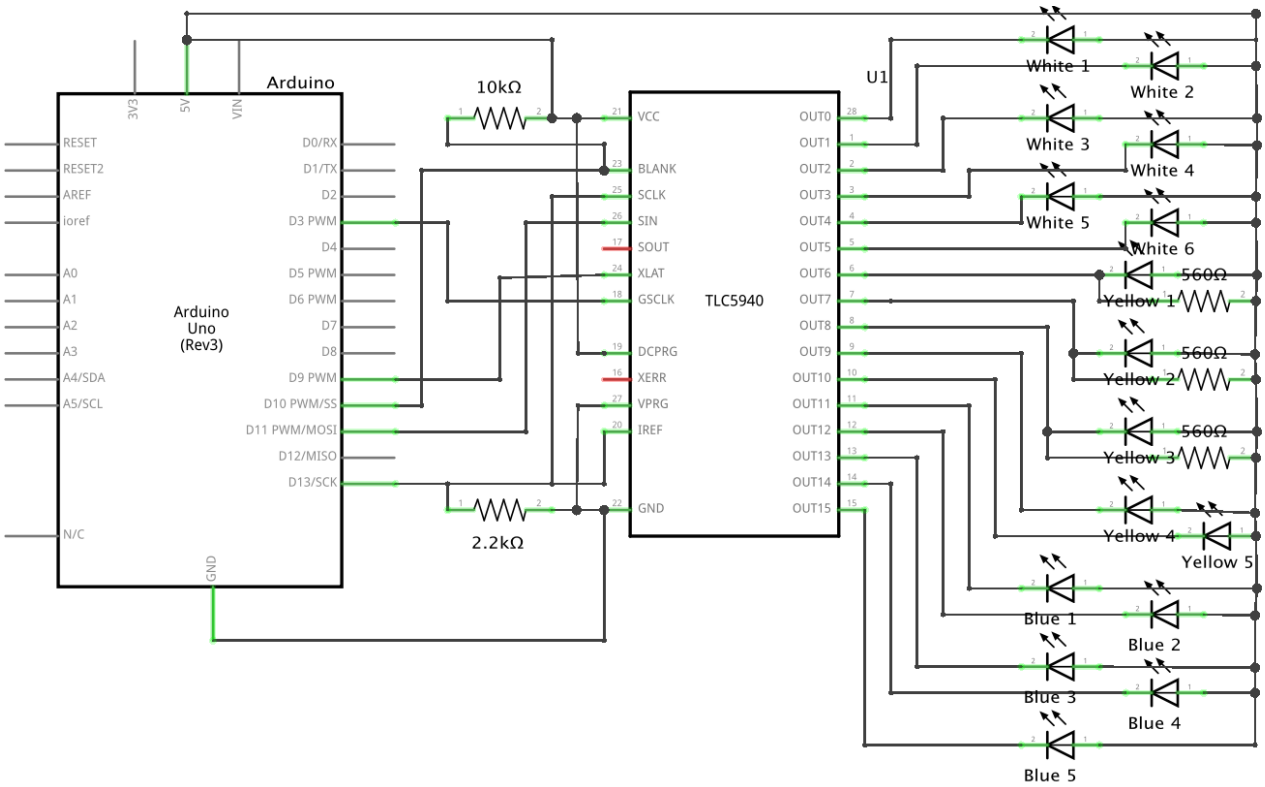

Figure 1

2. Heartbeat Sensor

I soldered all the necessary parts on a PC board according to the schematic found on the *Makezine* [2] which is shown in Figure 2. In order to avoid touching the circuit except the infrared LED emitter and detector, a little plastic box was made by laser cutter to pack up the sensor.

