

# Teclado Interativo em Braille

Sistema de auxílio à alfabetização de crianças deficientes visuais baseado em Raspberry Pi.

Bianca F. F. Teixeira  
Faculdade do Gama  
Universidade de Brasília  
Distrito Federal, Brasil  
biancaffteixeira@hotmail.com

Joyce da Costa Santos  
Faculdade do Gama  
Universidade de Brasília  
Distrito Federal, Brasil  
joyceeletronica3@gmail.com

**Resumo** — O projeto visa a criação de um teclado interativo, que auxilia crianças na aprendizagem do método Braille com respostas sonoras.

**Palavras-chave** — *raspberry pi; braille; teclado.*

## 1. INTRODUÇÃO

### 1.1 REFERÊNCIA BIBLIOGRÁFICA

O sistema Braille representa, para deficientes visuais, uma emancipação e uma porta de entrada no mundo da leitura e da escrita de forma autônoma, este é baseado na combinação de 6 pontos, formando letras, números e pontuação gráfica. O sistema tornou-se tão popular que ganhou espaço também nos meios musical, possuindo códigos para notações musicais, e matemático, com códigos para manipulações avançadas em cálculo integral e diferencial [1][2][3].

O Braille é bastante eficaz, sendo que, um deficiente visual bem treinado consegue ler, utilizando a ponta dos dedos com cerca de metade da velocidade da leitura que usamos com os olhos, porém, mesmo sendo um método eficiente, a alfabetização nesse sistema enfrenta várias dificuldades, como por exemplo:

- O discernimento tátil deve ser intensivamente treinado
- Existência ínfima de materiais de apoio que permitam a assimilação do alfabeto, antes da aptidão mecânica para escrita
- A enorme resistência de professores mais velhos, em especial nas escolas de cegos tradicionais, alegando que tecnologias de apoio, como o computador, não estarão sempre disponíveis [4].

Quanto às tecnologias de apoio pedagógico, já existem iniciativas como o sistema DOSVOX, que permite que uma pessoa cega, a um custo relativamente baixo, use um computador convencional, através de uma tecnologia em que o computador fala em português, porém, para uso desta ferramenta a criança já deve ser alfabetizada e necessita de um amigo vidente para auxílio [5].

Como consequência da necessidade de ferramental tecnológico como apoio e facilitação da aprendizagem no processo de alfabetização de crianças deficientes visuais,

surgiu a ideia de um teclado interativo em Braille, que possibilita que a criança conheça os grafemas em Braille, fazendo assimilação com o som emitido pelo sistema do teclado.

### 1.2. JUSTIFICATIVA

A escolha do projeto foi motivada pela possibilidade de oferecer ao alfabetizando em Braille uma plataforma interativa e portátil, que dê liberdade para este utilizá-la onde quiser, sem necessidade de auxílio ou supervisão, fazendo assim com que a criança descubra seu próprio ritmo de aprendizagem, tanto quanto perceba empiricamente a relação entre fonema e letra.

O alfabetizador é auxiliado na tarefa de ensino, fazendo com que o aluno escute e associe os sons a determinados grafemas, bem como a imitação dos sons ajudar a criança na aquisição da linguagem. As crianças com deficiência visual necessitam de estímulos no ambiente de aprendizagem para aguçar os outros sentidos e o alfabetizador deve atraí-las trazendo conhecimentos com recursos didáticos e materiais adaptados.

### 1.3. OBJETIVOS

- Desenvolver um sistema embarcado como apoio para crianças com deficiência visual em fase de alfabetização, utilizando um teclado adaptado aos códigos em Braille e um microcomputador de baixo custo, Raspberry Pi 3;
- Auxiliar no aprendizado da leitura em Braille acarretando uma melhora tátil e resposta rápida à identificação das letras do alfabeto.

### 1.4. REQUISITOS

- Desenvolvimento ou adaptação de uma plataforma para interação com o usuário, que baseia-se em um teclado com teclas adaptadas ao sistema Braille.
- Ligação teclado/Raspberry Pi
- Criação de um programa responsável:
  - Pela leitura de dados advindos do teclado.
  - Análise para assimilação código-fonema

- Acionamento de dispositivo sonoro.

