PCS3732 - Laboratório de Processadores

ARM-Playground

Lucas Alexandre Tavares - 11322900
Otávio Vacari Martins - 11808130
Silas Lima e Silva - 11262233

São Paulo
2023

1. Objetivo

Nosso projeto tem como principal objetivo facilitar o aprendizado básico do usuário sobre as instruções ARM32. Para isso, nosso grupo buscou abordar tanto a parte teórica, quanto a parte prática. Na próxima seção, detalharemos cada abordagem.

2. Implementação

O desenvolvimento do projeto foi feito inteiramente em *Python3*, buscando principalmente aproveitar a praticidade da linguagem para que a implementação do nosso projeto e eventuais *updates* sejam feitos de maneira simples e eficiente.

A execução do projeto é feita diretamente no terminal, onde buscamos atingir um visual mais semelhante ao que geralmente é encontrado quando vamos programar em assembly. Apesar de representar uma simplificação (comparado a um eventual desenvolvimento de telas), pensamos na experiência do usuário neste caso, criando um menu com *inputs* para selecionar dentre as opções disponíveis e tratamento de erros para evitar *inputs* inválidos.

2.1. Execução do programa

Para executar o programa, basta rodar o arquivo *main.py*. Ele importa as funções e constantes dos demais arquivos, e executa tudo no mesmo terminal. Para encerrar a execução do programa, basta selecionar a opção 'X' no menu principal.

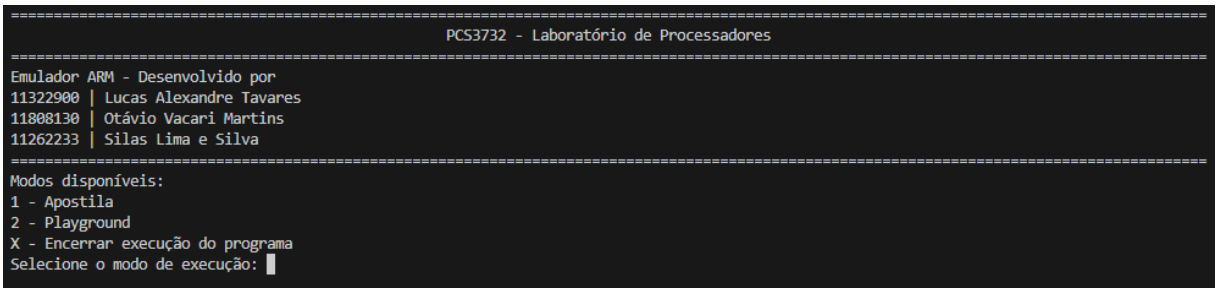

Imagem 1: Menu inicial do programa

2.1.1 Modo 1 - Apostila

Neste primeiro modo, listamos as principais instruções de ARM32, tais como as seguintes informações que julgamos necessárias:

- Mnemônico da instrução
- Descrição da instrução
- OPCode (quando aplicável)
- Sintaxe da instrução
    - Explicação de cada operando utilizado na instrução

Como próximos passos para nosso projeto, vemos tanto a possibilidade de adicionar novas instruções dentro do nosso arquivo *instrucoes.py*, como também adicionar mais informações a respeito das instruções.

