UNIVERSIDADE DE SÃO PAULO
ESCOLA POLITÉCNICA
DEPARTAMENTO DE ENGENHARIA DE COMPUTAÇÃO E SISTEMAS DIGITAIS

PCS3635 - LABORATÓRIO DIGITAL I

PLANEJAMENTO SEMANA 3 - POLI-ASTEROIDS

Felipe Luis Korbes - NUSP: 13682893
Henrique Eduardo dos Santos de Souza - NUSP: 13679972
João Felipe de Souza Melo - NUSP: 13682913

Turma: 5
Professor: Reginaldo Arakaki
Data da experiência: 27/03/2024

São Paulo
2024

Sumário

1. Introdução e Objetivos

O objetivo deste laboratório é de desenvolver as atividades do projeto, especialmente na implementação dos requisitos funcionais previstos pelo cronograma para essa semana. Deve-se então partir para a finalização do projeto lógico do circuito, com o diagrama de blocos e o modelo da máquina de estados finita, e programar os arquivos de descrição de hardware em Verilog. Além disso, procura-se prosseguir com o estudo da serialização UART para a comunicação da placa FPGA com o computador.

2. Progresso da Semana 3

Durante a transição da semana 2 para a semana 3, houve um avanço significativo no desenvolvimento do projeto. Foram introduzidas três novas unidades de controle, das quais duas estão encarregadas de renderizar o jogo em uma matriz de LED, enquanto a terceira assume a responsabilidade pela geração de asteroides de forma mais completa do que a previamente implementada.

Além dessas unidades de controle, foi incorporado um gerador de números aleatórios para a criação dos asteroides, juntamente com a introdução de três novos níveis de dificuldade no jogo. Foi também estabelecido um sistema de pontuação, além de ser implementado o controle, registro e movimentação dos tiros e asteroides em intervalos de tempo predefinidos.

Em seguida, para assegurar o funcionamento adequado da lógica do jogo, foram realizadas alterações e refinamentos em todas as unidades de controle. Adicionalmente, o jogo foi adaptado para funcionar no Digital, facilitando o processo de depuração e testes.

Por fim, o grupo investiu bastante tempo no estudo e na implementação no sistema de comunicação serial, que deve estabelecer um canal de comunicação entre o circuito da placa FPGA e o computador, para que os dados sejam interpretados pelo software.

2.1 Fluxo de Dados

O fluxo de dados permaneceu, em grande parte, consistente em relação à semana anterior do projeto. Ainda estamos lidando com três fluxos de dados distintos: um para os asteroides, outro para os tiros e um terceiro para registrar e processar as diferentes ações realizadas pelo jogador.

Nos fluxos de dados responsáveis pelos tiros e pelos asteroides, foram incorporados módulos de geração de números aleatórios. Esses módulos têm a função de preencher a memória com informações sobre os asteroides a serem renderizados na tela, contribuindo para a aleatoriedade e diversidade do jogo.

Enquanto isso, no fluxo de dados destinado a registrar as jogadas do jogador, foram introduzidos *edge detectors* para os movimentos realizados pelo jogador. Essa adição visa aprimorar o tratamento desses movimentos, garantindo uma melhor sincronização e evitando possíveis problemas decorrentes de desalinhamentos ou inconsistências.

2.2 Unidades de Controle

Em relação às unidades de controle, atualmente temos um total de 10 unidades distintas. Elas incluem a unidade de controle do jogo principal, responsável por coordenar todas as operações, a que gerencia a comparação entre tiros e asteroides, a que efetua essa comparação, a que lida com a comparação entre os asteroides e a nave, bem como aquela encarregada de gerar os frames do jogo. Além disso, temos unidades que renderizam esses frames, registram os tiros, movem os tiros e, por fim, duas unidades dedicadas à geração e movimentação dos asteroides.

