

Badly Programmed Ice Cream

Arthur Araújo - 232000472

Arthur Silva - 232000490

João Carlos - 232009511

University of Brasília, Dept. of Computer Science, Brazil

Abstract

O projeto "Badly Programmed Ice Cream" foi criado como um remake do jogo Bad Ice Cream feito em Assembly RISC-V utilizando o RARS como ferramenta de desenvolvimento. O projeto foi desenvolvido durante as aulas da disciplina de Introdução a Sistemas Computacionais. RISC-V é uma arquitetura de conjunto de instruções (ISA) baseada em princípios de design de processadores de redução de conjunto de instruções (RISC). Ela é caracterizada por ser livre e aberta, o que significa que seu design está disponível publicamente e pode ser usado livremente por qualquer pessoa ou organização. Foi desenvolvida na Universidade da Califórnia, em Berkeley, e é amplamente reconhecida por sua flexibilidade e eficiência.

1 Introdução

"Bad Ice-Cream" é um jogo de arcade desenvolvido pela Nitrome, lançado em 10 de Dezembro de 2010. No jogo, os jogadores controlam um personagem que deve coletar frutas em um labirinto enquanto evita inimigos. O objetivo é coletar todas as frutas para completar o nível. O jogo é conhecido por seu estilo de arte pixelizada e jogabilidade simples, mas desafiadora. No jogo original, o jogador deve escolher um dos sorvetinhos para controlar, sendo capaz de criar blocos de gelo para bloquear inimigos ou abrir caminho através do labirinto, mas também devem evitar que os inimigos quebrem os blocos de gelo ou os atinjam diretamente. Novos desafios e elementos são introduzidos ao longo das fases, tornando cada nível mais complexo e exigindo estratégia e habilidade para serem concluídos.



Figure 1: Jogo original Bad Ice Cream

Na seção 2 será apresentada a metodologia utilizada. A seção 3 apresenta os resultados obtidos. A seção 4 conclui este trabalho.

2 Metodologia

Inicialmente, o projeto foi desenvolvido no RARS, apresentado pelo professor Lamar. Contudo, durante o semestre, atualizações

foram feitas e a testagem passou a ser feita com a ferramenta FPGRARS criada pelo ex-aluno Leonardo Riether. A ferramenta facilita a execução e garante uma velocidade superior em comparação ao montador original RARS.

De maneira análoga aos projetos de alunos anteriores, foi implementado um sistema baseado em matrizes, as quais determinam o comportamento posicional do jogador, como também dos inimigos e dos coletáveis.

2.1 Planejamento

Toda comunicação durante o desenvolvimento do projeto foi realizada via Discord e Whatsapp. O projeto foi iniciado no dia 12/11 e finalizado no dia 21/12. Ao longo de cada semana, houveram reuniões para organização e definição de tarefas semanais. Para o versionamento de código, foi utilizado o GitHub, o qual possibilitou uma melhora significativa na organização e implementação de cada etapa do jogo.

Cada sprite utilizada para representação visual no jogo foi feita em conjunto, caracterizada pelas artes do aluno Arthur Araújo.

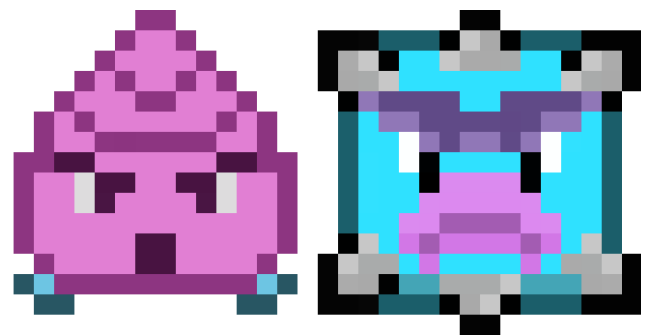


Figure 2: Sprites do player e do inimigo

2.2 Renderização

Para renderizar uma imagem no bitmap display do RARS/FPGRARS, foi necessário entender como os dados são interpretados. O bitmap consiste em dois grandes vetores de bytes, cada um representando uma camada de cor na tela. Cada byte do vetor codifica uma cor específica, e a tela tem uma resolução de 320x240 pixels (resolução 4:3). A interpretação dos dados do bitmap é feita de forma sequencial, começando pelo primeiro byte (pixel) na quina superior esquerda da tela e avançando ordenadamente até atingir a quina inferior direita, pulando para a próxima linha a cada 320 pixels. Nesse viés, com base no entendimento do funcionamento do bitmap display, e no tutorial feito pelo ex-aluno Davi Paturi, foi possível implementar o sistema inicial de renderização do mapa e do player.

3 Resultados Obtidos

Apesar de algumas dificuldades enfrentadas durante o processo de desenvolvimento, é válido afirmar que obtivemos um resultado final geral satisfatório em relação ao projeto. De uma forma simples, a jogabilidade foi aplicada com todas as ferramentas disponibilizadas pelo Assembly RISC-V. Seguem abaixo alguns problemas e desafios que enfrentamos:

- A dificuldade inicial foi entender como começar, qual o primeiro passo? A partir dos vídeos de ex-alunos e algumas monitorias, essa etapa foi vencida.
- Logo em seguida, nossa dificuldade era implementar os coletáveis e impedir que o jogador "quebrasse" o jogo invadindo regiões que não devia, problema que foi solucionado com a criação da matriz de representação.
- Implementar música no RARS foi um desafio que infelizmente não foi possível concluir devido ao tempo insuficiente conforme a entrega final se aproximava.
- A mecânica do gelo foi um grande desafio, visto que gerava muitos bugs, alguns deles ainda ocultos.

4 Conclusão

Desse modo, é visível que o projeto não foi uma tarefa fácil para alunos do primeiro semestre, e, apesar das dificuldades, o resultado foi muito gratificante. Desde o início, sabíamos que desenvolver um jogo em uma linguagem de baixo nível seria uma tarefa árdua. Além disso, o aprendizado foi coletivo e guiado por todos os conteúdos e materiais disponibilizados em inúmeras fontes, como repositórios antigos do github, slides do professor e vídeos escondidos no youtube.



Figure 8: Tela de Game Over

References

<https://github.com/victorlisboa/LAMAR>
<https://github.com/LeoRiether/FPGRARS>
Aulas do professor Marcus Vinicius Lamar
Computer Organization and Design RISC-V Edition, por David A. Patterson e John LeRoy Hennessy