Centro de Informática
Universidade Federal da Paraíba

Relatório Final

Jogo Flappy Bird e o Uso do VGA no Kit Altera

Almir Cassimiro Queiroga

Ian Victor Luna Soares

Henrique Elpídio Rufino Araújo

Júlio Gusmão Carlos de Mendonça

João Pessoa, 2018

Centro de Informática
Universidade Federal da Paraíba

Jogo Flappy Bird e o Uso do VGA no Kit Altera

Relatório elaborado para o projeto final da disciplina Circuito Lógicos II, ministrada pelo Professor Eudisley Gomes dos Anjos do Centro de Informática da Universidade Federal da Paraíba.

João Pessoa, 2018

Resumo

Esse objeto consiste em uma apresentação da descrição do código Verilog criado para construção de uma versão do jogo Flappy Bird na placa FPGA DE2-115 da Altera e uma explanação sobre o funcionamento da porta VGA. Nas próximas seções, serão discutidos ambos os assuntos focando no que tange à lógica de programação do sistema.

Palavras-chave: Verilog, Flappy Bird, FPGA, VGA.

Lista de siglas

VGA - Video Graphics Array (Padrão de disposição gráfica para vídeo)

HDL - Hardware Description Language (Linguagem de descrição de hardware)

FPGA - Field Programmable Gate Array (Arranjo de portas programáveis em campo)

IDE - Integrated Development Environment (Ambiente de desenvolvimento integrado)

RAM - Random-access memory (Memória de acesso aleatório)

Sumário

1. Introdução

Flappy Bird é um jogo para dispositivos móveis desenvolvido por pelo programador vietnamita Dong Nguyen e pela empresa dotGEARS. Popularizado em 2014, ele rendeu mais de cinquenta milhões de downloads e tornou-se o jogo grátis número um da App Store em cinquenta e três países, nesse período.

Depois que o projeto foi finalmente desativado e deletado das lojas de aplicativos, usuários de smartphones que já haviam feito o download do game amontoaram-se em sites de venda online, como o eBay, para anunciarem seus dispositivos. Aparelhos esses que, segundo matéria do jornal The Economist de 14 de fevereiro de 2014 intitulada "*The value of Flappy Bird*" (O valor do Flappy Bird), passaram a superar o dobro de seu valor de mercado.

O produto original possuía um estilo gráfico 2D retrô que remetia à obras como Super Mario Bros. de 1985. Nele, o objetivo do jogador é de fazer com que o pássaro "Faby" navegue entre os canos verdes que aparecem na tela e, assim, marcando um ponto para cada conjunto passado com sucesso.

Em fevereiro de 2018, foi lançado um sistema de entretenimento da Nintendo sob o nome *Analogue Super Nt* (Super Nintendo Analógico) que permite aos usuários jogar todos os lançamento do antigo Super Nintendo, mas agora em televisores modernos. Foi pensando nesse sistema baseado em FPGAs e tendo em vista o fim do Flappy Bird das lojas de aplicativos, que foi decidido dissecar o jogo e reconstruí-lo agora em Verilog na placa FPGA DE2-115 da Altera. A escolha deste dispositivo deu-se por apresentar uma grande capacidade de processamento devido ao paralelismo das operações por ele realizada.

Na próxima seção, detalhes acerca da metodologia utilizada para construção do projeto serão expostos bem como uma introdução à utilização do conector de vídeo.

2. Metodologia

Neste capítulo serão apresentados conceitos básicos necessários para uma melhor compreensão dos blocos implementados no módulo de interface VGA em FPGA.

Para uma melhor análise dos blocos desenvolvidos, optou-se por simular cada bloco e, em alguns casos, a interação destes blocos. Simulação é uma metodologia para detectar possíveis problemas e analisar todos os fatores do projeto, permitindo um melhor estudo do que vai acontecer e de como consertar erros que gerariam imperfeições ou até mesmo danificariam o kit (SHANNON, 1975 apud BRANDAO, 2005).

2.2 Espaço de Cores

Um espaço de cores é um método pelo qual podemos especificar, criar e visualizar a cor. Um computador pode descrever uma cor usando as quantidades de vermelho, verde e azul necessários para chegar a uma cor. Usa-se um modelo abstrato matemático para formalizar a descrição de cores através de tuplas de números, tipicamente formadas por três ou quatro elementos. Pode ser visto como um sistema definido por uma base representativa dos componentes, de acordo com a definição do espaço considerado, onde a representação de qualquer cor pode ser feita à custa da combinação desses componentes. São normalmente tridimensionais. RGB e CMYK são sistemas de cores (FORD; ROBERTS, 1998).

