LASER HAARP

レーザーハープ

団体

数学研究同好会・電子工作部門

第63回 星月祭

2024. 6/7~8

レーザーハープとは？

WHAT IS LASER HARP?

A.

電子楽器の一種

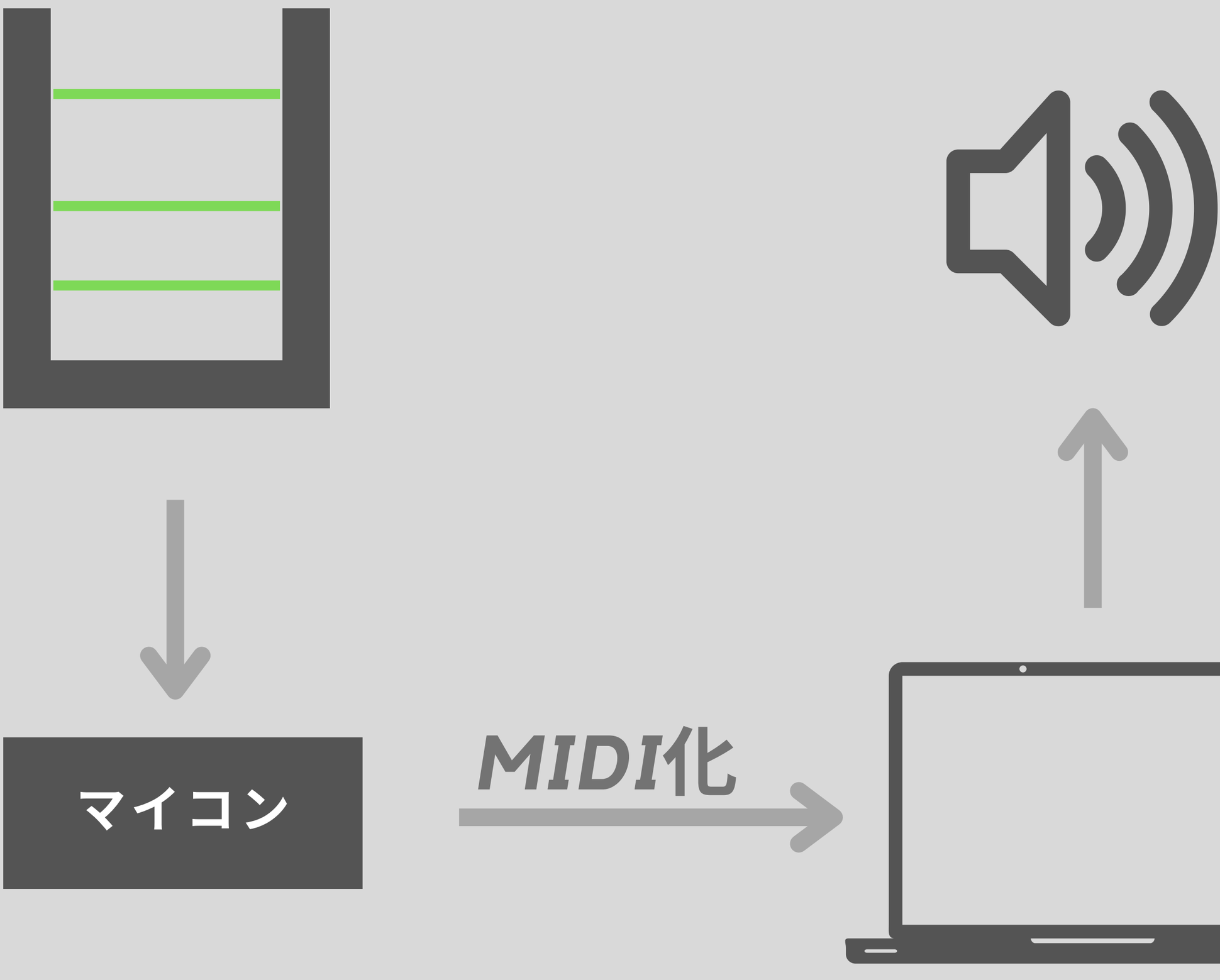
IT IS AN ELECTRONIC INSTRUMENT !