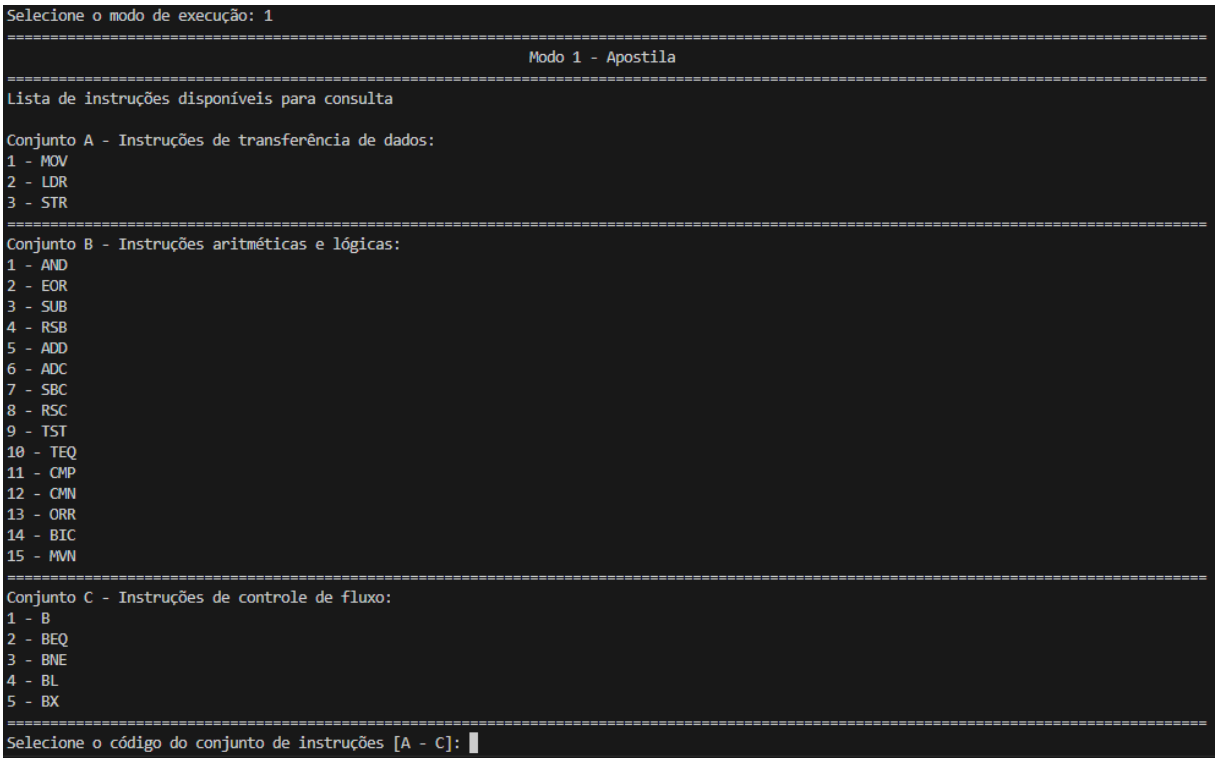

Imagem 2: Menu do modo 1 - Apostila

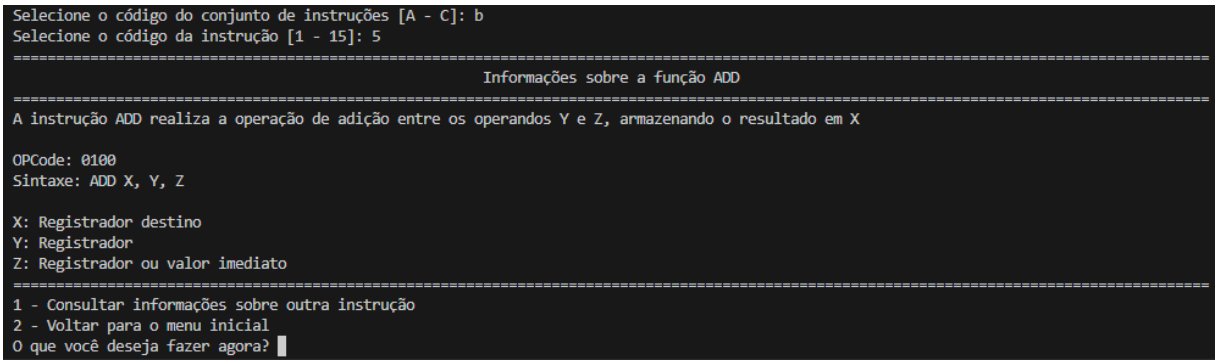

Imagem 3: Exemplo de saída ao selecionar a instrução B5

2.1.1.1. Instruções disponíveis para consulta

- Instruções de transferência de dados
    - MOV
    - LDR
    - STR
- Instruções aritméticas e lógicas
    - AND
    - EOR
    - SUB
    - RSB

    - ADD
    - ADC
    - SBC
    - RSC
    - TST
    - TEQ
    - CMP
    - CMN
    - ORR
    - BIC
    - MVN
- Instruções de controle de fluxo
    - B
    - BEQ
    - BNE
    - BL
    - BX

2.1.2. Modo 2 - Playground

Neste segundo modo, oferecemos aos usuários um ambiente de simulação onde eles podem executar instruções ARM32 e observar os resultados em tempo real. O Playground oferece uma experiência prática para entender o funcionamento das instruções e como elas afetam os registradores e a memória do sistema.