As três unidades de controle adicionadas nesta semana são aquelas que geram e renderizam os frames em uma matriz de LED, e a que gera os asteroides. Para a lógica das unidades de controle responsáveis por gerar e renderizar os frames na matriz de LED, o processo envolve percorrer a memória dos tiros, nave e asteroides. Em seguida, essas informações são salvas e todos os elementos são renderizados na matriz. Vale ressaltar que a unidade de controle que gera os frames é a única máquina de estados paralela ao restante do circuito.

Quanto à unidade de controle responsável pela geração de asteroides, ela também percorre a memória dos asteroides, verificando quais deles ainda não estão em jogo e se podem ser incluídos na partida.

Uma máquina de transição de estados que engloba todas as unidades de controle de maneira bastante simplificada se encontra na Figura 1.

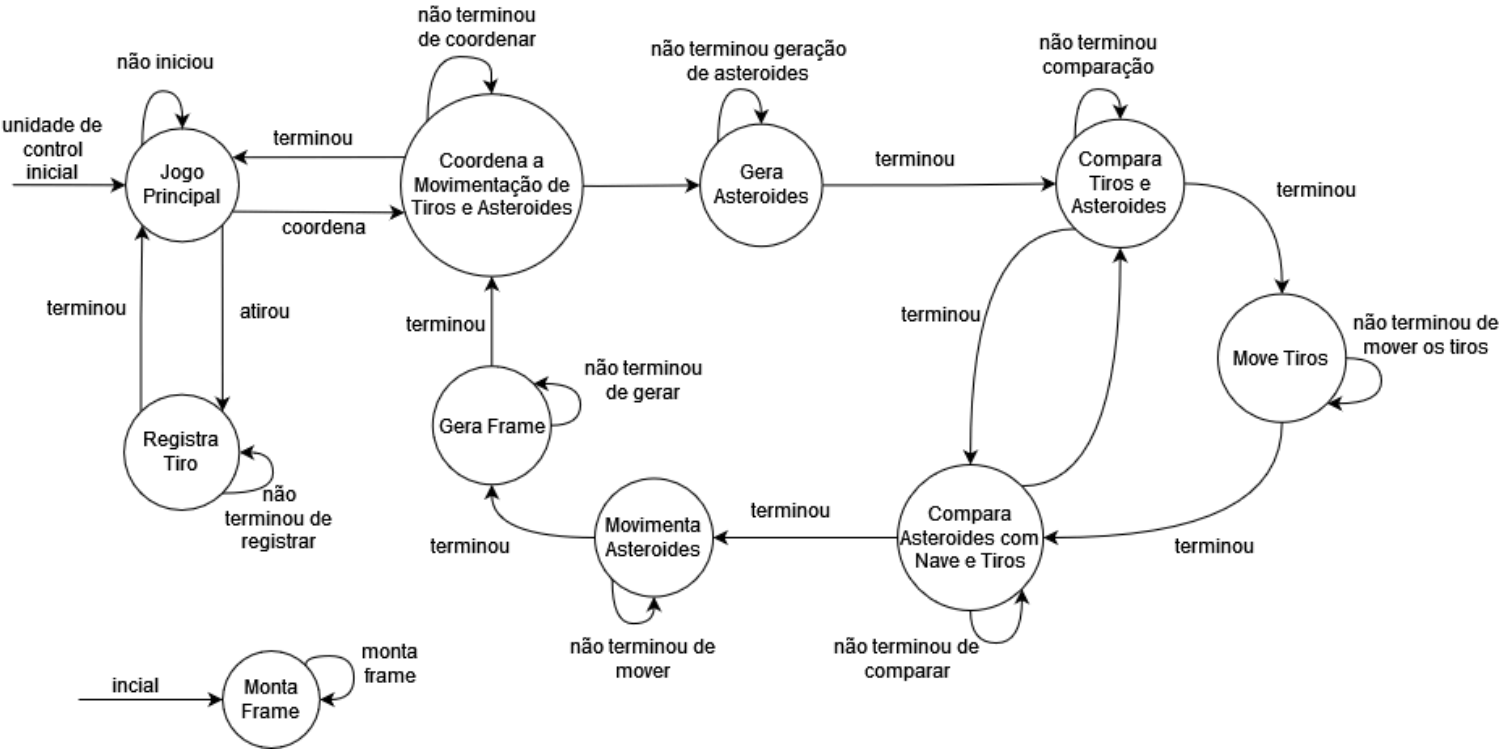

Figura 1 - Diagrama de transição das unidades de controle

2.3 Requisitos

Durante esta última semana de desenvolvimento, foram implementados requisitos extras que adicionam novas funcionalidades e aprimoramentos ao projeto.

Primeiramente, a geração de números aleatórios está sendo realizada utilizando o conceito de LFSR (Linear Feedback Shift Register), que consiste em um registrador de deslocamento aplicando operações XOR em alguns de seus bits para criar um processo de feedback. Para os propósitos deste projeto, podemos considerar esse gerador como verdadeiramente randômico, além de ser sintetizável na placa FPGA.

No que diz respeito à pontuação, foi incorporado um contador simples que é incrementado sempre que um asteroide é destruído. As dificuldades do jogo foram implementadas utilizando contadores 163 de limite variável dependendo da dificuldade escolhida pelo jogador. Esses contadores controlam a velocidade em que os tiros e os asteroides se movem, bem como o intervalo de geração de novos asteroides, tornando o jogo mais desafiador conforme a dificuldade aumenta.

Além disso, todo o circuito foi implementado no Digital, utilizando uma matriz de LEDs para permitir uma verificação mais aprofundada do circuito como um todo,

