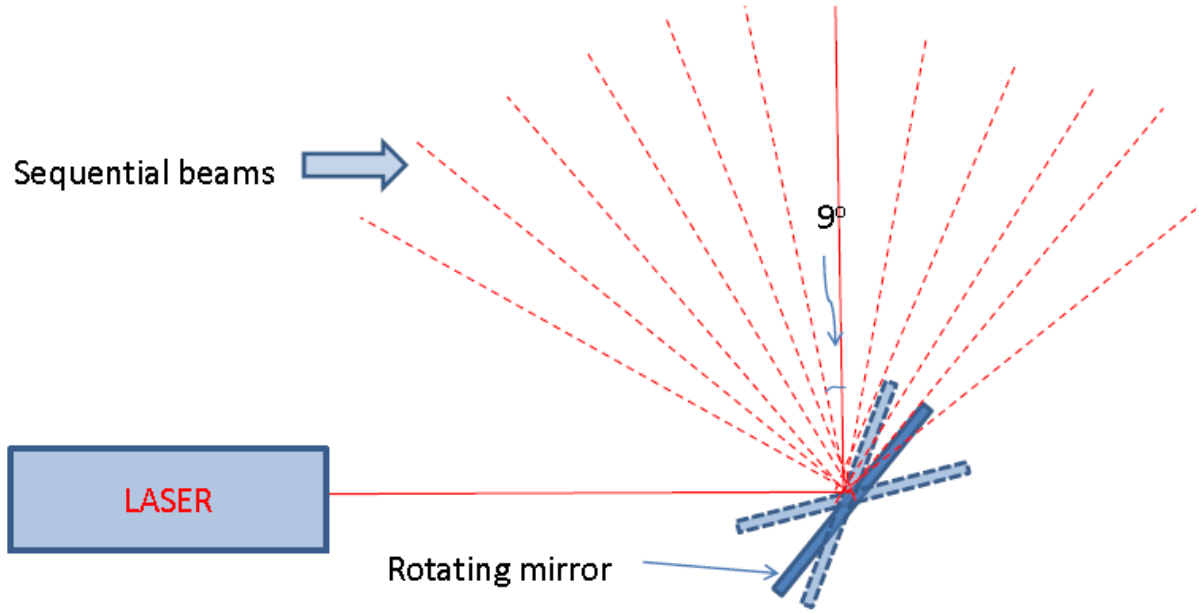
Laser harp

Prasad P. Bandarkar (08026006)
Sukhdeep Singh (08026020)

Laser harp is made using a single laser source, a reflecting mirror rotated by a stepper motor controlled by Arduino microcontroller, an LDR as a pick and a speaker to produce sound.

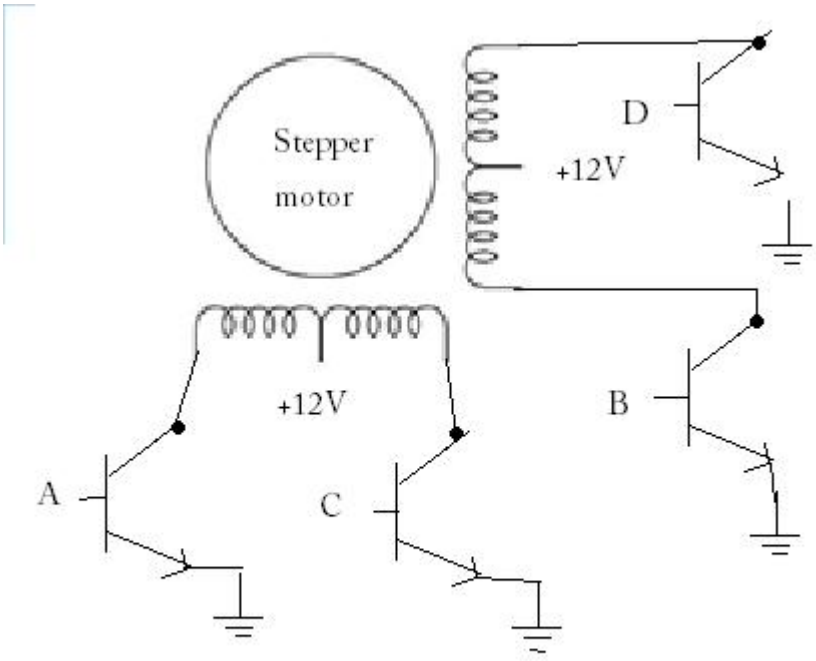
Strings of the harp:

As shown in the diagram below, the strings of the harp are created by sequentially reflecting the beam at different angles.



