Pratica Sistemas Digitais

SSC0108

Professor Mauricio Acconcia Dias

Fabio Fogarin Destro - 10284667

Vitor Henrique Gratiere Torres - 10284952

Paulo André de Oliveira Carneiro - 10295304

Sumário

1. Introdução

O projeto final da disciplina de Prática em Sistemas Digitais teve como objetivo, a criação de um jogo, utilizando a linguagem de descrição de hardware VHDL, fazendo com que fosse necessário a utilização de conceitos teóricos, bem como o uso do conceito de máquinas de estados, assim como conceitos práticos, como a utilização do módulo gráfico juntamente com a lógica de programação e a linguagem VHDL.

Os principais desafios do projeto foram a utilização do módulo gráfico para exibir na tela as informações do jogo e também ideia da execução em paralelo que ocorre em hardware em processos distintos, o que dificulta o raciocínio e exige certo cuidado ao programar e definir comportamentos.

Vale ressaltar que foi utilizado como exemplo o Template AP9, com o jogo atropele o tomate, disponibilizado pelo Bonato como exemplo de uso do módulo gráfico, tal exemplo esta disponivel em: http://wiki.icmc.usp.br/index.php/SSC-108(bonato)_2017

2. O jogo

O jogo desenvolvido, inicialmente era um jogo da memória, em que 8 peças seriam exibidas na tela e o usuário poderia escolher através do teclado um par de cartas para virar, e caso as cartas combinasse, tais cartas se manteriam e um ponto seria ganho. No entanto, devido a execução em paralelo de processos, alguns problemas na lógica utilizada foram ocasionados, uma vez que o paralelismo impedia o diferenciamento de forma simples entre a primeira tecla digitada, referente a primeira carta, e a segunda tecla. Por esse motivo, o jogo da memória foi abandonado, já que apesar de uma ideia aparentemente simples, chegamos a conclusão que é de certa forma complicado em termos de lógica.

Sendo assim, decidimos iniciar outro jogo, uma releitura de Pac-Man, que julgamos mais possível de ser executado, mesmo que mais complexo em termos gráficos, com mais estados e comportamentos, por se assemelhar a ideia do jogo dos tomates (dado como exemplo) foi trabalhoso, mas possível de ser finalizado com êxito.

Dessa forma, nosso projeto é uma adaptação de Pac-Man, um jogo arcade clássico, em que o personagem (Pac-Man) tem como objetivo comer todas as bolinhas presentes no campo sem que toque em nenhum dos fantasmas que andam pelo mapa, caso o Pac-Man coma todas as bolinhas, então uma tela de “You win” é exibida, caso contrário, é escrito no meio do mapa uma mensagem de “Game over”.

2.1 Como jogar

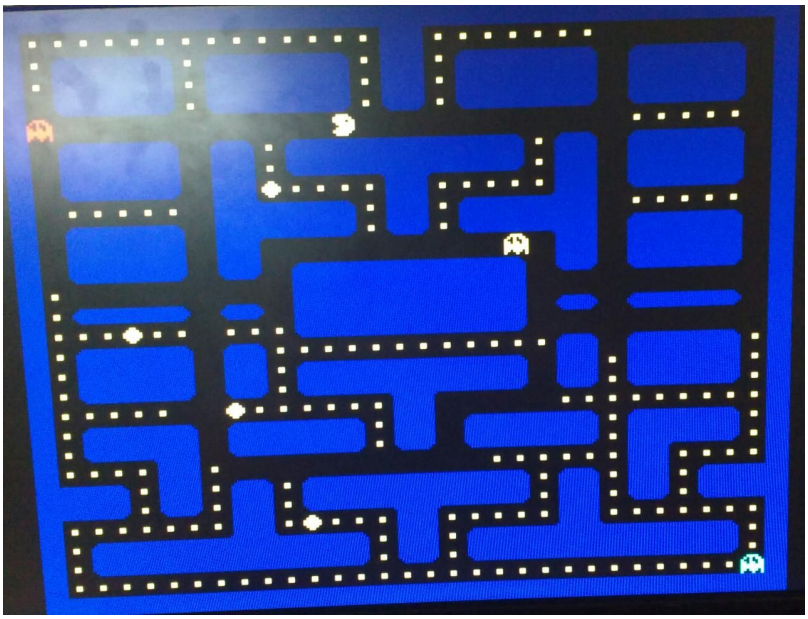
Primeiramente, para executar o jogo, é necessário abrir o jogo no Quartus, através da pasta “DE0_CV” e um duplo clique no arquivo “AP9.qpf”, a partir disso, é necessário sintetizar o projeto clicando na seta roxa, por fim é preciso passar o jogo para a placa, em “Programmer” e finalmente, é possível iniciar o jogo.