_レーザーハープとは電子楽器の一種であり、テクノやテクノポップなどで用いられる。
_主な使用者としてP-MODELの平沢進などが有名である

レーザーハープの仕組み

LASER HARPS STRUCTURE

_図として表すと以下の通り

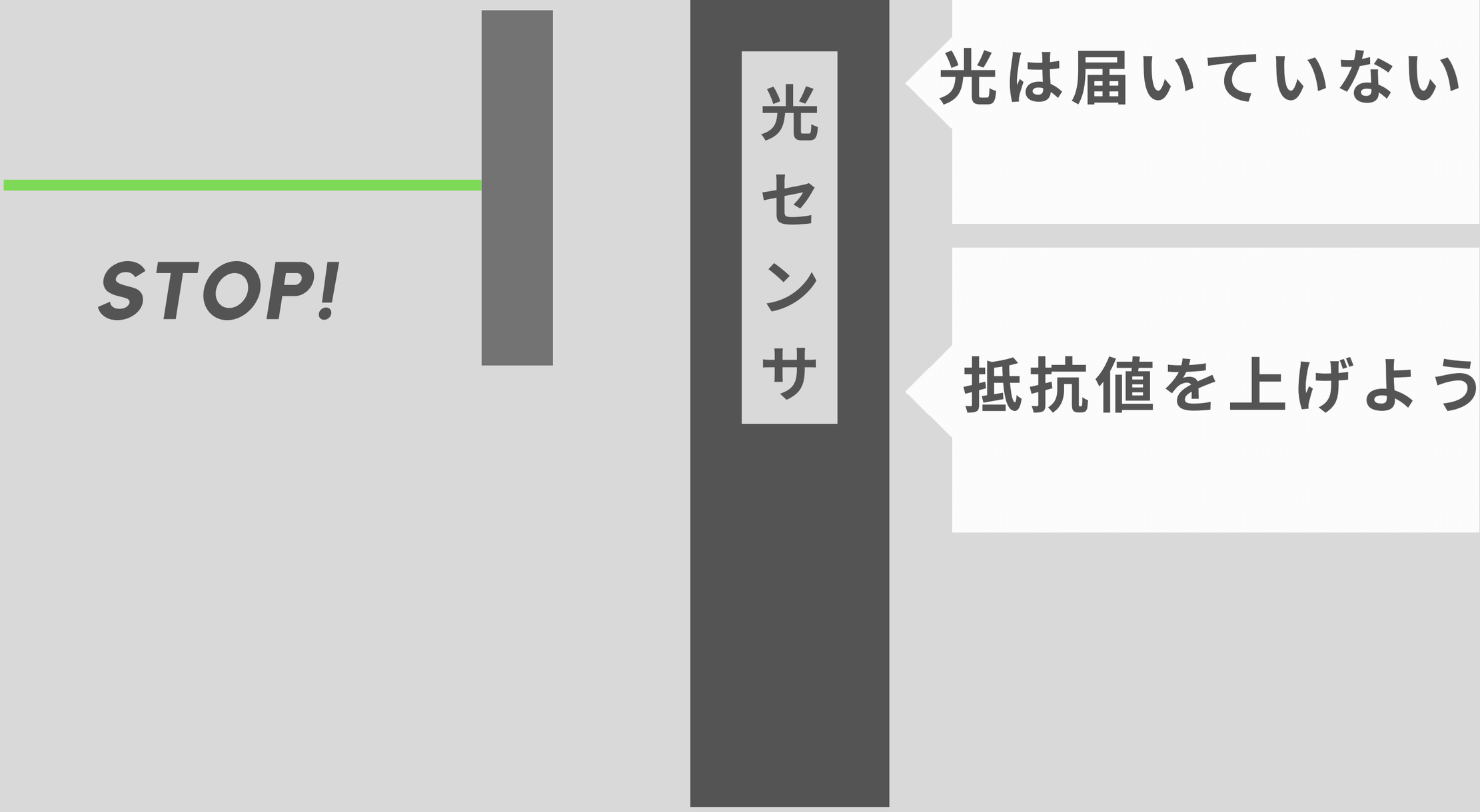


POINT

1. レーザーハープ内センサ