com vários sinais de depuração e display de LEDs. No entanto, surgiram alguns problemas durante essa implementação, como a matriz de LED não se comportando de maneira correta, apresentando piscadas constantes, o que dificulta a visualização dos elementos em jogo, especialmente quando há muitos deles na tela. Além disso, em frequências de clock muito elevadas, o Digital enfrenta dificuldades em realizar a simulação do circuito, resultando muitas vezes em um funcionamento mais lento do que o esperado. Esses problemas são comuns e são limitações do próprio software, também relatadas por outros grupos.

É importante ressaltar que espera-se que muitos desses problemas sejam resolvidos quando a porta serial for implementada, proporcionando uma plataforma mais estável e confiável para testes e depuração.

2.4 Serialização

Na semana 3, o grupo investiu muito no estudo da comunicação serial, que permitirá a recepção de dados da placa FPGA por um computador, que produzirá a interface gráfica com a qual o player pode interagir. O objetivo estabelecido para essa iniciativa era de ao menos receber via serial um dado fixo de 1 byte, armazenado na memória da placa FPGA, numa aplicação em python, que exibiria no terminal a informação recebida. Isso seria o ponto de partida fundamental para escalar o projeto e implementar a comunicação completa entre a placa e o computador.

A investida na comunicação serial deu-se em três momentos distintos: (a) estudo teórico da comunicação serial via UART e do protocolo MQTT, para decisão final de qual será implementado, (b) programação em verilog de uma UART (Universal Asynchronous Receiver / Transmitter) de transmissão e recepção de dados e (c) implementação do sistema de comunicação em laboratório, para um caso simples de transmissão de um dado fixo.

O passo (a) corresponde a uma decisão de projeto importante, orientada na escolha da tecnologia que ofereça o melhor conjunto de benefícios e que seja a mais viável e pertinente ao projeto. Após um estudo preliminar de ambos, que incluiu conversas com o monitor, chegou-se à conclusão de que o protocolo MQTT era demasiadamente complicado para o projeto e que a serialização seria o método mais fácil para o tráfego de dados de um bloco ao outro, logo optou-se pela serialização.

O passo (b) diz respeito à programação dos módulos receptores e transmissores da UART, para uso na comunicação serial. Trata-se então da etapa mais importante e difícil da comunicação serial, dada a complexidade dessa transmissão. Para fins de experimentação, utilizou-se uma UART pronta, do site NANDLAND (nandland.com).