Vale lembrar, que o jogo foi desenvolvido com base na placa Cyclone V, DE0-CV, que são as placas FPGA menores do laboratório no bloco 6 do ICMC e só funcionarão corretamente nelas. Além disso, a fim de ajustar o clock corretamente, é necessário que todos os switches da placa estejam para baixo, com exceção do último e antepenúltimo.

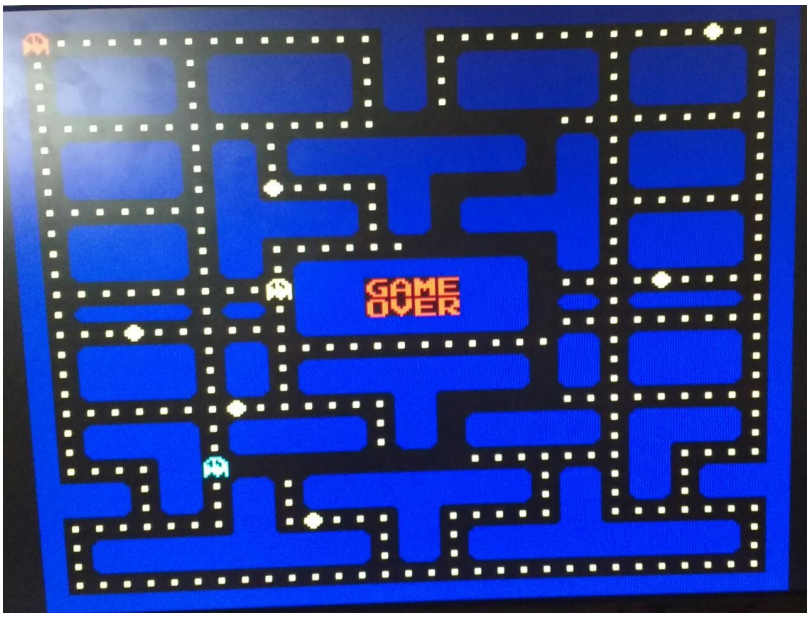
A jogabilidade consiste no movimento para os quatro lados, que podem ser efetuados pelas teclas “W” (cima), “A” (esquerda), “S” (baixo) e “D” (direita) que são comumente utilizadas em jogos para direcionar os personagens a partir do teclado. Tais teclas faz o Pac-Man se movimentar pelo mapa, que deve coletar as bolinhas e evitar encostar nos fantasmas, e caso encoste, o jogo é finalizado e o jogador perde. Em adição, as bolas maiores no mapa, são meramente de enfeite, com o intuito de simular o jogo clássico, que no entanto, não permite que o Pac-Man coma os fantasmas nem fique imune a eles.

2.2 Telas do jogo

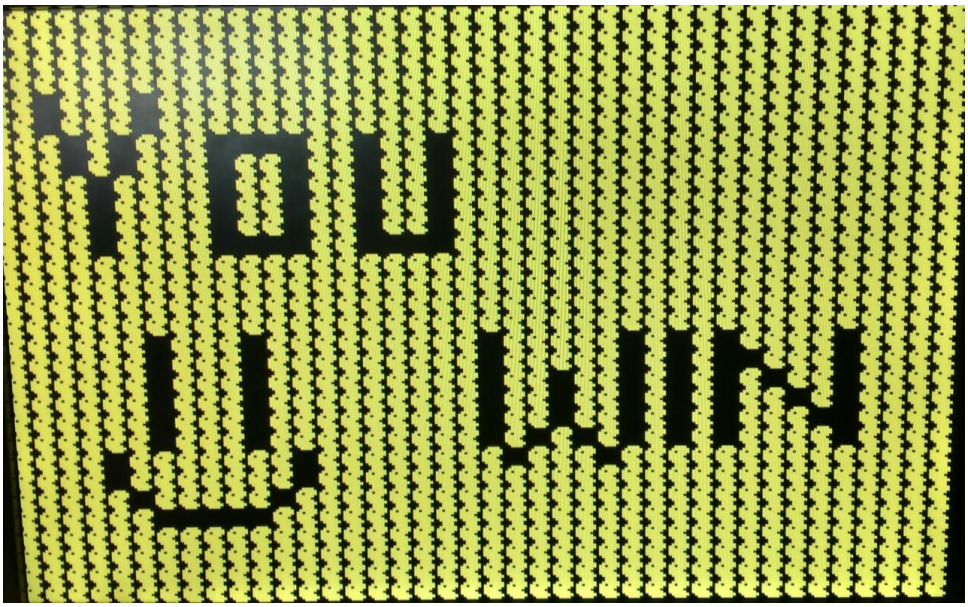
2.2.1 Jogando



2.2.2 Game over



2.2.3 You win



2.2.4 Antigo jogo da memória

3. Desenvolvimento

A primeira etapa do desenvolvimento do jogo foi a elaboracao do campo, para qual geramos uma matriz que representa todos os pixels da tela e, a partir dessa matriz e com o auxílio de um código em C que transformava os dados dessa matriz em um vetor, vetor esse que é lido pelo jogo para desenhar o mapa, conseguimos elaborar o plano de fundo como desejado. Os dados gerados nessa matriz para a elaboração do mapa inicial foram colocados nos arquivos video_mem1.mif e video_mem2.mif.

