UNIVERSIDADE FEDERAL DE MINAS GERAIS

Teclado Vocalizador Braille

Escola de Engenharia / Depto Eletrônica
LSI - Laboratório de Sistemas Inteligentes
CEFALA – Centro de Estudos da Fala, Música e Linguagem

Estudante Wanderson da Silva Maciel Filho
Prof. Maurílio Nunes Vieira

Faculdade de Educação
LAPOA - Laboratório Interdisciplinar de Produção de Objetos de Aprendizagem para a pessoa com deficiência

GEINE - Grupo Interdisciplinar de Estudos sobre Educação Inclusiva e Necessidades Educacionais Especiais

Estudante Beatriz Fonseca Torres
Profa. Regina Célia Passos Ribeiro de Campos

1. **Introdução** 3
2. **Funcionamento** 5
3. **Placa de controle** 6
3.1. **Hardware** 6
3.2. **Software** 8
4. **Desafios enfrentados** 8
5. **Possíveis melhorias** 9
6. **Conclusão** 9
7. **Apêndice: Código fonte** 10

1. Introdução

Este trabalho descreve o desenvolvimento do protótipo de um teclado para auxílio à alfabetização de uma criança com deficiência visual. O objetivo é o aprendizado dos símbolos Braille para as letras e números. O usuário está matriculado numa classe com alunos sem perda visual e outra meta é a inclusão do deficiente nesta classe. Para isso, o teclado permite não só a sonorização dos caracteres como também o registro e visualização de teclas digitadas para acompanhamento do professor ou dos colegas (Figura 1).

Figura 1: Estudante deficiente visual utilizando o protótipo.

A figura 2 mostra detalhes do protótipo. As teclas estão dispostas em um grupo alfabético e outro numérico. Círculos de madeira com os símbolos Braille foram colados sobre as chaves. Há dois alto-falantes (SP1 e SP2), um display de cristal líquido (LCD) e chaves para ligar/desligar os alto-falantes (mute) e ajustar o volume sonoro (Vol- e Vol+). O usuário também pode ouvir o som das letras por meio de um fone de ouvidos, cujo conector está abaixo da chave mute. As quatro teclas circulares pretas na parte inferior (não identificadas) são reservadas para uso futuro na fase de aquisição da escrita.

Figura 2: Visualização do exterior do protótipo. Ver detalhes no texto.

A Figura 3 mostra o interior do teclado onde estão a placa i2c controladora do display LCD, as fontes de alimentação (AC/DC) e a placa de controle principal. Esta versão do teclado não permite a utilização de baterias.

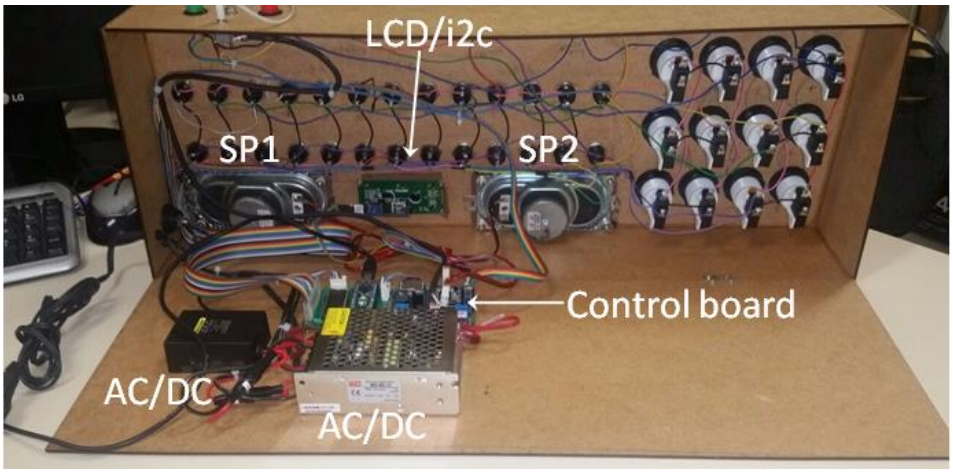

Figura 3: Visualização do interior do protótipo. Ver detalhes no texto.

