Arduino – ArduiJ.

Descripción.

El presente trabajo expone la creación de un reproductor básico de melodías. Donde se intenta mostrar una interfaz amigable y entendible para un usuario. Proponiendo de esta forma, botones en la forma tradicional de un reproductor.

Este reproductor consta de un lcd, donde se inicia mostrando un mensaje de bienvenida, para luego derivar en un menú donde se puede seleccionar el modo esperado, estos modos son Manual, donde se puede ingresar mediante un puerto serie, letras separadas por coma las que posteriormente serán traducidas en notas para ser reproducidas, y el otro modo Automático, que consta de un menú predefinido de seis melodías alojadas en el código, donde el usuario puede desplegarse por el menú y seleccionar la que desea.

A medida que se van reproduciendo las melodías, tanto en el modo manual como en el automático, se pondrá en movimiento un juego de leds que varia en cada nota.

Componentes utilizados.

- Cables.
- Resistencias.
- Piezo.
- Lcd 16 x 2.
- Placa de pruebas.
- Botones.
- Leds.
- Potenciometro.
- Arudino UNO r3.

Código – Liquid Crystal.

Se utilizo la librería LiquidCrystal para el uso del LCD 16 x 2.

```
#include <LiquidCrystal.h>
```

Constructor.

```
LiquidCrystal lcd(rs, e, db4, db5, db6, db7);
```

Código – Constantes de frecuencia.

```
#define NOTE_B0  31
#define NOTE_C1  33
#define NOTE_CS1 35
#define NOTE_D1  37
#define NOTE_DS1 39
#define NOTE_E1  41
#define NOTE_F1  44
#define NOTE_FS1 46
#define NOTE_G1  49
#define NOTE_GS1 52
#define NOTE_A1  55
#define NOTE_AS1 58
#define NOTE_B1  62
#define NOTE_C2  65
#define NOTE_CS2 69
#define NOTE_D2  73
#define NOTE_DS2 78
#define NOTE_E2  82
#define NOTE_F2  87
#define NOTE_FS2 93
#define NOTE_G2  98
#define NOTE_GS2 104
#define NOTE_A2  110
#define NOTE_AS2 117
#define NOTE_B2  123
#define NOTE_C3  131
#define NOTE_CS3 139
#define NOTE_D3  147
#define NOTE_DS3 156
#define NOTE_E3  165
#define NOTE_F3  175
#define NOTE_FS3 185
#define NOTE_G3  196
#define NOTE_GS3 208
```

```
#define NOTE_A3  220
#define NOTE_AS3 233
#define NOTE_B3  247
#define NOTE_C4  262
#define NOTE_CS4 277
#define NOTE_D4  294
#define NOTE_DS4 311
#define NOTE_E4  330
#define NOTE_F4  349
#define NOTE_FS4 370
#define NOTE_G4  392
#define NOTE_GS4 415
#define NOTE_A4  440
#define NOTE_AS4 466
#define NOTE_B4  494
#define NOTE_C5  523
#define NOTE_CS5 554
#define NOTE_D5  587
#define NOTE_DS5 622
#define NOTE_E5  659
#define NOTE_F5  698
#define NOTE_FS5 740
#define NOTE_G5  784
#define NOTE_GS5 831
#define NOTE_A5  880
#define NOTE_AS5 932
#define NOTE_B5  988
#define NOTE_C6  1047
#define NOTE_CS6 1109
#define NOTE_D6  1175
#define NOTE_DS6 1245
#define NOTE_E6  1319
#define NOTE_F6  1397
#define NOTE_FS6 1480
#define NOTE_G6  1568
```

```
#define NOTE_GS6 1661
#define NOTE_A6  1760
#define NOTE_AS6 1865
#define NOTE_B6  1976
#define NOTE_C7  2093
#define NOTE_CS7 2217
#define NOTE_D7  2349
#define NOTE_DS7 2489
#define NOTE_E7  2637
#define NOTE_F7  2794
#define NOTE_FS7 2960
#define NOTE_G7  3136
#define NOTE_GS7 3322
#define NOTE_A7  3520
#define NOTE_AS7 3729
#define NOTE_B7  3951
#define NOTE_C8  4186
#define NOTE_CS8 4435
#define NOTE_D8  4699
#define NOTE_DS8 4978
```

Se definen los nombres de las notas musicales basados en frecuencias que se utilizaran posteriormente,.

Código – constantes de pines.