Na implementação deste projeto fez-se uso dos modelos de cores RGB, os quais serão abordados com maiores detalhes. RGB é a abreviatura do sistema de cores básicas deste padrão formado por Vermelho (Red), Verde (Green) e Azul (Blue), e o propósito principal deste sistema é a reprodução de cores em dispositivos ópticos eletrônicos como monitores, scanners e câmeras digitais. É um modelo aditivo no qual o vermelho, o verde e o azul (usados em modelos aditivos de luzes) são combinados de várias maneiras para reproduzir outras cores. Uma cor no modelo de cores RGB pode ser descrita pela indicação da quantidade de vermelho, verde e azul que contém.

Cada pixel na tela pode ser representado no computador ou na interface do hardware como valores para vermelho, verde e azul, que são convertidos em intensidades, para que as intensidades procuradas sejam reproduzidas nos displays com fidelidade. O modelo de cor RGB é implementado de diferentes maneiras, dependendo da capacidade do sistema utilizado, sendo o mais comum de 24-bits de informação, com 8 bits, ou 256 níveis discretos de cor por pixel . Qualquer espaço de cor baseado neste modelo fica limitado a uma faixa de ≈ 16,7 milhões de cores (256 × 256 × 256). As imagens, no sistema RGB, são compostas por três planos independentes de imagem - um para cada componente desse sistema de cor. Quando exibidas em um monitor de vídeo, essas três imagens são combinadas para que o vídeo reproduza a imagem com o resultado da composição das três cores.

2.3 O VGA

A interface VGA foi desenvolvida pela IBM (International Business Machines) em 1987. Após o padrão VGA, novos padrões foram surgindo como por exemplo: UXGA, SXGA, WQXGA. Esses padrões são

capazes de exibir milhões de cores e suportar resoluções maiores. O nome VGA é também usado para designar o conector DE-15, que possui o formato de um D, conforme mostra a Figura 1.

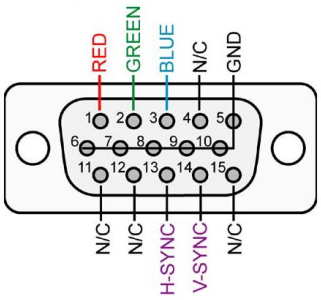

Figura 1 - Conector VGA fêmea. Fonte: Wikipédia (2018).

Dos pinos mostrados na Figura 2 somente cinco são utilizados no controle de vídeo. São os pinos: 1, 2, 3, 13 e 14. Os pinos 1, 2 e 3 são utilizados para determinar a cor (RGB) de um determinado pixel, já os pinos 13 e 14 são utilizados na sincronização do vídeo. Cabe frisar que R, G e B são sinais analógicos, isto é, a cor varia conforme a intensidade do sinal. Por padrão os pinos 1, 2 e 3 tem uma resistência de 75Ω conectados ao terra.

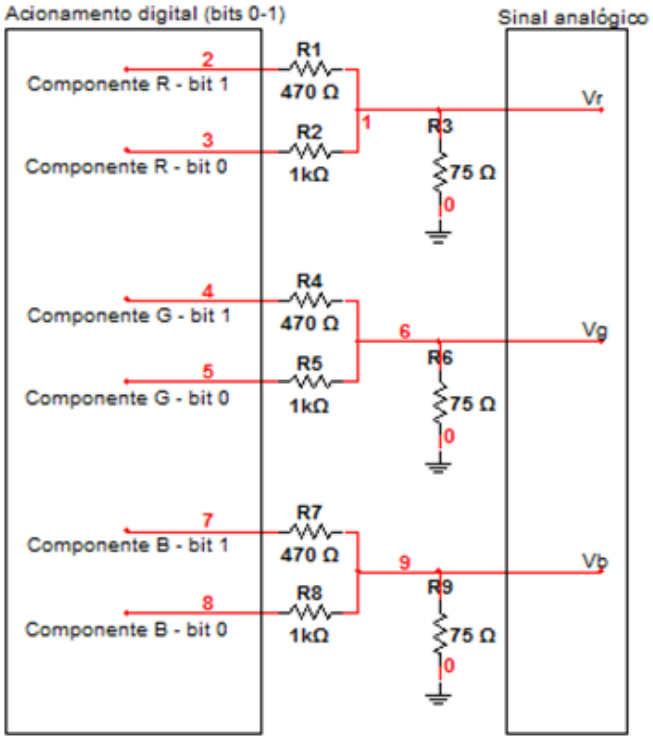

Figura 1.2 - VGA DAC com 2 bits.

Assim o número de bits em cada componente implica na geração de sinais com intensidades diferentes. A Figura 4 mostra o acionamento das componentes utilizando 2 bits. O divisor resistivo é responsável por manter a tensão do sinal numa faixa de 0V a 0,7V, sendo 0V a menor intensidade e 0,7V a intensidade máxima.

