Hearing for All

Mukund Venkatakrishnan
duPont Manual High School


360,000,000


324,000,000


Affordability
&
Accessibility

$1,500


$616

Sweden: 3.926

UK: 2.809

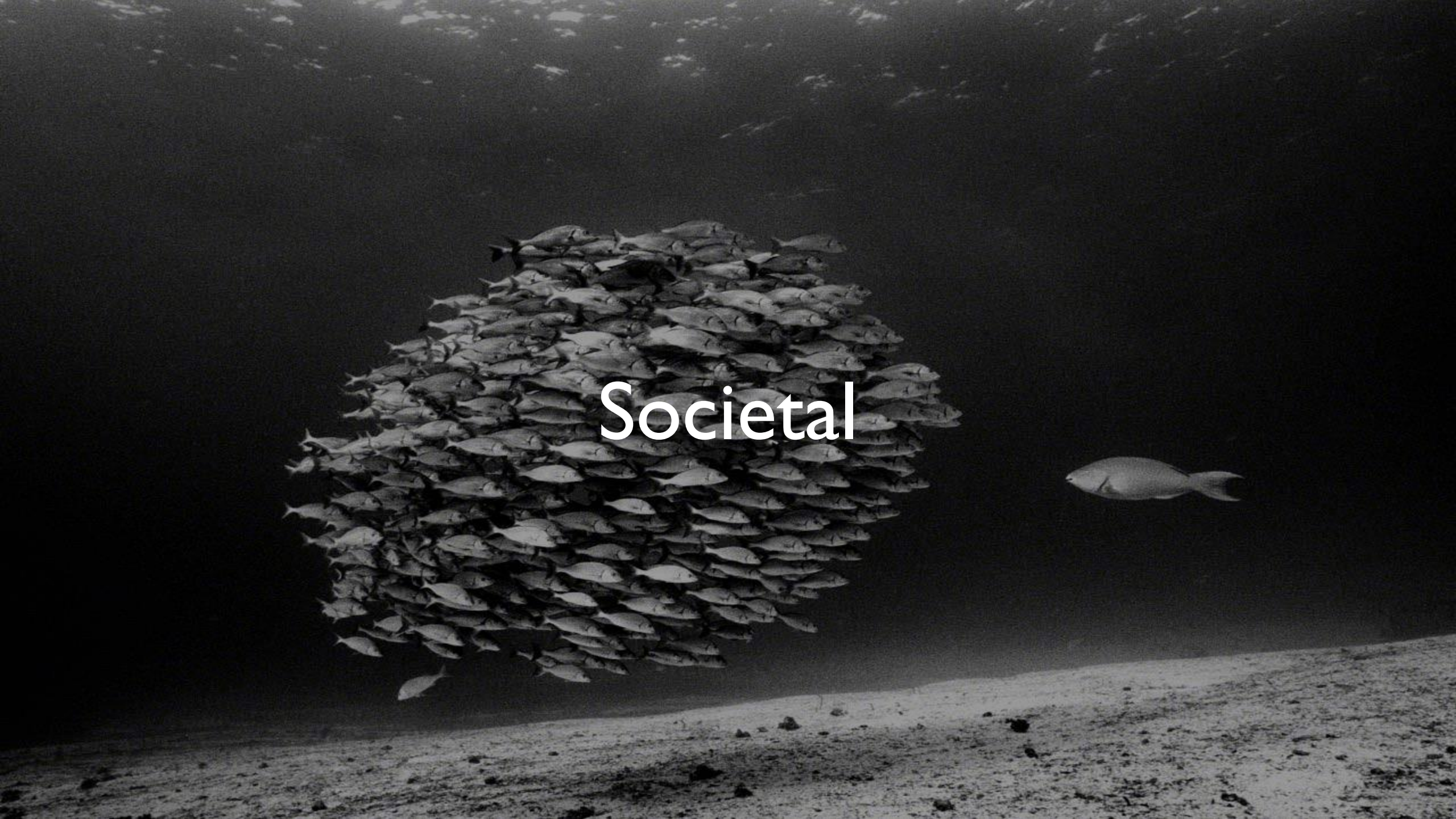
United States: 2.452

India: 0.702

Bolivia: 0.473

Liberia: 0.014

Functional


Societal


ALL SPICE
SUMAC
Sumak
HOT PAPRICA
POWDER
ACI
TOZ BIBER
MIXED
PEPPER
PAPRICA
POWDER
TOZ BIBER
WHITE
PEPPER
BEYAZ
KARABIBER
HOT PAPRICA
ACI BIBER
GREEN
PEPPER
YEŞİL BİBER
MIXED
TEA
ROSE
HIP TEA

Economic

Measures hearing loss

Treats hearing loss

Solution?

Solution

Reduce price

Self-reliant

Solution

Hardware | Software

Solution

Signal processor

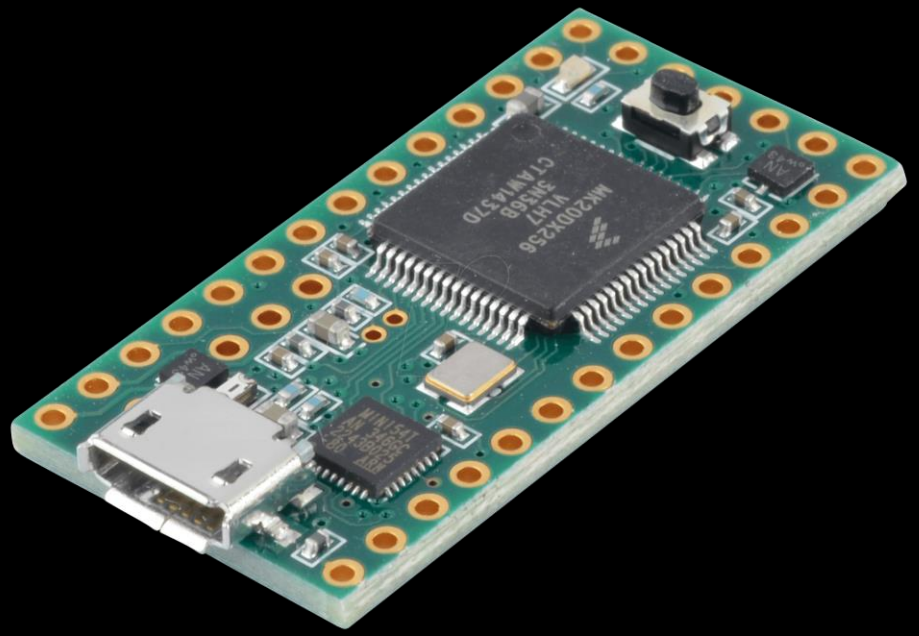
Code

How do they work?

