Universidade Tecnológica Federal do Paraná – UTFPR
Departamento Acadêmico de Eletrônica – DAELN
Departamento Acadêmico de Informática – DAINF
Engenharia de Computação
Oficina de Integração 3 (IF66J) – S71 – 2017/1

Relatório Técnico

Guitar LED: Plataforma Independente de Jogo Rítmico

Elis C. N. Santos – elisan@alunos.utfpr.edu.br
Helena S. Arins – lena.arins@gmail.com
Jhonatan A. Oliveira – jhonatan@alunos.utfpr.edu.br
Luiz H. Pereira – luizp@alunos.utfpr.edu.br
Vinicius A. Brenner – vini.brenner7@gmail.com

Julho de 2017

Resumo

Com o avanço da tecnologia, a popularização de jogos eletrônicos se tornou inevitável, proporcionando à seus jogadores momentos de lazer e mudando totalmente o cenário atual. Antes, o que era visto como algo para crianças, se tornou um mercado forte e importante, ficando cada vez mais comum encontrar pessoas jogando em todos os lugares. Entre os diversos gêneros existentes os jogos do gênero musical se destacam, pois desafiam o jogador a melhorar suas habilidades motoras e mentais para acompanhar o ritmo de uma música, seja através de botões, movimentos ou algum outro artifício, sempre tentando superar a pontuação da partida anterior. Sendo assim, este trabalho apresenta o desenvolvimento de um jogo eletrônico inspirado em jogos musicais, aqui chamado de Guitar LED, que tem como objetivo criar uma plataforma independente que permita aos jogadores simular a experiência de tocar uma guitarra através de botões, exibindo a "partitura"da música através de uma matriz de LEDs. Para alcançar este objetivo, foi utilizado um Raspberry como estação base, controlando todo o núcleo do jogo e sistemas áudio-visuais, e dois controles-guitarra, tendo como central um Arduino cada.

1 Introdução

A palavra jogo vem do latim "*incus*", que pode ser traduzida como diversão ou brincadeira[1]. Jogar é sair da rotina e convergir as atenções para uma situação desafiadora; é sentir e pensar [2]. Jogar também promove o autoconhecimento, ajuda a vencer a timidez, cria sentimento de segurança e faz desaparecer o receio, além de ser uma forma de aprender a considerar os sentimentos e individualidades dos outros [3]. No caso dos jogos musicais, jogar gera a oportunidade de desenvolver

aptidões, capacidade de concentração, escuta, comunicação e até mesmo trabalho em grupo.

Neste contexto, o projeto Guitar LED foi desenvolvido visando criar uma plataforma independente que permita ao jogador simular a experiência de tocar uma guitarra, pressionando cinco botões, com o objetivo de acompanhar o ritmo da música imposto pelo jogo. A Figura 1 ilustra sua visão geral.

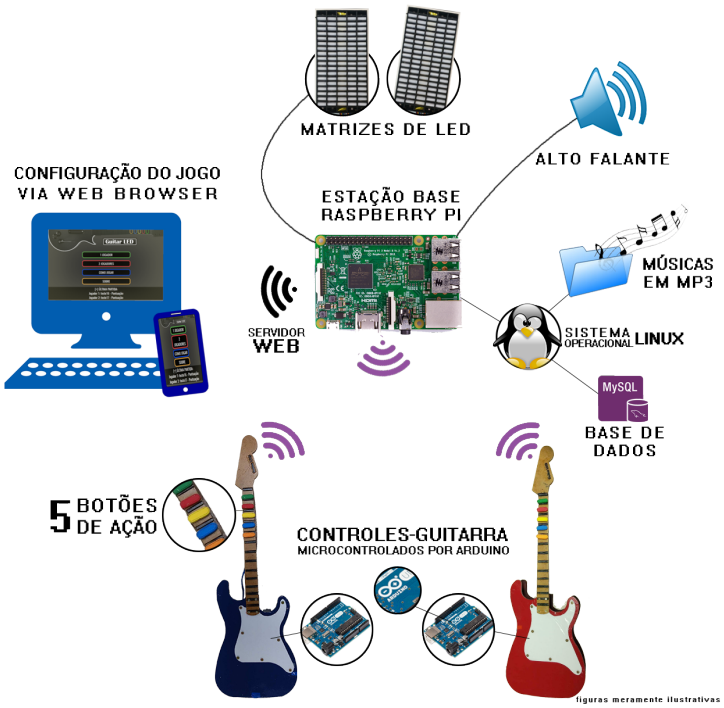


Figura 1: Visão Geral do Projeto. Fonte: Autoria Própria.

A particularidade do Guitar LED é exibir as informações durante a partida através de duas matrizes de LEDs, uma para cada jogador. As matrizes de LEDs são compostas por 100 LEDs que são dispostos em cinco fileiras, cada uma de uma cor diferente representando os botões do controle guitarra, com 20 LEDs cada. Possuem também uma estrutura de MDF que protege e separa cada um dos LEDs, evitando assim que suas luzes se interfiram.

