Simulador do Processador-ICMC

Recurso educacional aberto para SSC0119

QUEIROZ, Breno Cunha
11218991

MOREIRA, Maria Eduarda Kawakami
11218751

23 de Junho de 2020

1 Introdução

O trabalho foi desenvolvido como um complemento para o projeto Processador-ICMC [1], criado pelo professor Eduardo Simões e alunos do ICMC, a fim de ensinar e aprender Organização e Arquitetura de Computadores.
O principal objetivo é adicionar uma ferramenta de visualização ao ambiente e facilitar o aprendizado dos alunos, familiarizando-os com o processador desenvolvido.
Desenvolvemos um simulador do processador que executa o código os códigos montados e mostra o estado atual do processador e os caminhos ativos em cada ciclo [4]. A arquitetura e seus caminhos são mostrados através de uma janela do OpenGL. As linhas de instruções, os valores dos registradores, e a saída de vídeo são apresentados através de uma interface de terminal desenvolvida com curses.

2 Desenvolvimento do trabalho

2.1 A disciplina SSC0119

A disciplina SSC0119, ministrada pelo professor Eduardo Simões, gira em torno de um processador desenvolvido por alunos do ICMC que executa em FPGA. Cada aluno inicialmente é apresentado à arquitetura deste processador e à linguagem em assembly que é montada para ele. Posteriormente, os estudantes são instruidos a como desenvolver um jogo para executar neste processador. É possível testar os jogos tanto no computador, por meio de um simulador, ou na propria FPGA, através do Quartus.

2.1.1 Atual estrutura de ensino

Atualmente, com a pandemia de coronavírus, a disciplina sofreu com a falta de um laboratório para o ensino. Como os alunos estão em suas casas e não possuem acesso ao laboratório, não é mais possível testar os códigos na própria FPGA, sendo necessário um foco maior no teste e ensino com o simulador. Uma das principais dificuldades nesta nova estrutura online é entender como o processador funciona na FPGA. Nas outros oferecimentos desta disciplina os alunos preenchiam o código da FPGA para executar cada instrução do código (LOAD, STORE, ADD, etc). Entretando, os alunos não conseguem mais testar o código na FPGA, um impecílio para validar se o que foi programado realmente funciona.

2.1.2 Atual simulador

No simulador atual que é utilizado na disciplina é possível que os alunos testem o código dos seus jogos sem precisar da FPGA. Entretando, a forma como este simulador foi programado (sequencial) não está de acordo a como o código é executado na FPGA (paralelo). Dessa forma, estudar o código do atual simulador não é muito interessante para o entendimento de como o código da FPGA funciona.

2.2 O Simulador desenvolvido

O desafio principal em desenvolver um novo simulador era conseguir simular o funcionamento em paralelo de uma FPGA, visto que compreender o funcionamento em paralelo das intruções é inevitável para um entendimento completo da execução na FPGA.

2.2.1 Estrutura do simulador

Para simular o paralelismo, utilizamos principalmente seletores para controlar em que momento os valores dos componentes serão atualizados (Figura 1a).
O código em C, portanto, roda as instruções e faz as seleções do que deve ser modificado (Figura 1b). E só no final do loop principal do processador, é que todos os valores são atualizados (simulando um clock).

```
// Executa Load dos Registradores
if(LoadIR) IR = DATA_OUT;

if(LoadPC) PC = DATA_OUT;

if(IncPC) PC++;

if(LoadMAR) MAR = DATA_OUT;
```

(a) Início do loop principal.

```
// Selecao do Mux1
if      (selM1 == sPC)  M1 = PC;
else if (selM1 == sMAR) M1 = MAR;
else if (selM1 == sM4)  M1 = M4;
else if (selM1 == sSP)  M1 = SP;
```

(b) Fim do loop principal.

Figure 1: Legenda a ser pensada.

As imagens acima e abaixo exemplificam o que ocorre no caso LOAD.

O selM1 = sPC (Figura 2a) é um seletor que, no final do ciclo (Figura 1b), realiza M1 = PC.
O LoadMAR = 1 (Figura 2a) é um seletor que, no início do ciclo (Figura 1a), realiza MAR = DATA_ OUT.