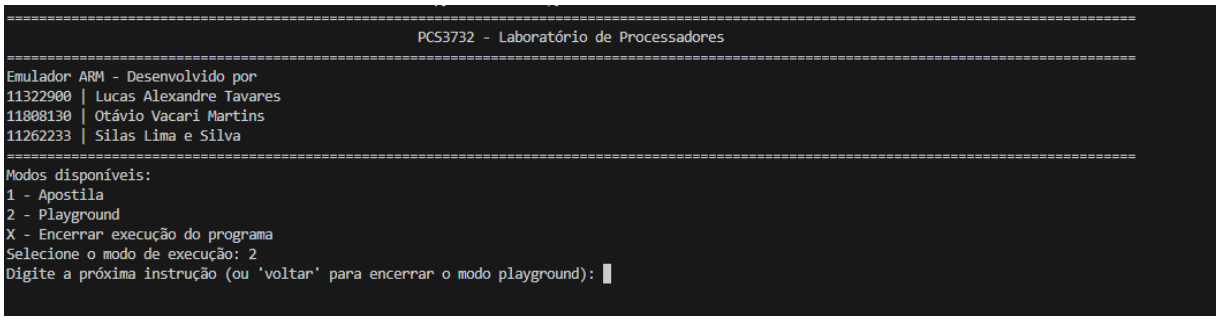

Imagem 4: Menu do modo 2 - Playground

No Playground, os usuários podem inserir instruções diretamente no terminal. Cada instrução é executada e seus efeitos nos registradores e na memória são mostrados na saída do programa. Isso permite que os usuários experimentem diferentes instruções e observem como elas afetam o estado do sistema.