Além das matrizes de LEDs, o jogo conta com uma interface *Web*, roteada pela estação base, que permite que o jogador configure a partida através da escolha da música, da dificuldade e do apelido, exibindo também os placares de partidas anteriores. As configurações realizadas, são armazenadas em uma base de dados, que se encontra na estação base, e permite que o programa principal tenha acesso a essas informações.

A estação base também armazena os áudios e arquivos com configurações dos botões para cada música, cuidando para que a matriz de LEDs e o alto falante fun-

cionem corretamente. Os controles possuem formato de guitarra e contém cinco botões pressionáveis de cores diferentes e um botão capacitivo, que simula a palheta.

Antes de iniciar o projeto realizou-se uma análise de requisitos, dividida entre requisitos funcionais e não funcionais para aumentar o rendimento durante o desenvolvimento, onde o resultado pode ser visto nas Tabelas 1 e 2.

Tabela 1: Requisitos Funcionais

Índice	**Requisitos Funcionais**
RF001	O dispositivo deverá suporta até dois jogadores.
RF002	O jogo deverá ser controlado através de controles sem fio.
RF003	O controle deverá conter botões físicos para a interação de usuário.
RF004	O "controle-guitarra" deverá fornecer ao usuário capacidade de confirmar início de partida, jogar, pausar, desistir e reiniciar a partida.
RF005	A interface *web* deverá ser capaz de configurar a partida (música e dificuldade) e exibir pontuações.
RF006	A interface *web* irá exibir uma lista das pontuações.
RF007	A interface *web* exibirá a pontuação da última partida em forma de placar.
RF008	O dispositivo deverá atuar no modo no-fail.
RF009	O dispositivo deverá ter duas matrizes de LED, uma para cada jogador.
RF010	O dispositivo deverá conter um alto falante de pelo menos 1 Watts.
RF011	A matriz possuirá as dimensões de 5 x 20 LEDs.

Tabela 2: Requisitos Não Funcionais

Índice	**Requisitos Não-Funcionais**
NF001	Os botões de jogo terão a combinação correspondente aos jogos que serviram de inspiração.
NF002	O banco de dados deverá ser escrito em MySql.
NF003	O canal de comunicação entre a interface *web* e o programa que controla o jogo será por bancos de dados MySQL.
NF004	O programa que controla o jogo será implementado utilizando C++ e orientação a objetos.
NF005	O dispositivo deverá dar suporte a arquivos compatíveis com Guitar Rage.
NF006	O jogo deverá oferecer quatro níveis de dificuldade.
NF007	Cada coluna da matriz possuirá LEDs de uma determinada cor.
NF008	O dispositivo usará MP3 como formato de arquivo de som executável.
NF009	O dispositivo usará arquivo de informação das notas compatível com o Guitar Rage.
NF010	O dispositivo funcionará na rede elétrica.

Este documento está organizado da seguinte maneira: a Seção 2 descreve os módulos de *hardware* e *software* que compõe a estação base do projeto; a Seção 3 apresenta os principais dispositivos utilizados, a integração da estação base com os módulos, o núcleo da estação base e os controles-guitarra; a Seção 4 lista os testes realizados, assim como as dificuldades encontradas e o orçamento; e a Seção 5 apresenta a conclusão resultante do desenvolvimento do projeto.

2 Módulos

Nesta seção são apresentados os módulos desenvolvidos, separados em *hardware* e *software*, que compõe a estação base do projeto Guitar LED, sendo estes os responsáveis pela comunicação com o ambiente externo à estação base.

2.1 Módulos de *Hardware*

Para que o Guitar LED cumpra com o seu objetivo de ser uma plataforma independente é necessário a exibição as notas durante a partida, bem como a reprodução do áudio do jogo sem a necessidade da conexão com dispositivos externos. Esta subseção apresenta o desenvolvimento dos módulos da matriz de LEDs e de áudio utilizados no projeto.

2.1.1 Matriz de LED

A matriz de LEDs é o destaque deste projeto, sendo ela a responsável por exibir a informação durante a partida para o jogador. Buscando a melhor experiência para o jogador, foi modelada uma estrutura em MDF, ilustrada na Figura 2, de forma que cada LED se encontre dentro de uma "célula" com um anteparo para tornar a luz uniforme. Cada matriz possui 100 células, dividas em 20 linhas e cinco colunas, onde cada coluna contém os LEDs de uma das cinco cores dos botões. Toda a estrutura da matriz foi modelada utilizando o *software* AutoCAD, e usinada com a máquina a comando numérico computadorizado (CNC).