A figura 4 mostra a placa de controle e seus componentes principais: Arduino Nano, decodificador do teclado (CD 4515), leitor/reprodutor de gravações de áudio (DfPlayer mini) e amplificador de potência (Pam 8610). A decodificação do teclado poderia ter sido feita usando o hardware de teclados seriais PS2,

obsoletos, mas a falta de padronização no layout das matrizes destes teclados motivou a utilização de um circuito baseado do circuito integrado CD 4515, um decodificador de 4 para 16 linhas.

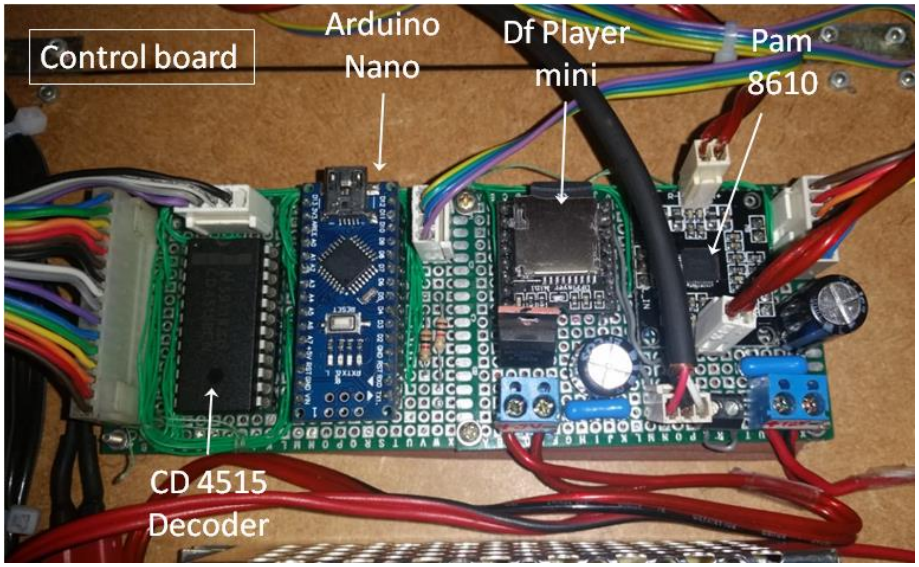

Figura 4: Placa de controle. Ver detalhes no texto.

2. Funcionamento

Basicamente (Figura 5), o microprocessador faz uma varredura do teclado, detecta se alguma tecla foi pressionada e reproduz o sinal de áudio (mp3 ou wav) correspondente. A letra correspondente também é registrada no display LCD. Chaves também permitem ajustar o nível do sinal de áudio.

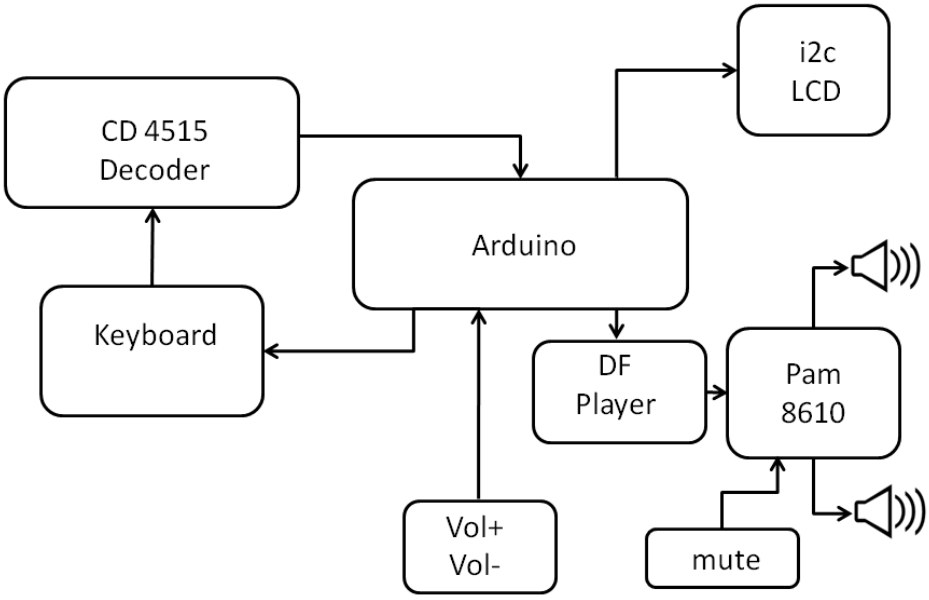

Figura 5: Diagrama esquemático da eletrônica de controle. Ver detalhes no texto.