Abaixo é possível ver a mesma instrução (LOAD) nos três códigos aqui comparados. É possível observar que o código desenvolvido em C no novo simulador (Figura 2a) se assemelha mais ao código em VHDL (Figura 2b) do que o do simulador que atualmente é utilizado na disciplina (Figura 2c)[2].

```
case LOAD:
    selM1 = sPC;
    RW = 0;
    LoadMAR = 1;
    IncPC = 1;
    state=STATE_EXECUTE;
    break;
```

(a) Simulador desenvolvido.

```
IF(IR(15 DOWNTO 10) = LOAD) THEN
    M1 <= PC;                -- M1
    Rw <= '0';               -- Rw
    LoadMAR := '1';      -- LoadMA
    IncPC := '1';
    state := exec;  -- Vai para o
END IF;
```

(b) Código VHDL da FPGA.

```
case LOAD:
  reg[rx] = mem[mem[pc]];
  pc++;
  break;
```

(c) Atual simulador da disciplina.

Figure 2: Execução da mesma instrução implementadas de diferentes formas.

2.2.2 Curses

A interface do terminal foi desenvolvida em Curses e consiste em 3 janelas: TOPBAR, onde são mostrados os oito registradores, CODE, aonde cada linha é uma instrução decodificada do arquivo .mif e WINDOW que consiste na saída de vídeo do simulador.
A janela TOPBAR representa os oito registradores $(R_0, R_1, R_2, ...)(Figura5)$ e mostra a mudança nos valores a cada instrução executada pelo simulador. A janela CODE decodifica cada linha do arquivo .mif para o nome da instrução e dos componentes, facilitando o entendimento da visualização em OpenGL e da arquitetura. Pode-se identificar também qual linha corresponde a qual instrução e em qual estado se encontra, o que norteia o aluno no código principal do simulador (simple_simulator.c).
A janela WINDOW consiste na saída de vídeo do simulador, que é crucial na disciplina para o desenvolvimento do jogo, que compõe um dos trabalhos finais. Na Figura 3 vê-se um exemplo de um jogo desenvolvido por um aluno da disciplina.

2.2.3 OpenGL

Foi utilizado Legacy OpenGL para apresentar a arquitetura ao usuário (Figura 4). Optamos por utilizar a versão legacy do OpenGL ao invés da mais moderna pois é mais simples e fácil ser entendida por quem teve pouco contato com computação gráfica (maioria dos estudantes da disciplina SSC0119). A cada instrução executada no simulador, os caminhos na arquitetura que são utilizados

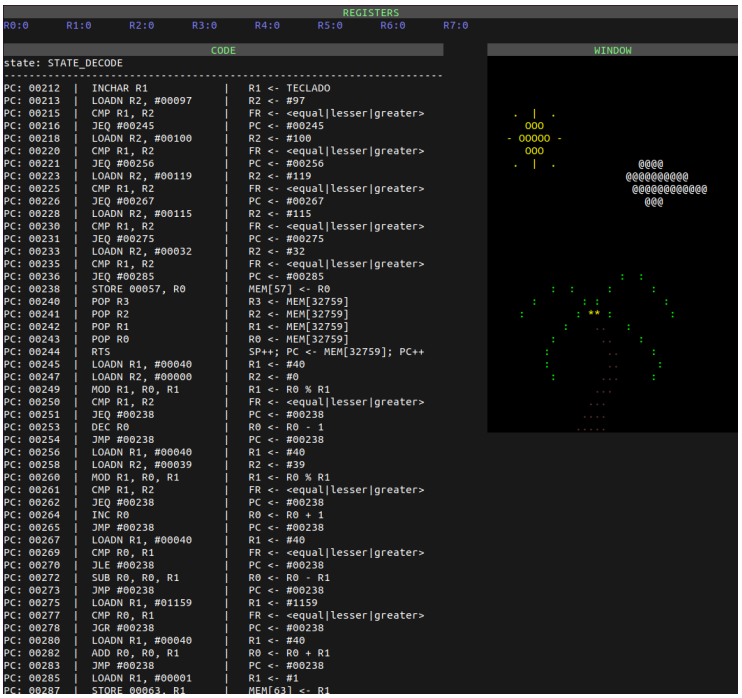

Figure 3: Interface do terminal em Curses

são destacados em verde. Acreditamos que esta pode ser uma poderosa ferramenta ao ministrar a disciplina pois facilita a visualização dos alunos do que está acontecendo a cada instrução. Tentamos manter a distribuição dos componentes fiel a que o professor Eduardo Simões utiliza na imagem de apresentação da arquitetura para os alunos (Figura 5).