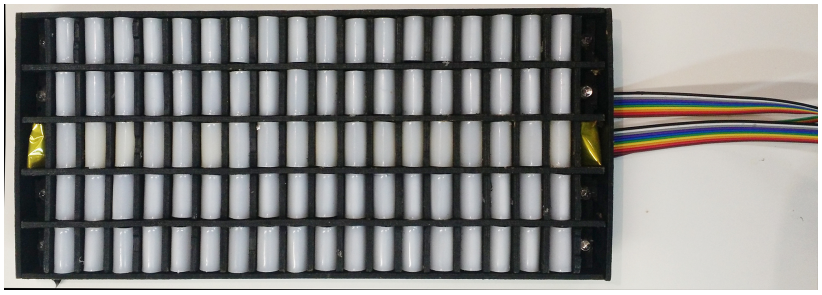


Figura 2: Matriz de LED. Fonte: Autoria Própria.

A eletrônica central de controle da matriz de LEDs é constituída por um circuito demultiplexador que utiliza os transistores BC548 e TIP122 como amplificador de corrente para as linhas e os transistores BC557 e o TIP127 como amplificador de corrente das colunas na matriz de LEDs. Os transistores TIP foram escolhidos para alimentar os LEDs pois transistores menores, como os BC548 e o BC547 não drenavam corrente suficiente para tornar os LEDs suficientemente brilhantes, mesmos os cálculos empregados coincidindo com os dados no datasheet destes.

Além do circuito demultiplexador, foi necessário utilizar o CI 4N25, pois é um componente capaz de separar a tensão de 5 volts utilizada pelo circuito dos 3 volts utilizados pelo Raspberry Pi 3. O acoplamento do circuito demultiplexador com o CI 4N25 foi realizado com o auxílio de um resistor de 470 ohms. O esquemático elétrico deste circuito é apresentando na Figura 3

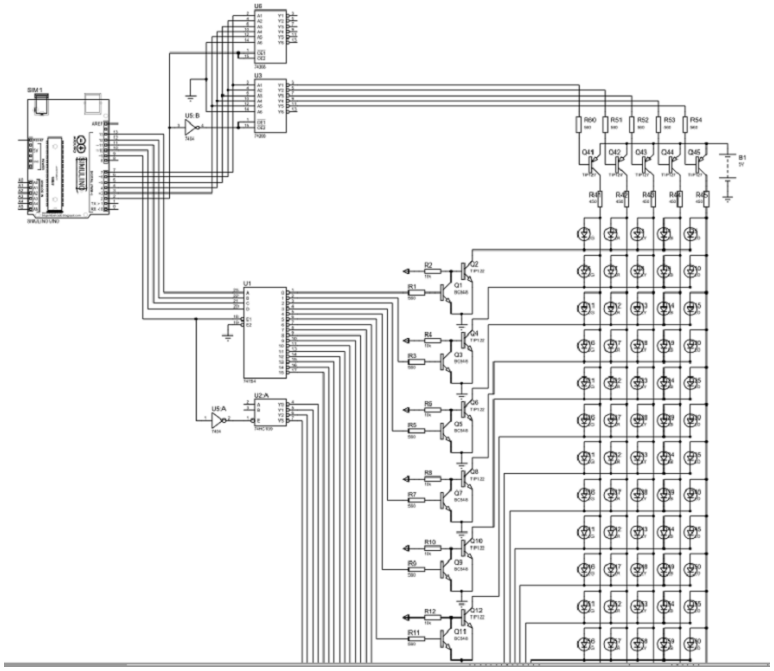


Figura 3: Circuito de controle da Matriz de LEDs. Fonte: Autoria Própria.

2.1.2 Áudio

Para continuar com o título de plataforma independente, o Guitar LED necessita de um sistema de áudio capaz de reproduzir as músicas contidas no jogo. Tendo em vista atender este requisito, foi desenvolvido um sistema de áudio estéreo, baseado no módulo amplificador Pam8403 com dois alto falantes conectados, como ilustra a Figura 4. Para a conexão com o Raspberry Pi 3, foi utilizada uma entrada do tipo P2.

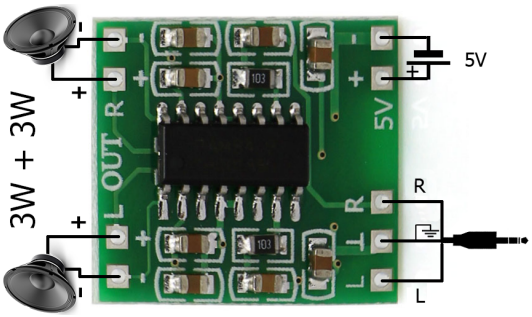


Figura 4: Amplificador PAM8403. Fonte: Autoria Própria.

2.2 Módulos de *Software*

Esta subseção apresenta a interface *web* e a base de dados, utilizadas para facilitar o funcionamento do Guitar LED.

2.2.1 Base de dados

Para que o projeto Guitar LED funcione de forma eficiente, a comunicação entre a interface *web* e a estação base precisa ser robusta, ou seja, ser contínua e não cair durante todo o processo. Visando resolver este problema, adotou-se uma base de dados simples, implementada através do MySQL, que atendesse as necessidades de ambas partes.