2. マイコンでのMIDI信号への変換
3. パソコン上でMIDI信号を音へ変換

レーザーハープ
内蔵センサ

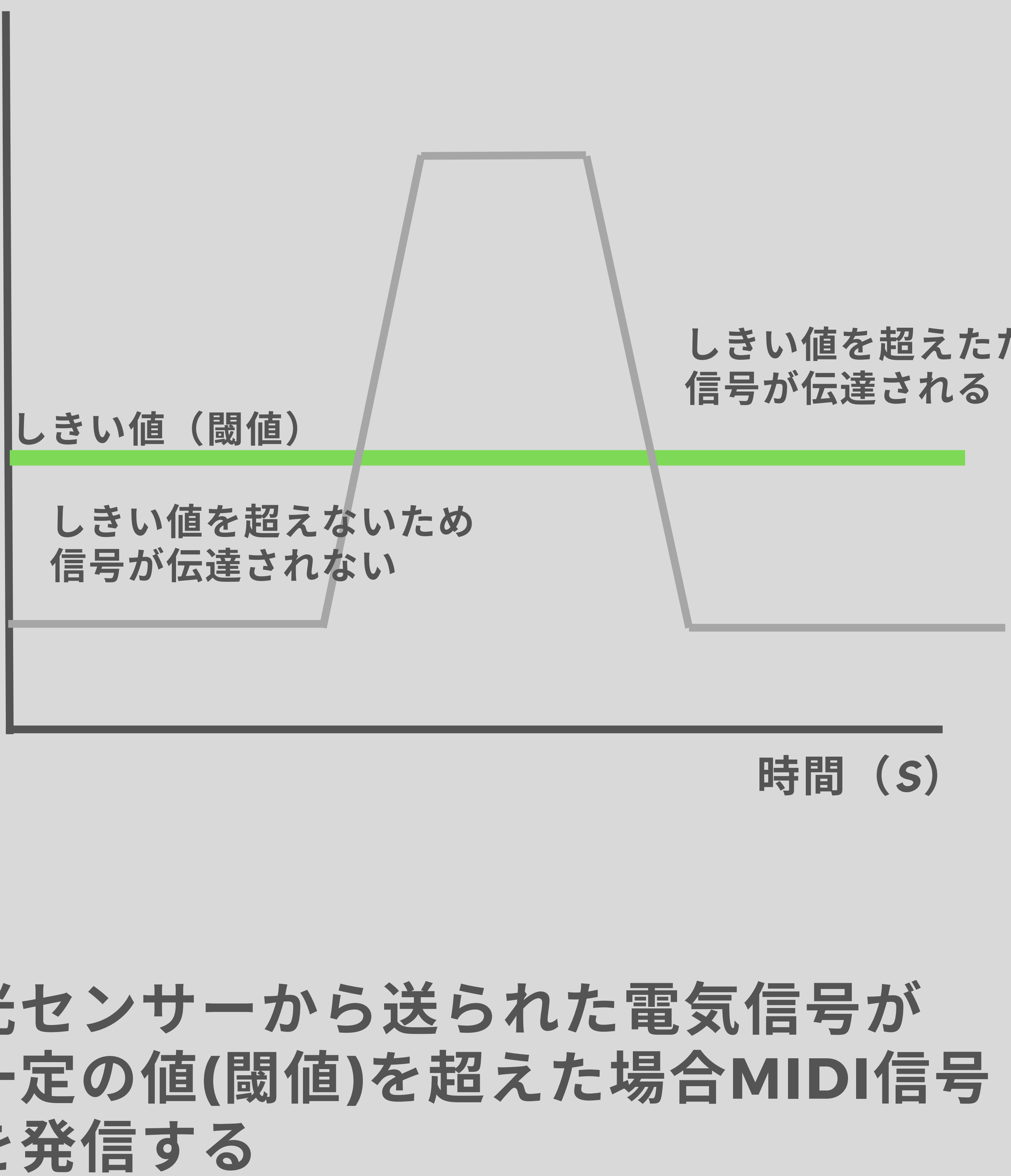
LASER HARPS CENSOR



光センサーを使いレーザーが弾かれた
行為を観測し電圧の強弱に変換する

MIDIへの変換

MIDI CONVERSION



