AVR Pong

KrpytonPlusPlus

27 de fevereiro de 2025

1 Introdução

Esse projeto visa projetar um clone do jogo *Pong* em um microcontrolador *avr* utilizando o padrão de comunicação *vga* para exibir a imagem em um monitor.

Para isso foi dimensionado os ciclos de *cpu* para comportar o envio das informações de varredura dos *pixels* e os ciclos sobressalentes foram utilizados para carregar os dados para os *buffers* com as cores de cada pixel em uma linha (preto ou branco) e fazer o tratamento das iterações do jogo.

Buscando facilitar os cálculos e aumentar a compatibilidade com os monitores foi utilizado um sistema de *clock* de $25{,}175MHz$ para o microcontrolador superior ao limite estipulado pelo fabricante no manual, porém como vai ser visto adiante é a mesma frequência de sinal utilizada pelo monitor para gerar a imagem.

2 VGA (Video Graphics Array)

2.1 Regiões de varredura

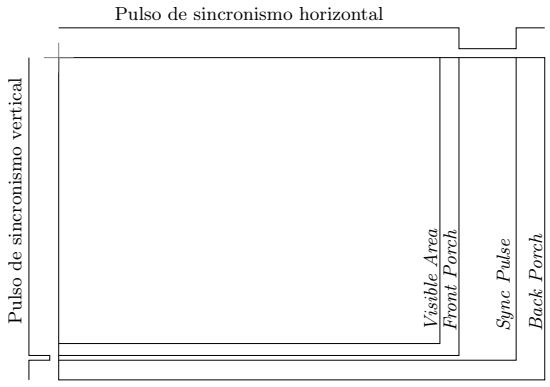

Figura 1: Regiões de varredura da tela

No monitor normalmente a *Back Porch* vem antes da *Visible Area*, porém para facilitar a apresentação e por ser um sinal cíclico foi apresentada no final.

Para facilitar a visualização as dimensões de cada região não estão proporcionais aos valores reais.

	Pixels	
	Horizontal	Vertical
Back Porch	48	33
Visible Area	640	480
Front Porch	16	10
Sync Pulse	96	2
Total	800	525

Tabela 1: Dimensão horizontal e vertical em *pixels* da tela para cada região de varredura

Normalmente os dados apresentados na tabela 1 são dados em micro segundos e quantidade de linhas, para a dimensão horizontal e vertical, respectivamente.

2.2 Pulsos de sincronismo

Os pulsos verticais são formados utilizando o *Timer 1* no modo *fast pwm* com *top* sendo o *icr1* ($wgm1[3:0] = 14_{10}$) e utilizando um *prescaling* com valor 8 ($cs1[2:0] = 2_{10}$), a escolha desse *Timer* foi feita pelo fato de possuir um contador de 16 *bits*. O *timer* possui 3 registradores de comparação (*ocr1a*, *ocr1b* e *icr1*), sendo dois de saída, para poder realizar a construção do pulso, utilizando o modo de comparação para quando o contador atingir o valor de *ocr1a* a saída ir para nível lógico baixo e quando voltar para *bottom* ir para nível lógico alto ($com1a[1:0] = 2_{10}$) e o terceiro, de entrada, para determinar o inicio da área visível da tela.

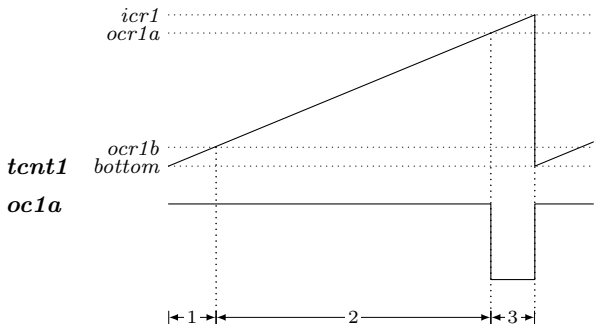

Figura 2: Funcionamento do *Timer 1* (*Vsync*)

Na figura é possível ver o funcionamento esperado da saída *oc1a* em função do contador *tcnt1*, as regiões 1, 2 e 3 são respectivamente, *Vertical Back Porch*, *Vertical Visible Area* junto com a *Vertical Front Porch* e *Vertical Sync Pulse*.

Os valores de comparação da contagem são dados a seguir:

icr1	52499
ocr1a	52299
ocr1b	3199

Tabela 2: Valores dos registradores de comparação do *Timer 1*

O valor do *ocr1b* foi calculado para uma linha a menos, para evitar conflitos com os pulsos horizontais, pois quando o seu valor é atingido ocorre uma interrupção para um função que ativa a interrupção para o *Timer 0* em *ocr0a*, que será desativada dentro do próprio algoritmo da função chamada pela comparação.

Em que *icr1*, por exemplo, é calculado da seguinte forma:

$$\frac{800 \cdot 525}{8} - 1 = 52499 \tag{1}$$

Pelo fato do *clock* utilizado ser de $25{,}175MHz$, igual a frequência dos *pixels* para o padrão de resolução utilizado, os valores de comparação são numericamente iguais menos um a contagem dos *pixels*, pelo fato da contagem iniciar em zero, considerando o valor do *prescaling*.

