Nyan Cat Game

Leonardo Miassi Netto (Nº 9326688)
Matheus Carvalho Raimundo (Nº 10369014)

Orientador: Prof. Mauricio Acconcia Dias

USP - São Carlos

Dezembro/2017

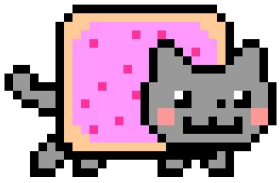
Índice Analítico

1. Considerações Iniciais

Como projeto final da disciplina, foi proposto que se faça um jogo qualquer escolhido pela dupla em Hardware FPGA. Será utilizado a linguagem de baixo nível VHSIC Hardware Description Language (VHDL), o módulo ‘AP9’ (fornecido pelo professor) e a placa FPGA Altera DE0-CV, além de acessórios como um teclado e monitor.

Foi optado por este grupo fazer o jogo “Nyan Cat Game”.

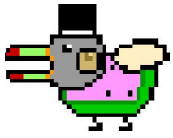
2. História e Contexto do Jogo



Nyan Cat é um conhecido personagem da internet (“meme”). Em 2011, foi postado em um popular serviço de *streaming* um vídeo de um gato cinza com estilo 8-bit voando sobre o espaço em um ciclo infinito (*loop*). O vídeo viralizou e foi um dos mais vistos do ano. Após isto, muitas derivações do gato voador no espaço se espalharam pela internet, criando-se histórias, inimigos, contexto e objetivos. Neste jogo, o Nyan Cat é o personagem que aparece no centro-esquerdo da tela de cor azul-claro.

Neste projeto, um dos inimigos do Nyan Cat é o Tac Cat. O Tac Cat é um gato de olhos vermelhos que é exatamente o oposto do Nyan Cat. E isso significa que ele também é mal. Neste jogo, ele está representado com a cor vermelha.

Além disso, há também o Bear, que é um urso que tenta perseguir o Nyan Cat. No jogo, ele está representado com a cor roxa.



Por fim, há o Melon Bird. O Melon Bird é um pássaro (no jogo representado pela cor azul-esverdeada) que também persegue o Nyan Cat. No jogo, o Melon Bird tem o diferencial de perseguir também na vertical, seguindo o Nyan Cat para tentar capturá-lo (inteligência artificial / “Bot”).

3. Módulos

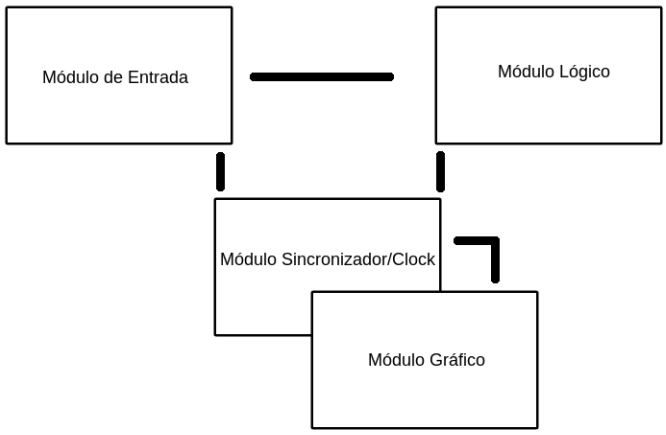
Há basicamente 4 módulos importantes neste trabalho: o gráfico, o de entrada (*IO/Stream*), o de clock e o lógico.

O módulo gráfico é responsável por desenhar na tela tudo o que o módulo lógico coordena. Ou seja, este vai desenhar os personagens, os textos e as cores na tela.

O módulo de entrada é responsável por sincronizar o clock do teclado e o clock do projeto, capturar todos os sinais de entrada do teclado, realizar a conversão para valores ASCII (código padrão americano para intercâmbio de informações) e retornar o resultado para o módulo lógico. A informação é recebida do teclado de forma sequencial (*serial*) e depois é convertido para paralelo. O módulo lógico vai, então, interpretar todas as teclas apertadas já com seu valor em ASCII.

O módulo de clock vai calcular o clock adequado para o projeto baseando-se no clock padrão e único da placa FPGA (50 MHz). Além disso, ele também inclui o “debounce” e o contador de atraso.