Pines digitales utilizados.

```
const int db7 = 2;          //LCD
const int db6 = 3;          //LCD
const int db5 = 4;          //LCD
const int db4 = 5;          //LCD
const int leds1 = 6;        //LED
const int pot = 7;          //POTENTIOMETER
const int rs = 8;           //LCD
const int leds2 = 9;        //LED
const int leds3 = 10;       //LED
const int e = 11;           //LCD
const int leds4 = 12;       //LED
const int leds5 = 13;       //LED
```

Pines analógicos utilizados.

```
const int upButton = 0;     //BUTTON
const int downButton = 1;   //BUTTON
const int backButton = 2;   //BUTTON
const int playButton = 3;   //BUTTON
```

Código – constantes generales.

```
int actualMelody = 0;
int minorNotes[] = {
    NOTE_AS3, 0, NOTE_CS3,
    NOTE_DS3, 0, NOTE_FS3,
    NOTE_GS3
};
int notes[] = {
    NOTE_A3, NOTE_B3, NOTE_C3,
    NOTE_D3, NOTE_E3, NOTE_F3,
    NOTE_G3
};
int actualMode = 0;
bool menuSelected = false;
int ledsSwitch = 1;
bool playingSong = false;
bool displayManualSongMessage = true;
```

Usos:

- Actual Melody y actual mode, indican la posición del puntero, y su uso se vincula con el botón “play”.
- Minor notes y notes, se utilizan para como base para ingresar por serial notas manualmente.
- Menu selected índice, si hay algún modo seleccionado en el momento, se utiliza para poder regular el uso de ambos menus.
- Playing song indica si en el actual momento hay alguna melodía en reproducción, se utiliza para poder detenerla.
- Led Switch, es un contador que regula el juego de leds.

Código – Melodías.

Se estructuraron seis melodías, que utilizan las notas definidas previamente, cada melodía consta de un array de sus notas, otro array del tempo de cada nota, y la cantidad total de las mismas.

```
//HappyBirthday----
int melody2[] = {
  NOTE_F4,NOTE_F4,NOTE_G4,NOTE_F4,NOTE_AS4,NOTE_A4,0,
  NOTE_F4,NOTE_F4,NOTE_G4,NOTE_F4,NOTE_C5,NOTE_AS4,0,
  NOTE_F4,NOTE_F4,NOTE_E5,NOTE_CS5,NOTE_A4,NOTE_GS4,NOTE_FS4,0,
  NOTE_CS5,NOTE_CS5,NOTE_C5,NOTE_GS4,NOTE_AS4,NOTE_GS4
};
int melody2Durations[] = { 4, 8, 4, 4, 4, 4, 4,
                           4, 8, 4, 4, 4, 4, 4,
                           4, 8, 4, 4, 4, 4, 4, 4,
                           4, 8, 4, 4, 4, 4, 4};
int melody2numbersOfNotes = 29;
```

Código – Funciones.

```
void ledsSwitcherOn(){
 switch (ledsSwitch) {
    case 0:
      digitalWrite(leds1, 255);
      break;
    case 1:
      digitalWrite(leds2, 255);
      break;
    case 2:
      digitalWrite(leds3, 255);
      break;
    case 3:
      digitalWrite(leds4, 255);
      break;
    case 4:
      digitalWrite(leds5, 255);
      break;
    case 5:
      break;
 }
}
```

Utilizando el contador de ledsSwitch, esta función utilizada en el método player, permite que por cada nota musical, se prenda una led distinta. De igual forma se apagan, en una función similar.

Código – Funciones.

```
void player(int melody[],int durations[],
            int sizeMelody,int speakerPin){

  for (int thisNote = 0; thisNote < sizeMelody; thisNote++) {

    if(analogRead(backButton)){
      break;
    }


    ledsSwitcherOn();

    int noteDuration = 1000 / durations[thisNote];
    tone(speakerPin, melody[thisNote], noteDuration);

    ledsSwitcherOff();

    int pauseBetweenNotes = noteDuration * 1.30;
    delay(pauseBetweenNotes);

    noTone(8);

  }
}
```

Esta función, recibe un array de melodías, su tempo, el tamaño y un pin(speaker).
Por cada nota del array, primero se fija si no se presiono el botón de “back”, de ser así se detiene la canción, volviendo al menú anterior. Caso contrario, por cada una se prende un led, y se utiliza la función tone del piezo para que este componente reproduzca la nota esperada con su tempo.

Código – Funciones.

Esta función a partir de la variable actualMelody, determina lo que se debe imprimir en el LCD, pudiendo así desplegarse por las distintas opciones. En este caso, mostrara un menú de canciones.

```
void updateMenuSongs() {
  switch (actualMelody) {
    case -1:
      actualMelody = 0;
      break;
    case 0:
      lcd.clear();
      lcd.print("> Sky Song");
      lcd.setCursor(0, 1);
      lcd.print("  Happy Birthday");
      break;
    case 1:
      lcd.clear();
      lcd.print("  Sky Song");
      lcd.setCursor(0, 1);
      lcd.print("> Happy Birthday");
      break;
    case 2:
      lcd.clear();
      lcd.print("> Game Of Thrones");
      lcd.setCursor(0, 1);
      lcd.print("  Super Mario");
      break;
    case 3:
      lcd.clear();
      lcd.print("  Game Of Thrones");
      lcd.setCursor(0, 1);
      lcd.print("> Super Mario");
      break;
    case 4:
      lcd.clear();
      lcd.print("> Tetris");
      lcd.setCursor(0, 1);
      lcd.print("  Take on me");
```

Código – Funciones.

```
void updateMenuMode() {
  switch (actualMode) {
    case -1:
      actualMode = 0;
      break;
    case 0:
      lcd.clear();
      lcd.print("> Saved Songs");
      lcd.setCursor(0, 1);
      lcd.print("  Manual Song");
      break;
    case 1:
      lcd.clear();
      lcd.print("  Saved Songs");
      lcd.setCursor(0, 1);
      lcd.print("> Manual Song");
      break;
    case 2:
      actualMode = 1;
      break;
  }
}
```

Funciona de igual manera que el método explicado anteriormente, permite desplegarse por las distintas opciones del menú, pero esta vez del menú de modos.