## 1.5. BENEFÍCIOS

O ambiente de aprendizagem proporciona atividades com objetos físicos estimulando os sentidos da criança, como o tato e a audição. Pequenos recursos tornam o ambiente diferenciado e adaptado às necessidades do aluno. Por isso, o teclado interativo traz benefícios como por exemplo a independência do aluno no processo de aquisição de conhecimento e adaptação ao ambiente, além de melhorar a resposta da criança ao assimilar a tecla com o som da letra ou combinação de sílabas. O auxílio de um alfabetizador pode ser substituído em curtos espaços de tempo pelo sistema embarcado a ser desenvolvido.

Outra vantagem está na utilização do Raspberry Pi. O espaço, que pode ser ocupado pelo teclado interativo juntamente com o Raspberry Pi, é reduzido contrapondo a um computador de mesa.

## 2. VISÃO GERAL DO SISTEMA

O projeto é dotado de duas partes que interagem entre si, um hardware, composto de um teclado adaptado ao Braille e a placa Raspberry Pi, os quais são responsáveis pela aquisição de dados do sistema (leitura de teclas), e o software, responsável pela análise dos dados de entrada, processamento e sonora do grafema (fonema). Abaixo, segue imagem com o fluxo de dados entre o hardware e software.

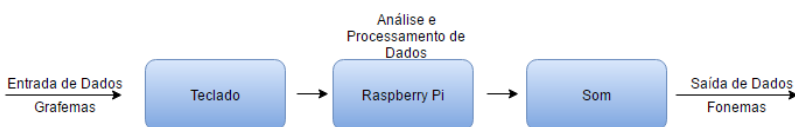


Figura 1 – Fluxo de Dados no Sistema

### 2.1 DESCRIÇÃO DO HARDWARE

O hardware é composto por um Raspberry Pi e por um teclado em Braille, composto de 16 teclas para esse protótipo. Para se obter um produto com tamanho e ergonomia adequados, optou-se pela fabricação deste, por não haver um modelo com a distribuição das teclas que permitirá um bom aprendizado e também ofereça conforto ao usuário. Este foi confeccionado a partir de madeira MDF, presente na Figura 2, botões de acrílico (que funcionam como chaves), e para o acionamento das teclas um sistema de teclado matricial.



Figura 2 - Teclado Matricial em MDF e botões de acrílico.

Abaixo, na Figura 3, é apresentado o sistema de teclado matricial.

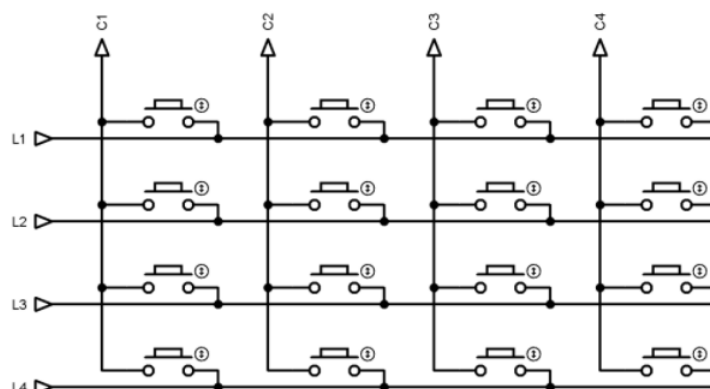


Figura 3 - Teclado Matricial

E na Figura 4, é apresentado o sistema de teclado matricial montado nas chaves de acrílico.

