UNIVERSIDADE DE ÉVORA

CURSO DE ENGENHARIA INFORMÁTICA
1º Ano

RELATÓRIO DE TRABALHO COMPUTACIONAL

Arquitetura de Computadores I

Prof. Miguel Barão

Trabalho Realizado Por:
Henrique Rosa - 51923
Rafaela Abade - 52246

Índice

1 Introdução

Este relatório consiste na explicação da organização do nosso trabalho. À semelhança com a calculadora feita na linguagem C, este programa terá que ler *strings* e analisá-las caractere a caractere, de modo a operar segundo as instruções do utilizador.

2 Funções

2.1 Secção de dados (.data)

buffer_image_rgb: contém um "espaço"em memória de 786432 (512x512x3) bytes (o número de bytes necessário para guardar uma imagem de 512x512 pixels em formato rgb);
buffer_image_gray: análogo ao *buffer_image_rgb*, tem menos espaço pois as imagens em formato gray só têm um byte por pixel;
rgb_path: contém o diretório da imagem rgb;
gray_path: contém o diretório da imagem gray (como é suposto o código criar a imagem gray, o directório serve como local para a criação);
sobel_h_path: contém o diretório da imagem de gray depois da convolução com o operador de sobel horizontal (análogo ao *gray_path*);
sobebl_v_path: o mesmo, mas para com o operador de sobel vertical;
final_path: análogo ao *gray_path*, mas para a imagem final;
sobel_h: array de bytes que contem os valores (por ordem) do operador de sobel horizontal;
sobel_v: análogo a *sobel_h*, mas para o operador de sobel vertical;
buffer_sobel_hbuffer_sobel_hbuffer_final: estes buffers são todos análogos ao *buffer_image_gray*;

2.2 *read_rgb_image*

Parametros: a0 - contém o endereço do caminho do ficheiro rgb, que vai ser lido.
Retorno: a0 - endereço do buffer da imagem rgb, agora com o conteudo do ficheiro.
Funcionamento: é guardado o valor do registo s5 na stack para ser reposto no fim. de seguida, guardamos o valor do resgisto a1 pois este vai sser necessario para a leitura do ficheiro, e guardamos no registo a7 o código sys_ call respectivo à abertura de ficheiros, e é guardado no registo a1 o valor zero, pois a flag de leitura é o 0 (análogamente, se quisessemos escrever, teriamos de guardar o valor 1 no registo a1).
É feita uma ecall e, como temos o código certo no registo a7, o ficheiro rgb é aberto, e de seguida repomos o valor de a1, pois este contem o buffer da imagem no qual vão ficar guardados os valores. O restante do código da função é análogo, a mudança do código guardado em a7 vai permitir a leitura e fecho do ficheiro. Na linha 34, é guardado o tamanho do ficheiro no registo a2, pois a função de leitura a partir de um ficheiro requer o tamanho deste. No final, repomos o valor de s5 e incrementamos o stack pointer, e retornamos.

2.3 *rgb_to_gray*

Parametros: a0 - contém o endereço do buffer no qual está a imagem rg que vamos converter para gray. a1 - contém o endereço do buffer da imagem gray.
Retorno: não retorna nada, apenas carrega os valores nos buffers.
Funcionamento: No início são guardados os valores de s0, s1, s2 e s4 na stack para serem repostos no fim, e são guardados nesses registos os valores que vão ser usados na operação da função, incluindo um divisor que, no caso, serve como "work arround" para a inexistencia de numeros fracionarios. É feita a operação como dita a fórmula no enunciado

2.4 right gray image

2.5 convolution

2.6 contour

2.7 main

3 Conclusão