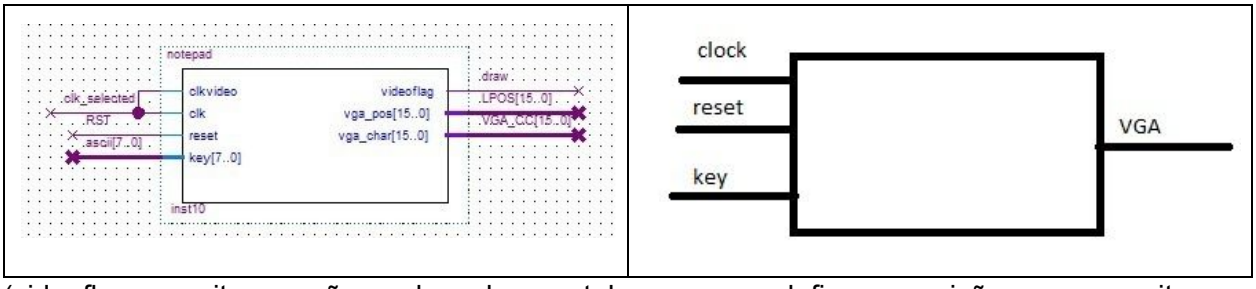
A segunda etapa do desenvolvimento foi modificar o arquivo charmap.mif incluindo as formas do Pac-Man, do fantasma e das letras que seriam utilizadas para o “Game Over” em sequência para facilitar a implementação. Logo em seguida elaboramos o processo responsável pela movimentação do Pac-Man, esse processo consiste em ler uma tecla e de acordo com a tecla lida, atualizar a posição atual do Pac-Man, redesenhando-a em outro lugar na tela, por exemplo se a tecla lida for um ‘w’ a posição do Pac-Man deveria ser decrementada 28 em hexadecimal e o charmap do Pac-Man deve ser atualizado para que sua boca fique virada para cima, fazendo assim que ele suba uma linha, gerando o movimento para o Pac-Man. Vale ressaltar que existe um vetor 1200 posições que é utilizado nesse processo que marca as posições válidas que o Pac-Man pode passar, por exemplo se na posição logo na frente do Pac-Man o vetor estiver marcando ‘1’ quer dizer que é uma parede, logo o Pac-Man não poderá passar por aquele lugar, mesmo que a tecla para andar naquela direção seja pressionada. Depois de muitos testes, conseguimos encontrar um delay que deixasse o movimento do Pac-Man com a velocidade ideal para que ele consiga fazer as curvas para coletar a pontuação e fugir dos fantasmas.

Logo em seguida, fizemos processos responsáveis pela movimentação dos fantasmas. Cada fantasma possui um respectivo processo já que cada um se movimenta de forma diferente. Os respectivos movimentos foram pensados para cobrir o maior espaço do mapa possível sem tornar o jogo muito difícil e sem ocasionar colisão entre fantasmas. Como cada fantasma possui um movimento específico cada um possui uma sequência de IF’s responsáveis por mudar a direção do fantasma quando ele atingir uma posição predefinida. Além disso cada processo de cada fantasma possui um delay diferente, responsável pela velocidade em que cada fantasma se locomove.

Além disso, para tornar o jogo mais próximo ao original, cada fantasma possui uma cor diferente (as mesmas cores que sao utilizadas no original).

