Universidade de Évora

Arquitetura e Sistemas de Computadores I

Ano Letivo 2016/2017

Trabalho Final ASC I

Trabalho realizado por:

Diogo Rafael nº37859

Vasco Crespo nº37913

26/5/2017

Introdução

Neste trabalho é pretendido converter uma imagem do formato RGB para GRAY com tons de cinzento, com fundo branco e com traços escuros onde haveria contornos através de um conjunto de funções em Assembly Mips.

Neste trabalho em particular só foi feita a parte em que a imagem RGB é convertida para GRAY e fica em preto e branco, para tal foram utilizadas as instruções sobre ficheiros e sobre como converter pixéis com cor para preto e branco dadas no enunciado do trabalho em várias funções.

Procedimento

.data e .text

Primeiro foi utilizado o segmento **.data** para definir todos os buffers que são necessários nas funções e as diretorias dos ficheiros utilizados e também uma mensagem de erro que será utilizada mais tarde, logo de seguida é utilizado o segmento **.text** para começar a escrever o código necessário para resolver o problema em questão.

Função *main*

Esta é a função principal do programa onde são efetuados vários **jal** para aceder às outras funções do programa, e finalmente um **j** para a função final ***END***.

Função *readRGB*

Nesta função são utilizadas as instruções fornecidas no enunciado para abrir e ler o ficheiro **lena.rgb** , é então realizado um **jr** de volta à função ***main*** (com a adição de uma pequena instrução **beq** caso algo corra mal, ou seja, caso o registo **$v0** seja igual a -1 irá ocorrer um branch para uma mensagem de erro).

Função *Conversion*

Após o **jal** realizado para esta função é guardado o endereço de **$ra** no primeiro byte de **$sp** , depois são carregadas as posições iniciais dos buffers nos registos **$t0** e **$t5** e em **$t6** o equivalente ao ultimo espaço do buffer, em seguida, a função entra num ***LOOP*** em que vai dar-se a conversão dos 3 bytes equivalentes a 1 pixel em rgb para 1 byte em gray(estes bytes são utilizados através da operação **lbu** no registo **$t0**), pixel a pixel, através de operações **mul**, **div** e **add**(multiplicando por 30 e dividindo por 10 no caso do vermelho, por exemplo, e adicionando todos os bytes uns aos outros, fazendo um pixel em gray) e são incrementados os buffers para poderem ser usados os bytes seguintes, quando o registo equivalente ao buffer de RGB que está a ser incrementado for igual ao registo **$t6** o loop para e dá-se um **jr** para a função ***main***.

Função *Write*

Mais uma vez, com as instruções dadas no enunciado foi criado um ficheiro e foram escritos nele todos os bytes presentes no BufferGRAY, criando um ficheiro *.gray* que está agora em preto e branco(e mais uma vez há um **beq** caso ocorra um erro), dá-se mais uma vez um **jr** para ***main***.

Função *ERROR*

Esta função encontra-se aqui caso seja encontrado um erro, tem como função "printar" a mensagem de erro "Erro".

Função *END*

Aqui são fechados ambos os ficheiros antes utilizados(o inicial e o criado) e dá-se o fim do programa.

Conclusão

Apesar de apenas ter sido realizado a primeira parte do trabalho, a conversão da imagem fornecida para preto e branco, podemos concluir que foi realizado com sucesso, não apresenta bugs visíveis e o único problema que encarámos foi devido a um problema no Windows.