3. Placa de controle

3.1. Hardware

O diagrama esquemático da placa controladora é apresentado na Figura 6. U2 é o Arduino Nano que faz o controle principal das funções. U1 é o circuito integrado CD4515 que decodifica 4 linhas de endereçamento binário (saídas digitais D8…D11 do Arduino) em até 16 linhas de acionamento individual (Y0…Y15). No projeto, são usadas 14 saídas do CD4515 (Y0…Y13), cada uma endereçando um grupamento de 3 chaves. Desta forma, a saída Y0 endereça as chaves das teclas A, B ou C; a saída Y1 endereça as teclas D, E ou F; e assim sucessivamente. Todas as teclas alfanuméricas são de contato momentâneo. Os grupos de três teclas estão interligados pelos sinais de retorno (Ret1, Ret2 e Ret3) de forma a permitir a detecção de qual tecla está acionada no grupo. As linhas de retorno correspondem a entradas digitais do Arduino. Em síntese, a detecção é feita com o endereçamento, por software, dos grupos de 3 teclas (Y0…Y13) seguida pela identificação da linha de retorno ativada em cada grupo.

Uma vez detectada a tecla, o software no Arduino fará o controle da reprodução do áudio e da escrita no display LCD. Será enviado um código ao DFPlayer que reproduz o arquivo correspondente ao nome da letra, previamente gravado, editado e armazenado em um cartão MicroSD alojado no DFPlayer. Para melhor controle da reprodução, as funções de controle de volume foram executadas no Arduino e não nos pinos do DFPlayer. O sinal de áudio na saída do DFPlayer tem potência adequada para uso em microfone (que pode ser conectado em J1), mas requer amplificação para reprodução em alto-falantes para que todos os estudantes ouçam. Esta amplificação é realizada pelo Pam8610, um amplificador classe D estéreo de 15 W por canal. Foi necessário o uso de uma fonte de alimentação isolada (12 V x 5 A) para este amplificador para evitar interferências com outros circuitos. A fonte de 9 V x 1 A alimenta o Arduino que, por sua vez, alimenta o CD4515, de tecnologia CMOS, pelo saída +5V. O circuito integrado U3 é o regulador 7805 que fornece 5 V x 1 A (+5Vpwr) para o display LCD e do DFPlayer.

O software gerencia a escrita das letras no display LCD (16 colunas x 2 linhas). Este registro permite que o professor ou colegas acompanhem as