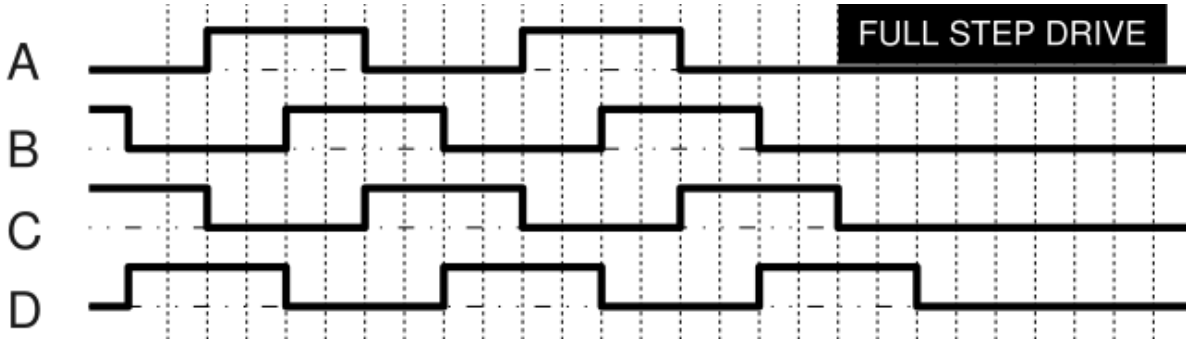
Due to persistence of vision, we may be able to see each string separately. The mirror is rotated by a stepper motor controlled by the Arduino board. Motor is rotated in steps of 9 degree by the motor. Each step is used to denote one frequency (or string of harp). Motor is rotated by the arduino using movement functions (two functions, one each for clockwise and anticlockwise motion) defined in the program. The movement functions rotate the motor once by 9 degrees, in respective direction. The movement functions are called from loop for each step movement, with some time delay, to allow user to see light corresponding to each frequency and also to let Arduino read if it is picked by the user.

The motor is controlled by the Arduino using 4 digital inputs. A driver circuit consisting of 4 transistors is necessary to drive the motor. The circuit connections as shown in the diagram.



Circuit Diagram of stepper motor

A,B,C,D are the four digital inputs. A pulse sequence as shown in the diagram below is applied to the driver circuit. The technique used to drive the above motor is called Full-Step Drive. In this case the supply to each motor is reversed alternately to drive the motor.



Sensor:

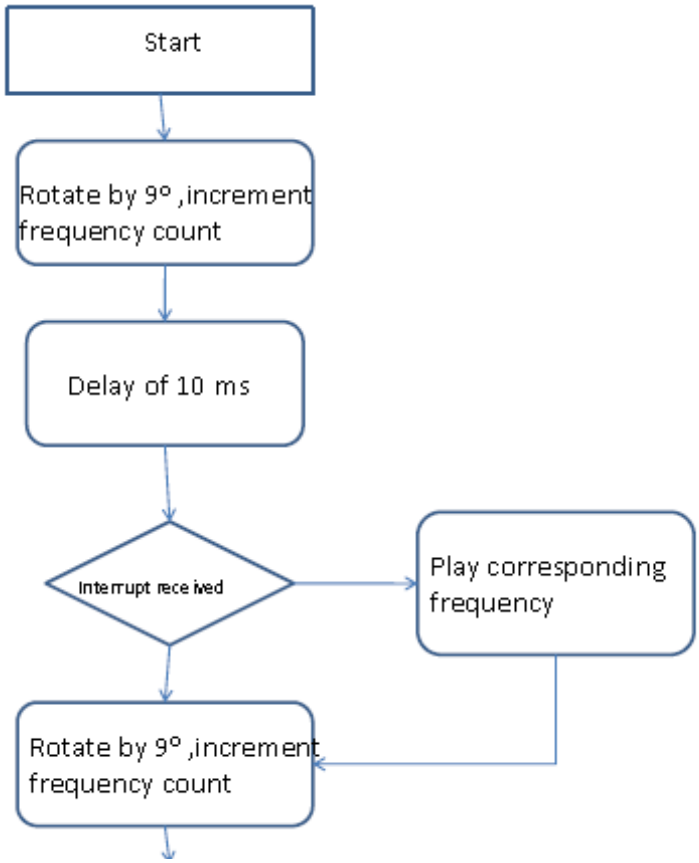
Pick for the harp uses an LDR for detecting the laser beam. The following circuit is used to connect the LDR. The voltage at between the resistor and the LDR is used as digital input to tan external interrupt pin