Name of DSP	Ability to run complex filters	Ability to receive inputs	Ability to process sounds under .2 seconds
miniDSP	No	No	Not Tested
Arduino DSP Shield	No	Yes	Not Tested
PlainDSP	No	No	Not tested
Teensy 3.1	Yes	Yes	Yes

Teensy 3.1

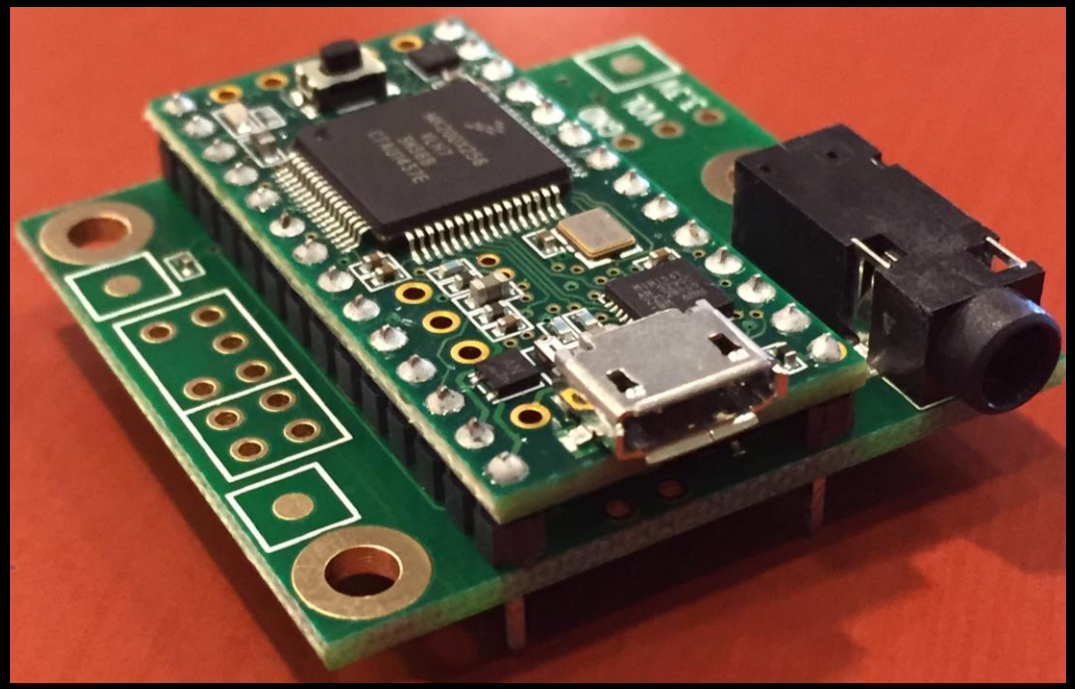
open-source



$15

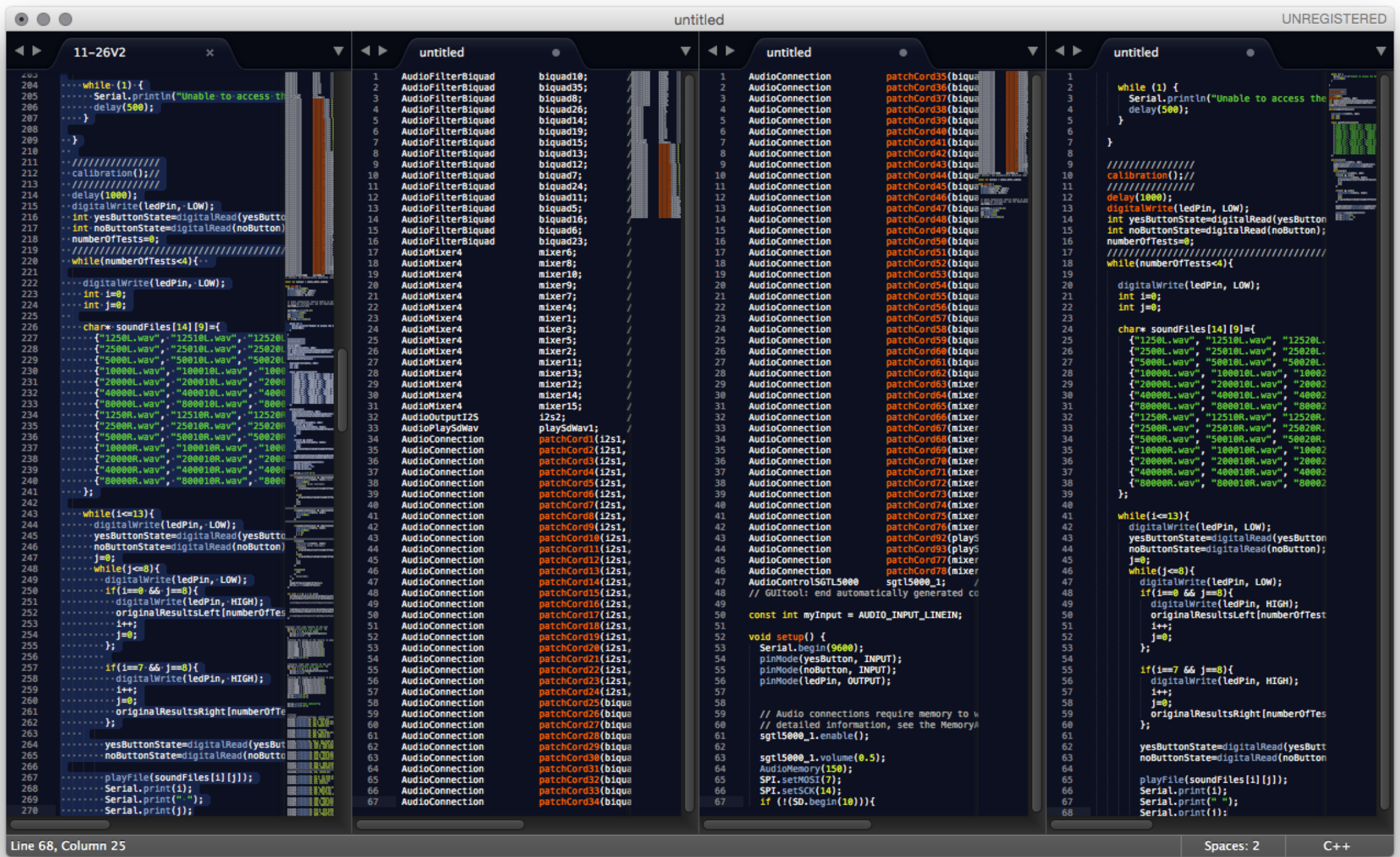
powerful processor

Hardware



Solution

DSP | Code


untitled
UNREGISTERED
11-26V2
untitled
untitled
untitled
while (1) {
Serial.println("Unable to access the
delay(500);
}
}
////////////////
calibration();//
////////////////
delay(1000);
digitalWrite(ledPin, LOW);
int yesButtonState=digitalRead(yesButton
int noButtonState=digitalRead(noButton);
numberOfTests=0;
////////////////////////////////////
while(numberOfTests<4){
digitalWrite(ledPin, LOW);
int i=0;
int j=0;
char* soundFiles[14][9]={
{"1250L.wav", "12510L.wav", "12520L.
{"2500L.wav", "25010L.wav", "25020L.
{"5000L.wav", "50010L.wav", "50020L.
{"10000L.wav", "100010L.wav", "10002
{"20000L.wav", "200010L.wav", "20002
{"40000L.wav", "400010L.wav", "40002
{"80000L.wav", "800010L.wav", "80002
{"1250R.wav", "12510R.wav", "12520R.
{"2500R.wav", "25010R.wav", "25020R.
{"5000R.wav", "50010R.wav", "50020R.
{"10000R.wav", "100010R.wav", "10002
{"20000R.wav", "200010R.wav", "20002
{"40000R.wav", "400010R.wav", "40002
{"80000R.wav", "800010R.wav", "80002
};
while(i<=13){
digitalWrite(ledPin, LOW);
yesButtonState=digitalRead(yesButton
noButtonState=digitalRead(noButton);
j=0;
while(j<=8){
digitalWrite(ledPin, LOW);
if(i==0 && j==8){
digitalWrite(ledPin, HIGH);
originalResultsLeft[numberOfTest
i++;
j=0;
};
if(i==7 && j==8){
digitalWrite(ledPin, HIGH);
i++;
j=0;
originalResultsRight[numberOfTes
};
yesButtonState=digitalRead(yesButt
noButtonState=digitalRead(noButton
playFile(soundFiles[i][j]);
Serial.print(i);
Serial.print(" ");
Serial.print(j);
AudioFilterBiquad biquad10;
AudioFilterBiquad biquad35;
AudioFilterBiquad biquad8;
AudioFilterBiquad biquad26;
AudioFilterBiquad biquad14;
AudioFilterBiquad biquad19;
AudioFilterBiquad biquad15;
AudioFilterBiquad biquad13;
AudioFilterBiquad biquad12;
AudioFilterBiquad biquad7;
AudioFilterBiquad biquad24;
AudioFilterBiquad biquad11;
AudioFilterBiquad biquad5;
AudioFilterBiquad biquad16;
AudioFilterBiquad biquad6;
AudioFilterBiquad biquad23;
AudioMixer4 mixer6;
AudioMixer4 mixer8;
AudioMixer4 mixer10;
AudioMixer4 mixer9;
AudioMixer4 mixer7;
AudioMixer4 mixer4;
AudioMixer4 mixer1;
AudioMixer4 mixer3;
AudioMixer4 mixer5;
AudioMixer4 mixer2;
AudioMixer4 mixer11;
AudioMixer4 mixer13;
AudioMixer4 mixer12;
AudioMixer4 mixer14;
AudioMixer4 mixer15;
AudioOutputI2S i2s2;
AudioPlaySdWav playSdWav1;
AudioConnection patchCord1(i2s1,
AudioConnection patchCord2(i2s1,
AudioConnection patchCord3(i2s1,
AudioConnection patchCord4(i2s1,
AudioConnection patchCord5(i2s1,
AudioConnection patchCord6(i2s1,
AudioConnection patchCord7(i2s1,
AudioConnection patchCord8(i2s1,
AudioConnection patchCord9(i2s1,
AudioConnection patchCord10(i2s1,
AudioConnection patchCord11(i2s1,
AudioConnection patchCord12(i2s1,
AudioConnection patchCord13(i2s1,
AudioConnection patchCord14(i2s1,
AudioConnection patchCord15(i2s1,
AudioConnection patchCord16(i2s1,
AudioConnection patchCord17(i2s1,
AudioConnection patchCord18(i2s1,
AudioConnection patchCord19(i2s1,
AudioConnection patchCord20(i2s1,
AudioConnection patchCord21(i2s1,
AudioConnection patchCord22(i2s1,
AudioConnection patchCord23(i2s1,
AudioConnection patchCord24(i2s1,
AudioConnection patchCord25(biqua
AudioConnection patchCord26(biqua
AudioConnection patchCord27(biqua
AudioConnection patchCord28(biqua
AudioConnection patchCord29(biqua
AudioConnection patchCord30(biqua
AudioConnection patchCord31(biqua
AudioConnection patchCord32(biqua