光センサーから送られた電気信号が
一定の値(閾値)を超えた場合MIDI信号
を発信する

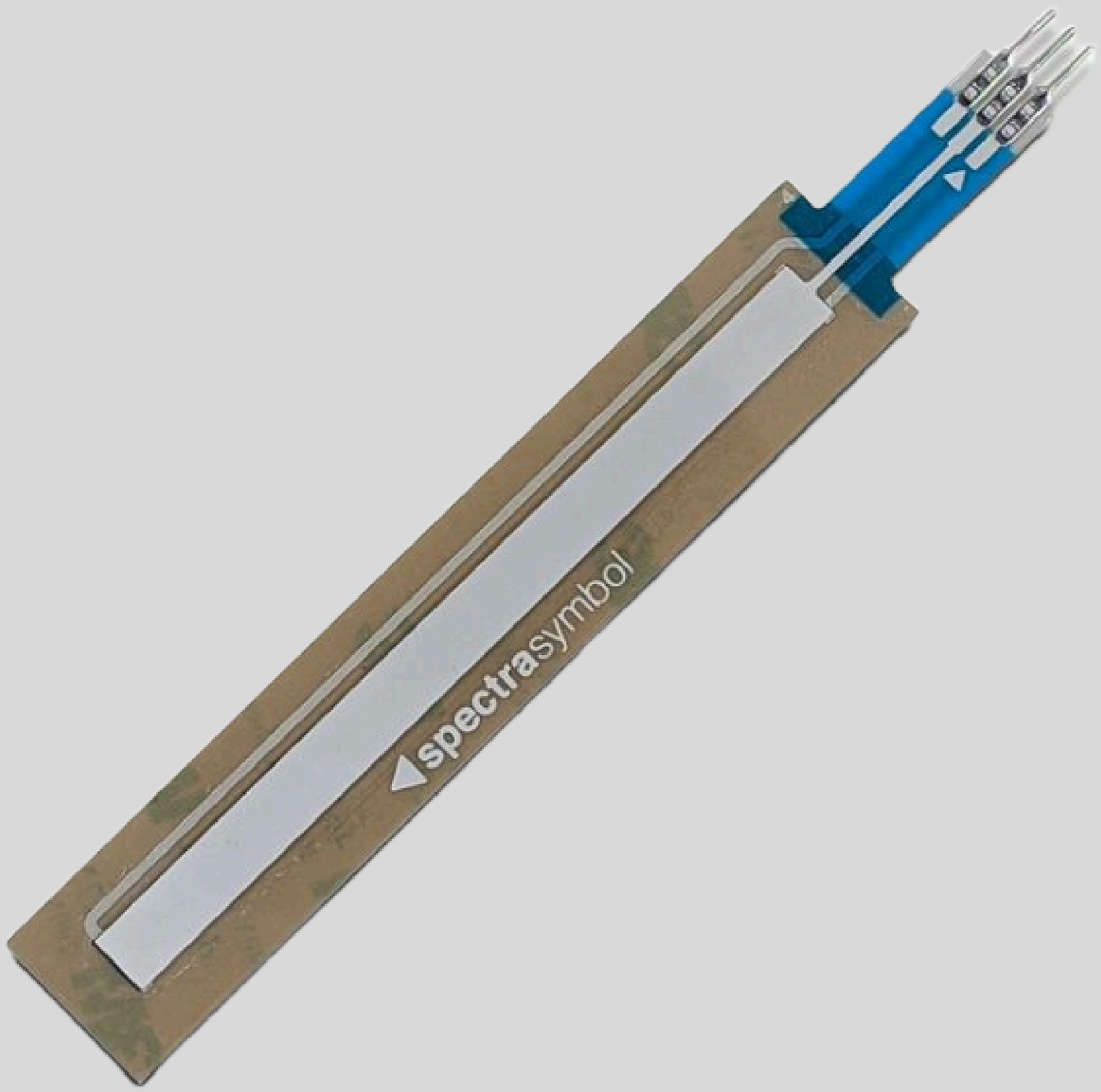
音色変更とオクターブ

音色変更

MIDIにはトラックという概念があります。
ボタンを押すたびにハープは別のトラックにノート(信号)を送信します。
トラックごとに楽器をPC側で指定することで音色の変更を実現しています。

