UNIVERSIDADE DE ÉVORA

Sistemas Operativos I

2º Ano Engenharia Informática

Diagrama de 5 Estados

Trabalho elaborado por:

- Diogo Rafael Nº 37859
- Vasco Crespo Nº 37913
- João Galvão Nº 37814.

Ano Letivo 2017/2017

Introdução

Este trabalho tem como objetivo a simulação do diagrama de 5 Estados tal como o seu respetivo escalonamento.

Os estados do diagrama (NEW; READY; RUN; BLOCKED; EXIT) tem todos funções diferentes.

Estado NEW, é o estado responsável pela criação dos novos processos.

Estado READY, é o estado que possui os processos que estão á espera de ir para o estado seguinte, o RUN.

Estado RUN, é aquele que corresponde ao CPU, e que vai correr os processos um a um.

Estado BLOCKED, que será um estado em que os processos irão aguardar a chegada de uma informação, seja esta a razão do destino dos processos.

Estado EXIT, é para onde os processos vão após acabarem de correr.

Desenvolvimento

1ª Parte

Nesta primeira parte do trabalho, tivemos de criar funções e estruturas que eram necessária. Inicialmente tivemos de criar uma função para ler o ficheiro de input. Esse ficheiro contém um primeiro valor que corresponde ao tempo de chegada do processo ao estado New, sendo que os outros valores, correspondendo às instruções que vão ser feitas ao processo, serão agrupados 2 a 2. Criamos um escalonamento RR com Quantum de 4, e definimos o tamanho da memória para 300 instruções.

- Função readFile():

Esta função vai ser responsável por adicionar cada linha do ficheiro a um array temporário.

- Função arraycreate():

Esta Função vai ser responsável por separar as instruções que se encontram dentro do array, já prontas para serem colocadas na memória.

- PCB:

O PCB vai descrever os processos, ou seja, que contem o id, o tempo inicial, a posição na memória, o PC e o tamanho.

- Queues:

As queues vão retratar os estados do nosso modelo (NEW; READY; BLOCKED; RUN; EXIT). Depois de as criarmos tivemos de criar funções das queues para podermos trabalhar com as mesmas.

- Função addNew():

Esta função pega nos processos e coloca-os na queue do New, consoante o seu tempo de chegada.

- Função addREADY()

Esta função realiza o dequeue do quele referente ao estado NEW e realiza o enqueue da queue referente ao estado READY, caso nenhum do processo esteja em espera no estado BLOCKED, pois os processos do estado BLOCKED tem prioridade em relação ao estado NEW

- Função addToMemory():

A função, dependendo do processo que entre no estado NEW, vai colocar o processo em memória, criando as variáveis e colocando as operações do processo em memória.

- Função ProcessStart():

A função ProcessStart através das funções previamente construídas, vai receber os processos e coloca-os nos seus devidos estados. Quando os processos se encontram no estado Run, vão ser executadas as operações referentes ás instruções. O ProcessStart vai parar o seu ciclo assim que todos os processos estiverem no estado EXIT

2ª Parte

Nesta segunda parte do trabalho introduzimos a nova instrução, a instrução FORK como instrução número 6, para além da nova instrução foram introduzidos 2 dos 3 gestores de memória pedidos o BEST FIT e o WORST FIT.

Para os gestores de memória utilizamos 4 funções.

- Função addToMemory():

Esta função vai verificar qual o gestor de memória a utilizar.

- Função addToMemoryBestFit():

Utiliza o método do BEST FIT para fazer a implementação na memória.

- Função addToMemoryWorstFit():

Utiliza o método do WORST FIT para fazer a implementação na memória.

- Função addToMemorypart2():

Vai implementar na memória utilizando um dos métodos anteriores o BEST FIT ou o WORST FIT

Conclusão

O trabalho não foi terminado totalmente com sucesso, pois alguns erros aconteceram já muito em cima do prazo de entrega, e como tivemos alguns problemas com o entendimento da linguagem utilizada não conseguimos resolver com facilidade os erros anteriores.

Conseguimos concluir teoricamente o objetivo do trabalho que era o funcionalidade da simulação do modelo de 5 estados, mesmo o trabalho não estando funcional percebemos o objetivo final.