AudioConnection patchCord33(biqua
AudioConnection patchCord34(biqua
AudioConnection patchCord35(biqua
AudioConnection patchCord36(biqua
AudioConnection patchCord37(biqua
AudioConnection patchCord38(biqua
AudioConnection patchCord39(biqua
AudioConnection patchCord40(biqua
AudioConnection patchCord41(biqua
AudioConnection patchCord42(biqua
AudioConnection patchCord43(biqua
AudioConnection patchCord44(biqua
AudioConnection patchCord45(biqua
AudioConnection patchCord46(biqua
AudioConnection patchCord47(biqua
AudioConnection patchCord48(biqua
AudioConnection patchCord49(biqua
AudioConnection patchCord50(biqua
AudioConnection patchCord51(biqua
AudioConnection patchCord52(biqua
AudioConnection patchCord53(biqua
AudioConnection patchCord54(biqua
AudioConnection patchCord55(biqua
AudioConnection patchCord56(biqua
AudioConnection patchCord57(biqua
AudioConnection patchCord58(biqua
AudioConnection patchCord59(biqua
AudioConnection patchCord60(biqua
AudioConnection patchCord61(biqua
AudioConnection patchCord62(biqua
AudioConnection patchCord63(mixer
AudioConnection patchCord64(mixer
AudioConnection patchCord65(mixer
AudioConnection patchCord66(mixer
AudioConnection patchCord67(mixer
AudioConnection patchCord68(mixer
AudioConnection patchCord69(mixer
AudioConnection patchCord70(mixer
AudioConnection patchCord71(mixer
AudioConnection patchCord72(mixer
AudioConnection patchCord73(mixer
AudioConnection patchCord74(mixer
AudioConnection patchCord75(mixer
AudioConnection patchCord76(mixer
AudioConnection patchCord92(playS
AudioConnection patchCord93(playS
AudioConnection patchCord77(mixer
AudioConnection patchCord78(mixer
AudioControlSGTL5000 sgtl5000_1;
// GUItool: end automatically generated co
const int myInput = AUDIO_INPUT_LINEIN;
void setup() {
Serial.begin(9600);
pinMode(yesButton, INPUT);
pinMode(noButton, INPUT);
pinMode(ledPin, OUTPUT);
// Audio connections require memory to w
// detailed information, see the MemoryA
sgtl5000_1.enable();
sgtl5000_1.volume(0.5);
AudioMemory(150);
SPI.setMOSI(7);
SPI.setSCK(14);
if (!(SD.begin(10))){
Line 68, Column 25
Spaces: 2
C++

11-26V2

```
    mixer7.gain(2,mixGain(result500L, resul
    mixer7.gain(3,mixGain(result500L, resul

    mixer8.gain(0,mixGain(result500L, resul
    mixer8.gain(1,mixGain(result1000L, resu
    mixer8.gain(2,mixGain(result1000L, resu
    mixer8.gain(3,mixGain(result1000L, resu

    mixer9.gain(0,mixGain(result2000L, resu
    mixer9.gain(1,mixGain(result2000L, resu
    mixer9.gain(2,mixGain(result2000L, resu
    mixer9.gain(3,mixGain(result4000L, resu

    mixer10.gain(0,mixGain(result4000L, res
    mixer10.gain(1,mixGain(result4000L, res


}

double mixGain(int dB1, int dB2, double B
  double mix = ((point*(dB2-dB1))/BW)+dB1
  mix=pow(10, mix/20);
  mix = (mix-0.1276)/1.7754;
  Serial.println(mix);
  return mix;
}

void calibration(){
  int yesButtonState=digitalRead(yesButto
  int noButtonState=digitalRead(noButton)

  while(yesButtonState==1 && noButtonStat
    yesButtonState=digitalRead(yesButton)
    noButtonState=digitalRead(noButton);

    Serial.print("Calibrating...");
    Serial.print("\n");

    sgtl5000_1.volume(.9);
    playFile("1234.WAV");
    delay(1000);
  };
  Serial.print("\n");
  Serial.print("Calibration complete.");
  digitalWrite(ledPin, HIGH);
}

void playFile(char* filename)
{
  Serial.print("Playing file: ");
  Serial.println(filename);

  // Start playing the file.  This sketch
  // run while the file plays.
  playSdWav1.play(filename);

  // A brief delay for the library read W
  delay(5);

  // Simply wait for the file to finish p
  while (playSdWav1.isPlaying()) {
    float vol = analogRead(1);
    vol = vol / 1024;
    sgtl5000_1.volume(vol);
  }
}
```