Os pulsos horizontais são formados pelo *Timer 0*, deixando o *Timer 2* livre, no modo *fast pwm* com *top* sendo o *ocr0a* ($wgm0[2:0] = 7_{10}$) e utilizando *prescaling* com valor 8 ($cs0[2:0] = 2_{10}$). Esse *Timer* possui dois registradores de comparação (*ocr0a*, *ocr0b*) utilizados para criar um pulso de polaridade negativa entre eles.

Durante o *overflow* do contador ocorre uma chamada de interrupção que faz o tratamento da região horizontal *Back Porch* e da área visível, realizando o envio dos *pixels* e durante a região *Front Porch* e *Sync Pulse* ocorre a construção do *buffer* da próxima linha.

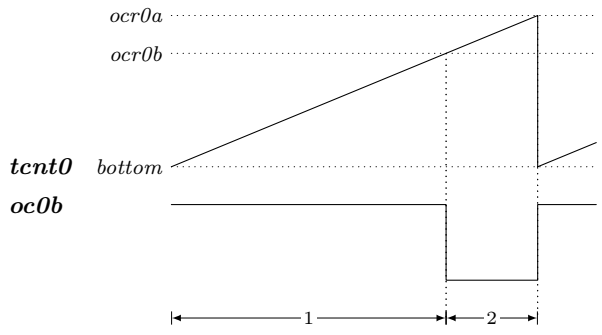

Figura 3: Funcionamento do *Timer 0* (*Hsync*)

Na figura é possível ver o funcionamento esperado da saída *oc0b* em função do contador *tcnt0*, as regiões 1 e 2 são respectivamente, horizontal *Back Porch* junto com a *Visible Area* e a *Front Porch*, e a *Sync Pulse*.

Os valores de comparação da contagem são dados a seguir:

ocr0a	99
ocr0b	87

Tabela 3: Valores dos registradores de comparação do *Timer 0*

Em que o calculo é similar ao realizado para o comparadores do pulso vertical, como é possível ver a seguir para o *ocr0a*, por exemplo:

$$\frac{800}{8} - 1 = 99 \tag{2}$$

2.3 Varredura dos *pixels*

Para realizar a varredura dos *pixels* foi utilizado o protocolo de comunicação *usart* no modo master *spi* do microcontrolador, pelo fato de ser possível fazer uma comunicação de no máximo duas vezes mais lenta que o *clock* do sistema de forma autônoma, e por ter a mesma frequência dos *pixels* implica que a resolução horizontal é reduzida pela metade, porém ocupando a mesma quantidade de *pixels*.

A comunicação é feita pela conexão `txd` e o `hardware` utiliza a conexão *xck* como *clock* de sincronismo, mesmo que ele não seja utilizado.

3 *Pong*

O jogo *Pong* se baseia em lançar uma bola de um lado ao outro da tela com uma "raquete"(*paddle*) tentando fazer com que o outro jogador erre a bola assim ganhando um ponto, parecido com o *ping pong*, porém esse projeto se baseia em apenas um jogador, portanto o outro *paddle* apenas segue a bola, e os elementos do jogo possuem algumas iterações com entre si, por exemplo sempre que a bola atinge o *paddle* da esquerda ela aumenta a velocidade além de possuir trajetórias diferentes dependendo de onde acertar.

A disposição dos elementos do jogo é apresentada na figura 4, onde é possível observar os dois *paddles*, a bola, a linha divisória central e a pontuação de cada jogador.

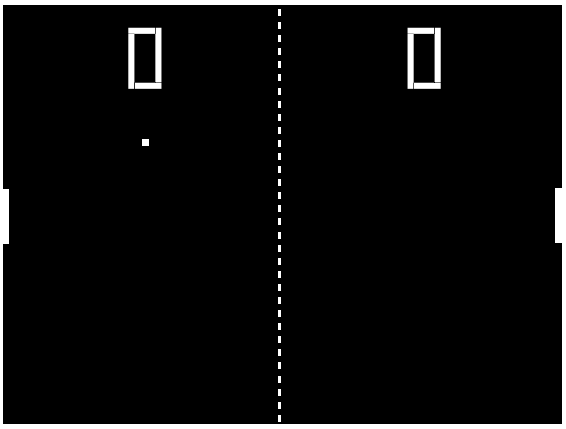

Figura 4: Janela do jogo

As dimensões de cada elemento da janela segue os valores apresentados na tabela 4, para o tracejado central esse é o valor de cada traço:

	H (*Pixels*)	V (Linhas)
Janela	640	480
Paddle	8	63
Bola	8	8
Tracejado central	4	6

Tabela 4: Dimensão de cada objeto, em que H é o tamanho horizontal e V é o tamanho vertical.

Como já apresentado, a bola possui trajetórias diferentes dependendo de onde ela atinge o *paddle*, quanto mais afastado do centro maior será o ângulo entre a normal e a trajetória refletida, na mesma direção de crescimento do *paddle*, em que no centro esse valor é igual a zero, como é possível ver na figura 5:

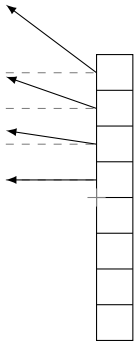

Figura 5: Colisões no *paddle* (as setas funcionam de forma simétrica)

Referências