A interface VGA foi bem aceita no mercado e por algumas décadas foi amplamente utilizada, atualmente vem perdendo espaço para padrões digitais, porém ainda se encontra o conector deste padrão em aparelhos de televisão, placa de vídeo de computadores, retroprojetores, entre outros dispositivos. Para transmitir uma imagem para um monitor através de uma interface VGA são necessários os sinais de sincronismo horizontal e vertical, o sinal de clock (relógio) e os sinais que representam a intensidade de cada cor primária (Vermelho, Verde e Azul) para cada pixel.

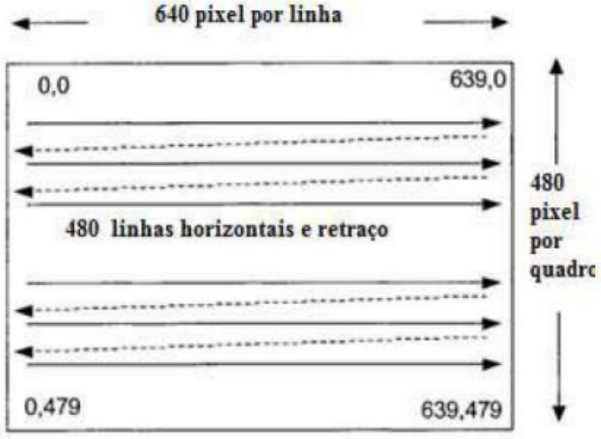

Figura 2 - Modo de escrita do padrão VGA. Fonte: Santos (2009).

No sistema VGA os valores de cada pixel são escritos no monitor da esquerda pra direita, e de cima para baixo, conforme mostrado na Figura 2. Neste trabalho foi utilizada uma resolução de 640x480, taxa de atualização de 60 Hz; e um sinal de clock de 50 MHz. O sinal de sincronismo horizontal (ver Figura 3) com polarização negativa corresponde a uma linha com duração total de 800 pulsos de clock (32,00 µs), possui uma área visível de 640 pulsos de clock (25,6 µs), tempo de guarda inicial de 16 pulsos de clock (0,64 µs), pulso de sincronismo com duração de 96 pulsos de clock (3,84 µs) e tempo de guarda final de 48 pulsos de clock (1,92 µs). O sinal de sincronismo vertical (ver Figura 4), com polarização negativa, corresponde a um quadro com duração de 525 linhas (16,78 ms), possuindo uma área visível de 480 linhas (15,25 ms), tempo de guarda inicial de 12 linhas (0,45 ms), pulso de sincronismo de 2 linhas (0,064 ms) e tempo de guarda final de 31 linhas (1,02 ms).

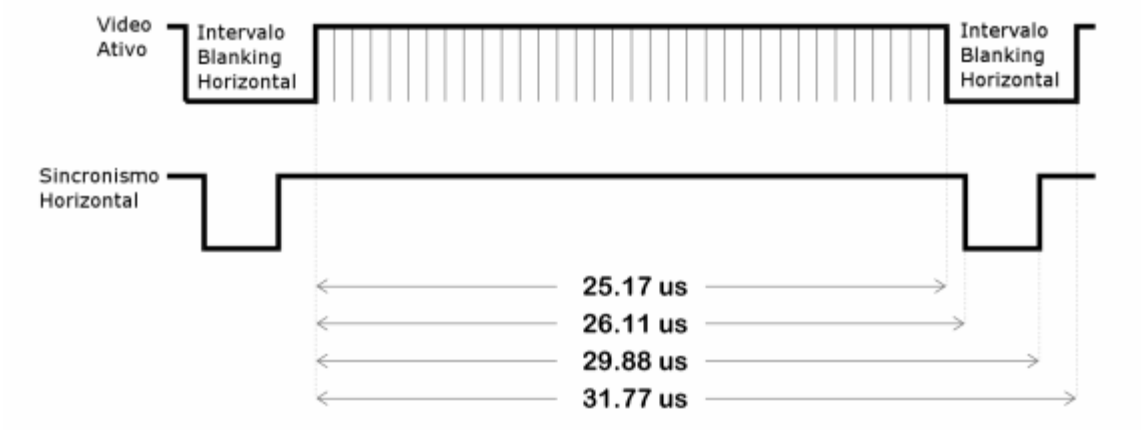

Figura 3 - Representação do sincronismo horizontal.
Fonte: Xess Corporation (1998).