O passo (c) consistiu em empregar os conhecimentos obtidos em (a) e em (b) na implementação física do dispositivo de comunicação serial, na placa FPGA. Isso envolveu utilizar os módulos da UART e conectar todos os componentes do sistema, desde a placa até o computador. A montagem ficou tal qual a figura 2.

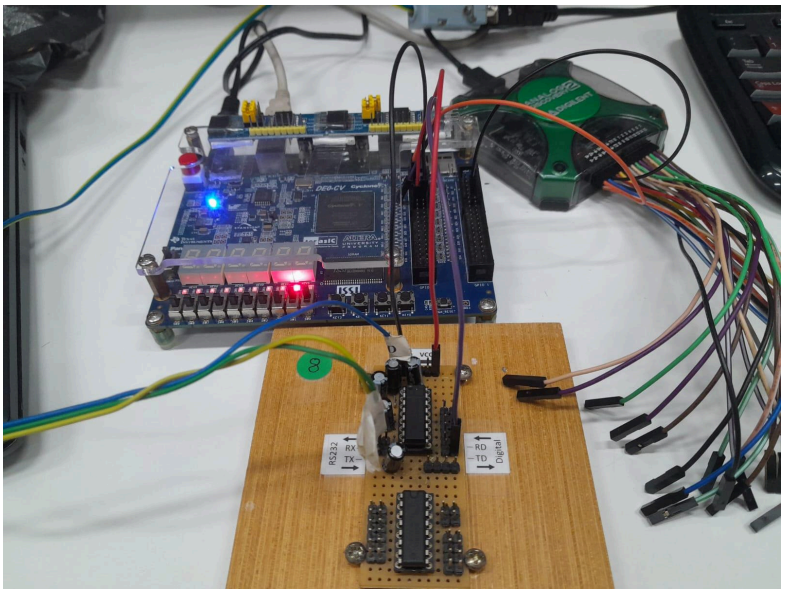

Figura 2 - Montagem física do sistema de comunicação serial

Infelizmente, o passo (c) não foi bem sucedido e o grupo não conseguiu realizar a transmissão dos dados da placa para o computador, via transmissão serial. O sistema, após os sinais de start e enable, entrava em modo busy, que indica transmissão de dados em curso, mas travava nesse estado. Suspeitou-se da corretude dos módulos prontos da UART da NANDLAND, que poderiam apresentar algum problema importante de lógica. No entanto, essa hipótese foi descartada após o testbench dos módulos individuais e do módulo da mais alta hierarquia, que instanciam todos os demais. Não foi possível encontrar o defeito do projeto ou da montagem física.

2.4 Testes

Para esta semana, o plano é levar adiante a implementação do circuito atual na placa FPGA. Apesar da ausência de uma interface gráfica para uma depuração em tempo real do programa, pretende-se utilizar essa implementação na placa FPGA para verificar se o circuito é realmente sintetizável. Mesmo sem a interface gráfica, ainda é possível realizar alguns testes com os sinais já presentes na máquina. Abaixo apresenta-se alguns casos de testes que serão executados durante a aula de laboratório.

Cenário de Teste 1 - Dificuldade hard e perder o jogo					
#	**Operação**	**Sinais de controle**	**Resultado esperado**	**Resultado observado**	**Veredito**
c.i.	Condições iniciais	reset=0 iniciar=0	pronto=0 vidas=3 pontos=0	pronto=0 vidas=3 pontos=0	
1	Resetar o circuito	reset=1 iniciar=0	pronto=0 vidas=3 pontos=0	pronto=0 vidas=3 pontos=0	
2	Iniciar e dificuldade hard	reset=0 iniciar=1 hard=1	pronto=0 vidas=3 pontos=0	pronto=0 vidas=3 pontos=0	
3	Deixa clock correr até perder	-	pronto=1 vidas=3-2-1-0 pontos=0	pronto=1 vidas=3-2-1-0 pontos=0	