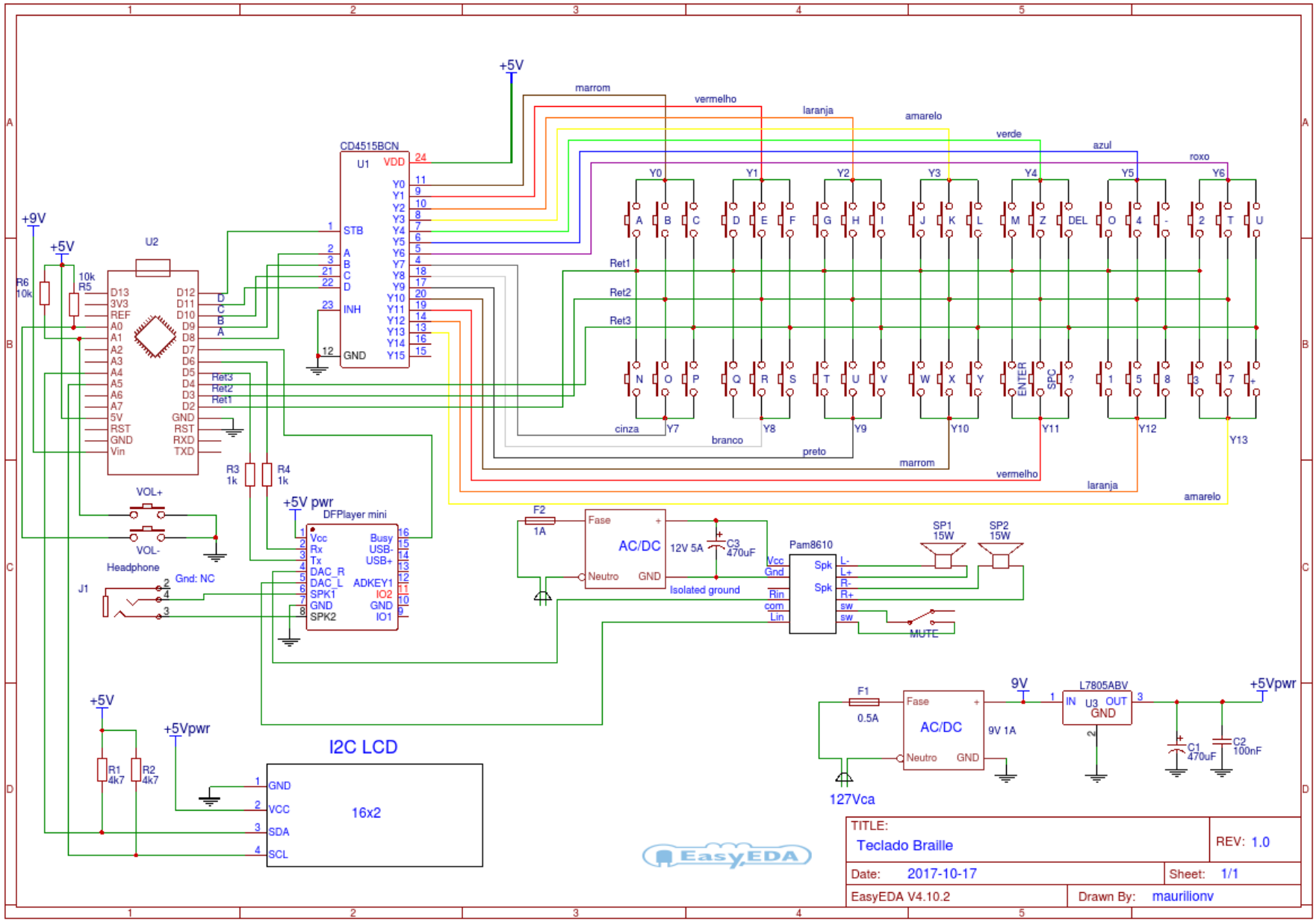

Figura 6: Esquemático do circuito eletrônico

ações do usuário do teclado. No futuro, quando o estudante estiver na fase de aquisição da escrita, poderão ser desenvolvidos recursos para auxílio à escrita, incluindo a reprodução sílabas, palavras, etc.

3.2. Software

Os comentários no código fonte, anexo, dão os detalhes do controle do hardware, links para bibliotecas, etc. O acesso direto às portas do microprocessador (PORTB) resultou em um código compacto. As teclas são endereçadas através das variáveis alfabeto[][] e Order[][], que já prevêem quatro teclas reserva (ENT, SPC, DEL, ?).

4. Desafios enfrentados

Na tentativa de realizar um projeto conciso e compacto, foram buscadas alternativas para se escrever um código que pudesse ser o mais sucinto possível. A melhor opção encontrada foi o uso do endereçamento PORTB para a leitura das teclas. Ao se tratar disso, é destacável o desafio de encarar uma forma de programação que não fora utilizada antes.

Ademais, sabe-se que sinais analógicos sofrem muito ruídos externos que apresentam diversas fontes. Isso se intensifica quando se trata de um projeto de baixo investimento como esse. Dessa forma, ao tentar reproduzir os áudios MP3 concluímos que a qualidade era baixa devido a algum tipo de interferência. Ao investigar de onde poderia estar vindo a distorção do sinal foi melhorada a qualidade da gravação do arquivo, refeita as conexões a fim de evitar mau contato, alterada a intensidade dos alto falantes, mas o que resultou em uma melhora eficaz na reprodução do áudio foi a inserção de uma fonte de tensão exclusiva para o circuito de áudio. A conclusão que é retirada desta solução é a existência de algum tipo de sinal que chegava aos alto falantes por meio do terminal negativo que era interligado por todo o circuito.

Por fim, pode-se citar a dificuldade na construção de uma placa projetada em softwares de design. Existe a dificuldade de representar a variedade de módulos que o circuito apresenta. A partir disso, foi escolhida a construção da placa de forma manual, já que o tempo para a conclusão do projeto era curto.