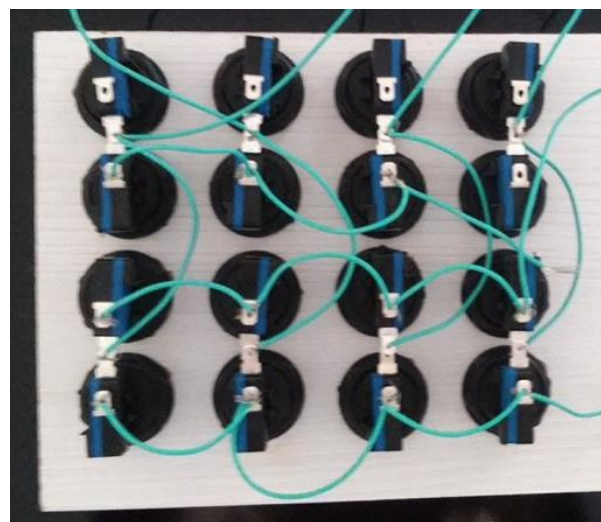


Figura 4 - Teclado matricial com chaves de acrílico.

O botão utilizado funciona como chave comum. O seguinte esquema abaixo foi montado em protoboard com o

Raspberry Pi 3 para testar uma chave, substituta temporária do botão que seria utilizado. Um resistor pull-down é ativado dentro do código. Neste esquemático, a chave foi ligada em 3.3 V e a outra entrada no pino 16 (GPIO 23). O código testado em Python foi chamado no terminal do Raspberry.

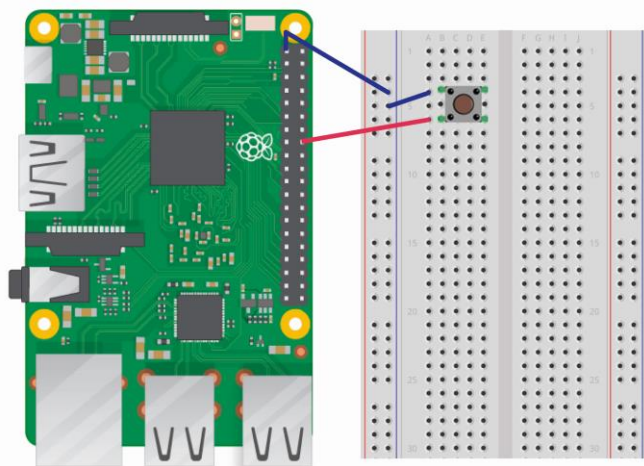


Figura 5 - Chave conectada à Raspberry com pull-down interno.

### 2.1.1. ENTRADAS E SAÍDAS UTILIZADAS DA PLACA DE DESENVOLVIMENTO

Uma das técnicas mais eficientes de multiplexação para leitura de teclados é o arranjo em formato matricial. Com esta configuração podemos, com N terminais, ler até  $(N/2)^2$  chaves, ou seja, com apenas 8 terminais é possível fazer uma leitura de 16 teclas [6].

Então serão utilizadas 8 portas GPIO do Raspberry para leitura dos botões, sendo estas: GPIO's 4, 17, 21, 22, 18, 23, 24 e 25, correspondentes aos pinos 7, 11, 13, 15, 12, 16, 18 e 22 e para saída de voz, será utilizada a saída de áudio da placa.

