

# Samsara: A canção das estrelas

Carlos Manoel de Sousa Lima  
João Pedro Rodrigues de Souza  
Karolina Caldeira Alves do Nascimento  
Marianna de Castro Gilberto Penha Ferreira

Universidade de Brasília, Depto. Ciência da computação, Brasil

## Resumo

Este trabalho documenta o processo de desenvolvimento de “Samsara: A canção das estrelas”, um jogo de aventura com elementos abstratos programado inteiramente em linguagem Assembly utilizando a arquitetura RISC-V 32 bits através do simulador FGPRARS. O projeto foi desenvolvido como trabalho final da disciplina de Introdução aos Sistemas Computacionais(ISC), com o objetivo de aplicar na prática os conceitos de programação de baixo nível. O jogo acompanha a protagonista Samsara em uma jornada através de mapas imaginários, onde utiliza uma flauta mágica que emite notas musicais como projéteis para enfrentar inimigos baseados em peças de xadrez.

**palavras-chave:**Assembly, RISC-V, FGPRARS, jogo, programação baixo nível.

## Introdução

A linguagem Assembly representa um nível fundamental de programação, oferecendo controle direto sobre o hardware. O desenvolvimento de “Samsara: A canção das estrelas” surgiu para consolidar conhecimentos sobre arquitetura RISC-V através de um projeto prático significativo. A escolha por criar um jogo completo permitiu explorar aspectos diversos da programação de sistemas.

A narrativa segue Samsara, uma criança que recebe de uma estrela cadente uma flauta mágica capaz de acessar o reino dos sonhos. Neste mundo, ela deve coletar carneiros mágicos enquanto enfrenta criaturas baseadas em peças de xadrez, acompanhada por seu gato. A história utiliza conceitos filosóficos sobre consciência e realidade.

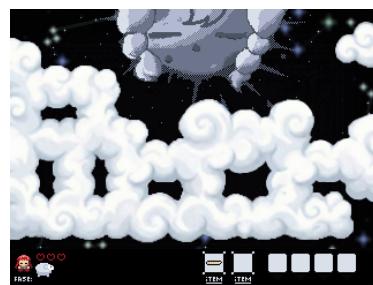
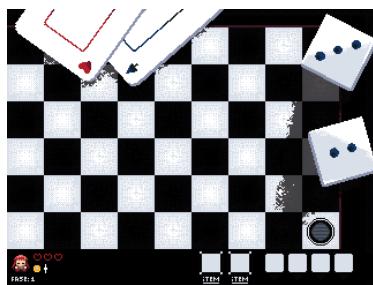


# Metodologia

O projeto foi desenvolvido utilizando o simulador FGPRARS(RISC-V Assembly and Runtime Simulator). Esta ferramenta oferece funcionalidades essenciais: framebuffer para renderização pixel a pixel em resolução 320x240, interface MMIO para captura de entrada de teclado, e capacidade de reprodução de áudio MIDI.

## Recursos gráficos

As imagens do jogo foram criadas em software de edição gráfica seguindo estética abstrata que invoca elementos de sonho. As sprites foram exportadas em PNG e convertidas diretamente para arquivos .s utilizando o conversor png2oac, que transforma os dados de pixels em diretivas Assembly compatíveis com o FGPRARS.



## Sistema de colisão

O sistema de colisão implementado utiliza abordagem baseada em grade onde o mapa é dividido em blocos de 8x8 pixels. Cada bloco pode ter um de 4 valores:

- 0: Espaço livre transitável
- 1: parede impenetrável
- 3: porta para transição entre cenários
- 4: loja que aciona interface de compras

A detecção converte coordenadas de pixel em coordenadas de grade (dividindo por 8) e consulta o valor da matriz de colisão.