A partir del valor de actualMelody, utiliza la función player, y le pasa la melodía correspondiente

```
void playSong() {
  switch (actualMelody) {
    case 0:
      player(melody1,melody1Durations,melody1numbersOfNotes,pot);
      break;
    case 1:
      player(melody2,melody2Durations,melody2numbersOfNotes,pot);
      break;
    case 2:
      player(melody3,melody3Durations,melody3numbersOfNotes,pot);
      break;
    case 3:
      player(melody4,melody4Durations,melody4numbersOfNotes,pot);
      break;
    case 4:
      player(melody5,melody5Durations,melody5numbersOfNotes,pot);
      break;
    case 5:
      player(melody6,melody6Durations,melody6numbersOfNotes,pot);
      break;
  }
}
```

```
void playManualMelody() {
    int newMelody[16];
    int newMelodyDurations[16];
    int i = 0;

    if (Serial.available() > 0) {
      char ch = Serial.read();
      if (ch >= 'a' && ch <= 'g'){
        newMelody[i] = minorNotes[ch - 'a'];
        newMelodyDurations[i] = 4;
        i++;
      }
      else if (ch >= 'A' && ch <= 'G') {
        newMelody[i] = notes[ch - 'A'];
        newMelodyDurations[i] = 4;
        i++;
      }
      lcd.print(ch);
      player(newMelody,newMelodyDurations,i,pot);
    }

  else{
    lcd.clear();
  }
}
```

Este método se utiliza para el ingreso manual de notas, toma desde el serial distintas letras separadas por “coma”, y las agrega a un array, con un nuevo valor traducido a las constantes de las notas musicales definidas para que luego el player pueda reproducirlas.

Código – Setup.

```
void setup(){

  lcd.begin(16, 2);

  pinMode(upButton, INPUT);
  pinMode(downButton, INPUT);
  pinMode(playButton, INPUT);
  pinMode(backButton, INPUT);
  pinMode(backButton, INPUT);
  pinMode(leds1, OUTPUT);
  pinMode(leds2, OUTPUT);
  pinMode(leds3, OUTPUT);
  pinMode(leds4, OUTPUT);
  pinMode(leds5, OUTPUT);
  pinMode(pot, OUTPUT);

  Serial.begin (9600);

  startDisplay();
}
```

En el setup, inicializamos el lcd (begin), luego inicializamos todos los pins utilizados.
Inicializamos también el puerto serial en el puerto 9600, que luego se utilizara para introducir notas. Y por ultimo utilizamos una función startDisplay(), que imprime un mensaje de bienvenida, y luego inicia el menú para seleccionar el modo.

Código – Loop

```
if (analogRead(upButton) && !menuSelected){
  goUpMenuMode();
}

if (analogRead(downButton) && !menuSelected) {
  goDownMenuMode();
}

if (analogRead(playButton)&& !menuSelected) {
  goSelectedMode();
}
```

Definimos en diferentes if, las posibilidades iniciales para elegir el modo de nuestro ArduiJ.
Es importante distinguir la variable de menuSelected, ya que a partir de esta podremos saber el menú en donde estamos parados.
En caso de presionar playButton se cambia de menu.

Código – Loop

Funciona de igual forma que el menú anterior, pero aquí si seleccionamos playButton, se reproduce la canción deseada. Este menu corresponde al modo automatico.

```
if(automaticMode() && menuSelected){
  if (analogRead(upButton)) {
    goUpMenuSongs();
  }

  if (analogRead(downButton)) {
    goDownMenuSongs();
  }

  if (analogRead(playButton)){
    playingSong = true;
    playSong();
    playingSong = false;
  }

}
```

Código – Loop

```
else if (manualMode() && menuSelected){
  if(displayManualSongMessage){
   Serial.println("Insert a melody using letters A to G (minus is for minor notes)");
   displayManualSongMessage = false;
  }
  playManualMelody();
}
```

En caso de haber elegido el modo manual, en el monitor serial nos aparecerá un mensaje indicando lo que podemos ingresar, y al enviar re reproducirá la melodía ingresada y se imprimirán en el display las letras ingresadas.

Problemas.

- El primer problema presentado, fue regular el volumen del piezo, el cual se resolvió haciéndole una conexión directa al potenciómetro. El problema de esta solución radica en que no podemos saber la frecuencia debido a la solución empleada.
- Otro problema fue pausar la melodía, este problema se resolvió derivándole la tarea al botón de back, pero no soluciona el problema al 100% ya que regresa al menú anterior.