The voltage at between the resistor and the LDR is used as digital input to interrupt the Arduino program and let it know that sound for that particular frequency is to be played. Note that string can only be picked. Keeping pick over light continuously will not keep the program in suspended state and sound

will not be continuously played. Key to let the user play continuously is to shift between different frequency quickly, so that he can pick a string regularly. To prevent multiple interrupts from generating very quickly and disrupting the flow of the program, interrupts are disabled after the first interrupt till the note is played.

Sound is played using freqout function. The freqout function produces a squarewave of the corresponding frequency at an output pin. It uses the delayMicroseconds() function to achieve this. This is signal is then fed to a 8 Ohm speaker which produces the note.

The flow of the program is represented in the diagram below. The flow continoues till all 12 beams are created. Then the rotation angle reverses to produce the beams in the reverse order.



Work Distribution

Sukhdeep:: Designed the basic skeleton of the code being used and implemented loop function and functionality using interrupts to read input from the photodiode and play sound using freqout function.

Prasad:: Worked on stepper motor and it's electrical parts to make sure that it rotates the mirror properly. Implemented movement functions in the code to rotate the motor. Helped in implementing freqout function.

We both worked together in deciding the geometry of the system and implementing it physically.

Implementation

Code:

```
//Project Code
//   Sukhdeep Singh 08026020
//   Prasad Bandarkar 08026006

int motorPin1 = 6;              // Pin for output to motor
int motorPin2= 5;
int motorPin3 = 4;
int motorPin4 = 3;

int soundpin = 9;             // output to speaker
int t = 100;   // duration of sound in msecs

int s_freq = 1;              // frequency activated
int t_freq = 12;         // Total number of frequencies

int state =1,move=0;           // state of motor and how steps it moved
int i=1;
int pick=0;                 // to store if string is picked
unsigned long dely=100;       // delay before system move to next frequency


int tones[] = {512,542,575,609,645,683,724,767,813,861,912,967};        // to denotes frequency tones

long delayTimeSmall = 10;               // delay in each motor step

void setup(){
pinMode(motorPin1,OUTPUT);
pinMode(motorPin2,OUTPUT);
pinMode(motorPin3,OUTPUT);
pinMode(motorPin4,OUTPUT);

attachInterrupt(0,picked,FALLING);

}

void loop()
{
  for(;s_freq<=t_freq;)            // loop to generate strings. motor moves in 1 direction determined by
move2anti function
  {
    if (s_freq >1)                  // no rotation for 1st frequency
    move2anti();                // rotate motor

    interrupts();          // interrupt on to check if string is plucked
```

```
   delay(dely);           // delay to be spent at each string
   noInterrupts();          // interrupts off as system starts moving to next string

   if (pick==1)                 // if string is plucked when interrupt was on, play sound
   {
    freqout(tones[s_freq-1]);
    pick=0;
   }
   s_freq ++;              // next frequency
  }

s_freq --;      // s_freq=13 after for loop ends. Bring back to 12


  for(;s_freq>=1;)          // loop to generate strings. motor back moves in opposite direction using
move2 function
  {


   if (s_freq <12)           // no rotation for 12th string
   move2();

  interrupts();         //turn interrupt on
  delay(dely);
   noInterrupts();                  //turn interrupt off

   if (pick==1)
   {
    freqout(tones[s_freq-1]);
    pick=0;
   }

  s_freq--;
  }

  s_freq ++;           // after for loop, s_freq=0, make it 1
}

void picked(){

   pick=1; // picking detected
      noInterrupts();

}

void freqout(int freq)  // freq in hz, t in ms
{
  int hperiod;                          //calculate 1/2 period in us
  long cycles, i;