O módulo lógico é o principal responsável pelo funcionamento do jogo. Ele vai receber todas as informações dos outros módulos e retornar algo para o módulo gráfico. O módulo lógico realiza, então, ações como calcular posição dos personagens, calcular cores, alternar entre estados “inicializado/jogando/pausado/morreu” e muito mais (colisão, movimento, efeitos, IA dos *bots*, …).



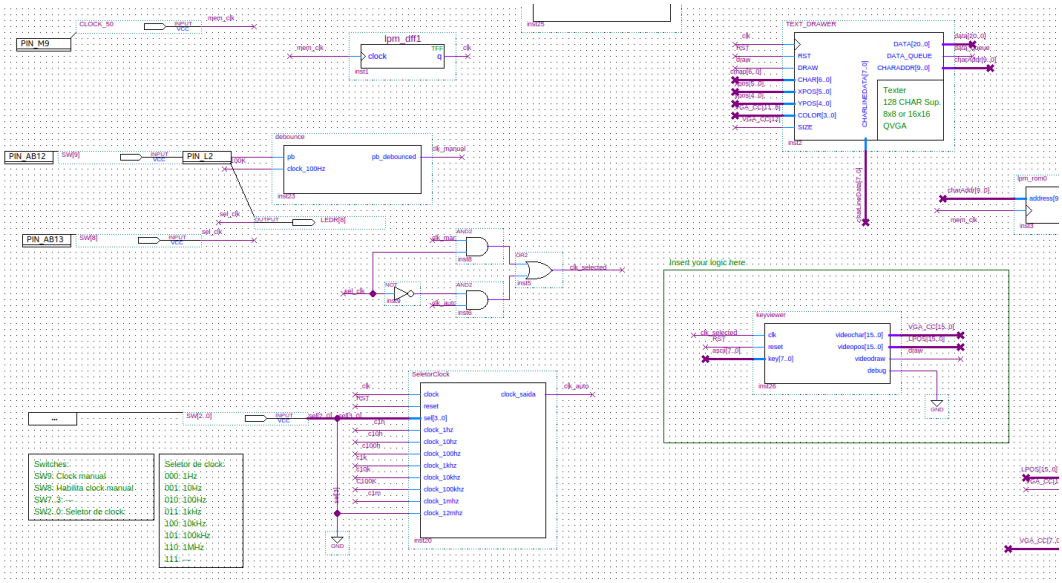


Imagem: Módulo ‘AP9’.

4. Requerimentos

Este jogo requer para funcionamento:

- Placa FPGA Altera DE0-CV, com clock configurado nos PINS (SW) para “101”;
- Monitor de resolução mínima 800:600 pixels (largura:altura);
- Teclado PS2.

5. Controles

Neste jogo os controles são bem simples:

- ➔ Quando iniciado, estará em uma tela de espera. A tecla [ESPAÇO] começa.
- ➔ Após isto, manter pressionado [ESPAÇO] faz com que o Nyan Cat suba na tela (voe). Soltar [ESPAÇO] faz com que ele desça na tela (caia).
- ➔ O objetivo é evitar todos os inimigos que virão da direita da tela e não ultrapassar os limites das bordas do jogo. Depois de iniciado, a tecla [ENTER] permite pausar/despausar o jogo.
- ➔ Quando o Nyan Cat perder (“morrer”), será exibido “fim de jogo” na tela e poderá começar o jogo novamente apertando [ESPAÇO].
- ➔ As teclas [W], [SETA PARA CIMA] têm as mesmas funções da tecla [ESPAÇO]

6. Obstáculos do Jogo

Com o tempo a dificuldade do jogo aumenta. A velocidade dos inimigos aumenta gradativamente, e depois de conseguir 2500 pontos, a borda da tela do jogador começa a piscar de diferentes cores, afim de desviar sua atenção. Em um momento, ela ficará transparente (preto), o que significa que o jogador deve ficar muito atento, pois mesmo transparente, o Nyan Cat não deve encostar nela, caso contrário o jogo acaba!

7. Implementação

Para produzir o módulo lógico, que processa e faz o cálculo de todo o funcionamento do jogo, foi utilizado algumas técnicas lógicas de VHDL.