Para modelar a base de dados do projeto, utilizou-se o Diagrama Entidade-Relacionamento (DER), ilustrado na Figura 5. A entidade Dificuldade é uma tabela que consiste das dificuldades de jogo selecionadas pelos jogadores em cada partida através da interface web, atribuindo-se valor nulo ao atributo "Dif2" para o modo de um único jogador. A entidade *Score* é a tabela que armazena o nome dos jogadores inseridos através da interface web e guarda também as pontuações dos jogadores, escritas pelo núcleo do jogo ao final de cada partida. A entidade Musica, como o nome sugere, armazena o nome das músicas disponíveis para jogar. Estes nomes são obtidos na inicialização do núcleo do jogo, que verifica ao diretório destinado às músicas. A entidade Controle funciona como um mecanismo de acionamento do jogo, uma vez que sua chave é na realidade uma *flag* que indica a configuração de uma partida através da interface web, de modo que o núcleo de jogo pode verificá-la e iniciar a execução da partida corretamente.

Esta base de dados permite que a interface *web* e a estação base, apesar de serem implementadas em plataformas diferentes, se comuniquem de forma direta.

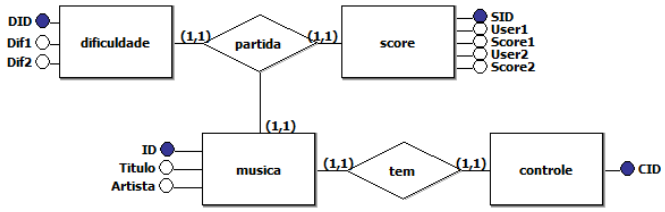


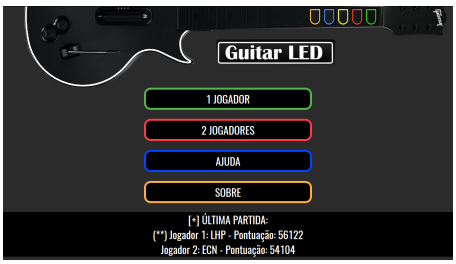
Figura 5: Diagrama Entidade Relacionamento (DER). Fonte: Autoria Própria

2.2.2 Interface Web

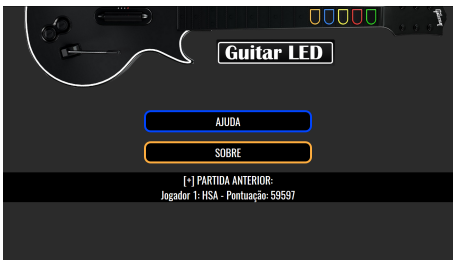
Através das matrizes de LEDs é possível visualizar as notas do jogo e seu ritmo, porém ela não é um meio ideal para que o usuário configure uma partida. Para tal, foi desenvolvida uma interface *web*, que é disponibilizada localmente via *WiFi*, de modo que usuários possam acessá-la através de dispositivos como *laptops* e *smartphones*.

A interface consiste em páginas HTML que utilizam a linguagem PHP para acessar a base de dados e inserir nela informações inerentes a partida (Figura 7.a). Desta forma, a estação base pode verificá-las de forma síncrona e inicializar corretamente o jogo. De forma semelhante, a interface *web* tem acesso as pontuações (Figura 7.b), que são geradas pelo núcleo do jogo. Também existe, através da interface, a possibilidade do jogador selecionar músicas para escutar antes de realizar a configuração de uma partida. A página foi estilizada através de comandos CSS para ter um *layout* consistente com o motivo do jogo, sendo que alguns comandos *JavaScript* permitem a exibição de mensagens de erro ao usuário.

A interface foi concebida juntamente à base de dados, de forma a permitir que apenas uma partida seja configurada por vez, exibindo as páginas e mensagens de erros adequadas ao usuário, como mostra a Figura 6.

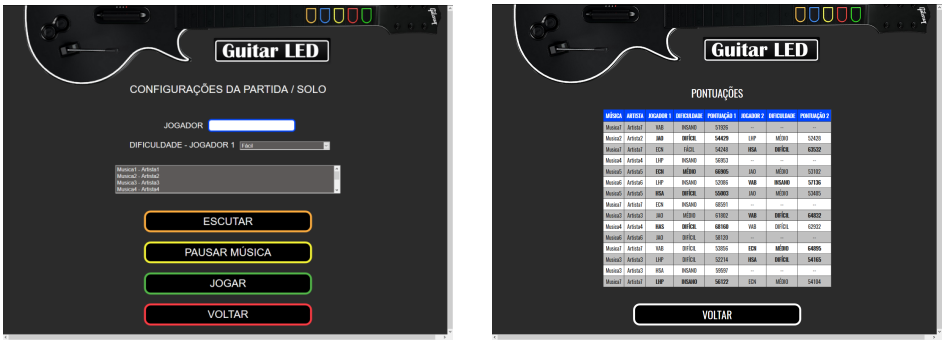


(a) Tela inicial da interface.



(b) Tela exibida durante uma partida ativa.