オクターブ変更

位置センサの値によって±2オクターブの範囲の音を出すことができます。

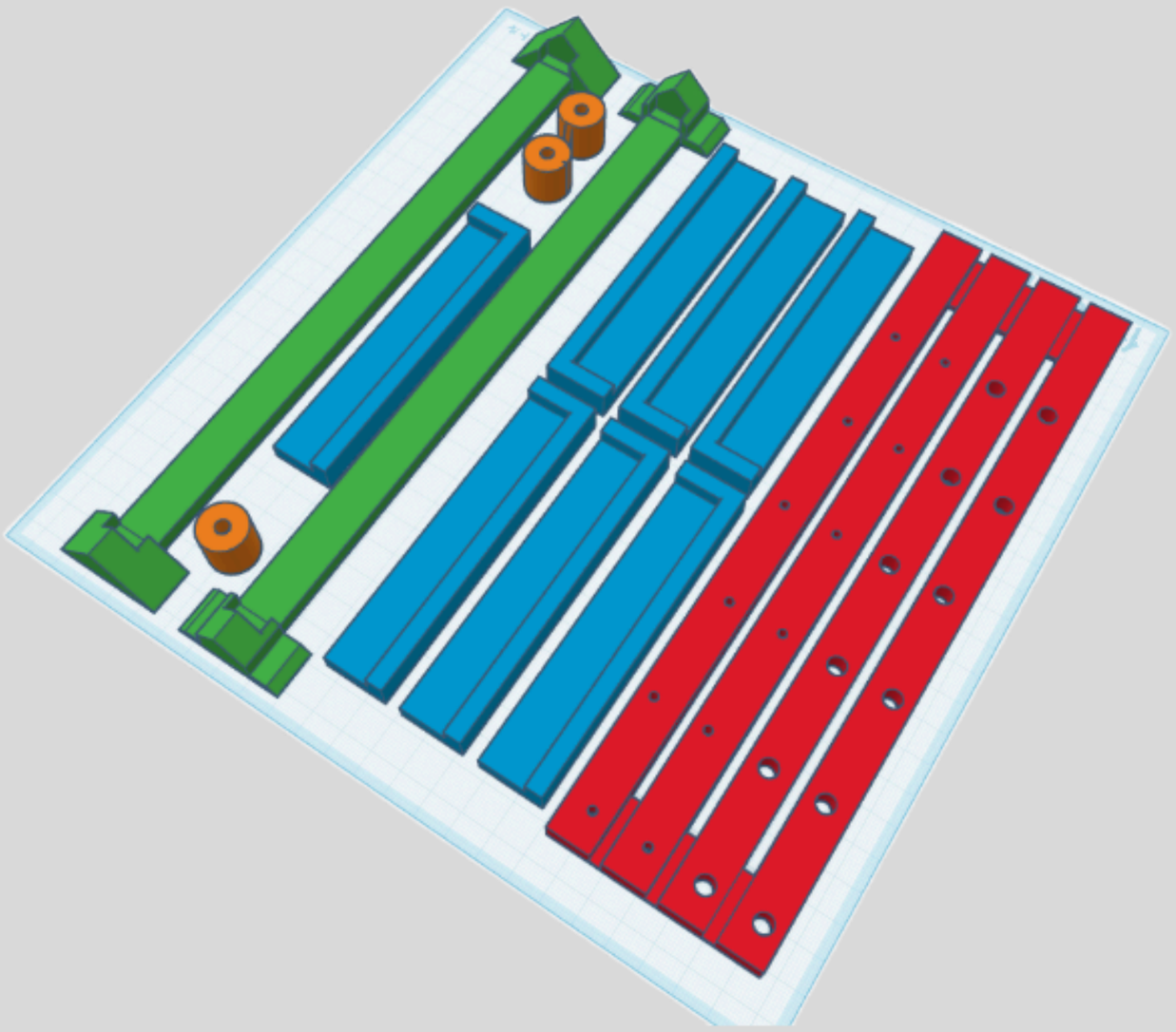


3Dモデリング

WEB上で動くTINKERCAD というものを利用して3Dモデルを作成しました。
BAMBU LAB P1Sという比較的グレードの高い3Dプリンターを先生からお借りして作成しました。

