Relatório do trabalho de Arquitetura de Computadores 1

João Santos nº 51966 | Diogo Matos nº 54466

27 de maio de 2023

Introdução

Neste trabalho pretendia-se desenvolver um programa em Assembly RISC-V para localizar personagens da saga Star Wars numa imagem. Dado um ficheiro com imagem no formato RGB, o programa gera uma nova imagem com a identificação da personagem escolhida pelo utilizador.

Data

Declarações:

imageRGB -> local onde está guardada imagem convertida para RGB;
imageFinalRGB -> local onde estará guardada a imagem final em RGB;
inicialBuffer -> buffer criado onde está a imagem em RGB (com tamanho 320*180*3 o que é o tamanho da imagem vezes 3 (bytes));
resultBuffer -> buffer criado onde estará a imagem final em Hue.

Funções

Iremos falar de cada função criada para este trabalho, das quais são:
- read_rgb_image;
- write_rgb_image;
- hue;
- indicator;
- location;
- main.

read_rgb_image

Descrição:

Lê um ficheiro com uma imagem no formato RGB para um array em memória. De realçar que esta função e a próxima mencionada foram feitos com auxilio da opção "help"do simulador.

Argumentos:

a0 -> String com o nome do ficheiro.

Retorna:

a0 -> Endereço de um buffer onde a imagem deverá ser escrita.

write_rgb_image

Descrição:

Cria um ficheiro novo com uma imagem no formato RGB.

Argumentos:

a0 -> Nome do ficheiro;
a1 -> Buffer da imagem;
a2 -> Comprimentro do buffer.

Retorna:

Não retorna nenhum valor.

hue

Descrição:

Calcula a componente Hue a partir das componentes R,G e B como pixel. Para o desenvolvimento desta mesma função utilizámos a informação disponibilizada no enunciado do trabalho onde explica como se calcula a componente Hue de um pixel (3 bytes), na função fazemos a verificação dos 3 bytes para saber que calculo devemos efetuar e retornamos o resultado do mesmo.

Argumentos:

a0 -> R;
a1 -> G;
a2 -> B.

Retorna:

a0 -> Retorna valor do Hue de um pixel.

indicator

Descrição:

Dado um personagem (e.g. 1,2,3) e um pixel com componentes, R, G, B, indica se esse pixel pertence ou não à personagem. Para o desenvolvimento assumimos que o Yoda é personagem 1, o Darth Maul é o 2 e o Boba Fett é o 3, escolhendo a personagem que queremos a função irá pegar no determinado pixel, vai utilizar a função Hue para calcular a componente Hue desse e vai determinar se pertence ou não à personagem pretendida.

Argumentos:

a0 -> R;
a1 -> G;
a2 -> B;
a3 -> Personagem pretendida.

Retorna:

a0 -> Retorna 1 se pertencer à personagem pretendida e 0 se não pertencer.

location

Descrição:

A função "location"calcula o centro de massa para um determinado personagem em uma imagem RGB. A função recebe dois argumentos: o personagem desejado (a3) e um buffer que contém a imagem RGB original (a4).

O centro de massa é calculado através da verificação dos pixels da imagem. A função utiliza dois contadores, s1 e s3, para controlar as coordenadas x e y, respectivamente. Também são utilizados os registos s4 e s5 para acumular a soma das coordenadas x e y multiplicadas pelo indicador de cada pixel.

A função possui um loop externo que itera sobre o eixo y, enquanto s3 for menor que 180. Dentro desse loop, há um loop interno que itera sobre o eixo x, enquanto o s2 for menor que 960 (320 * 3). Em cada iteração, o indicador do pixel é calculado através da chamada da função "indicator"e é armazenado em a0.

Em seguida, s1 é atualizado ao somar o valor de a0, que representa o resultado do indicador. As coordenadas x e y são multiplicadas pelo indicador que por sua vez são armazenadas em a1 e a2, respectivamente. Os valores de a1 e a2 são somados aos acumuladores s4 (cx) e s5 (cy), que representam a soma das coordenadas x e y multiplicadas pelo indicador, respectivamente. O contador s2 (x) é incrementado em 1 a cada iteração.