Figura 6: Tela Inicial da Interface *Web*. Fonte: Autoria Própria.



(a) Tela de Configuração de Partida: Solo. (b) Tela de exibição de pontuação.

Figura 7: Aspecto da Interface *Web*. Fonte: Autoria Própria.

3 Desenvolvimento

Nesta seção são apresentados os três principais componentes utilizados no projeto Guitar LED, sendo estes os responsáveis pela criação da estação base, do sistema embarcado e comunicação entre os dispositivos. Em seguida, também é apresentado o desenvolvimento da estação base e controle-guitarra do projeto.

3.1 Dispositivos Utilizados

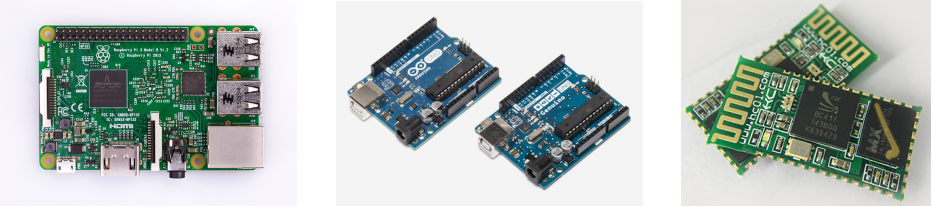
O Raspberry Pi 3 [4], ilustrado na Figura 8.a, foi escolhido por ser um microcontrolador capaz de suprir as demandas da estação base do projeto. Ele possui 40 pinos de entrada/saída, quatro portas USB 2.0 e um armazenamento realizado através de um cartão micro SD, além de ter a vantagem de consumir uma quantidade menor de energia se comparado com seus antecessores.

Este microcontrolador é capaz de executar o sistema operacional Xubuntu, fato que tornou-se crucial para o projeto, pois a disponibilidade de recursos e a execução nativa da linguagem C/C++ proporcionam um ambiente ideal para o desenvolvimento do *software* do projeto.

O microcontrolador escolhido pela equipe para o sistema embarcado são os dispositivos Arduino[5], ilustrados na Figura 8.b. Este recurso foi utilizado para compor os controles, pois é capaz não só de tratar os botões acionados pelo usuário, transformando o estado das chaves em informação, como também transmitir esta informação para a estação base.

A comunicação entre os controles-guitarra e a estação base é realizada através do protocolo *bluetooth*, pois este antede as necessidades do projeto e possui o melhor custo-benefício.

Para estabelecer a comunicação são utilizados dois módulos *bluetooth* (Figura 8.c), cada um deles se localiza em um dos controles-guitarra, para enviar as informações obtidas através do Arduino UNO sobre os botões até a estação base (Raspberry), onde estas informações serão processadas da forma correta.



(a) (b) (c)

Figura 8: Dispositivos utilizados no projeto. (a) Raspberry Pi 3. Fonte: Site Oficial do Raspberry. (b) Arduino UNO. Fonte: Site Oficial do Arduino (acessado em 02/04/2017). (c) HC-05. Fonte: Loja Virtual HC01 (acessado em 10/06/2017).

3.2 Estação Base

Criada a partir do Raspberry Pi 3, a estação base é a parte central do projeto, armazenando a base de dados e a interface *web*, e controlando o sistema de áudio e das matrizes de LEDs através de suas conexões. Em outras palavras, a estação base (Figura 9) é o local onde os módulos de *software* e *hardware*, citados anteriormente, são conectados, possibilitando o funcionamento em conjunto para a existência do jogo.

Figura 9: Estação Base. Fonte: Autoria Própria

Além dos módulos que comunicam com o ambiente externo (matriz de LEDs e sistema de áudio), a estação base também possui o núcleo do jogo, onde ocorre todo o controle sobre o sistema. Na estação base é onde ocorre a integração de todos os subsistemas do jogo em si, responsáveis em realizar as funções de controle e execução do jogo. A seguir, são descritas as implementações de cada parte que integram o núcleo do jogo.

3.2.1 Controle do *Buffer* dos LEDs

Para evitar problemas de sincronismo ou erro na exibição de informações na matriz de LEDs, criou-se uma classe apenas para acessar à memória que representa os LEDs da matriz. Sendo assim, também foi necessário criar uma classe de *buffer* para interagir com o *display* de LEDs mais facilmente.

O *buffer* possui métodos com os quais é possível zerar a matriz de LEDs, ligar todos os LEDs da matriz, ler as teclas apertadas para "imprimi-las", inserir linha a linha e retornar a última linha da matriz (notas que deveriam estar acesas). Além disso, esta classe gerencia um semáforo, para evitar que existam conflitos durante o acesso simultâneo de dois componentes de *software* a uma matriz.

3.2.2 MP3 Player

Utilizando as bibliotecas *libao* e *libmpg123*, disponíveis no repositório *apt-get* do *Linux*, e a biblioteca *pthreads*, sendo esta nativa nos sistemas Unix, vindas das normas POSIX, foi desenvolvido um *software* em linguagem $C++$ capaz de realizar a manipulação e execução destes arquivos de áudio [6], [7].