かなり精度は高いですが、穴部分のサイズや組み合わせる部分はノズルの太さを考慮して設計する必要がありました。

また、FDM3Dプリンターの性質上、宙に浮いているようなものの積層は苦手であったり、なるべくノズルの動く量を少なくするように設計するなど工夫しています。



レーザーハープ
プログラム

ライブラリインクルード、変数定義

```
#INCLUDE <MIDIUSB.H>

CONST INT ANALOGPINS[] = {A0, A1, A2, A3, A4, A5};
CONST BYTE MIDINOTES[] = {60, 62, 64, 65, 67, 69}; // ド, レ, ミ, ファ, ソ, ラ
BOOL NOTESTATE[6] = {FALSE, FALSE, FALSE, FALSE, FALSE, FALSE};

CONST INT BUTTONPIN = 13;
INT CURRENTCHANNEL = 0;
INT PREVIOUSCHANNEL = 0;
BOOL LASTBUTTONSTATE = HIGH;

CONST INT LEDPINS[] = {2, 3, 4, 5, 6, 7, 8};
CONST INT PATTERNS[10][7] = {
  {1, 1, 1, 1, 0, 1, 1},
  {1, 1, 0, 0, 0, 0, 0},
  {1, 0, 1, 0, 1, 1, 1},
  {1, 1, 1, 0, 1, 0, 1},
  {1, 1, 0, 1, 1, 0, 0},
  {0, 1, 1, 1, 1, 0, 1},
  {0, 1, 1, 1, 1, 1, 1},
  {1, 1, 1, 0, 0, 0, 0},
  {1, 1, 1, 1, 1, 1, 1},
  {1, 1, 1, 1, 0, 1, 1}
};

INT CURRENTOCTAVESHIFT = 0;
INT PREVIOUSOCTAVESHIFT = 0;
```

ここではPCにMIDI信号を送信するためのライブラリと音程の設定、７セグメントLEDに表示するパターンなどを定義しています。

セットアップ関数

```
VOID SETUP() {
 SERIAL.BEGIN(9600);
 FOR (INT I = 0; I < 7; I++) {
 PINMODE(LEDPINS[I], OUTPUT);
 }
 PINMODE(BUTTONPIN, INPUT_PULLUP);
 DISPLAYNUMBER(CURRENTCHANNEL);
 PREVIOUSOCTAVESHIFT = GETOCTAVESHIFT(); // 初期値設定
}
```