untitled

```
Serial.print("\n");
  ////////////////////////////////////
        if(yesButtonState==0 && noButtonSt
          digitalWrite(ledPin, HIGH);//abl
          delay(1000);
          //Results being recorded//
          if(i<=6){
            originalResultsLeft[numberOfTe
          }

          else{
            originalResultsRight[numberOfT
          };
          j=0;
          i++;
        };
  ////////////////////////////////////
        if(yesButtonState==1 && noButtonSt
          digitalWrite(ledPin, HIGH);
          delay(1000);
          j++;
        };
  ////////////////////////////////////

        if(yesButtonState==0 && noButtonSt
          digitalWrite(ledPin, HIGH);
          delay(1000);
          i = 14;
          j = 9;
        };
  ////////////////////////////////////
        if(j==9){
          digitalWrite(ledPin, HIGH);
          //Results being recorded//
          if(i<=5){
            originalResultsLeft[numberOfTe
          }
          else{
            originalResultsRight[numberOfT
          }
          i++;
          j=0;
        };

        if(i>=14){
          j=9;
        };
  //    delay(100);
      };
    };
    numberOfTests=numberOfTests+1;
    Serial.print(numberOfTests);
  };


  for (int i = 0; i < 7; ++i){
    leftResultsFinal[i]=(originalResultsLe
    originalResultsLeft[2][i]+originalResu

    //leftResultsFinal[i]=leftResultsFinal


    rightResultsFinal[i]=(originalResultsR
    originalResultsRight[2][i]+originalRes

    //rightResultsFinal[i]=rightResultsFin
  };
```

untitled

```
//Outputs left side results to the user
  Serial.print("Left side results: ");
  for (int i = 0; i < 7; ++i){
    Serial.print(leftResultsFinal[i]);
    Serial.print(", ");
  }
  //Stores the values of the results to pa
  result125L = leftResultsFinal[0];
  result250L = leftResultsFinal[1];
  result500L = leftResultsFinal[2];
  result1000L = leftResultsFinal[3];
  result2000L = leftResultsFinal[4];
  result4000L = leftResultsFinal[5];
  result8000L = leftResultsFinal[6];
  Serial.print("\n");


  //Outputs right side results to the user
  Serial.print("Right side results: ");
  for (int i = 0; i < 7; ++i){
    Serial.print(rightResultsFinal[i]);
    Serial.print(", ");
  }
  //Stores the values of the results to pa
  result125R = rightResultsFinal[0];
  result250R = rightResultsFinal[1];
  result500R = rightResultsFinal[2];
  result1000R = rightResultsFinal[3];
  result2000R = rightResultsFinal[4];
  result4000R = rightResultsFinal[5];
  result8000R = rightResultsFinal[6];
  Serial.print("\n");
  Serial.print("\n");


  Serial.print("TEST COMPLETE");
  Serial.print("\n");



  //format
  //biqaud.setBandpass(stage, center frequ
  biquad1.setBandpass(0, 125, 125/28.9);
  biquad2.setBandpass(0, 157.5, 157.5/36.5
  biquad3.setBandpass(0, 198.4, 198.4/45.9
  biquad4.setBandpass(0, 250, 250/57.9);

  biquad5.setBandpass(0, 315, 315/73);
  biquad6.setBandpass(0, 396.9, 396.9/88.8
  biquad7.setBandpass(0, 500, 500/115.8);
  biquad8.setBandpass(0, 630, 630/145.9);

  biquad9.setBandpass(0, 793.7, 793.7/183.
  biquad10.setBandpass(0, 1000, 1000/231.6
  biquad11.setBandpass(0, 1259.9, 1259.9/2
  biquad12.setBandpass(0, 1587.4, 1587.4/3

  biquad13.setBandpass(0, 2000, 2000/463.1
  biquad14.setBandpass(0, 2519.8, 2519.8/5
  biquad15.setBandpass(0, 3174.8, 3174.8/7
  biquad16.setBandpass(0, 4000, 4000/926.2

  biquad17.setBandpass(0, 5039.7, 5039.7/1
  biquad18.setBandpass(0, 6349.6, 6349.6/1
  biquad19.setBandpass(0, 8000, 8000/1852.
  ////////////////////////////////////
  biquad20.setBandpass(0, 125, 125/28.9);
```