Para iniciar a execução de uma música com esta classe, o parâmetro de entrada necessário seria a *string* que contém a informação do diretório de onde o arquivo de áudio está localizado. O nome do arquivo em si não é necessário pois, na estrutura de arquivo, as informações nos subdiretórios de músicas são padronizadas. Existem métodos para interromper e parar a execução da música, que não possuem nenhum parâmetro de entrada. Também foram implementadas as funções *Beep* e *Beep1*, ambas destinadas à executar as faixas de bipe no *player* para sinalizar a inicialização do jogo.

A execução de músicas ocorre em paralelo ao sistema de comando, sendo a comunicação entre elas ditada por variáveis estáticas que ambas as *threads* possuem direito de acesso.

3.2.3 Interfaceamento C - SQL

Enquanto as partidas do jogo são configuradas através da interface *Web*, o núcleo do jogo está rodando na estação base. Desta forma o principal meio de comunicação entre ambos é a base de dados, onde os dados de inicialização das partidas (como a música e dificuldades selecionadas) e das pontuações ao final de cada jogada são escritos. Para tal utilizou-se a biblioteca *MySQL Connector*, disponibilizada pela *Oracle*.

O interfaceamento consiste numa classe em C++ que contém métodos para a obtenção dos dados da partida. Estes dados são: a música selecionada para escutar, a música selecionada para partida, a *flag* de controle que indica se o usuário deseja iniciar a partida ou não, a dificuldade selecionada pelos jogadores, além de métodos para a escrita das pontuações no banco de dados. Os métodos permitem o uso dos dados no formato de *strings* sem a preocupação com as consultas à Base de Dados,

deixando-as "transparentes"no momento da programação dos algoritmos de ritmo e controle dos LEDs.

3.2.4 Algoritmo de Ritmo

O projeto Guitar LED optou em adotar a estrutura de representação de notas no formato do jogo *open source Guitar Rage*. A estrutura das notas arquivadas em .dat (Figura 10) representam as notas individualmente, facilitando sua identificação e filtragem dependendo da aplicação.

```
1  [NOTE]
2  <DIFFICULTY>:3
3  <TIME>:3.764706
4  <CLICKTIME>:3840
5  <DURATION>:0.117647
6  <CLICKDURATION>:120
7  <TRACK>:0
8  [/NOTE]
```

Figura 10: Estrutura das Notas. Fonte: Autoria Própria.

No caso do projeto Guitar LED, somente as estruturas de *DIFFICULTY* (dificuldade do jogo que a nota pertence), *TIME* (o tempo em que a nota ocorre em sincronia com a música), *DURATION* (duração de uma nota) e *TRACK* (posição que uma nota ocorre) foram utilizadas.

Com a estrutura bem definida, foi desenvolvido um *parser* que filtra as notas de acordo com a dificuldade, sendo que em seguida, as notas são alocadas em um vetor. Como muitas notas ocorrem em um mesmo período de tempo, montaram-se “acordes”, vetores organizados pelo tempo de acontecimento.

Todos os acordes são inseridos no *buffer* matricial, descrito na Seção 3.2.1, num determinado tempo antes de seu tempo de execução, sendo isso variável de acordo com a velocidade da música. Para realizar a execução destes acordes, um *timer* preciso é necessário para que ocorra em sincronia com a música. Possuindo o *timer* e obtendo o menor valor de tempo de ocorrência entre notas, é possível determinar um valor de passo aceitável para que a música seja executada na matriz de forma fluída e intuitiva sem confundir o jogador.

3.3 Controle-Guitarra

Antes de iniciar a implementação do *hardware* e do *software* do controle, foi necessário construir a estrutura física do mesmo. Como base para o modelo do controle do Guitar LED foi escolhida uma imagem de guitarra do estilo *Stratocaster*. Neste caso, o clássico modelo desenvolvido pela fabricante Fender foi usado como base para modelagem no *software* AutoCAD, sendo usinada em uma máquina a comando numérico computadorizado (CNC). A Figura 11 ilustra o resultado final da confecção dos controles-guitarra.

Figura 11: Controles-guitarra. Fonte: Autoria Própria.

A eletrônica do controle-guitarra (Figura 12) está espalhada por toda sua estrutura mecânica. Em seu braço, existem 15 botões do tipo tátil, divididos em cinco colunas, onde cada uma representa uma das cores do jogo. Já o seu corpo possui um botão capacitivo que simula o local das palhetadas e um botão tátil para inicializar e pausar o jogo. Nele, também se encontra o Arduino, dispositivo onde são ligados todos os componentes da guitarra, de forma a criar um vetor com seus sinais recebidos, transmitindo-o através do módulo *bluetooth* para a estação base.

