

Sistemas Operacionais Embarcados

Ponto de controle 2

Controlador MIDI

Igor de Alcantara Rabelo

Matrícula: 14/0143751

Universidade de Brasília

Gama, Brasil

E-mail:rabelo.alcantara.igor@gmail.com

Rodrigo Magalhaes

Matrícula: 14/0031154

Universidade de Brasília

Gama, Brasil

E-mail:rodrigomacr2@gmail.com

I. JUSTIFICATIVAS

Dispositivos MIDI são uma solução de software e hardware para problemas musicais. Sua vantagem se encontra na capacidade de simular diversos tipos de sons e instrumentos com grande facilidade, não sendo necessário possuir ou saber tocar vários tipos de instrumentos para produzir uma música rica e complexa. Controladores MIDI ainda podem oferecer funções de mixagem e edição de áudio durante a gravação, oferecendo ao músico a liberdade de experimentar diversas sonoridades e até mesmo de tocar ao vivo.

O controlador MIDI, como representado na figura abaixo, é uma ferramenta musical que permite aos usuários fazerem diversas coisas em um estúdio de música. É de muita importância para quem trabalha em estúdios musicais pois o MIDI dá um grande poder de edição e sincroniza todo o estúdio em uma fábrica de tracks super eficiente, além de outros benefícios.^[1]



Figura 1. Controlador MIDI.

II. OBJETIVOS

A intenção com este projeto é construir uma plataforma em hardware para produção de músicas on-the-go. A interface deverá possuir botões para representar cada nota da escala musical e para variação dos parâmetros do sinal (frequência, delay, volume) e deverá suportar diversos sintetizadores.

Considerando o alto preço de Pads MIDI no mercado, toma-se como um dos objetivos deste projeto a tentativa de redução do custo da plataforma enquanto mantendo uma qualidade profissional.

III. BILL OF MATERIALS

1. Raspberry Pi 3B+;
2. Cartão de memória;
3. Teclado de computador USB;
4. Display LCD;
5. Saídas para speaker (P2) e computador (USB);

IV. BENEFÍCIOS

Um dos grandes benefícios ao se usar um controlador MIDI é o de acionar sons musicais e tocar instrumentos musicais e também estes controladores podem ser usados para controlar outros dispositivos compatíveis com o MIDI, como luzes de palco, mixers de áudio digital e unidades complexas de efeitos de guitarra.^[3]

V. DESENVOLVIMENTO

Para o desenvolvimento do protótipo inicial, foi utilizado um teclado de computador para testar os botões como se fosse um teclado MIDI. O código desenvolvido foi feito utilizando a

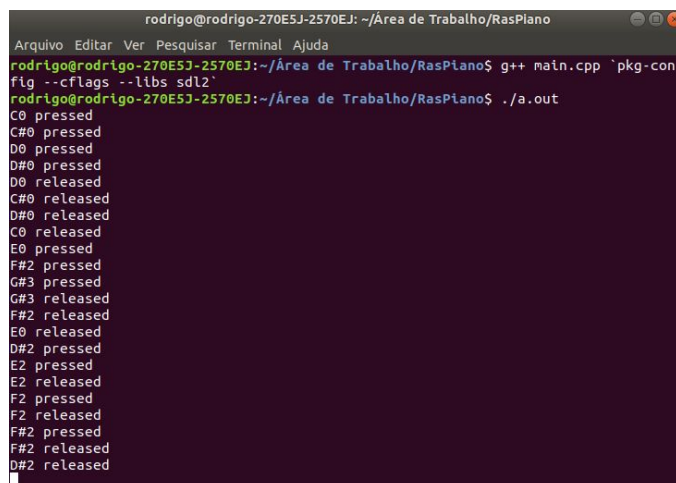
linguagem C++ onde foi possível utilizar a biblioteca SDL que fornece uma plataforma simples para gráficos, som, e dispositivos de entrada.