Quando o loop interno é concluído, o contador s3 (y) é incrementado com 1 e o programa retorna ao início do loop externo, onde continua a iteração até que s3 seja igual a 180.

Após os loops terminarem, os valores finais do centro de massa são calculados onde é dividido os acumuladores s4 e s5 pelos valores do contador s1. O resultado para o centro de massa em x é armazenado em a0 e o resultado para o centro de massa em y é armazenado em a1.

Por fim, os registos e a pilha são recuperados para os seus valores originais e a função retorna, onde fornece o valor do centro de massa para o personagem pretendido.

Argumentos:

a3 -> personagem pretendida;
a4 -> Buffer com a imagem RGB original.

Retorna:

Retorna o valor do centro de massa para certo personagem.

cross

Descrição:

A função "Cross"é responsável por desenhar uma cruz no buffer inicial de uma imagem RGB. A função recebe vários argumentos, incluindo as coordenadas Cx e Cy, que determinam a posição da personagem pretendida, o identificador da personagem (a3) e o buffer com a imagem RGB original (a4).

A função começa por alocar espaço na pilha e salva alguns registos importantes. Em seguida, são definidos alguns valores para os identificadores das personagens (t1, t2, t3) e outras constantes (t5, t6).

Os próximos passos da função envolvem cálculos para determinar a posição da cruz no buffer. As coordenadas Cx e Cy são multiplicadas por valores pré-definidos (t0, t5) e subtraídas de outros valores (t4, s3) para obter a posição correta da cruz no buffer.

A função passa então para uma verificação condicional usando a instrução "beq"para verificar qual é o identificador da personagem pretendida (a3). Dependendo do valor de a3, o fluxo do programa é redirecionado para diferentes seções de código que desenham a cruz correspondente à personagem.

Cada seção de código começa por definir os valores das componentes RGB para a cruz. Em seguida, há um procedimento repetitivo para desenhar a cruz no buffer.

No final de cada seção do código, há um trecho que lida com a movimentação para a próxima parte da cruz. Finalmente, a função retorna o buffer final com a identificação da personagem pretendida.

Argumentos:

a0 -> Cx;
a1 -> Cy;
a3 -> Personagem pretendida;
a4 -> buffer com a imagem RGB original.

Retorna:

Retorna o buffer final com a identificação da personagem pretendida.

main

Descrição:

A função main solicita ao utilizador o número da personagem pretendida, lê uma imagem RGB correspondente, realiza operações com base na personagem escolhida onde escreve a imagem final. A função chama as funções read_rgb_image para ler a imagem, location para determinar a localização da personagem na imagem, cross para fazer um cruzamento com outros elementos e write_rgb_image para escrever a imagem final. Em seguida, ela recupera os valores originais da pilha.

Argumentos:

Sem argumentos.

Retorna:

Sem Valor de retorno.

Output

(a) Yoda (b) Maul (c) Boba

Figura 1: Output de cada personagem

Instruções para execução:

Para testar o programa deve inicialmente correr no terminal o comando "convert starwars.png starwars.rgb"para converter a imagem PNG para RGB. Posteriormente deve correr o código no simulador Rars escolhendo também qual o personagem que quer identificar. Depois do programa terminar a execução deve correr no terminal "convert -size 320x180 -depth 8 result.rgb result.png"para converter RGB para PNG. No final já terá a imagem final no formato PNG para ser visualizada.

Conclusão:

Com este trabalho evoluimos o nosso conhecimento e capacidade em implementar em Assembly Risc-V. Não se obteve um resultado totalmente perfeito não sendo o output esperado, ainda houve vários testes e tentativas para se obter o resultado esperado, como por exemplo a alteração de loops e alteração da função "cross", alterando o valor do registo t5 para 2880 e aí sim o resultado já seria o esperado.