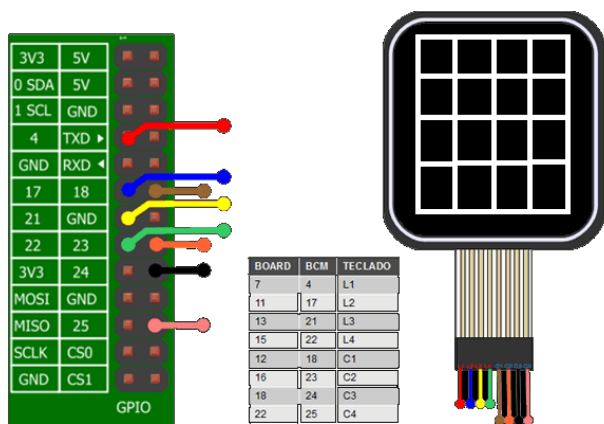


Figura 6 - Esquema de Ligação Teclado Matricial 4x4 e GPIO Raspberry

## 2.2. DESCRIÇÃO DO SOFTWARE

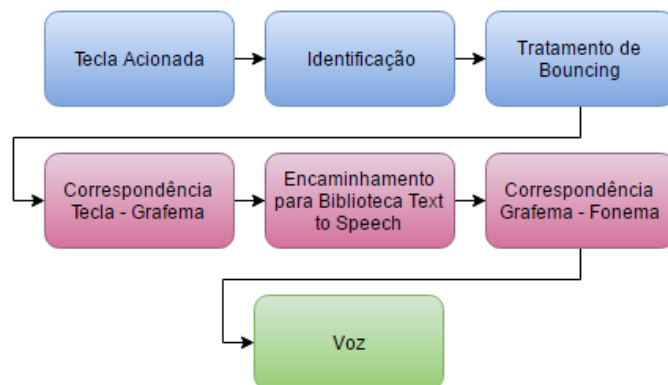


Figura 6 - Fluxograma de Dados do sistema

O algoritmo do programa é responsável:

1. Pela recepção e tratamento de dados do teclado: realizar a leitura dos botões acionados pelo usuário, tanto quanto, realizar um tratamento do sinal recebido, pois em chaves mecânicas há um erro conhecido como bouncing, que é uma “trepidação” que causa oscilações no sinal, e necessita de um algoritmo de debounce.
2. Pelo processamento: correspondência entre a tecla acionada e a letra correspondente (grafema) e encaminhamento para a biblioteca Text to Speech e correspondência entre o grafema e o fonema.
3. Pela execução: Saída em voz no fone de ouvido da tecla acionada.

### 2.2.1. VARREDURA DE TECLAS

A leitura matricial é feita por botões dispostos separadamente em linhas e colunas, como apresentado na Figura 3, de modo que a identificação da tecla pressionada é possível a partir de sua posição na matriz. A leitura, em si, é realizada por um processo conhecido como varredura de teclas, onde uma coluna é acionada por vez e é verificado qual botão daquela coluna foi pressionada, dessa forma, o programa responsável por esse processo percorre toda matriz de botões.

Diante disto, o terceiro código em anexo é responsável pela varredura das teclas, nele a matriz de letras é declarada conforme o alfabeto, e a correspondência das letras com os terminais da placa são feitos a partir do número da GPIO. As GPIO's da placa são definidas como saída para o acionamento de cada coluna, e definidas como entrada para leitura das linhas. Um loop também é criado para percorrer as linhas e as colunas.

## 2.2.2. DETECÇÃO DE EVENTOS E INTERRUPÇÕES

Como é possível perceber pelo primeiro código em anexo, este possui um delay de 1 segundo até a próxima leitura, desprezando assim qualquer acionamento da tecla que poderia ser feito dentro desse tempo. Para resolução deste problema, um evento é criado sempre que a tecla é pressionada durante o intervalo de tempo, ou seja, há a utilização da função detecção de eventos (event\_detect) em qualquer borda de descida, que já faz parte do RPi.GPIO. Esse sistema é responsável por “guardar” o acionamento da tecla, e coloca como prioridade, para quando se der o início do próximo loop, baseados na função callback, retornar o evento que ocorreu, sendo um segundo processo ou thread.

Para o tratamento de bounce, podemos aproveitar o parâmetro callback e requisitar que este ignore os primeiros 300 milissegundos da leitura, usando o parâmetro bouncetime. [7]

## 2.2.3. BIBLIOTECAS

A principal biblioteca a ser utilizada neste projeto é a Text-to-Speech (TTS) que é um mecanismo capaz de realizar síntese de voz, ou seja, converter texto em linguagem de voz. [8]. Este mecanismo produz uma voz artificial humana e pode ser usada em diversas aplicações, neste caso, um sistema voltado aos deficientes visuais que tem sido usado desde a década de 1980. [9] Desde então, alguns projetos de software livres começaram a ser implementados e por isso, foram sendo cada vez mais disponibilizados para download. Uma delas é a Festival Text-to-Speech que converte textos para áudios em inglês, espanhol e galês. [10]