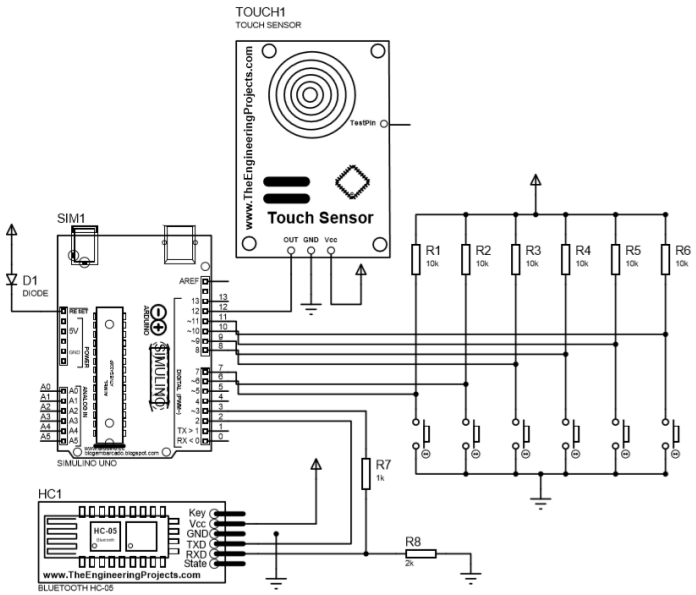


Figura 12: Diagrama elétrico do controle-guitarra. Fonte: Autoria Própria

3.4 Sistema de Pontuação

Apesar do sistema de pontuação fazer parte do núcleo da estação base, seu funcionamento depende diretamente de informações enviadas pelos controles-guitarra. Deste modo, o sistema de pontuação é onde são realizadas as checagens dos controles, assimilando ao que está sendo exibido na matriz de LEDs, em outras palavras, ele integra a estação base com os controles-guitarras.

O controle-guitarra envia ao Raspberry, através de *bluetooth*, um *byte* de caracteres, seguindo uma estrutura proposta que representasse os botões pressionados no controle, para facilitar a transmissão e verificação de dados. A estrutura pode ser observada na Figura 13.

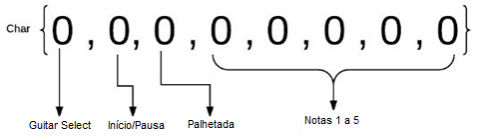


Figura 13: Estrutura do *byte* enviado. Fonte: Autoria Própria.

Como a informação está sendo enviada numa frequência considerada instantânea, é possível comparar as posições das notas com o *buffer* ditado pelo algoritmo de ritmo. Caso as entradas das notas coincidam, o jogador é recompensado com 50 pontos por número de notas e o jogo identifica que ele está em um *streak*, uma sequência de acertos. Essa verificação é necessária para identificar quando ocorrem notas longas, que necessitam ser pressionadas, mas não palhetadas mais de uma vez, pois caso haja uma palhetada ou uma alteração de botões, a recompensa pela nota é perdida. Ao final da partida a pontuação final é enviada para o banco de dados, atualizando o registro da partida, juntamente com o nome do usuário, e as escolhas de música e de dificuldade.

4 Resultados

Esta seção apresenta as dificuldades encontradas juntamente com as soluções adotadas durante o desenvolvimento e integração do sistema, os resultados obtidos durante os testes finais realizados e o orçamento do projeto realizado.

4.1 Dificuldades Encontradas

Durante a primeira integração de *software*, foi encontrado um problema com o armazenamento das *strings* resultante dos métodos de consulta SQL, sendo estas armazenadas em espaços de memória referenciados por ponteiros. O problema foi solucionado com o armazenamento dos resultados em objetos do tipo *std::string*.

A segunda integração não trouxe nenhum problema real, apenas notou-se a falta de um botão na interface *web* que permitisse interromper a música após

seleciona-la para ouvir, sendo assim, a interface foi ajustada e recebeu um novo botão.

Na terceira integração, verificou-se um problema na sincronização entre a indicação de início de partida pela interface *web* e a consulta da música escolhida pelo algoritmo de jogo. Para tal fez-se um atraso no algoritmo, suficiente para a escrita de todas as informações de partida no banco de dados.

4.2 Testes Finais

Após a conclusão do desenvolvimento do projeto, foi medido o desempenho do jogo criado no cumprimento do seu objetivo mais importante: entreter os jogadores. Para isto, foi realizada uma bateria de testes a fim de analisar dois aspectos essenciais: adaptação aos controles-guitarra e o conforto durante a partida. Os testes realizados consistiram na execução de partidas com a participação de jogadores com diferentes níveis de habilidade nos jogos usados como base deste projeto e, conhecimento sobre o funcionamento do projeto Guitar LED.

Para o primeiro aspecto analisado, como esperado, o resultado obtido foi que jogadores mais familiarizados com os "jogos base" tiveram menos dificuldade para se adaptar aos controles do jogo se comparados aos jogadores menos familiarizados. Também notou-se que jogadores com menos experiência, esqueciam de palhetar as notas, invalidando-as na contagem da pontuação.