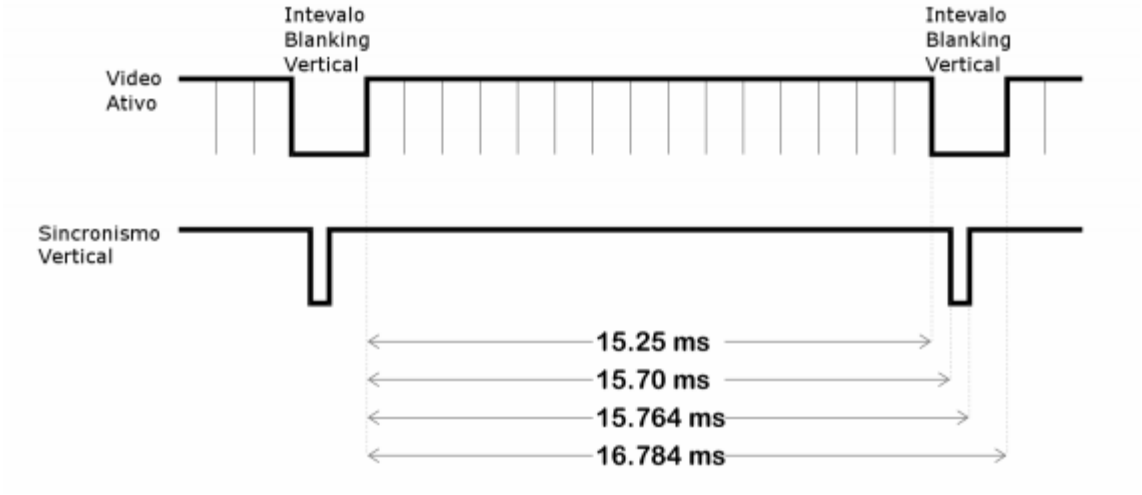

Figura 4 - Representação do sincronismo vertical.
Fonte: Xess Corporation (1998).

2.3.1 Pixels

Antes de analisar como os sinais de controle são utilizados é necessário ter conhecimento sobre como os pixels são dispostos na tela, especificamente como eles são representados em um frame. Se considerarmos uma tela (monitor) como uma matriz, podemos dividi-la em um conjunto de linhas (N) e colunas (M). Para cada unidade linha-coluna temos um pixel com suas componentes RGB.

Na Figura 5 temos o modelo de uma tela, sendo o primeiro pixel alocado no canto superior esquerdo. Para construir uma tela é necessário seguir o processo de varredura de pixels do monitor. A varredura da tela é iniciada da esquerda para a direita e de cima para baixo, sendo os pixels determinados de forma sincronizada. Ao chegar no final de uma linha o processo é iniciado novamente da esquerda para direita. O tempo necessário para estabelecer os pixels de um frame VGA é determinado conforme a resolução da tela. Esses parâmetros são discutidos a seguir.

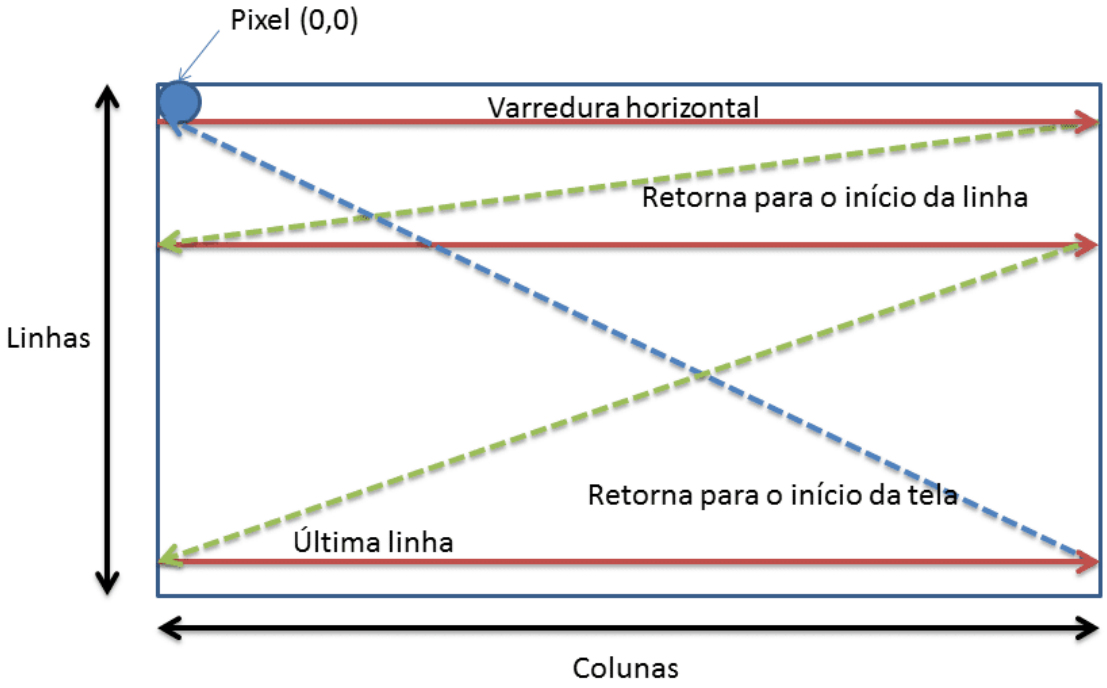

Figura 5 - Varredura do quadro VGA.