  pinMode(soundpin, OUTPUT);                // turn on output pin

  hperiod = (500000 / freq) - 7;            // subtract 7 us to make up for digitalWrite overhead

  cycles = ((long)freq * (long)t) / 1000;    // calculate cycles
```

```
  for (i=0; i<= cycles; i++){            // play note for t ms
    digitalWrite(soundpin, HIGH);
    delayMicroseconds(hperiod);
    digitalWrite(soundpin, LOW);
    delayMicroseconds(hperiod - 1);     // - 1 to make up for digitaWrite overhead
  }
pinMode(soundpin, INPUT);               // shut off pin to avoid noise from other operations
}



void move2()              // function to move motor
{
  if (state==1){
 digitalWrite(motorPin1, HIGH);
digitalWrite(motorPin2, LOW);
digitalWrite(motorPin3, LOW);
digitalWrite(motorPin4, HIGH);
delay(delayTimeSmall);
state=2;                // change to next state
move++;
if (move==4)          // return when motor has moved 5 steps
{
  move=0;
  return ;
}
  }

  if (state==2){
  digitalWrite(motorPin1, HIGH);
digitalWrite(motorPin2, HIGH);
digitalWrite(motorPin3, LOW);
digitalWrite(motorPin4, LOW);
delay(delayTimeSmall);
state=3;
move++;
if (move==4)
{
  move=0;
  return;
}
  }


if (state==3){
digitalWrite(motorPin1, LOW);
digitalWrite(motorPin2, HIGH);
digitalWrite(motorPin3, HIGH);
digitalWrite(motorPin4, LOW);
delay(delayTimeSmall);
state=4;
move++;
if (move==4)
{
  move=0;
```

```
  return;
}

  }


if (state==4){
digitalWrite(motorPin1, LOW);
digitalWrite(motorPin2, LOW);
digitalWrite(motorPin3, HIGH);
digitalWrite(motorPin4, HIGH);
delay(delayTimeSmall);
state=1;
move++;
if (move==4)
{
  move=0;
  return;
}
}
}


void move2anti()              //move in opposite direction to move2
{
  if (state==1){
  digitalWrite(motorPin1, LOW);
digitalWrite(motorPin2, LOW);
digitalWrite(motorPin3, HIGH);
digitalWrite(motorPin4, HIGH);
delay(delayTimeSmall);
state=2;
move++;
if (move==4)
{
  move=0;
  return;
}
}

if (state==2){
digitalWrite(motorPin1, LOW);
digitalWrite(motorPin2, HIGH);
digitalWrite(motorPin3, HIGH);
digitalWrite(motorPin4, LOW);
delay(delayTimeSmall);
state=3;
move++;
if (move==4)
{
  move=0;
  return;
}
}
```

```
if (state==3){
digitalWrite(motorPin1, HIGH);
digitalWrite(motorPin2, HIGH);
digitalWrite(motorPin3, LOW);
digitalWrite(motorPin4, LOW);
delay(delayTimeSmall);
state=4;
move++;
if (move==4)
{
  move=0;
  return;
}
}

if (state==4){
digitalWrite(motorPin1, HIGH);
digitalWrite(motorPin2, LOW);
digitalWrite(motorPin3, LOW);
digitalWrite(motorPin4, HIGH);
delay(delayTimeSmall);
state=1;
move++;
if (move==4)
{
  move=0;
  return;
}
}
}
```

We have succesfully implemented the code and produced different frequencies in the time domain, as was shown in project demonstration. However, we could not implement the rotating mirror part using motor, due to some problems with motors. We tried to implement it using an old stepper motor provided by electronics lab. We were able to rotate that motor successfully in one direction using the functions for movement in above code. The motor however would get struck in reverse direction. After trying to make it work for few days, we decided to change the motor. Due to lack of time, we couldn't make new driver circuit for the new motor and decided to go ahead with the diver circuit of old motor, which probably was not a good idea but we still decided to go for it as both motor's power ratings were not too different. During the transfer, something went wrong unfortunately and the driver circuit hasn't been able to successfully transfer signal from microcontroller to the motor.

Acknowledgement

We would like to take this opportunity to acknowledge the workshop in Physics department IIT-B, to provide us with the mirror setup that can be attached to the motor.

We also thank Staff members of electonics lab in Physics department for their continous support and particularly Prof. Pradeep Sarin for his guidance at the start of the project.

References:

1. http://www.arduino.cc/en/Tutorial/StepperUnipolar
2. http://en.wikipedia.org/wiki/Stepper_motor