No aspecto de conforto, durante as primeiras partidas, houveram relatos sobre a localização dos botões no braço dos controles-guitarra. Com isso, nestas partidas, os jogadores tendiam a ter pequenos momentos de interrupção para descansar seus dedos. Para melhorar o conforto, os botões foram realocados de forma que ficassem mais próximos. Após isso, houveram novos testes, onde relatos de melhorias na questão conforto foram realizados.

4.3 Orçamento

A Tabela 3 apresenta os custos totais para a elaboração deste projeto.

Tabela 3: Orçamento.

Item	Qtd.	Valor Unitário (R$)	Valor Total (R$)
Kit 2 Alto Falantes [3 W – 4 Ohms]	1	19,99	19,99
Arduino Uno R3 + Pinos	2	19,60	39,60
Chapa de Acrílico	1	26,00	26,00
Componentes Eletrônicos	Vários	223,14	223,14
LEDs - Várias Cores	200	0,19	37,50
MDF 3mmx500mmx800mm	1	7,00	7,00
MDF 9mmx500mmx800mm	5	10,00	50,00
Modulo Amplificador Som Estéreo	1	2,50	2,50
Módulo *Bluetooth* Hc-05	2	19,80	39,60
Tintas e Verniz	6	23,00	138,00
Usinagem - Acrílico	1	20,00	20,00
Usinagem - MDF	5	5,00	25,00
Raspberry Pi 3	1	179,00	179,00
Outros	Vários	52,23	52,23
TOTAL			**859,56**

5 Conclusão

A execução do Guitar LED mostrou-se suficiente para trazer a experiência dos jogos de referência. Como proposto, a interface web permite a configuração das partidas através de dispositivos do cotidiano, capazes de conectar-se à rede WiFi, e o banco de dados funcionou de fato como um canal de comunicação com a estação-base. As cadeias de notas dos jogos de referência foram "traduzidas" com sucesso para a matriz de LEDs, e também há o suporte aos dois jogadores, para reforçar a experiência do jogo original, porém com a independência de um *videogame* ou computador.

Uma decisão crucial para o sucesso do projeto, foi sua "modularização" durante o desenvolvimento. Com isto, foi possível desenvolver independentemente boa parte do projeto, sem a necessidade de aguardar o término de outro módulo. Como resultado, após a conclusão de cada etapa, era possível realizar uma bateria de testes para verificar seu funcionamento, visando buscar erros e corrigi-los imediatamente. Sendo assim, na integração final, muitos dos erros já estavam solucionados, diminuindo grande quantidade dos problemas a serem enfrentados.

Apesar do funcionamento do Guitar LED ser de acordo com o desejado, visto os requisitos propostos durante o seu projeto, após sua conclusão verificou-se que existem melhorias que podem ser realizadas visando melhorar a experiência do jogador. A primeira proposta de melhoria encontrada durante o desenvolvimento foi a de oferecer a opção de cadastro ao jogo, assim o jogador cadastrado teria acesso à uma nova área na interface *web* onde poderia acompanhar seu desempenho, e até mesmo ser criado um modo de treinamento para melhorar o desempenho do jogador. Outra melhoria proposta é a facilitação na inserção de novas músicas ao jogo. A ideia desta proposta é de possibilitar a renovação do repertório quando conectasse um *pendrive* à estação base do Guitar LED, sendo necessário que os

arquivos estivessem todos no formato correto.

Contudo, mesmo existindo modos de melhorar a plataforma, o Guitar LED obteve um desempenho excelente, oferecendo a releitura de um jogo clássico de forma criativa e inovadora, proporcionando ao jogador uma experiência única.

Agradecimentos

Agradecimentos ao GIP3D por permitir o uso da máquina de CNC, essencial para a realização deste projeto e ao professor João Alberto Fabro pelo empréstimos de componentes e por ceder o espaço (LASER) para realização do projeto.

Referências

[1] Ana Cristina Alves de Jesus. *Histórico dos Jogos na Educação*. Rio de Janeiro: Brasport, 2010.

[2] Inaldo Mendes de Mattos Junior. *Jogos Musicais: Implicações e Incentivo Baseados em um Relato de Experiência*. São Luis, 2014.

[3] Ger Storms. *100 Jogos Musicais*. Rio Tinto – Portugal, Asa, 1996.

[4] Pedro Cipoli. Saiba tudo sobre o raspberry pi 3 e o que ele representa para o mercado. `https://canaltech.com.br/materia/processador/saiba-tudo-sobre-o-raspberry-pi-3-59065/`. Publicado em 10 de março de 2016 às 17h30.

[5] Arduino. Arduino UNO & Genuino UNO. `https://www.arduino.cc/en/main/arduinoBoardUno`.

[6] Stan Seibert. Libao Documentation. `https://xiph.org/ao/doc//`, 2001.

[7] Thomas Orgis & Michael Hipp. Api documentation for libmpg123 and libout123. `https://www.mpg123.de/api/`.