A partir do SDL, pode-se interceptar os eventos do teclado (pressionamento e soltura de botões). Separou-se então quatro linhas do teclado com 12 teclas, suportando um alcance de 4 oitavas incluindo os semitons. Durante o processo de mapeamento entre os botões e as notas, percebeu-se que o layout do teclado pode afetar diretamente o reconhecimento de teclas. O SDL utiliza o layout QWERTY americano como base, que não possui referências para botões de acentuação muito comuns em teclados brasileiros. Felizmente, a biblioteca oferece identificações tanto por Keycode, dada pelo valor do símbolo em UTF-8, quanto por Scancode, dada pela posição física no teclado, sendo as duas utilizadas quando conveniente.

Após resolvido o problema de mapeamento, notou-se que o reconhecimento do pressionamento de múltiplas teclas simultaneamente não era realizado nativamente pela biblioteca. Criou-se então um vetor de armazenamento de estados para as 48 teclas e uma função condicional, semelhante aos eventos de borda de subida e de descida, que as impedia de executar as mesmas linhas de código antes que houvesse uma mudança de estados. Assim que o botão é pressionado, a posição do vetor referente à tecla recebia valor 1, e quando solto o valor volta a 0. Por organização, transferiu-se a função criada para um header SimpleEvent.h, reduzindo aproximadamente 600 linhas do código principal. Ambos os códigos podem ser vistos na referência 1.

VI. RESULTADOS

Para a fase inicial do projeto, utilizou-se a impressão do nome das notas e seu estado de ativação (pressionado ou solto) para verificar o funcionamento do código (Figura 2). Verificou-se que o reconhecimento de teclas simultâneas funcionou corretamente e o vetor de armazenamento de estados será utilizado no próximo estágio do projeto para a reprodução das formas de onda. Com a função KeyboardWatchdog, da biblioteca SimpleEvent.h, adaptada para receber apenas uma struct como argumento será possível criar uma thread para rodar a interceptação de teclas paralelamente a função que reproduz áudio, esperando-se assim a redução do tempo de resposta do programa.



```
rodrigo@rodrigo-270E5J-2570EJ: ~/Área de Trabalho/RasPiano
Arquivo Editar Ver Pesquisar Terminal Ajuda
rodrigo@rodrigo-270E5J-2570EJ:~/Área de Trabalho/RasPiano$ g++ main.cpp -c -std=c++11 -cflags -lSDL2
rodrigo@rodrigo-270E5J-2570EJ:~/Área de Trabalho/RasPiano$ ./a.out
C0 pressed
D0 pressed
D0 released
C0 released
E0 pressed
F#2 pressed
G#3 pressed
G#3 released
F#2 released
E0 released
D#2 pressed
E2 pressed
E2 released
F2 pressed
F2 released
F#2 pressed
F#2 released
D#2 released
```

Figura 2. Reconhecimento de teclas.

A primeira etapa do projeto foi realizada com êxito. As próximas etapas consistem na utilização do ALSA(Advanced Linux Sound Architecture) que é basicamente um componente do núcleo que irá fornecer o driver para a placa de som da Raspberry Pi. Um dos objetivos do ALSA é a configuração automática do hardware da placa de som e a manipulação de múltiplos dispositivos em um sistema.

Também será utilizado o TiMidity++ que é um sintetizador de software que pode reproduzir arquivos MIDI sem um sintetizador de hardware. Nesse software é possível renderizar a placa de som em tempo real e salvar o resultado em um arquivo .wav.

VII. REFERÊNCIAS

- [1] RasPiano:
<https://github.com/RodrigoMac/Sistemas-Embarcados/tree/master/RasPiano>
- [2] SDLwiki:
<https://wiki.libsdl.org/FrontPage>
- [3]Raspberry Pi Midi-KeyBoard:
<https://www.youtube.com/watch?v=mpeTHLr1vIc>
- [4]ALSA(Advanced Linux Sound Architecture):
https://www.alsa-project.org/wiki/Main_Page
- [5]Timidity:
<https://wiki.archlinux.org/index.php/Timidity>