Tabela 1 – Cenário de Teste 1 - Dificuldade hard e perder o jogo

Cenário de Teste 2 - Dificuldade easy e pontuar					
#	**Operação**	**Sinais de controle**	**Resultado esperado**	**Resultado observado**	**Veredito**
c.i.	Condições iniciais	reset=0 iniciar=0	pronto=0 vidas=3 pontos=0	pronto=0 vidas=3 pontos=0	
1	Resetar o circuito	reset=1 iniciar=0	pronto=0 vidas=3 pontos=0	pronto=0 vidas=3 pontos=0	

2	Iniciar e dificuldade easy	reset=0 iniciar=1 easy=1	pronto=0 vidas=3 pontos=0	pronto=0 vidas=3 pontos=0	
3	Dar alguns tiros para ganhar pontos	-	pronto=1 vidas=3-2-1-0 pontos=incrementar	pronto=1 vidas=3-2-1-0 pontos=incrementar	

Tabela 2 – Cenário de Teste 2 - Dificuldade easy e pontuar

3. Planejamento da aula prática

Inicialmente, baixa-se todos os arquivos do github para o diretório local. Em seguida, inicia-se a discussão sobre o projeto e divisão das tarefas entre os membros do grupo, para que se demonstre os requisitos importantes do dia. Deve-se alinhar o cronograma, os requisitos prioritários, em que o grupo vai trabalhar, tomar decisões de projeto, dentre outros aspectos importantes. Da parte de código, deve-se continuar a programação dos módulos verilog do jogo, de acordo com os diagramas feitos pelo grupo, e dos componentes relacionados à serialização.

Da parte de projeto, desde a implementação física até a parte lógica, deve-se discutir em grupo, para verificar-se a validade das ideias, plausibilidade dentro do cronograma e, por fim, tomar a decisão final. Se houver tempo hábil, deve-se também adiantar a discussão a respeito da montagem física do projeto, em termos de componentes, cabos, conexões, maquete etc.

Para implementação na placa FPGA, definiu-se a associação de pinos e sinais entre a placa FPGA e a Analog Discovery tal qual na tabela abaixo.

Sinal	**Pino na Placa DE0-CV**	**Pino na FPGA**	**Analog Discovery**
CLOCK	GPIO_0_D0	PIN_N16	StaticIO – LED – DIO0 e Patterns – Clock Fio 0 (rosa)
RESET	GPIO_0_D1	PIN_B16	StaticIO – Button 0/1 – DIO1 Fio 1 (verde)

INICIAR	GPIO_0_D2	PIN_M16	StaticIO – Button 0/1 – DIO2 Fio 2 (roxo)
EASY	GPIO_0_D3	PIN_C16	StaticIO – SWITCH – DIO3 Fio 3 (marrom)
MEDIUM	GPIO_0_D4	PIN_D17	StaticIO – SWITCH – DIO4 Fio 4 (rosa)
HARD	GPIO_0_D5	PIN_K20	StaticIO – SWITCH – DIO5 Fio 5 (verde)
PRONTO	GPIO_0_D6	PIN_K21	StaticIO – LED – DIO7 Fio 7 (marrom)
UP	SW0	PIN_U13	-
DOWN	SW1	PIN_V13	-
RIGHT	SW2	PIN_T13	-
LEFT	SW3	PIN_T12	-
ESPECIAL	KEY0	PIN_U7	-
TIRO	KEY1	PIN_W9	-
DB_PONTOS	Display HEX0	DB_PONTOS[0]=PIN_U21 DB_PONTOS[1]=PIN_V21 DB_PONTOS[2]=PIN_W22 DB_PONTOS[3]=PIN_ W21 DB_PONTOS[4]=PIN_Y22 DB_PONTOS[5]=PIN_ Y21 DB_PONTOS[6]=PIN_ AA22	–
DB_VIDAS	Display HEX1	DB_VIDAS[0]=AA20 DB_VIDAS[1]=AB20 DB_VIDAS[2]=AA19 DB_VIDAS[3]=AA18 DB_VIDAS[4]=AB18 DB_VIDAS[5]=AA17 DB_VIDAS[6]=U22	-
DB_TIRO_Y	Display HEX2	DB_TIRO_Y[0]=PIN_Y19 DB_TIRO_Y[1]=PIN_AB17 DB_TIRO_Y[2]=PIN_AA10 DB_TIRO_Y[3]=PIN_Y14 DB_TIRO_Y[4]=PIN_V14 DB_TIRO_Y[5]=PIN_AB22 DB_TIRO_Y[6]=PIN_AB21	–
DB_TIRO_X	Display HEX3	DB_TIRO_X[0]=PIN_Y16 DB_TIRO_X[1]=PIN_W16 DB_TIRO_X[2]=PIN_Y17	-