Outra variante é a eSpeak Text-to-Speech que converte para 51 línguas, incluindo a língua portuguesa. Além disso, a eSpeak TTS suporta plataformas como Android, MAC OSX e Solaris e é um software escrito em C que produz um arquivo tipo .WAV, (assim como a Festival TTS) compactado. [11]

A GnuSpeech, também um pacote livre, possui um sistema de criação de base de dados, onde é possível modificar os dados fonéticos. Dessa forma, a saída poderá ser adaptada, com suas limitações, à diversas línguas. Este tipo de software pode ser utilizado em pesquisa psicoacústica (estímulo das sensações auditivas) e linguística. [12]

Ainda há o mecanismo Google Text-to-Speech, desenvolvido em meados de 2014, oferecido pela Google e desenvolvido para Android que é um aplicativo leitor de tela. A limitação desse aplicativo é referente ao sistema operacional, sendo este Android e a necessidade de uma conexão de internet potencialmente ótima para obter a resposta rápida ao usuário. [13]

Portanto, o pacote que atende às necessidades do projeto é a eSpeak TTS, onde é possível obter respostas de áudio em língua portuguesa utilizando espaço reduzido e

compactado na Raspberry podendo ser utilizado offline, ou seja, sem a necessidade de uma conexão de internet.

### 2.2.3.1. ESPEAK TEXT-TO-SPEECH

A biblioteca eSpeak, utilizada no projeto, no Raspberry Pi pode ser chamada do próprio terminal do Raspberry Pi ou dentro do código com a chamada correta e na língua desejada, português do Brasil. Para instalar o pacote, o seguinte comando foi utilizado:

```
$ sudo apt-get install espeak
```

Para a chamada em inglês da eSpeak do terminal do Raspberry Pi:

```
$ espeak "Hello world!"
```

Para a chamada em português da eSpeak do terminal do Raspberry Pi:

```
$ espeak -vpt "Olá mundo!"
```

Para averiguar a saída um fone de ouvido 3.5mm é utilizado.

## 3. RESULTADOS FUTUROS

Para o projeto final é necessário que o código realize a varredura de teclas, tanto quanto, realize o debouncing destas e a criação de eventos para correção de leitura rápida. Neste caso, o código principal, presente no anexo 3, necessita que o código presente no anexo 2 seja incorporado ao algoritmo.

## 4. REFERÊNCIAS

- [1]. A criança cega e o Braille. Disponível em: <[https://www.maxwell.vrac.puc-rio.br/8602/8602\\_3.PDF](https://www.maxwell.vrac.puc-rio.br/8602/8602_3.PDF)>. Acesso em 3 de abril de 2017.
- [2]. Banco de escola. Educação para todos. Disponível em: <<http://www.bancodeescola.com/entrevista-rbc-agosto-2008.htm>>. Acesso em 3 de abril de 2017.
- [3]. O sistema Braille no Brasil. Disponível em: <<http://www.apadev.org.br/dv/pages/workshop/Osistemabrailenobrasil.pdf>>. Acesso em 3 de abril de 2017.
- [4]. Projeto DEDINHO. Alfabetização de crianças cegas com ajuda do computador. Disponível em: <<http://intervox.nce.ufrj.br/dosvox/textos/dedinho.doc>>. Acesso em 3 de abril de 2017.
- [5]. Projeto DOSVOX. Disponível em: <<http://intervox.nce.ufrj.br/dosvox/>>. Acesso em 3 de abril de 2017.
- [6]. Teclado Matricial. Disponível em: <<https://www.embarcados.com.br/teclado-matricial-e-varredura-de-teclas/>>. Acesso em: 05 de abril de 2017.

[7]. Sourceforge. Raspberry GPIO Python. Disponível em: <<https://sourceforge.net/p/raspberry-gpio-python/wiki/Inputs/>> Acesso em 6 de junho de 2017.