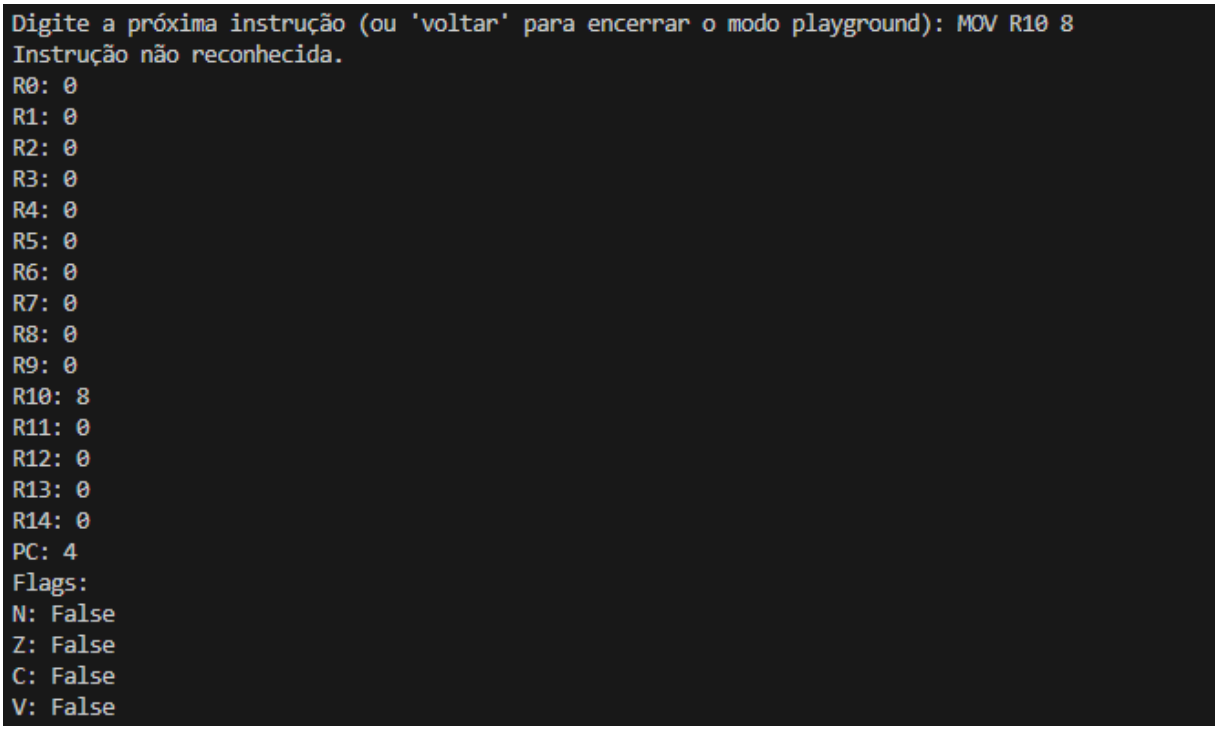

Imagem 5: Exemplo de entrada e saída no Playground

Para cada instrução inserida, o Playground exibirá o estado atual dos registradores e das flags após a execução da instrução. Isso proporciona uma maneira interativa de aprender e entender como as instruções ARM32 funcionam e como elas influenciam o comportamento do sistema.

2.2. Componentes e Funcionalidades

Nosso projeto é dividido em vários arquivos, cada um responsável por um aspecto específico do funcionamento do programa:

- main.py: Arquivo principal que contém a função main(), responsável por apresentar o menu inicial e direcionar o usuário para o modo escolhido (Apostila ou Playground).
- apostila.py: Arquivo que contém a função apostila(), que exibe as instruções disponíveis para consulta no modo Apostila. Ele interage com o arquivo instrucoes.py para obter informações sobre as instruções.
- playground.py: Arquivo que contém a classe ARM32Simulator, responsável por simular a execução das instruções no modo Playground. Ele interage com os arquivos play_alu_operations.py, play_memory_operations.py e play_jump_operations.py para realizar as operações específicas de cada tipo de instrução.
- play_alu_operations.py: Arquivo que contém a classe Operations, que implementa as operações aritméticas e lógicas das instruções.
- play_memory_operations.py: Arquivo que contém a classe MemoryOperations, que implementa as operações de leitura e escrita na memória das instruções.
- play_jump_operations.py: Arquivo que contém a classe JumpOperations, que implementa as operações de salto das instruções.
- instrucoes.py: Arquivo que contém as informações sobre as instruções ARM32, como mnemônicos, descrições, OPCode, sintaxe e detalhes dos operandos.

- funcoes.py: Arquivo que contém funções auxiliares para formatação de saídas no terminal.

3. Próximos Passos

O projeto atual é uma versão inicial que aborda instruções básicas de ARM32 e oferece uma plataforma para aprender e praticar a execução dessas instruções. No entanto, há várias melhorias e expansões que podem ser consideradas para futuras iterações:

- Adição de mais instruções: Expandir a lista de instruções disponíveis no modo Apostila e no Playground para abranger um conjunto mais amplo de operações.
- Melhoria da interface: Criar uma interface gráfica mais amigável para o Playground, permitindo uma experiência mais visual e interativa.
- Salvamento de sessões: Adicionar a funcionalidade de salvar e carregar sessões do Playground, permitindo que os usuários voltem ao estado anterior da simulação.
- Detalhes avançados: Incluir mais informações detalhadas sobre as instruções, como exemplos de uso, cenários de aplicação e possíveis armadilhas.
- Tutoriais interativos: Implementar tutoriais interativos que guiem os usuários pelo processo de execução de instruções específicas, fornecendo uma maneira mais estruturada de aprender.
- Testes automatizados: Desenvolver testes automatizados para verificar a precisão das simulações e garantir que as instruções se comportem conforme o esperado.

4. Conclusão

Nosso projeto "ARM-Playground" é uma ferramenta educacional que visa auxiliar estudantes e entusiastas a entender e praticar as instruções ARM32 de forma interativa e prática. Esperamos que esta ferramenta possa contribuir para a aprendizagem de programação em assembly e para o entendimento dos conceitos por trás das operações de processadores ARM32.