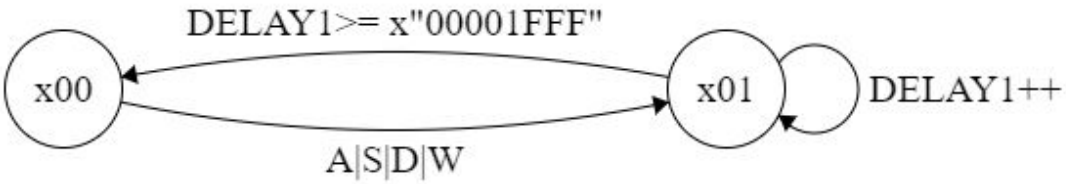
Por fim, tivemos que elaborar o último processo, responsável pela impressão do jogo na tela. Além disso é nesse processo que checamos se houve colisão entre algum dos fantasmas e o Pac-Man, parando o jogo e imprimindo game over, e checa se o jogador conseguiu coletar todas as moedas de pontuação e caso ele já tenha conseguido, pare imediatamente o jogo e imprima a tela “You Win”. Para desenhar o Pac-Man é necessário 4 estados. O primeiro apaga a posição antiga do Pac-Man da tela (ligando a videoflag e pintando essa região de preto que é a mesma cor do fundo), evitando os possíveis rastros. O próximo estado, desliga a videoflag e chama o próximo estado. O terceiro estado checa se houve colisão entre o Pac-Man e algum fantasma, caso essa colisão não ocorra redesenha o Pac-Man em sua nova posição, ligando a videoflag e atualizando os componentes do vga_char e setando o vetor de pontuação nessa posição como 1 (quando esse vetor inteiro estiver setado como 1, significa que o Pac-Man já colheu todos os pontos possíveis) e o último estado é responsável apenas em desligar a videoflag. Para desenhar os fantasmas, segue-se o mesmo padrão que para desenhar o Pac-Man, porém para apagar os rastros da fantasma não basta simplesmente pintar a posição antiga de preto, pois dessa forma iria sobrepor a pintura da pontuação, então checa se o Pac-Man já colheu o ponto dessa região observando o vetor de pontuação, se ele estiver em 1 pode pintar o fundo de preto, se estiver em 0 deve pintar a pontuação novamente.

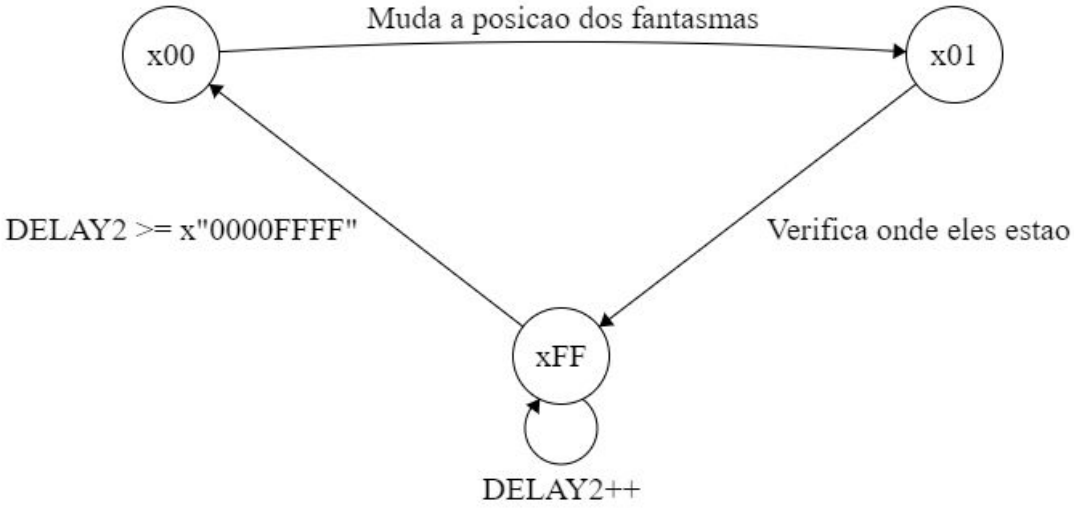
Por fim, é possível ainda visualizar o diagrama de blocos que contém as entradas e as saídas do jogo, em que basicamente possuímos o clock, o reset e a key (tecla pressionada pelo usuário que indica a direção desejada do Pac-Man) e como saída, o desenho no VGA. Segue abaixo o diagrama de blocos fornecido pelo Quartus e também uma simplificação:



(videoflag permite ou não o desenho na tela, vga_pos define a posição a ser escrito e o vga_char define o tipo de char que será escrito)

Abaixo estão duas simulações das máquinas de estado que movimentam o Pac-Man e os fantasmas respectivamente





Note que não são os estados que apagam e sobrescrevem os personagem, estes podem ser encontrados na descrição do hardware por serem muitos estados com muitas condicionais

(O código da lógica desenvolvida pode ser acessado em “notepad.vhd”)

4. Conclusão

No final do desenvolvimento do trabalho, conseguimos criar um jogo clássico e completamente funcional. Apesar dos contratempos com o jogo da memória, o grupo foi capaz de desenvolver uma remasterização do Pac-Man que utilizasse nossos conhecimentos adquiridos nas aulas práticas e teóricas, utilizando a lógica em VHDL para a implementação do que aprendemos na teoria. Apesar das tentativas de adicionar som ao jogo, desistimos dessa ideia pois seria necessário um módulo extra no qual não teríamos tempo de aprender.

Desta forma, o objetivo do projeto foi alcançado, uma vez que os conhecimentos em VHDL foram aplicados, além do aprendizado na utilização do módulo gráfico para a saída VGA que possibilitaram ao fim do projeto um jogo com começo meio e fim, e boa jogabilidade, assim como proposto no início.