## Inimigos

No jogo foi implementado cinco tipos de inimigos inspirados em peças de xadrez:

- Cavalo: Se move em padrão de “L”
- Bispo: Desliza diagonalmente
- Torre: Se movimenta na horizontal
- Dama: Persegue o jogador e precisa de doze acertos para ser derrotada
- Carneiro: fogeativamente do jogador



## Música

A trilha sonora utilizada é La Valse d'Amélie – Version Orchestre, de Yann Tiersen. A conversão da música foi realizada através de um script automatizado em Python criado por nós, responsável por transformar cada nota da composição em instruções MIDI no formato exigido pelo FPGRARS, gerando diretamente o código Assembly correspondente à música.

## Resultados obtidos

O desenvolvimento apresentou diversos obstáculos característicos da programação de baixo nível. A limitação de registradores foi um desafio constante, exigindo gestão rigorosa de cada um deles para otimizar o uso dos recursos disponíveis. O debugging foi particularmente desafiador, exigindo execução passo a passo do código e inspeção manual de registradores para identificar problemas.

As branches no RISC-V possuem alcance limitado de apenas 12 bits, o que frequentemente exigiu reorganização do código para manter os saltos dentro do alcance permitido. O gerenciamento de memória mostrou-se ainda mais complexo devido ao grande volume de sprites, buffers e estruturas que precisavam coexistir simultaneamente. Isso requereu planejamento meticuloso, implementando estratégias como reutilização de buffers temporários e compartilhamento de dados entre entidades similares.

Coordenar múltiplos sistemas (renderização, colisão, movimento de inimigos) foi outro ponto crítico, exigindo comunicação explícita através de convenções estritas sobre quais registradores utilizar e como organizar as regiões de memória. Essa coordenação foi fundamental para garantir que todos os subsistemas funcionassem harmoniosamente sem conflitos.

Apesar das dificuldades, o projeto foi concluído incorporando todas as funcionalidades planejadas: sistema completo de renderização com sprites animados, três mapas distintos, sistema robusto de colisão, cinco tipos de inimigos com comportamentos únicos, sistema funcional de loja com economia baseada em moedas, HUD informativo, e integração de trilha sonora com MIDI.

## Conclusão

O desenvolvimento de “Samsara: A canção das estrelas” representou uma experiência desafiadora e pedagogicamente rica em programação de baixo nível. O projeto demonstrou que é possível criar aplicações interativas complexas em assembly que combinam aspectos técnicos robustos com expressão artística.

A implementação dos inimigos com comportamentos distintos baseados em peças de xadrez criou variedade interessante de gameplay. A inversão de dinâmica no terceiro mapa, onde o objetivo muda de combate para captura, demonstra como mecânicas simples podem ser recontextualizadas.

Este trabalho serviu como aplicação prática extensiva dos conceitos estudados na disciplina de Introdução aos Sistemas Computacionais, proporcionando compreensão

profunda de arquitetura de computadores. Todos os requisitos estabelecidos para o projeto foram cumpridos.

## Referências

FPGRARS - Simulador RISC-V com suporte gráfico e MIDI:

<https://github.com/LeoRiether/FPGRARS>

LAMAR - Simulador educacional para RISC-V:

<https://github.com/victorlisboa/LAMAR?tab=readme-ov-file>

Gerenciador de Conversão - Ferramenta para converter imagem em dados Assembly:

<https://github.com/gss214/Gerenciador-de-Conversao>

img2riscv - Conversor de imagem BMP para Assembly RISC-V:

<https://github.com/mateusap1/img2riscv>

png2oac – Conversor de PNG para Assembly:

<https://github.com/ABMHub/png2oac>

Playlist sobre RISC-V e arquitetura de computadores:

<https://www.youtube.com/playlist?list=PLLOKob75DU32afhLBN5nY2KzOJ5k6lw-Q>

Playlist sobre projetos práticos em Assembly:

<https://www.youtube.com/watch?v=AGLKNB2pC6E&list=PLLOKob75DU3389JeYb-z-N5KBbwNWpa&index=1>

Yann Tiersen – La Valse d'Amélie (Version Orchestre):

▶ [La valse d'Amélie \(Version orchestre\)](#)