5. Possíveis melhorias

Como o objetivo dessa primeira abordagem era apenas um protótipo do vocalizador, claramente existem pontos a serem melhorados até que se chegue num produto final que corresponda com todas as expectativas. A seguir pode-se conferir uma lista de pontos que devem ser considerados nos futuros protótipos:

- Compactação da estrutura de forma que ofereça maior confiabilidade mecânica e praticidade para o aparelho.
- Concepção da placa por via de software, a fim de padronizar a construção e fácil substituição
- Inserção de um sinal sonoro indicando para o usuário que o aparelho está pronto para utilização após ser iniciado. O mesmo serve para o momento em que o aparelho for desligado.
- Adicionar uma função para que além do aparelho representar letras e números ele seja capaz de representar sílabas e pequenas palavras. O mesmo pensamento pode ser aplicado quando se trata dos números e operações simples (soma, subtração, multiplicação e divisão).
- Tornar o aparelho portátil, através de uma bateria que possa ser recarregada e apresenta uma autonomia interessante.

6. Conclusão

Portanto, a partir do que foi apresentado, pode-se constatar que pequenas aplicações da engenharia são o suficiente para facilitar a vida do ser humano, principalmente daqueles portadores de deficiência física. Esse protótipo foi testado com alunos e apresentou uma alternativa aos professores no momento de ensinar a linguagem Braille.

Além disso, vale ressaltar que o protótipo cumpriu com o propósito de validar a ideia inicial que, a partir de agora, pode ser otimizada a fim de tornar sua utilização mais impactante na educação dos deficientes visuais.

7. Apêndice: Código fonte

Disponível em:

https://drive.google.com/open?id=1iKNauiLZP-CNw-wMV79ck8yiDLUtnyXp

```
  /*   UFMG / Esc.Engenharia  / Depto Eletrônica
   *   Prof. Maurílio Nunes Vieira, Estudante Wanderson Maciel
   *
   *  UFMG / Fac Educação/ LAPOA - GEINE
   *  Profa. Regina Célia Passos Ribeiro de Campos,  Estudante Beatriz Fonseca Torres
   */

// i2c LCD
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

// MP3 Player
#include <SoftwareSerial.h>
#include "DFPlayer_Mini_Mp3.h"

// CD4515 (4 to 16 decoder) address lines
#define strobe 12
#define A       8        // D8  = PORTB PB0 ... Data 1 => 4515/pin 2  (lsb)
#define B       9        // D9  = PORTB PB1 ... Data 2 => 4515/pin 3
#define C       10       // D10 = PORTB PB2 ... Data 3 => 4515/pin 21
#define D       11       // D11 = PORTB PB3 ... Data 4 => 4515/pin 22 (msb)
#define strobe  12       //        => 4515 pino 1

//                    ------------------------------------------
//           PORTB:   PB7  PB6  PB5   PB4  PB3   PB2   PB1   PB0
// Arduino UNO/NANO:            D13   D12  ~D11  ~D10  ~D9   D8
// CD 4515                           Srobe  D     C     B    A
//                    ------------------------------------------

// kebyboard return lines
#define Ret1    2                   // keyboard return 1
#define Ret2    3                   // keyboard return 2
#define Ret3    4                   // keyboard return 1

#define DEL     127                 // ASCII for "delete"
#define ENT     13                  // ASCII for CR = Carriage Return ('enter')
#define SPC     32                  // ASCII for Space

#define COL     14                  // nb of columns in alfabeto[3][COL]

// Mp3Player
#define Rx      5
#define Tx      6
#define Busy    7
#define VOL     15                   // default Volume (0 ... 30)
#define VolUp   1                    // A1 input (pull up resistor)
#define VolDown 0                    // A0 input (pull up resistor)
#define EQ      2                    // 0/1/2/3/4/5  =  Normal/Pop/Rock/Jazz/Classic/Bass

void mp3_set_EQ (uint16_t eq);
// i2C LCD Display at address 0x3f
LiquidCrystal_I2C lcd(0x3f,2,1,0,4,5,6,7,3, POSITIVE);

// creates character for "enter" (carriage return)
byte EnterSymbol[8] = {
  B00000,
  B00001,
  B00001,
  B00101,
  B01001,
  B11111,
  B01000,
  B00100
};

// MP3 Player
SoftwareSerial mySerial(Rx, Tx);
```

```
void setup(){
  // LCD i2C   powered by extenal supply
  // i2c uses pins A4 and A5 (with 4.7 kohm pull up resistors)
  lcd.begin (16,2);
  lcd.setBacklight(HIGH);
  lcd.setCursor(0,0);
  lcd.clear();
  lcd.blink();
  lcd.print("UFMG - Out/2017");
  lcd.setCursor(0,1);
  lcd.print(" Inicializando ");
  lcd.createChar(ENT,EnterSymbol);

  // CD4515 connections
  pinMode(A, OUTPUT);                // Data 1 A (pino 2 4515)
  pinMode(B, OUTPUT);                // Data 2 B (pino 3 4515)
  pinMode(C, OUTPUT);                // Data 3 C (pino 21 4515)
  pinMode(D, OUTPUT);                // Data 4 D (pino 22 4515)
  pinMode(strobe, OUTPUT);           // Strobe (pino 1 4515)

  // keyboard return lines
  pinMode(Ret1, INPUT_PULLUP);       // selected by alfabeto[0,i], i = 0 ... 11
  pinMode(Ret2, INPUT_PULLUP);       // selected by alfabeto[1,i], i = 0 ... 11
  pinMode(Ret3, INPUT_PULLUP);       // selected by alfabeto[2,i], i = 0 ... 11

  // MP3 Player
  // http://www.picaxe.com/docs/spe033.pdf
  /* MP3 Player
    http://www.picaxe.com/docs/spe033.pdf
    ".mp3" ou ".wav" files:
     - MUST BE in folder /mp3
     - MUST BE numbered with 4 numbers:
       0001.mp3   (a)
       0002.mp3   (b)
        ...       ...
       0038.mp3   (-)

     - File name ca have other text after inintial 4 numbers:
       0001a.wav      (a)
       0002b.wav      (b)
        ...           ...
       0038menos.wav  (-)
  */

  /***********************************************
  **Arduino          DFPlayer:
  *Pin5 -- 1kohm --   TX (3);
  *Pin6 -- 1kohm --   RX (2);
  *pin7               BUSY (16)
  *5V                 1 (Vcc)
  *GND                7 (Gnd)
  *                   6 headphone +
  *                   8 headphone -
  *                   4 DAC_R  => PAM 8610 power amp
  *                   5 DAC_L  => PAM 8610 power amp
  ***********************************************/
  pinMode(Busy, INPUT);
  Serial.begin (9600);
  mySerial.begin (9600);
  mp3_set_serial (mySerial);  //set softwareSerial for DFPlayer-mini mp3 module
  delay(5000); // delay 1ms to set volume
  mp3_set_volume (VOL); // value 0~30; default VOL = 25
  delay(10);

  mp3_set_EQ (EQ);    // default Equalizer  0/1/2/3/4/5  Normal/Pop/Rock/Jazz/Classic/Bass


  // PAM 8610 (2 x 15W)?
  // https://tronixlabs.com.au/breakout-boards/amplifier/mini-amplifier-breakout-board-2-x-3w-class-d-pam8403/