7.1 Constante Redesenho

A tela está em constante redesenho, a cada ciclo de clock. Ela começa do primeiro pixel (x0, y0) e continua até o último pixel visível (xn, yn). Mesmo que não haja nenhuma alteração dos elementos dispostos na tela, ela é redesenhada a todo momento. Isso elimina tarefas como “apagar posição antiga” de personagens, ou “limpar lixo visível”. Há um contador que começa em i = 0, e vai até i = n. A cada ciclo de clock, ele verifica se o bloco em (xi, yi) possui algum elemento a ser pintado (como Nyan Cat, inimigos ou texto). Se sim, ele pinta esse elemento, caso contrário, pinta tudo de preto. A seguir, i é incrementado. Quando i chega a ‘n’, i volta a ser 0 e o ciclo se repete.

7.2 Processo de Teclado

O processo (*process*) de teclado captura a entrada do teclado (em ASCII) e salva em variáveis se o Nyan Cat deve subir ou descer (se [ESPAÇO] está sendo apertado ou não) e se deve ou não pausar o jogo. Esse processo é responsável apenas por realizar estas ações relacionadas ao teclado.

7.3 Processo Principal

Foi utilizado um único processo para realizar os cálculos e desenhar na tela. Neste primeiro processo, há diferentes contadores de atraso que realizam diferentes ações.

Desenhar na tela não exige um atraso, por isso não possui um contador de atraso (apesar de ter um contador de posição do pixel a ser desenhado).

A velocidade do Nyan Cat é controlada por um contador de atraso, visto que o clock alto faria ele se mover muito rapidamente. Toda vez que esse contador chega em seu máximo, ele é zerado, e o Nyan Cat muda de posição.

Há também um contador de atraso (com maior atraso), para controlar a velocidade dos inimigos e a pontuação.

7.4 Aleatório

Em alguns momentos durante o jogo, é necessário utilizar o aleatório. O aleatório é dado por um contador (que começa em 0) e triplica de tamanho a cada ciclo de clock, além de somar consigo mesmo a posição Y do Nyan Cat. Apenas isso já é o bastante para desordenar os bits e gerar um aleatório aceitável.

7.5 Início de Jogo

Inicialmente o jogo está no estado “inicializado”, onde ele aguarda [ESPAÇO] ser pressionado para ir para o estado “jogando”.

7.6 Jogando

Neste estado, a posição Y do Nyan Cat incrementa ou diminui (dependendo se [ESPAÇO] está ou não sendo pressionado). Ao mesmo tempo, a posição X dos inimigos diminui e a pontuação incrementa mais lentamente (com contador de atraso maior). A inteligência do Melon Bird é programada para seguir o Nyan Cat na posição Y também (é necessário “enganar” ele para despistar). Aleatoriamente e ocasionalmente, surge um novo inimigo na tela (Tac Cat, Bear ou Melon Bird), na extrema direita, com uma posição Y também aleatória. Quando neste estado, pressionar [ENTER] levará ao estado “pausado”.

7.7 Pausado

Quando pausado, o jogo aguarda pelo [ESPAÇO] ou pelo [ENTER] para retornar ao estado “jogando”.

7.7 Fim de Jogo

O Fim de Jogo é dado quando, a qualquer momento do estado “jogando”, a posição de algum dos inimigos coincide com a posição do Nyan Cat (colisão), ou a posição Y do Nyan Cat ultrapassa os limites impostos pela borda da tela. O jogo vai então para o estado “morreu”. Neste estado, o jogo aguarda por [ESPAÇO] para iniciar novamente.

8. Conclusão

Com esse trabalho, chegou-se a conclusão de que desenvolver um jogo em VHDL para FPGA pode não ser tão complicado e difícil como parece. Ao mesmo tempo, é algo completamente diferente do usual. VHDL é uma linguagem fortemente tipada, e constantemente foi utilizado funções de conversão (inteiro, vetor, *arrays*), além do fato de que deve-se “pensar em hardware”. É por isso que há muitas diferenças em como as variáveis são armazenadas e modificadas, além do funcionamento de loops (“for”) e borda de subida do clock.

Mas sem sombra de dúvidas, a maior dificuldade se deu com a implementação. O software utilizado para desenvolver o projeto demorava exponencialmente a cada recurso adicionado no jogo para compilar. Além de gerar arquivos extremamente largos, que com o tempo não podiam mais ser enviados via e-mail, por exemplo. Muitas vezes, levava mais de 5 minutos para descobrir-se erros simples (como um ponto-e-vírgula faltando ou fora do lugar). Contudo, o projeto foi desenvolvido com sucesso, e no fim o jogo estava funcionando exatamente como esperado.