[8]. RPi Text to Speech (Speech Synthesis). Disponível em: <[http://elinux.org/RPi\\_Text\\_to\\_Speech\\_\(Speech\\_Synthesis\)](http://elinux.org/RPi_Text_to_Speech_(Speech_Synthesis))>. Acesso em 2 de maio de 2017.

[9]. Transformando texto escrito em texto narrado com .NET e a biblioteca System.Speech. Disponível em: <<http://www.devmedia.com.br/transformando-texto-escrito-em-texto-narrado-com-net-e-a-biblioteca-system-speech/24240>> Acesso em 3 de maio de 2017.

[10]. The Centre for Speech Technology Research. The University of Edinburgh. Disponível em: <<http://www.cstr.ed.ac.uk/projects/festival/>> Acesso em 3 de maio de 2017.

[11]. eSpeak text to speech. Disponível em: <<http://espeak.sourceforge.net/>> Acesso em 2 de maio de 2017.

[12]. GnuSpeech. Disponível em: <<https://www.gnu.org/software/gnusppeech/>> Acesso em 2 de maio de 2017.

[13]. Google Text-to-Speech engine arrives to Google Play. Disponível em: <<http://www.androidauthority.com/google-text-to-speech-314826/>> Acesso em 9 de maio de 2017.

#### 4. ANEXOS

- 1) Código em Python para testar um botão:

```
import RPi.GPIO as gpio
import time

gpio.setmode(gpio.BCM)

gpio.setup(23, gpio.IN, pull_up_down = gpio.PUD_DOWN)

while True:
    if(gpio.input(23) == 1):
        print('Botão pressionado')
    else:
        print('Botao desligado')

    time.sleep(1)

gpio.cleanup()
exit()
```

2) Código em Python para criação de thread e tratamento de bouncing:

```
import RPi.GPIO as gpio
import time

# Define o pino que iremos utilizar

PIN = 23

# Funções

def action_press_button_loop(gpio_pin):
    print ('o botao no pino %d foi pressionado' % gpio_pin)
    print ('Saindo...')

def action_press_button(gpio_pin):
    print ('Tratando o botão no pino %d que foi pressionado' % (gpio_pin))

# Configuração da GPIO
# Configurando o modo dos pinos como BCM

gpio.setmode(gpio.BCM)

# Configurando PIN como INPUT e modo pull-down interno

gpio.setup(PIN, gpio.IN, pull_up_down = gpio.PUD_DOWN)

# Adicionando um evento ao GPIO 23 na mudança de nível alto 3.3V para o nível baixo 0V,
# assim como, adicionar o parâmetro bouncetime de 300 ms, que significa que os primeiros
# 300 ms serão ignorados da leitura.

gpio.add_event_detect(PIN, gpio.FALLING, callback=action_press_button, bouncetime=300)

while True:
    print ('Polling...')

    # chama a thread se a borda de descida for detectada

    if gpio.event_detected(PIN):
        action_press_button_loop(PIN)

# tempo para reiniciar o loop
time.sleep(1)

gpio.cleanup()
exit()
```

3) Código em Python para varredura de teclas:

```
import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BOARD)

MATRIZ = [
    ["A", "B", "C", "D"],
    ["E", "F", "G", "H"],
    ["I", "J", "K", "L"],
    ["M", "N", "O", "P"]
]

LINHA = [7, 11, 13, 15]
COL = [12, 16, 18, 22]

## definição de coluna como saída ##

for j in range(4):

    GPIO.setup(COL[j], GPIO.OUT)
    GPIO.output(COL[j], 1)

# definição de linha como entrada ##

for i in range(4):
    GPIO.setup(LINHA[j], GPIO.IN, pull_up_down = GPIO.PUD_DOWN)

try:
    while(True):
        for j in range(4):
            GPIO.output(COL[j],0)

            for i in range(4):
                if GPIO.input(LIN[i]) == 0:
                    print MATRIZ[i][j]
                    while(GPIO.input(LIN[i]) == 0):
                        pass

            GPIO.output(COL[j], 1)
except KeyboardInterrupt:
    GPIO.cleanup()
```