UNIVERSIDADE DE BRASÍLIA
INSTITUTO DE CIÊNCIAS EXATAS
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO
116394 ORGANIZAÇÃO E ARQUITETURA DE COMPUTADORES
TURMA: C

Trabalho 3: JOGO DA VIDA em Assembly RISC-V

João Pedro de Oliveira Silva 190057807

1. Objetivos e Configurações do RARS

O trabalho tem como objetivo a prática da programação em assembly do RISC-V utilizando os recursos do simulador RARS. O trabalho consiste em desenvolver uma implementação do Jogo da Vida, exibindo o resultado de sua evolução através da interface gráfica do RARS, com o auxílio da ferramenta de exibição de saída gráfica mapeada em memória.

A configuração da memória é compacta com o text adress iniciando no zero.

O Bitmap Display por sua vez, possui as unidades de largura e altura no valor 8, o display da largura e da altura dos pixels se encontra em 128, e o endereço base para o display é o 0x00003000 (heap).

A matriz de dados 1, tem as células iniciais, a matriz 2 carregará as gerações seguintes.

2. Descrição geral do programa

```
while(1){
        Carrega a matriz 1 no Display
        Chama a função que gera os novos pixels para a matriz seguinte
        Copia a matriz 2 para a matriz 1
        espera 50ms
}
return 0;
```

3. Funções implementadas

3.1. readm

readm(i, j, mat) - onde a0 = i (linhas), a1 = j (colunas) e a2 = endereço inicial da matriz: retorna o valor da célula da matriz em a0.
Verifica se as coordenadas são válidas, caso sejam, percorre a matriz, byte a byte, até encontrar as coordenadas desejadas.
Chegando no ponto desejado, salva em a3 o valor que foi encontrado lá.

3.2. plotm

plotm(mat) - onde a0 = endereço inicial da matriz: desenha a matriz na tela gráfica

Função que recebe como entrada apenas a cópia do endereço da matriz de bytes como entrada e não tem nada como saída pois ela apenas atualiza o display. Percorre a matriz de bytes, pixel a pixel, atualizando o display com os valores encontrados onde ele colore pixels cujo byte da matriz for “1” e descolore se for “0”. Ela repete este procedimento em todos os pixels da matriz.

3.3. geracion

Função principal do programa, a qual irá criar na mat2 a próxima geração de células. Percorre pixel a pixel a matriz de bytes, em seguida, chama-se a função morto_vivo, para saber se o pixel em questão irá viver ou morrer na próxima geração. Os valores das células na próxima geração serão salvos na mat2.

3.4. morto vivo

Recebe 4 parâmetros como entrada, sendo eles a5 (n° de linhas), a6 (n° de colunas) a4 (cópia da mat1). A saída é armazenada na variável a7 que determina se a célula presente nas coordenadas a5 e a6 vive ou morre na próxima geração (0 = morre, 1 = vive/nasce).

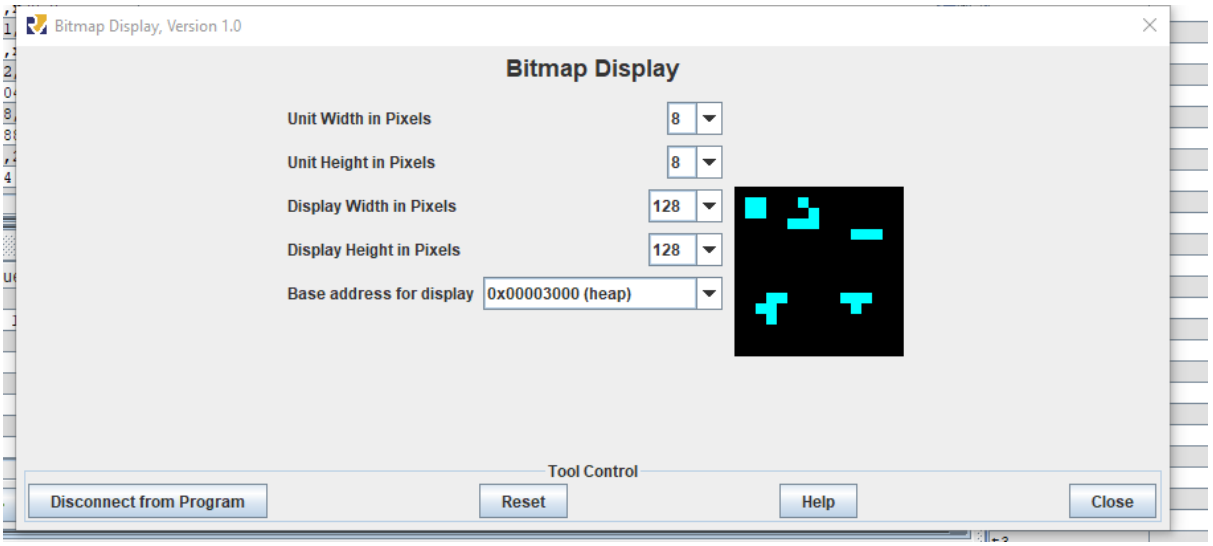
A implementação foi feita para chamar a função readm em cada um dos 8 vizinhos de uma célula e incrementa um contador sempre que um desses vizinhos estiver vivo, ou seja, têm o valor “1” armazenado nas suas coordenadas na mat1.

Finalmente verifica-se se a célula tem 2 ou 3 vizinhos para que neste caso ela continue viva e, caso tenha 3 vizinhos e não tenha célula na posição informada, na próxima geração, uma nova célula nascerá nessas coordenadas e a7 será “1”.

3.5. atualiza

Recebe como parâmetros os endereços de memória da primeira e segunda matrizes em a0 e a1, sem retorno. Essa apenas copia o conteúdo da mat2 para a mat1.

4. Conclusão



O código ocorreu como o esperado, uma vez que chegado ao fim, não apresenta novas atualizações no Bitmap Display.