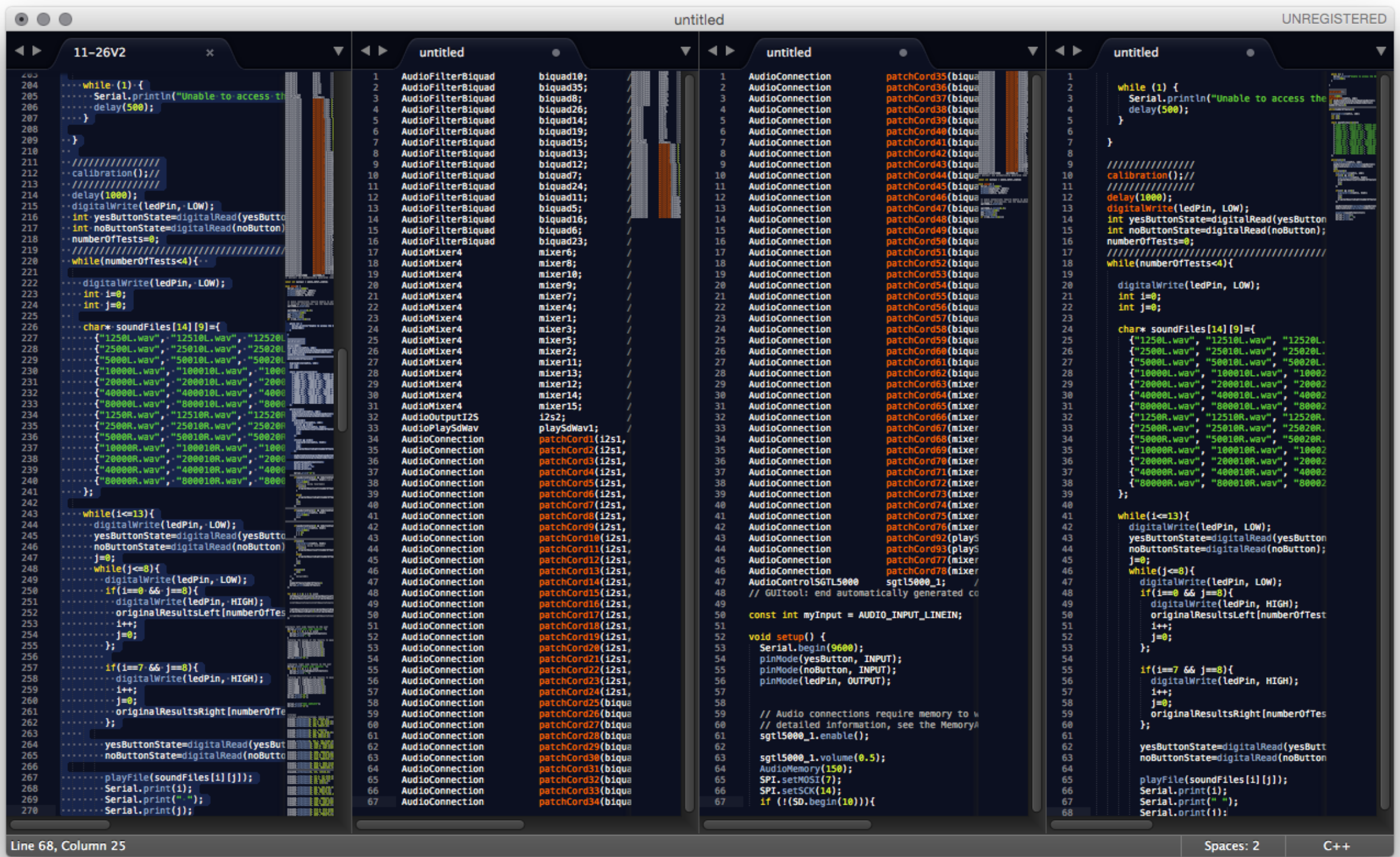
untitled

```
biquad21.setBandpass(0, 157.5, 157.5/36.5)
  biquad22.setBandpass(0, 198.4, 198.4/45.
  biquad23.setBandpass(0, 250, 250/57.9);
  biquad24.setBandpass(0, 315, 315/73);

  biquad25.setBandpass(0, 396.9, 396.9/88.
  biquad26.setBandpass(0, 500, 500/115.8);
  biquad27.setBandpass(0, 630, 630/145.9);
  biquad28.setBandpass(0, 793.7, 793.7/183

  biquad29.setBandpass(0, 1000, 1000/231.6
  biquad30.setBandpass(0, 1259.9, 1259.9/2
  biquad31.setBandpass(0, 1587.4, 1587.4/3
  biquad32.setBandpass(0, 2000, 2000/463.1

  biquad33.setBandpass(0, 2519.8, 2519.8/5
  biquad34.setBandpass(0, 3174.8, 3174.8/7
  biquad35.setBandpass(0, 4000, 4000/926.2
  biquad36.setBandpass(0, 5039.7, 5039.7/1

  biquad37.setBandpass(0, 6349.6, 6349.6/1
  biquad38.setBandpass(0, 8000, 8000/1852.
}

unsigned long last_time = millis();

void loop(){
  sgtl5000_1.inputSelect(myInput);
  if (millis() - last_time >= 2500) {
    Serial.print("Proc = ");
    Serial.print(AudioProcessorUsage());
    Serial.print(" (");
    Serial.print(AudioProcessorUsageMax())
    Serial.print("),  Mem = ");
    Serial.print(AudioMemoryUsage());
    Serial.print(" (");
    Serial.print(AudioMemoryUsageMax());
    Serial.println(")");
    last_time = millis();
  }

  //mixGain(int dB1, int dB2, double BW, d
  mixer1.gain(0,mixGain(0, result125L, 125
  mixer1.gain(1,mixGain(result125L, result
  mixer1.gain(2,mixGain(result125L, result
  mixer1.gain(3,mixGain(0, result250L, 125

  mixer2.gain(0,mixGain(result250L, result
  mixer2.gain(1,mixGain(result250L, result
  mixer2.gain(2,mixGain(result250L, result
  mixer2.gain(3,mixGain(result500L, result

  mixer3.gain(0,mixGain(result500L, result
  mixer3.gain(1,mixGain(result500L, result
  mixer3.gain(2,mixGain(result1000L, resul
  mixer3.gain(3,mixGain(result1000L, resul

  mixer4.gain(0,mixGain(result1000L, resul
  mixer4.gain(1,mixGain(result2000L, resul
  mixer4.gain(2,mixGain(result2000L, resul
  mixer4.gain(3,mixGain(result2000L, resul

  mixer5.gain(0,mixGain(result4000L, resul
  mixer5.gain(1,mixGain(result4000L, resul
  mixer5.gain(2,mixGain(result4000L, resul
  ////////////////////////////////////
  mixer5.gain(3,mixGain(0, result125R, 125
```

```
573  void playFile(char* filename)
574  {
575    Serial.print("Playing file: ");
576    Serial.println(filename);
577
578    // Start playing the file.  This sketch continues to
579    // run while the file plays.
580    playSdWav1.play(filename);
581
582    // A brief delay for the library read WAV info
583    delay(5);
584
585    // Simply wait for the file to finish playing.
586    while (playSdWav1.isPlaying()) {
587      float vol = analogRead(1);
588      vol = vol / 1024;
589      sgtl5000_1.volume(vol);
590    }
591  }
```

```
553  void calibration(){
554    int yesButtonState=digitalRead(yesButton);
555    int noButtonState=digitalRead(noButton);
556
557    while(yesButtonState==1 && noButtonState==1){
558      yesButtonState=digitalRead(yesButton);
559      noButtonState=digitalRead(noButton);
560
561      Serial.print("Calibrating...");
562      Serial.print("\n");
563
564      sgtl5000_1.volume(.9);
565      playFile("1234.WAV");
566      delay(1000);
567    };
568    Serial.print("\n");
569    Serial.print("Calibration complete.");
570    digitalWrite(ledPin, HIGH);
571  }
```

```
520  double mixGain(int dB1, int dB2, double BW, double point){
521
522    //Calculating dB loss based on linear relationship
523    double mix = ((point*(dB2-dB1))/BW)+dB1;
524
525    //Kneepoints
526    if(mix<=10 && mix>=0){
527      mix=mix-20;
528    }
529    if(mix<=20 && mix>=10){
530      mix=mix-20;
531    }
532    if(mix<=30 && mix>20){
533      mix=mix;
534    }
535    if(mix<=40 && mix>30){
536      mix=mix-2.5;
537    }
538    if(mix<=50 && mix>=40){
539      mix=mix-10;
540    }
541    if(mix<=70 && mix>=50){
542      mix=mix-25;
543    }
544
545    mix=
546    //Converion from dB to mixer coefficient
547    mix=0.4637*pow(2.71828, mix*0.1277);
548    Serial.println(mix);
549    return mix;
550    delay(2000);
551  }
```