Na Figura 5 temos o modelo de uma tela, sendo o primeiro pixel alocado no canto superior esquerdo. Para construir uma tela é necessário seguir o processo de varredura de pixels do monitor. A varredura da tela é iniciada da esquerda para a direita e de cima para baixo, sendo os pixels determinados de forma sincronizada. Ao chegar no final de uma linha o processo é iniciado novamente da esquerda para direita. O tempo necessário para estabelecer os pixels de um frame VGA é determinado conforme a resolução da tela. Esses parâmetros são discutidos a seguir.

2.3.2 Frame

Um frame VGA define um conjunto de parâmetros que devem ser seguidos para sincronizar o vídeo com o mecanismo de varredura da tela. A Figura 6 exibe a região ativa de vídeo que corresponde à tela exibida (vide Figura 5).

Convém observar que além da região ativa de vídeo outros parâmetros de tempo devem ser respeitados. Tanto na varredura horizontal quanto na vertical são estabelecidos alguns parâmetros como: *Back Porch* e *Front Porch*. Esses parâmetros estão relacionados com os sinais *Horizontal Sync* (H_SYNC) e *Vertical Sync* (V_SYNC) que são utilizados para sincronizar a varredura da tela.

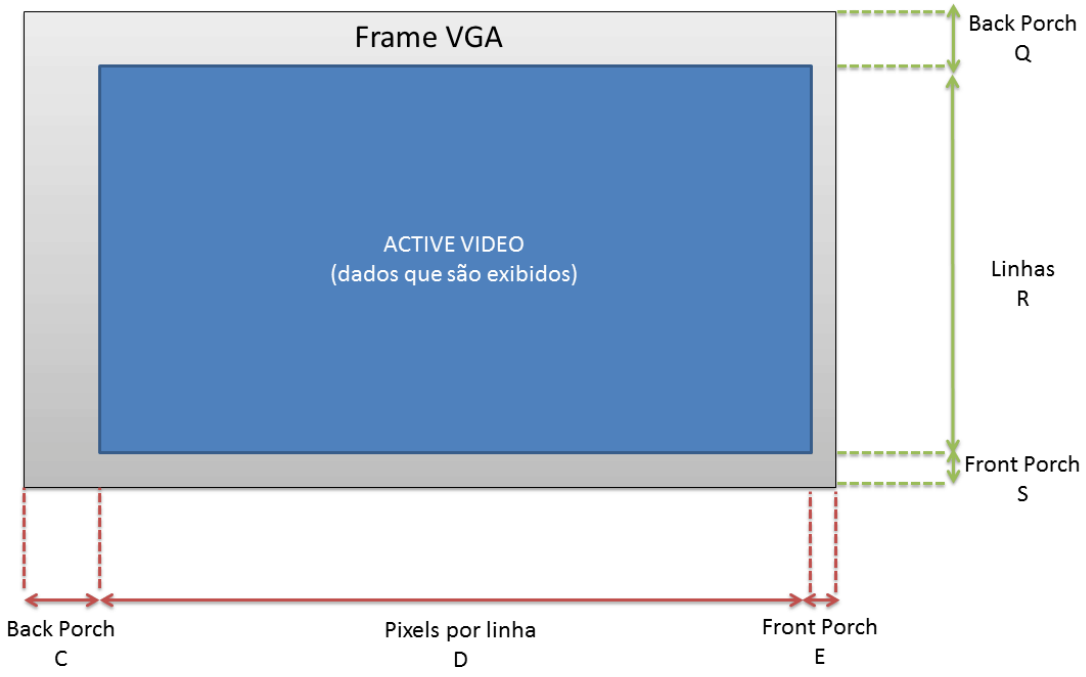

Figura 6 - Frame VGA.

O sincronismo de varredura horizontal é determinado pelo sinal H_SYNC. O sinal H_SYNC é ativado em nível lógico 0 e indica que uma linha foi finalizada e que próxima linha será iniciada.