  hello(2000);            // mensagens iniciais 2000 = delay in ms

}
```

```
/*   saída do 4515
               Y0 =    0    1    2    3    4      5     6    7    8     9   10    11     12    13 */

char alfabeto[3][COL]={{'a', 'd', 'g', 'j', 'm',   '0',  '2', 'n', 'q', 't', 'w', ENT,   '1',  '3'},
                       { 'b', 'e', 'h', 'k', 'z',   '4',  '6', 'o', 'r', 'u', 'x', SPC,   '5',  '7'},
                       { 'c', 'f', 'i', 'l', DEL,   '-',  '9', 'p', 's', 'v', 'y', '?',   '8',  '+'}};

uint16_t Oder[3][COL] = {1,   4,   7,  10,   13,   27,   29,  14,  17,  20,  23,  40,    28,   30,
                         2,   5,   8,  11,   26,   31,   33,  15,  18,  21,  24,  41,    32,   34,
                         3,   6,   9,  12,   39,   38,   36,  16,  19,  22,  25,  42,    35,   37};

/*
ordem dos arquivos de áudio (podem ser inseridos helps controlados, por exemplo, com pulsadas do botão '?'
a b c d e f g h i j  k  l  m  n  o  p  q  r  s  t  u  v  w  x  y  z  0  1  2  3  4  5  6  7  8  9  +  -  del ent spc ?
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39  40  41  42

*/

int lcd_count = 0;    // Global variable (usar static?)
void loop() {
  Address4515();
  VolCntr();
}


void Address4515() {
  int  j;
  byte i;

  for( i=0; i < COL; i++) {
     /*
     4515 input: 0000 DCBA   (i)
                 0000 0000   (0)
                 0000 0001   (1)
                 0000 0010   (2)
                 0000 ...    ...
                 0000 1100   (COL)
   */
    digitalWrite(strobe, HIGH);             // Enable addressing

    PORTB = PORTB & B11110000;              // clear Data1 ... Data4 of CD4515
    PORTB = PORTB | i;                      // Address CD 4515

    digitalWrite(strobe, LOW);              // disable addressing

    j = KeyScan();                          // key pressed?

    if(j<3){                                // yes
      if(alfabeto[j][i] != DEL && alfabeto[j][i] != ENT && alfabeto[j][i] != SPC && alfabeto[j][i] != '?' ) {
         LcdDisplay(j,(int)i);
         Audio(Oder[j][i]);
       }
    }
  }
}



int KeyScan()
/*
 * CD4515 applies LOW to addressed output
*/
{
 int j;
 for(j = 0; j<3; j++){
  if(digitalRead(j+Ret1) == 0) {
    delay(200);                             // empirical
    while(digitalRead(j+Ret1) == 0) {
      // wait while key is pressed
    }
    break;
  }
 }
 return j;  // j = 3 ==> no key was pressed
}
```

```
void LcdDisplay(int j, int i) {
  // lcd_count: global variable
  if(lcd_count==16){                      // last column?
    lcd.setCursor(0,1);                   // begin of 2nd line
  }

  if(lcd_count > 31){                     // is last position in LCD?
    lcd.clear();                          // clear and move cursor to (0,0)
    lcd_count=0;
  }

  if(alfabeto[j][i] == ENT) {
    lcd.print((char) ENT);
    lcd_count++;
  }
  else {
    lcd.print(alfabeto[j][i]);
    lcd_count++;
  }

  if(i==25 && lcd_count != 16 && lcd_count != 32) {
     lcd.print(" ");
  }
  delay(50);
}


void Audio(uint16_t num) {
 do{
 }while (!digitalRead(Busy));
 mp3_play(num);
 delay(10);                               // Is this line necessary?
}


void hello(int tempo) {
  // mensagens iniciais
  // lcd_count: global variable

  lcd.clear();
  lcd.noBlink();
  lcd.setCursor(0,0); lcd.print("Alfabeto Vocali-");
  lcd.setCursor(0,1); lcd.print("zador em Braille");
  delay(tempo);

  lcd.clear();
  lcd.setCursor(0,0); lcd.print("FAE/LAPOA/GEINE");
  lcd.setCursor(0,1); lcd.print(" Profa Regina");
  delay(tempo);

  lcd.clear();
  lcd.setCursor(0,0); lcd.print("FAE/LAPOA/GEINE");
  lcd.setCursor(0,1); lcd.print("Beatriz Fonseca");

  delay(tempo);
  lcd.clear();
  lcd.setCursor(0,0); lcd.print("DElt/Cefala/LSI");
  lcd.setCursor(0,1); lcd.print(" Prof Maurilio");
  delay(tempo);

  lcd.clear();
  lcd.setCursor(0,0); lcd.print("DElt/Cefala/LSI");
  lcd.setCursor(0,1); lcd.print("Wanderson Maciel");
  delay(tempo);

  lcd.clear();
  lcd.blink();
  lcd_count=0;                          // global variable
}
```

```
void VolCntr(){
  static int vol = VOL;                 // initialize first time only
  int x;

  if (x = analogRead(VolUp) < 100) {  // check VolUp key
    do{
      if ((vol++) > 30) {
        vol = 30;
      }
      mp3_set_volume (vol); // value 0~30;
      delay(100);
      // LcdDisplay(2,13); // +

    } while ((x = analogRead(VolUp) < 100));
  }

  if (x = analogRead(VolDown) < 100) {  // check VolDown key
    do{
      if ((vol--) < 0) {
        vol = 0;
      }
      mp3_set_volume (vol); // value 0~30;
      delay(100);
      // LcdDisplay(2,5); // -

    } while ((x = analogRead(VolDown) < 100));
  }
}
```