untitled
UNREGISTERED
11-26V2
untitled
untitled
untitled
while (1) {
Serial.println("Unable to access the
delay(500);
}
}
////////////////
calibration();//
////////////////
delay(1000);
digitalWrite(ledPin, LOW);
int yesButtonState=digitalRead(yesButton
int noButtonState=digitalRead(noButton);
numberOfTests=0;
////////////////////////////////////
while(numberOfTests<4){
digitalWrite(ledPin, LOW);
int i=0;
int j=0;
char* soundFiles[14][9]={
{"1250L.wav", "12510L.wav", "12520L.
{"2500L.wav", "25010L.wav", "25020L.
{"5000L.wav", "50010L.wav", "50020L.
{"10000L.wav", "100010L.wav", "10002
{"20000L.wav", "200010L.wav", "20002
{"40000L.wav", "400010L.wav", "40002
{"80000L.wav", "800010L.wav", "80002
{"1250R.wav", "12510R.wav", "12520R.
{"2500R.wav", "25010R.wav", "25020R.
{"5000R.wav", "50010R.wav", "50020R.
{"10000R.wav", "100010R.wav", "10002
{"20000R.wav", "200010R.wav", "20002
{"40000R.wav", "400010R.wav", "40002
{"80000R.wav", "800010R.wav", "80002
};
while(i<=13){
digitalWrite(ledPin, LOW);
yesButtonState=digitalRead(yesButton
noButtonState=digitalRead(noButton);
j=0;
while(j<=8){
digitalWrite(ledPin, LOW);
if(i==0 && j==8){
digitalWrite(ledPin, HIGH);
originalResultsLeft[numberOfTest
i++;
j=0;
};
if(i==7 && j==8){
digitalWrite(ledPin, HIGH);
i++;
j=0;
originalResultsRight[numberOfTes
};
yesButtonState=digitalRead(yesButt
noButtonState=digitalRead(noButton
playFile(soundFiles[i][j]);
Serial.print(i);
Serial.print(" ");
Serial.print(j);
AudioFilterBiquad biquad10;
AudioFilterBiquad biquad35;
AudioFilterBiquad biquad8;
AudioFilterBiquad biquad26;
AudioFilterBiquad biquad14;
AudioFilterBiquad biquad19;
AudioFilterBiquad biquad15;
AudioFilterBiquad biquad13;
AudioFilterBiquad biquad12;
AudioFilterBiquad biquad7;
AudioFilterBiquad biquad24;
AudioFilterBiquad biquad11;
AudioFilterBiquad biquad5;
AudioFilterBiquad biquad16;
AudioFilterBiquad biquad6;
AudioFilterBiquad biquad23;
AudioMixer4 mixer6;
AudioMixer4 mixer8;
AudioMixer4 mixer10;
AudioMixer4 mixer9;
AudioMixer4 mixer7;
AudioMixer4 mixer4;
AudioMixer4 mixer1;
AudioMixer4 mixer3;
AudioMixer4 mixer5;
AudioMixer4 mixer2;
AudioMixer4 mixer11;
AudioMixer4 mixer13;
AudioMixer4 mixer12;
AudioMixer4 mixer14;
AudioMixer4 mixer15;
AudioOutputI2S i2s2;
AudioPlaySdWav playSdWav1;
AudioConnection patchCord1(i2s1,
AudioConnection patchCord2(i2s1,
AudioConnection patchCord3(i2s1,
AudioConnection patchCord4(i2s1,
AudioConnection patchCord5(i2s1,
AudioConnection patchCord6(i2s1,
AudioConnection patchCord7(i2s1,
AudioConnection patchCord8(i2s1,
AudioConnection patchCord9(i2s1,
AudioConnection patchCord10(i2s1,
AudioConnection patchCord11(i2s1,
AudioConnection patchCord12(i2s1,
AudioConnection patchCord13(i2s1,
AudioConnection patchCord14(i2s1,
AudioConnection patchCord15(i2s1,
AudioConnection patchCord16(i2s1,
AudioConnection patchCord17(i2s1,
AudioConnection patchCord18(i2s1,
AudioConnection patchCord19(i2s1,
AudioConnection patchCord20(i2s1,
AudioConnection patchCord21(i2s1,
AudioConnection patchCord22(i2s1,
AudioConnection patchCord23(i2s1,
AudioConnection patchCord24(i2s1,
AudioConnection patchCord25(biqua
AudioConnection patchCord26(biqua
AudioConnection patchCord27(biqua
AudioConnection patchCord28(biqua
AudioConnection patchCord29(biqua
AudioConnection patchCord30(biqua
AudioConnection patchCord31(biqua
AudioConnection patchCord32(biqua
AudioConnection patchCord33(biqua
AudioConnection patchCord34(biqua
AudioConnection patchCord35(biqua
AudioConnection patchCord36(biqua
AudioConnection patchCord37(biqua
AudioConnection patchCord38(biqua
AudioConnection patchCord39(biqua
AudioConnection patchCord40(biqua
AudioConnection patchCord41(biqua
AudioConnection patchCord42(biqua
AudioConnection patchCord43(biqua
AudioConnection patchCord44(biqua
AudioConnection patchCord45(biqua
AudioConnection patchCord46(biqua
AudioConnection patchCord47(biqua
AudioConnection patchCord48(biqua
AudioConnection patchCord49(biqua
AudioConnection patchCord50(biqua
AudioConnection patchCord51(biqua
AudioConnection patchCord52(biqua
AudioConnection patchCord53(biqua
AudioConnection patchCord54(biqua
AudioConnection patchCord55(biqua
AudioConnection patchCord56(biqua
AudioConnection patchCord57(biqua
AudioConnection patchCord58(biqua
AudioConnection patchCord59(biqua
AudioConnection patchCord60(biqua
AudioConnection patchCord61(biqua
AudioConnection patchCord62(biqua
AudioConnection patchCord63(mixer
AudioConnection patchCord64(mixer
AudioConnection patchCord65(mixer
AudioConnection patchCord66(mixer
AudioConnection patchCord67(mixer
AudioConnection patchCord68(mixer
AudioConnection patchCord69(mixer
AudioConnection patchCord70(mixer
AudioConnection patchCord71(mixer
AudioConnection patchCord72(mixer
AudioConnection patchCord73(mixer
AudioConnection patchCord74(mixer
AudioConnection patchCord75(mixer
AudioConnection patchCord76(mixer
AudioConnection patchCord92(playS
AudioConnection patchCord93(playS
AudioConnection patchCord77(mixer
AudioConnection patchCord78(mixer
AudioControlSGTL5000 sgtl5000_1;
// GUItool: end automatically generated co
const int myInput = AUDIO_INPUT_LINEIN;
void setup() {
Serial.begin(9600);
pinMode(yesButton, INPUT);
pinMode(noButton, INPUT);
pinMode(ledPin, OUTPUT);
// Audio connections require memory to w
// detailed information, see the MemoryA
sgtl5000_1.enable();
sgtl5000_1.volume(0.5);
AudioMemory(150);
SPI.setMOSI(7);
SPI.setSCK(14);
if (!(SD.begin(10))){
Line 68, Column 25
Spaces: 2
C++

11-26V2

```
    mixer7.gain(2,mixGain(result500L, resul
    mixer7.gain(3,mixGain(result500L, resul

    mixer8.gain(0,mixGain(result500L, resul
    mixer8.gain(1,mixGain(result1000L, resu
    mixer8.gain(2,mixGain(result1000L, resu
    mixer8.gain(3,mixGain(result1000L, resu

    mixer9.gain(0,mixGain(result2000L, resu
    mixer9.gain(1,mixGain(result2000L, resu
    mixer9.gain(2,mixGain(result2000L, resu
    mixer9.gain(3,mixGain(result4000L, resu

    mixer10.gain(0,mixGain(result4000L, res
    mixer10.gain(1,mixGain(result4000L, res


}

double mixGain(int dB1, int dB2, double B
  double mix = ((point*(dB2-dB1))/BW)+dB1
  mix=pow(10, mix/20);
  mix = (mix-0.1276)/1.7754;
  Serial.println(mix);
  return mix;
}

void calibration(){
  int yesButtonState=digitalRead(yesButto
  int noButtonState=digitalRead(noButton)

  while(yesButtonState==1 && noButtonStat
    yesButtonState=digitalRead(yesButton)
    noButtonState=digitalRead(noButton);

    Serial.print("Calibrating...");
    Serial.print("\n");

    sgtl5000_1.volume(.9);
    playFile("1234.WAV");
    delay(1000);
  };
  Serial.print("\n");
  Serial.print("Calibration complete.");
  digitalWrite(ledPin, HIGH);
}

void playFile(char* filename)
{
  Serial.print("Playing file: ");
  Serial.println(filename);

  // Start playing the file.  This sketch
  // run while the file plays.
  playSdWav1.play(filename);

  // A brief delay for the library read W
  delay(5);

  // Simply wait for the file to finish p
  while (playSdWav1.isPlaying()) {
    float vol = analogRead(1);
    vol = vol / 1024;
    sgtl5000_1.volume(vol);
  }
}
```

