Sistemas Operacionais Embarcados

Ponto de controle 3

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:rodrigomacrt2@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, 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 boa qualidade.

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);
6. Caixa de som Mini Speaker com entrada/saída p2;
7. Softwares utilizados: Raspbian e TiMidity++;

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

Foram feitas algumas pesquisas para fazer a compra de um teclado MIDI que se encaixasse em um modelo mais próximo do profissional, porém, os custos seriam mais altos e também iria demorar mais para que o teclado chegasse porque teria que fazer o pedido em sites de outro país. Foi utilizado um teclado de computador de modelo ABNT2 para funcionar de modo que substituísse o teclado MIDI.

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 clique de múltiplas teclas em simultâneo 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 recebe 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.

Para a segunda etapa de desenvolvimento do projeto, implementou-se o envio de mensagens MIDI, adicionando funcionalidades básicas ao projeto. Utilizando-se a biblioteca RtMidi.h, criou-se um canal de comunicação entre o driver de áudio do Raspbian (ALSA) e um sintetizador open source third party (TiMidity++), possibilitando a reprodução de áudio.

Seguindo o protocolo de mensagens MIDI [6], definiu-se as funções responsáveis por ligar e desligar as notas musicais, aumentar e diminuir o volume, variar a frequência das teclas e alterar o instrumento sintetizado. As mensagens são basicamente compostas por entre 2 e 3 bytes, sendo o primeiro byte de status (informa a operação a ser realizada) enquanto os remanescentes são parâmetros dessa instrução.

O layout do teclado em relação às funções pode ser visto a seguir:

Figura 2. Layout para teclado do tipo ABNT2.

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 foi utilizado 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.

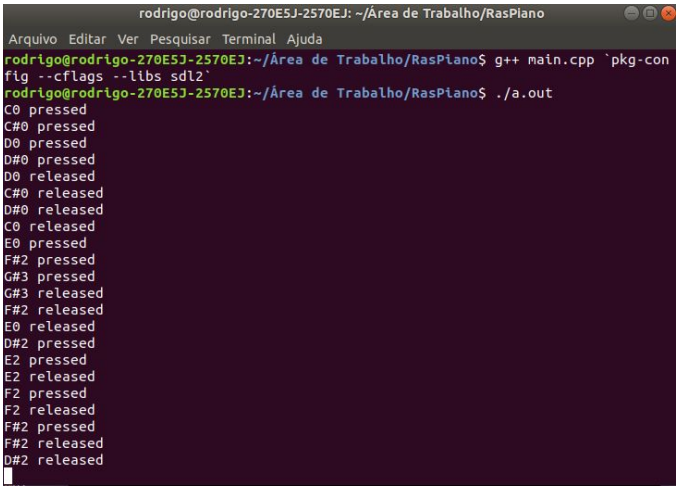

Figura 2. Reconhecimento de teclas.

Para a segunda fase do projeto, conclui-se a interface de áudio. Agora, ao invés de imprimir as notas no terminal, o programa envia mensagens MIDI ao sintetizador que agora reproduz o som em tempo real. Ao ligar e desligar as notas, passa-se um valor específico para a frequência e outro para o volume. Neste caso, nenhum parâmetro desta mensagem pode ser alterado retroativamente, o que significa que para mudar de volume ou a frequência base da nota, deve desligá-la e ligá-la novamente.

A função de bend de frequência, por ser uma mensagem separada da instrução de ativação de notas, é capaz de distorcer a frequência da nota em tempo real (retroativamente). Por fim, adicionou-se a opção de troca de instrumentos. Até definir-se uma solução melhor, utilizou se os números de 1 a 9 do Keypad para selecionar entre 9 instrumentos dos mais comuns para teste. Como a lista oficial possui 128 opções possíveis de timbre, provavelmente se utilizará uma espécie de menu para a troca.

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=mpeTHLr1vlc

[4]ALSA(Advanced Linux Sound Architecture):
https://www.alsa-project.org/wiki/Main_Page

[5]Timidity:
https://wiki.archlinux.org/index.php/Timidity

[6]Summary of MIDI Messages:
https://www.midi.org/midi/specifications/item/table-1-summary-of-midi-message