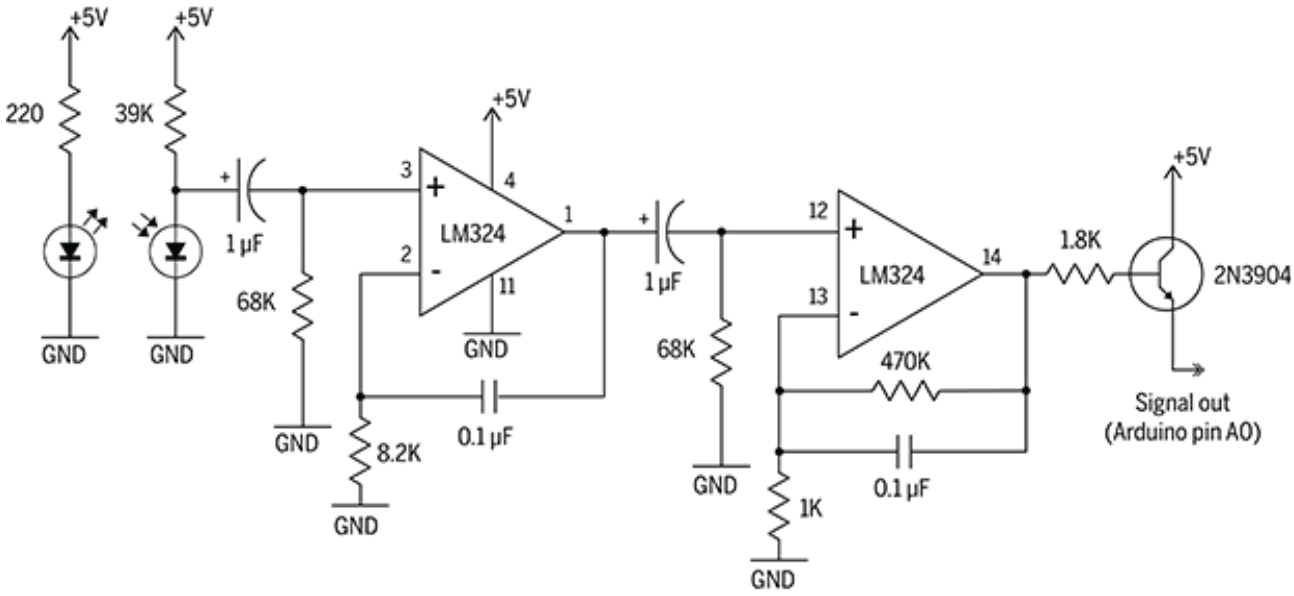

Figure 2

Programming

1. Arduino

The Arduino works as a medium for the communication between the Beat Dress and computer. It is used for receiving the value sent from heartbeat sensor and sending it to the Processing, as well as controlling the on and off of LEDs.

Only one analog input pin is needed for reading the voltage changes in the heartbeat sensor, which the value will be continually sent out to the serial port for Processing to draw the heart rate graph.

When the dance music is played, Arduino will read the value stored in an array sent by Processing. The value indicates which drum beat has been detected, according to which the LED with different colours will blink. In our case, turn the white LEDs on at the moment when the kick drum beat is detected. Yellow LEDs stand for the snare drum beat, and blue LEDs stand for the hat drum beat. Instead of using *digitalWrite()* or *analogWrite()* to control the LED, *tlc.set()* is handy for controlling the lightness of more PWM outputs, but note that the *Tlc5940* library should be downloaded into Arduino's library firstly.

2. Processing

Having not had much experience with Java prior to this project, the main difficulty was programming a function for drum beat detection. Fortunately, I found that there was a class called *BeatDetect* for doing beat detection in an audio library called *Minim* [3].

Thus the main work completed in the Processing were implementing an *AudioListener* in the class of *BeatListener* so as to call *detect* on every buffer of audio. Inside of *setup()*, create the objects and load the song for detection. Inside of *draw()*, use the *isXXX* functions to find out what beats have occurred in that frame. Return true if a beat corresponding to the frequency range of a XXX drum beat has been detected, and send the results in an array to the serial port for controlling the

corresponding LEDs by Arduino. Meanwhile, the dance music loaded into the Processing can be paused, played again, and rewinded by pressing specified key of the computer's keyboard.
Another important function is to draw the heart rate graph in real time, which was realised by calling *serialEvent()*. Since the heart rate was stored as *string* in the serial port from Arduino, it should be converted into *int* at first so as to draw the graph by *line()* in Processing.

Discussion

The challenges I met were mainly concentrated on the following four sectors:
1. Amplifier Circuit
There has already been a well-designed pulse sensor for Arduino sold on the Internet. Considering my background in integrated circuit, I would rather DIY than buy one. Taking the schematic shown in Figure 1 as a reference, I assembled the circuit on a PC board and replaced the aluminium electrolytic capacitors with tantalum ones for simplicity and stability. But the sensor was still too sensitive to the change of surrounding lightness. Also, if there are oils or sweat on the skin touching the circuit, it is possible for a weak short which can cause discomfort. Thus I used the laser cutter to make a little box that can pack up the circuit, and only the infrared emitter and detector were exposed in order to sense the blood pressure of the fingertip. However, since the sensor output is exposed, crazy noise are sometimes put into the output, which is still a problem need to be solved.
2. Soldering
16 LEDs were inserted in the dress, so there were 32 pins being soldered, which took me a lot of time to finish. Multimeter were used to check whether the LEDs had been well connected or not. Finally, I used the insulating tape to wrap all the joints.
3. Serial Communication
There is only one serial port for the communication between Arduino and Processing. In my project, multiple datas were sent and received at the same time. Therefore, in Processing, I stored the results of drum beat detection in the array to send to Arduino. However, there is still a problem unfixed that Processing cannot draw the heart rate graph as usual when the music is on, because the results of drum beat detection are also stored in the serial port, which will cause confusion on the values of heart rate.
4. Beat Detection Algorithm
The algorithm worked well with dance or techno music, but it may not perform well with other styles of music. In the future work, I would focus on how to apply the algorithm to more types of music, as well as increase the accuracy of kick/snare/hat drum beat detection.

Share the Project Online

Read the report and download the code on my website:

http://beiciliang.weebly.com/blog/beat-dress-my-idmt-final-project

Reference

1. http://article.programmersheaven.com/Patin/BeatDetectionAlgorithms.pdf
2. http://makezine.com/video/visualize-your-heartbeat-with-this-homemade-pulse-sensor/
3. http://code.compartmental.net/tools/minim/