untitled

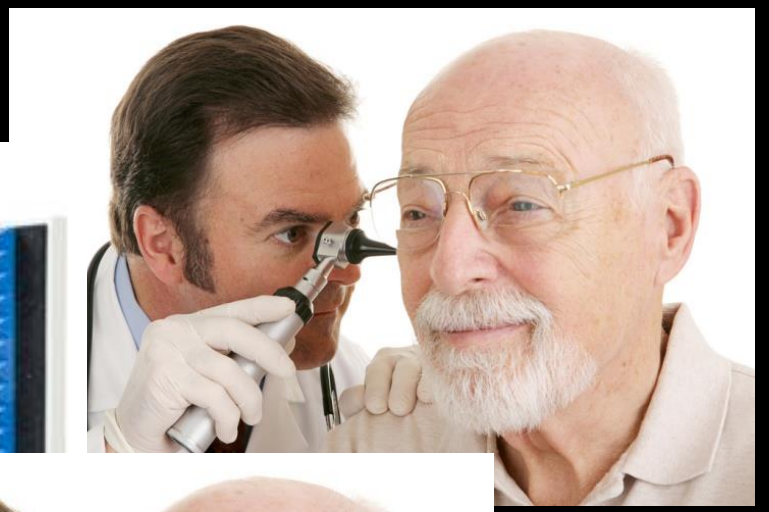
```
Serial.print("\n");
  ////////////////////////////////////
        if(yesButtonState==0 && noButtonSt
          digitalWrite(ledPin, HIGH);//abl
          delay(1000);
          //Results being recorded//
          if(i<=6){
            originalResultsLeft[numberOfTe
          }

          else{
            originalResultsRight[numberOfT
          };
          j=0;
          i++;
        };
  ////////////////////////////////////
        if(yesButtonState==1 && noButtonSt
          digitalWrite(ledPin, HIGH);
          delay(1000);
          j++;
        };
  ////////////////////////////////////

        if(yesButtonState==0 && noButtonSt
          digitalWrite(ledPin, HIGH);
          delay(1000);
          i = 14;
          j = 9;
        };
  ////////////////////////////////////
        if(j==9){
          digitalWrite(ledPin, HIGH);
          //Results being recorded//
          if(i<=5){
            originalResultsLeft[numberOfTe
          }
          else{
            originalResultsRight[numberOfT
          }
          i++;
          j=0;
        };

        if(i>=14){
          j=9;
        };
  //    delay(100);
      };
    };
    numberOfTests=numberOfTests+1;
    Serial.print(numberOfTests);
  };


  for (int i = 0; i < 7; ++i){
    leftResultsFinal[i]=(originalResultsLe
    originalResultsLeft[2][i]+originalResu

    //leftResultsFinal[i]=leftResultsFinal


    rightResultsFinal[i]=(originalResultsR
    originalResultsRight[2][i]+originalRes

    //rightResultsFinal[i]=rightResultsFin
  };
```