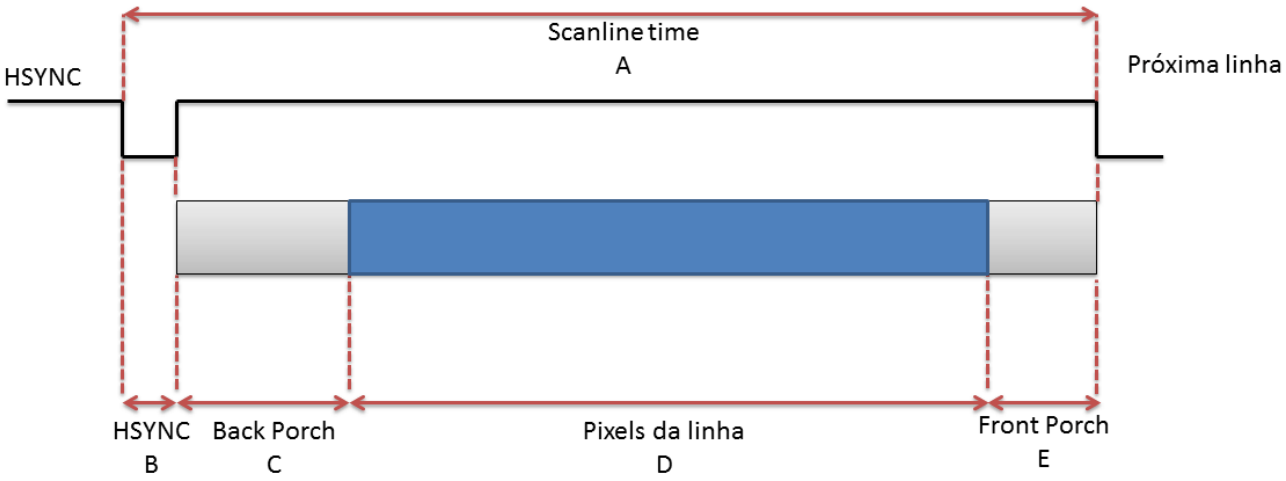

Figura 7 - Sincronização horizontal do frame VGA.

Conforme a Figura 7, o sinal H_SYNC deve permanecer em 0 por um período B. Após o período B um tempo C, chamado de *Back Porch*, deve ser aguardado antes de iniciar a região ativa da linha (pixels correspondentes à linha). Ao transmitir todos os pixels da linha um tempo E, chamado de *Front Porch*, deve ser aguardado até que o sinal H_SYNC seja colocado em 0 novamente. Veremos adiante que os parâmetros B, C, D e E dependem da resolução e frequência de atualização da tela.

O sincronismo de varredura vertical é determinado pelo sinal V_SYNC. O sinal V_SYNC é ativado em nível lógico 0 e indica que um quadro foi finalizado e que o próximo será iniciado. Um quadro é um conjunto N de linhas, sendo que cada linha possui M colunas.

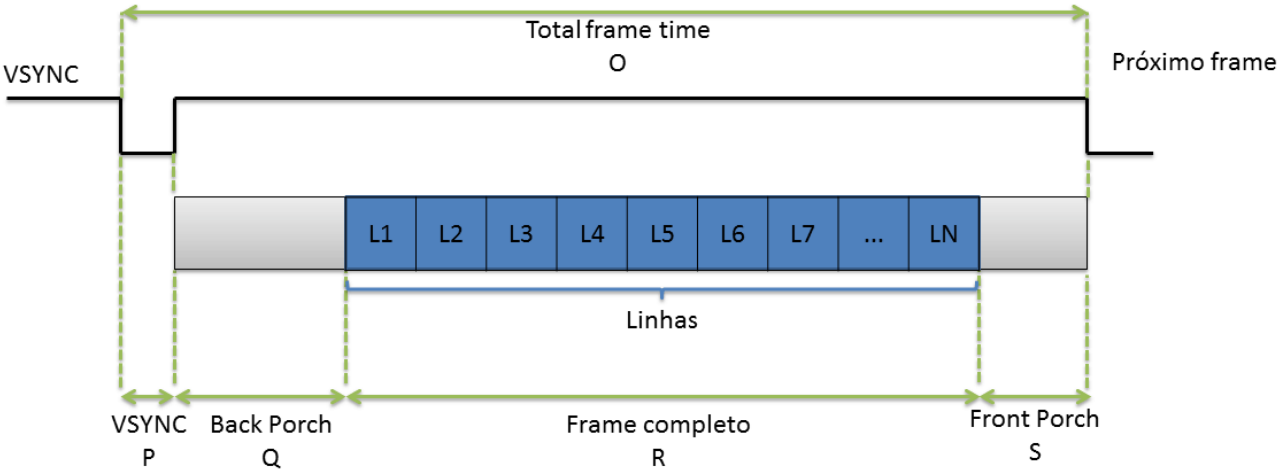

Figura 8 - Sincronização vertical do frame VGA.