		DB_TIRO_X[3]=PIN_V16 DB_TIRO_X[4]=PIN_U17 DB_TIRO_X[5]=PIN_V18 DB_TIRO_X[6]=PIN_V19	
DB_ASTEROIDE_Y	Display HEX4	DB_ASTEROIDE_Y[0]=PIN_U20 DB_ASTEROIDE_Y[1]=PIN_Y20 DB_ASTEROIDE_Y[2]=PIN_V20 DB_ASTEROIDE_Y[3]=PIN_U16 DB_ASTEROIDE_Y[4]=PIN_U15 DB_ASTEROIDE_Y[5]=PIN_Y15 DB_ASTEROIDE_Y[6]=PIN_P9	
DB_ASTEROIDE_X	Display HEX5	DB_ASTEROIDE_X[0]=PIN_N9 DB_ASTEROIDE_X[1]=PIN_M8 DB_ASTEROIDE_X[2]=PIN_T14 DB_ASTEROIDE_X[3]=PIN_P14 DB_ASTEROIDE_X[4]=PIN_C1 DB_ASTEROIDE_X[5]=PIN_C2 DB_ASTEROIDE_X[6]=PIN_W19	

Tabela 3 - Designação de Pinos

4. Relatório

5. Cronograma

Apesar do relatório destacar que muitas implementações significativas foram realizadas durante a última semana, é importante reconhecer que essas atividades não foram refletidas no cronograma do projeto apresentado abaixo. No entanto, fica evidente que muitas das implementações realizadas foram direcionadas para a melhoria do circuito como um todo, tornando sua lógica mais concisa e robusta.

Assim como nas semanas anteriores, o plano é manter o cronograma inalterado e prosseguir com as tarefas planejadas. Isso inclui realizar pequenos ajustes, correções de bugs e, se houver tempo disponível, realizar a refatoração do código. Além disso, está prevista a conclusão da conexão serial da placa com o computador, bem como a montagem e a finalização da maquete do projeto.

Cronograma do projeto Resumido	
Atividade	**Semana**

Finalizado - RF1 (Implementação Física na placa FPGA) Finalizado - RF2 (Controle da Nave Espacial) Finalizado - RF5 (Sistema de Geração de Asteroides) Finalizado - RF3 (Sistema de Vidas)	06/03 a 13/03
Finalizado - RF7 (Tiro Realizado) Finalizado - RF6 (Destruição do Asteroide) Finalizado - RF4 (Sistema de Pontuação) Finalizado - RNF3 (Escalabilidade) Finalizado - RNF4 (Manutenção) Finalizado - RF11 (Hitbox)	13/03 a 20/03
Em andamento - RF8 (Registro de Pontuação dos Players na memória) Finalizado - Implementação de 3 Níveis de Dificuldade Em andamento - RF9 (Menu de Interação) Em andamento - RF10 (Monitor)	20/03 a 27/03
Não iniciado - Correções de bugs e implementações finais Não iniciado - RFN5 (Som atrativo) Não iniciado - Montagem do protótipo para o produto final	27/03 a 03/04