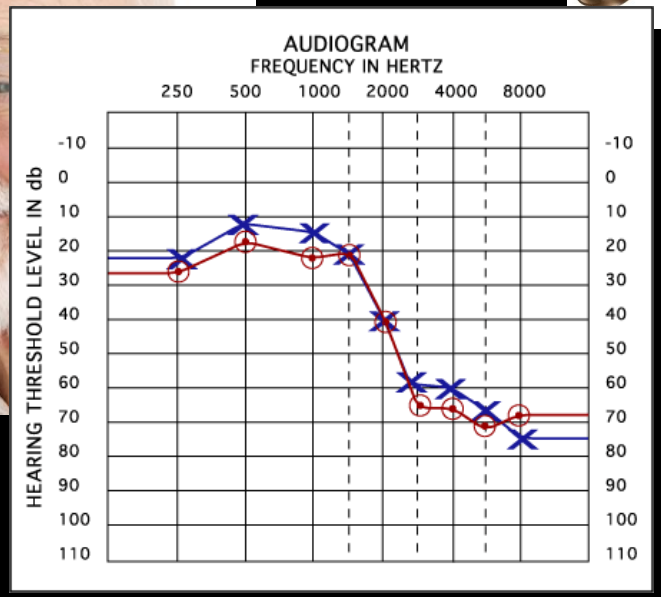
untitled

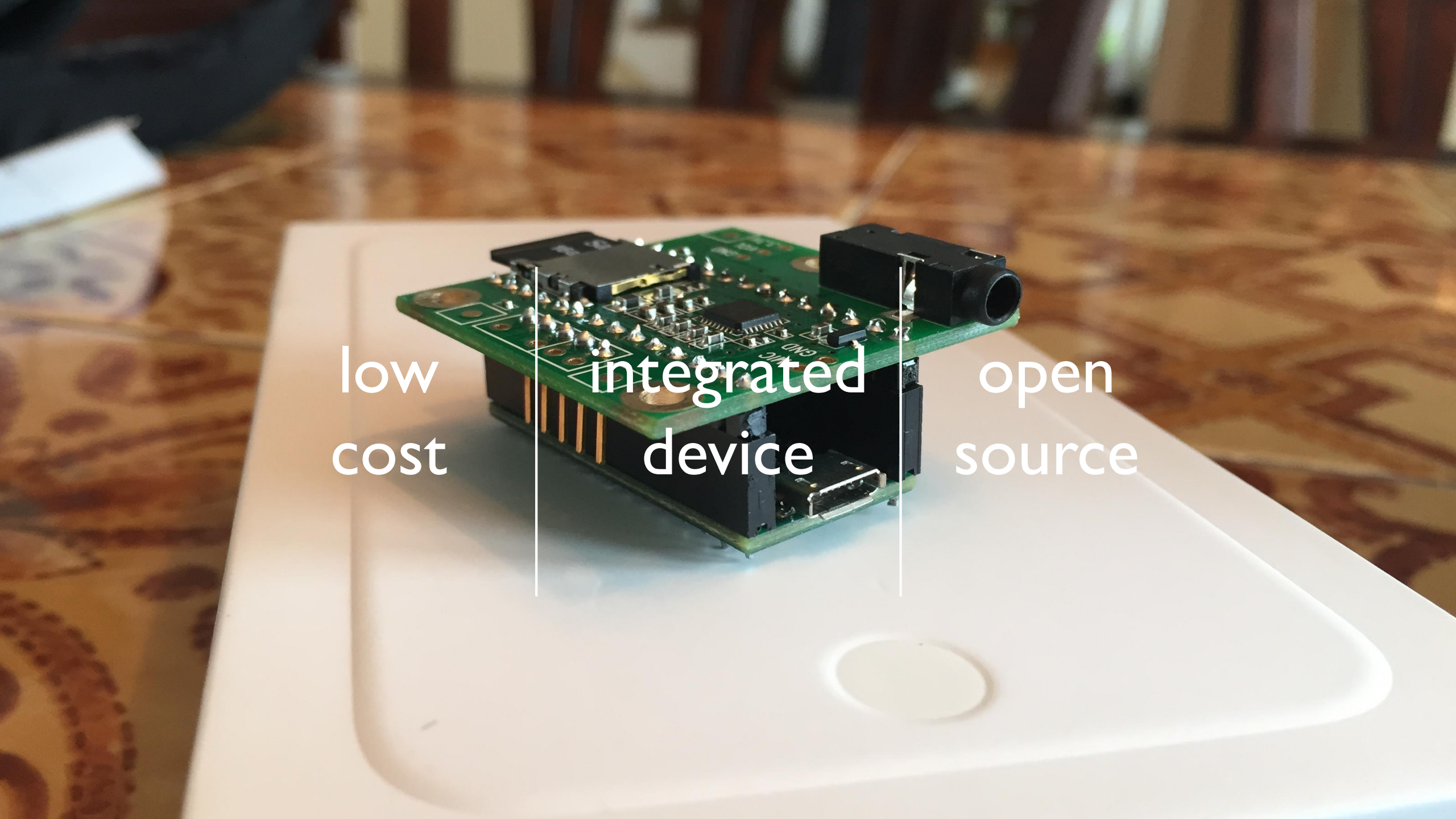
```
//Outputs left side results to the user
  Serial.print("Left side results: ");
  for (int i = 0; i < 7; ++i){
    Serial.print(leftResultsFinal[i]);
    Serial.print(", ");
  }
  //Stores the values of the results to pa
  result125L = leftResultsFinal[0];
  result250L = leftResultsFinal[1];
  result500L = leftResultsFinal[2];
  result1000L = leftResultsFinal[3];
  result2000L = leftResultsFinal[4];
  result4000L = leftResultsFinal[5];
  result8000L = leftResultsFinal[6];
  Serial.print("\n");


  //Outputs right side results to the user
  Serial.print("Right side results: ");
  for (int i = 0; i < 7; ++i){
    Serial.print(rightResultsFinal[i]);
    Serial.print(", ");
  }
  //Stores the values of the results to pa
  result125R = rightResultsFinal[0];
  result250R = rightResultsFinal[1];
  result500R = rightResultsFinal[2];
  result1000R = rightResultsFinal[3];
  result2000R = rightResultsFinal[4];
  result4000R = rightResultsFinal[5];
  result8000R = rightResultsFinal[6];
  Serial.print("\n");
  Serial.print("\n");


  Serial.print("TEST COMPLETE");
  Serial.print("\n");



  //format
  //biqaud.setBandpass(stage, center frequ
  biquad1.setBandpass(0, 125, 125/28.9);
  biquad2.setBandpass(0, 157.5, 157.5/36.5
  biquad3.setBandpass(0, 198.4, 198.4/45.9
  biquad4.setBandpass(0, 250, 250/57.9);

  biquad5.setBandpass(0, 315, 315/73);
  biquad6.setBandpass(0, 396.9, 396.9/88.8
  biquad7.setBandpass(0, 500, 500/115.8);
  biquad8.setBandpass(0, 630, 630/145.9);

  biquad9.setBandpass(0, 793.7, 793.7/183.
  biquad10.setBandpass(0, 1000, 1000/231.6
  biquad11.setBandpass(0, 1259.9, 1259.9/2
  biquad12.setBandpass(0, 1587.4, 1587.4/3

  biquad13.setBandpass(0, 2000, 2000/463.1
  biquad14.setBandpass(0, 2519.8, 2519.8/5
  biquad15.setBandpass(0, 3174.8, 3174.8/7
  biquad16.setBandpass(0, 4000, 4000/926.2

  biquad17.setBandpass(0, 5039.7, 5039.7/1
  biquad18.setBandpass(0, 6349.6, 6349.6/1
  biquad19.setBandpass(0, 8000, 8000/1852.
  ////////////////////////////////////
  biquad20.setBandpass(0, 125, 125/28.9);
```

untitled

```
biquad21.setBandpass(0, 157.5, 157.5/36.5)
  biquad22.setBandpass(0, 198.4, 198.4/45.
  biquad23.setBandpass(0, 250, 250/57.9);
  biquad24.setBandpass(0, 315, 315/73);

  biquad25.setBandpass(0, 396.9, 396.9/88.
  biquad26.setBandpass(0, 500, 500/115.8);
  biquad27.setBandpass(0, 630, 630/145.9);
  biquad28.setBandpass(0, 793.7, 793.7/183

  biquad29.setBandpass(0, 1000, 1000/231.6
  biquad30.setBandpass(0, 1259.9, 1259.9/2
  biquad31.setBandpass(0, 1587.4, 1587.4/3
  biquad32.setBandpass(0, 2000, 2000/463.1

  biquad33.setBandpass(0, 2519.8, 2519.8/5
  biquad34.setBandpass(0, 3174.8, 3174.8/7
  biquad35.setBandpass(0, 4000, 4000/926.2
  biquad36.setBandpass(0, 5039.7, 5039.7/1

  biquad37.setBandpass(0, 6349.6, 6349.6/1
  biquad38.setBandpass(0, 8000, 8000/1852.
}

unsigned long last_time = millis();

void loop(){
  sgtl5000_1.inputSelect(myInput);
  if (millis() - last_time >= 2500) {
    Serial.print("Proc = ");
    Serial.print(AudioProcessorUsage());
    Serial.print(" (");
    Serial.print(AudioProcessorUsageMax())
    Serial.print("),  Mem = ");
    Serial.print(AudioMemoryUsage());
    Serial.print(" (");
    Serial.print(AudioMemoryUsageMax());
    Serial.println(")");
    last_time = millis();
  }

  //mixGain(int dB1, int dB2, double BW, d
  mixer1.gain(0,mixGain(0, result125L, 125
  mixer1.gain(1,mixGain(result125L, result
  mixer1.gain(2,mixGain(result125L, result
  mixer1.gain(3,mixGain(0, result250L, 125

  mixer2.gain(0,mixGain(result250L, result
  mixer2.gain(1,mixGain(result250L, result
  mixer2.gain(2,mixGain(result250L, result
  mixer2.gain(3,mixGain(result500L, result

  mixer3.gain(0,mixGain(result500L, result
  mixer3.gain(1,mixGain(result500L, result
  mixer3.gain(2,mixGain(result1000L, resul
  mixer3.gain(3,mixGain(result1000L, resul

  mixer4.gain(0,mixGain(result1000L, resul
  mixer4.gain(1,mixGain(result2000L, resul
  mixer4.gain(2,mixGain(result2000L, resul
  mixer4.gain(3,mixGain(result2000L, resul

  mixer5.gain(0,mixGain(result4000L, resul
  mixer5.gain(1,mixGain(result4000L, resul
  mixer5.gain(2,mixGain(result4000L, resul
  ////////////////////////////////////
  mixer5.gain(3,mixGain(0, result125R, 125
```




AUDIOGRAM
FREQUENCY IN HERTZ
250
500
1000
2000
4000
8000
HEARING THRESHOLD LEVEL IN db
-10
0
10
20
30
40
50
60
70
80
90
100
110

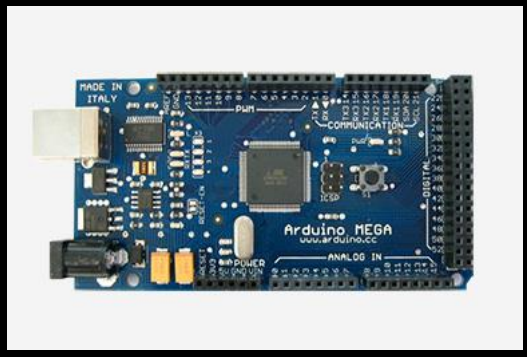

low
cost
integrated
device
open
source

So what?

YOU CAN DO
ANYTHING WITH THE
INTERNET

Learn how to program

ARDUINO
OPEN-SOURCE
COMMUNITY



MAKE SOMETHING


Amplify life.