ピンの設定を行い、デバッグ情報送信用のシリアル通信を開始します。

メインの処理

```
VOID LOOP() {
  HANDLEBUTTON();
  HANDLESENSORS();
  PRINTSENSORVALUES();
  DELAY(20);
}

VOID HANDLEBUTTON() {
  BOOL CURRENTBUTTONSTATE = DIGITALREAD(BUTTONPIN);
  IF (LASTBUTTONSTATE == HIGH && CURRENTBUTTONSTATE == LOW) {
    ALLNOTESOFF(PREVIOUSCHANNEL, CURRENTOCTAVESHIFT);
    CURRENTCHANNEL = (CURRENTCHANNEL + 1) % 10;
    DISPLAYNUMBER(CURRENTCHANNEL);
    SERIAL.PRINT("チャンネル変更: ");
    SERIAL.PRINTLN(CURRENTCHANNEL + 1);
    PREVIOUSCHANNEL = CURRENTCHANNEL;
    DELAY(200);
  }
  LASTBUTTONSTATE = CURRENTBUTTONSTATE;
}

VOID HANDLESENSORS() {
  CURRENTOCTAVESHIFT = GETOCTAVESHIFT();

  // オクターブが変更された場合、以前のノートをすべてNOTEOFF
  IF (CURRENTOCTAVESHIFT != PREVIOUSOCTAVESHIFT) {
    ALLNOTESOFF(CURRENTCHANNEL, PREVIOUSOCTAVESHIFT);
    PREVIOUSOCTAVESHIFT = CURRENTOCTAVESHIFT;
    SERIAL.PRINT("オクターブ変更: ");
    SERIAL.PRINTLN(CURRENTOCTAVESHIFT);
  }

  FOR (INT I = 0; I < 6; I++) {
    INT VAL = ANALOGREAD(ANALOGPINS[I]);
    BYTE SHIFTEDNOTE = MIDINOTES[I] + CURRENTOCTAVESHIFT * 12;
    SHIFTEDNOTE = CONSTRAIN(SHIFTEDNOTE, 0, 127);

    IF (VAL < 50 && !NOTESTATE[I]) {
      SENDNOTEON(CURRENTCHANNEL + 1, SHIFTEDNOTE, 127);
      NOTESTATE[I] = TRUE;
    } ELSE IF (VAL >= 50 && NOTESTATE[I]) {
      SENDNOTEOFF(CURRENTCHANNEL + 1, SHIFTEDNOTE, 127);
      NOTESTATE[I] = FALSE;
    }
  }
}
INT GETOCTAVESHIFT() {
 INT VAL = ANALOGREAD(A11);
 IF (VAL > 740) RETURN 2;
 ELSE IF (VAL > 650) RETURN 1;
 ELSE IF (VAL > 300) RETURN 0;
 ELSE IF (VAL > 100) RETURN -1;
 ELSE RETURN -2;
}

VOID ALLNOTESOFF(INT CHANNEL, INT OCTAVESHIFT) {
 FOR (INT I = 0; I < 6; I++) {
 BYTE SHIFTEDNOTE = MIDINOTES[I] + OCTAVESHIFT * 12;
 SHIFTEDNOTE = CONSTRAIN(SHIFTEDNOTE, 0, 127);
 IF (NOTESTATE[I]) {
 SENDNOTEOFF(CHANNEL + 1, SHIFTEDNOTE, 0);
 NOTESTATE[I] = FALSE;
 }
 }
}
```

各種センサーの値、ボタンの情報などを取得し、MIDI信号をコンピューターに送信します。

デバッグ出力

```
VOID PRINTSENSORVALUES() {
  SERIAL.PRINT("SENSORS: ");
  FOR (INT I = 0; I < 6; I++) {
    SERIAL.PRINT("A");
    SERIAL.PRINT(I);
    SERIAL.PRINT("=");
    SERIAL.PRINT(ANALOGREAD(ANALOGPINS[I]));
    SERIAL.PRINT(", ");
  }
  SERIAL.PRINT("A11=");
  SERIAL.PRINTLN(ANALOGREAD(A11));
}

VOID SENDNOTEON(BYTE CHANNEL, BYTE PITCH, BYTE VELOCITY) {
  MIDIEVENTPACKET_T NOTEON = {0X09, BYTE(0X90 | ((CHANNEL - 1) & 0X0F)), PITCH, VELOCITY};
  MIDIUSB.SENDMIDI(NOTEON);
  MIDIUSB.FLUSH();
}

VOID SENDNOTEOFF(BYTE CHANNEL, BYTE PITCH, BYTE VELOCITY) {
  MIDIEVENTPACKET_T NOTEOFF = {0X08, BYTE(0X80 | ((CHANNEL - 1) & 0X0F)), PITCH, VELOCITY};
  MIDIUSB.SENDMIDI(NOTEOFF);
  MIDIUSB.FLUSH();
}

VOID DISPLAYNUMBER(INT NUM) {
  FOR (INT I = 0; I < 7; I++) {
    DIGITALWRITE(LEDPINS[I], PATTERNS[NUM][I] ? HIGH : LOW);
  }
}
```

センサーの値をシリアルでコンピューターに送信します。
また7セグLEDを制御するコードも含まれます。