Nota-se que na Figura 8 o mesmo mecanismo de sincronização horizontal é utilizado. A diferença está na região ativa que representa todas as linhas do frame. Na figura abaixo é mostrada a relação do sinal V_SYNC com a atualização das linhas (vide Figura 7) do quadro.

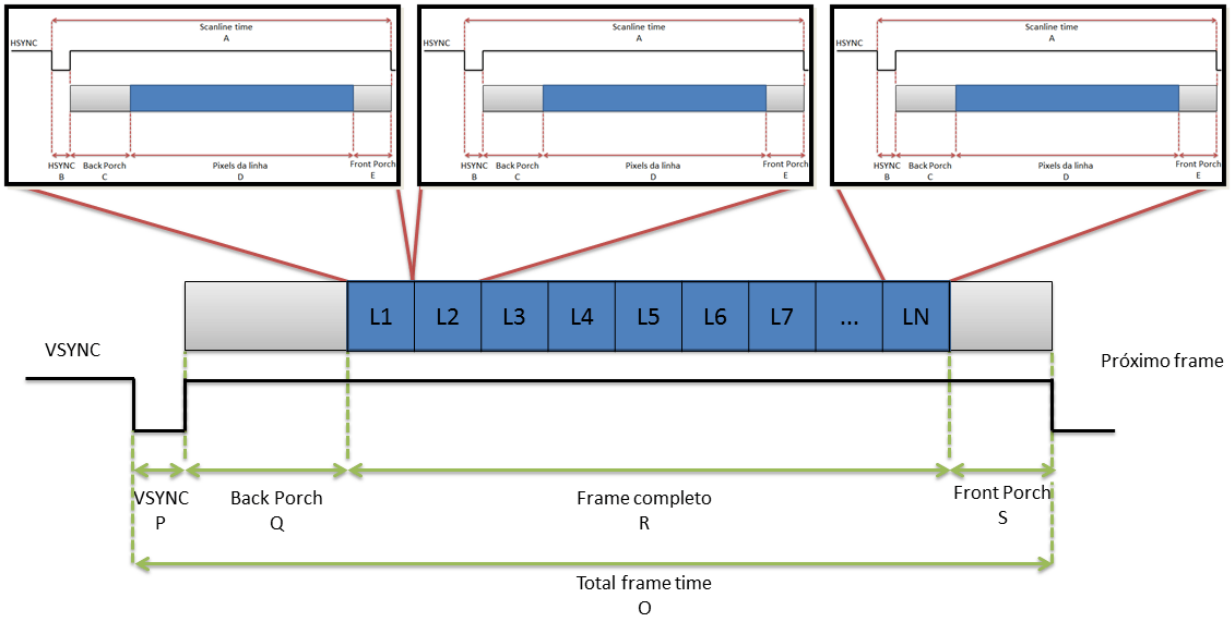

Figura 9 - Sincronização do frame VGA.

Convém observar que os parâmetros P, Q, R e S também dependem da resolução e frequência de atualização da tela.

2.3.3 Temporização

Considerando que um sistema para controlar um dispositivo VGA tenha uma base de tempo precisa, pode-se utilizar a tabela abaixo para determinar o período dos parâmetros de controle. A Tabela 1 exibe a relação de pulsos de clock necessários para cada parâmetro (B, C, D, E, P, Q, R e S) conforme a resolução da tela e a taxa de atualização.

Format	Pixel Clock (MHz)	Horizontal (in Pixels)				Vertical (in Lines)			
		Active Video	Front Porch	Sync Pulse	Back Porch	Active Video	Front Porch	Sync Pulse	Back Porch
640x480, 60Hz	25.175	640	16	96	48	480	11	2	31
640x480, 72Hz	31.500	640	24	40	128	480	9	3	28
640x480, 75Hz	31.500	640	16	96	48	480	11	2	32
640x480, 85Hz	36.000	640	32	48	112	480	1	3	25
800x600, 56Hz	38.100	800	32	128	128	600	1	4	14
800x600, 60Hz	40.000	800	40	128	88	600	1	4	23
800x600, 72Hz	50.000	800	56	120	64	600	37	6	23
800x600, 75Hz	49.500	800	16	80	160	600	1	2	21
800x600, 85Hz	56.250	800	32	64	152	600	1	3	27
1024x768, 60Hz	65.000	1024	24	136	160	768	3	6	29
1024x768, 70Hz	75.000	1024	24	136	144	768	3	6	29
1024x768, 75Hz	78.750	1024	16	96	176	768	1	3	28
1024x768, 85Hz	94.500	1024	48	96	208	768	1	3	36

Tabela 1 - Temporização do VGA.