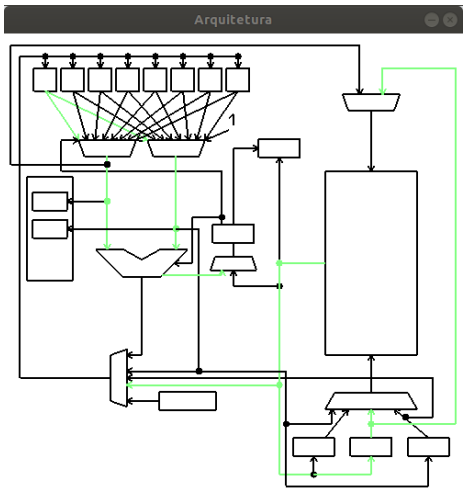

Figure 4: Arquitetura do processador OpenGL com caminhos em uso destacados em verde.

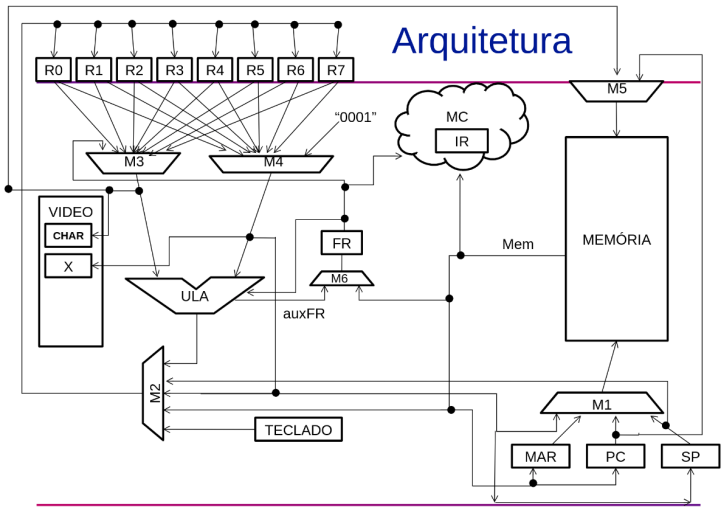

Figure 5: Imagem da arquitetura usada durando o ensino na disciplina.

3 Agradecimentos

Ao Professor Eduardo Simões por nos dar suporte no desenvolvimento do projeto, nos explicando a arquitetura do Processador-ICMC e a ideia do paralelismo no código do simulador, que serviu de base para todo o trabalho. Agradecemos a oportunidade de complementar este projeto e a confiança.

4 Conclusão

Neste trabalho foi desenvolvido um simulador para o Processador-ICMC que é programado com estruturas que se assemelham com a estrutura paralela encontrada no VHDL. Além disso, a ferramenta desenvolvida utiliza OpenGL e curses para criar uma interface com o usuário, onde é possível visualizar quais trilhas da arquitetura são utilizada a cada instrução.
Atualmente a visualização está funcionando como descrito, porém, a velocidade do simulador ainda precisa ser melhorada, visto que a execução dos jogos está bem mais lenta do que o encontrado na FPGA e no atual simulador da disciplina. Acreditamos que isto esta relacionado com a taxa de atualização da janela do OpenGL e do curses. Outro ponto a ser melhorado é a apresentação da arquitetura, que está *hardcoded*, o ideal seria que esta fosse variável de acordo com o código do simulador, de forma a permitir analisar outras arquiteturas construídas pelos alunos.
Postamos no youtube um video do simulador desenvolvido em execução [3].

References

[1] Simões, E. Et al. *Repositporio do Processador-ICMC* .
`https://github.com/simoesusp/Processador-ICMC`
ICMC-USP, 2020.

[2] Simões, E. *Manual do processador utilizado na disciplina SSC0119.*
`https://github.com/simoesusp/Processador-ICMC/blob/master/Manual/ProcessadorICMC2.pdf`
ICMC-USP, 2020.

[3] Queiroz, B; Moreira, M. *Video do simulador em funcionamento.*
`https://youtu.be/eYd-wCxaJ6Q`
ICMC-USP, 2020.

[4] Queiroz, B; Moreira, M. *Repositório do Github.*
`https://github.com/madukm/arquiteturaProcessador-ICMC`
ICMC-USP, 2020.