Como foi visto, o padrão VGA especifica uma série de parâmetros que devem ser respeitados para que um quadro seja exibido corretamente. A próxima seção explicará o funcionamento dos principais arquivos do projeto.

3. Descrição do Projeto

Os principais arquivos que sustentam o universo do jogo são “flappy_bird_controls.qsys”, “Flappy_Bird.sv” e “main.c”. Cada qual com sua importância relacionada à física, controles e visual.

3.1 “flappy_bird_controls.qsys”

Esse arquivo é a chave para entender o funcionamento das interligações do código. É através dele que é possível gerar automaticamente conexões lógicas que permitem unir funções à subsistemas. Configurou-se ele através do sistema de software Qsys para que fosse possível configurar a relação de clock e memória RAM.

3.2 “Flappy_Bird.sv”

Nesse arquivo, é onde a construção do visual do jogo é definida e onde a lógica de comandos que servem para a movimentação do pássaro e do cenário são estabelecidos. Essa é a base para comunicação da porta VGA com o monitor. Aqui é, também, onde definimos a relação da memória RAM com clock.

3.3 “main.c”

Essa parte da construção do jogo foi escrita na linguagem de programação C. Aqui é onde as regras do jogo são estabelecidas. Definimos como os canos vão ser gerados na tela, suas posições e o quanto de espaço deve haver entre eles. Aqui também é definida a física do jogo, delimitando as ações de movimentação do pássaro após a tecla “espaço” ser pressionada.

São definidas, também, as condições de inicialização de jogo como a tecla “enter” que precisa ser acionada no teclado que fará com que o pássaro seja inicializado no espaço do cenário assim como a posição dos canos. Além disso, a condição da tabela de marcação de pontos também está inclusa nesse arquivo. A medida que o pássaro vai atravessando o cenário entre os canos, novos canos vão sendo gerados a partir de uma função “randomHeight” que continua sendo chamada se as condições de pássaro morto não forem satisfeitas ainda.

4. Execução do Projeto, Testes e Resultados

Foi utilizado o Quartus II, Qsys e um editor de texto para implementar a arquitetura do código, bem como uma placa de desenvolvimento DE2-115 da Altera, um teclado PS2 e um monitor VGA.

Durante a fase de testes, foram encontrados problemas no que cerca a utilização do arquivo “.qsys” para realizar a interligação dos arquivos que se vinculam à memória RAM. Contudo, após corretamente adicionar os arquivos com as informações de ligação completas no Qsys, o problema de interligação e utilização de memória foi, então, solucionado possibilitando que o mesmo realizasse a compilação dos arquivos que compõem o sistema do jogo.

O resultado final foi a esperada reprodução do Flappy Bird original, mantendo a estrutura estética do mesmo, levando em consideração o design dos elementos do jogo como canos, pássaro e cenário, e reprodução da física similar com o do projeto original de 2014 feito pela dotGEARS.

É esperado que, para uma implementação futura, possa-se trabalhar em cima da arquitetura atual e modifica-la de maneira a substituir o teclado PS2 por uma dispositivo de captura de áudio para que o mesmo funcione a partir de comandos de voz possibilitando, assim, uma nova dinâmica de jogabilidade que pode ser melhor estudada futuramente.

5. Conclusões

O jogo Flappy Bird foi um sucesso sem igual e aprender sobre a construção do mesmo foi muito significativo para cimentar os conhecimentos adquiridos sobre circuitos lógicos e a linguagem de descrição de hardware Verilog, bem como suas várias funcionalidades.

Apesar dos problemas durante a fase final de compilação dos arquivos, o sistema construído funcionou como esperado.

6. Referências

- https://zefanxu.github.io/Flappy_Bird_on_DE2_115/<Acesso em: 30 de maio de 2018>
- https://www.economist.com/graphic-detail/2014/02/14/game-over<Acesso em: 23 de maio de 2018>
- https://en.wikipedia.org/wiki/Flappy_Bird<Acesso em: 20 de maio de 2018>
- https://github.com/dcarr622/FlappyFPGA
- https://github.com/themaxaboy/Flappy-Bird-Verilog
- https://byuu.org/articles/what-is-emulation/<Acesso em: 3 de maio de 2018>
- https://timetoexplore.net/blog/arty-fpga-vga-verilog-01<Acesso em: 3 de maio de 2018>
- BRANDÃO, C. et al. Aplicação da simulação dinâmica em uma linha de sopro de garrafas pets em uma indústria de refrigerantes. In: XXV Encontro Nac. de Eng. de Produção, Porto Alegre, 2005
- FORD, A.; ROBERTS, A. Colour Space Conversions. [S.l.], 1998.