1ª Parte

Suponha uma arquitectura sobre o modelo de 5 estados que consome programas constituídos por operações definidas por dois dígitos, exemplo de programa:

2 01 11 11 11 21 71 91

O primeiro valor, "2" é um valor que indica o instante em que o processo é inserido no estado NEW. Esse primeiro valor é descartado quando o programa é escrito em memória.

Escalonamento Round Robin, Quantum 4 mas configurável (#define). O estado ready só permite no máximo 4 processos (#define). Limitação esta aplicável apenas a processos vindos do estado NEW. Quando no mesmo instante, um processo vindo do NEW e do BLOCKED querem entrar no READY, o vindo do BLOCKED tem prioridade.

Nas restantes operações do programa, o primeiro dígito indica a instrução e o segundo indica a variável onde se aplica essa operação. Assim existem 10 operações diferentes e 10 variáveis disponíveis por programa.

O programa é guardado em memória, ficando o PCB apenas com a indicação da localização do programa e com o PC (Program Counter) do respectivo programa.

Tabela das instruções:

Códigos	Instruções	Significado
0X	ZERO X	X = 0
1X	ADD X	X ++
2X	SUB X	X --
3X	IF X	If X ==0, goto next line (PC++); else goto next line +1 (PC += 2)
4Y	BACK Y	Goto PC -= Y, em que Y é logo o valor do salto, não se vai consultar o valor da variável
5Y	FORW Y	Goto PC += Y, em que Y é logo o valor do salto, não se vai consultar o valor da variável
6X	FORK X	X = Fork() Apenas para implementar na 2ª parte do trabalho
7X	DISK SAVE X	Wait…
8X	COPY X	X0 = X
9X	EXIT	Exit

- O desenvolvimento das instruções deve modular e distribuído pelos elementos do grupo de trabalho.
- Todas as instruções consomem um ciclo temporal
- As instruções de 0 a 5 e a 8 mantêm o processo em estado RUN
- A instrução 6 faz um fork duplicando o processo que irá para o estado NEW (o novo processo, o processo original continua no RUN caso ainda não tenha esgotado o seu Quantum), **esta função é para implementar apenas na 2ª parte do trabalho.**
- A instrução 7 simula uma escrita em disco envia o processo para estado BLOCK e consome 3 ciclos temporais em espera.

- A instrução 9 passa o processo para o estado EXIT

A memória onde as instruções são colocadas é estática e é representada por um array MEM[] que têm um limite de **300 posições**. (Este limite deve estar "#define" no vosso programa para permitir futuros ajustes).

Tenha em conta que qualquer programa em memória reserva sempre espaço para as 10 variáveis, mesmo que não as use todas.

O programa acima referido seria representado no array da memória MEM[] da seguinte forma:

	10 Variáveis reservadas					Instruções do Programa							
…	V0	V1	V2	…	V9	01	11	11	11	21	71	91	…

Input

No Moodle encontra-se o ficheiro com o input de testes. Esse ficheiro contém 8 linhas, sendo cada linha um processo.

A instrução fork apenas é evocada uma vez na última instrução, **como é para implementar apenas na 2ª parte**, para a ignorar basta remover do último processo a operação 61.

Output

Deve haver 2 modos de output que são escritos em ficheiro devidamente formatado (*scheduler_simples.out e scheduler_complexo.out*).

- Ficheiro *scheduler_simples.out* com o seguinte formato:

  INSTANTE | ESTADO_P1 | ESTADO_P2 | . . . | ESTADO_Pn

  Exemplo:

  . . .
  9 | exit | exit | ready | block | run | block | block | new
  10 | exit | exit | run | ready | exit | block | block | ready
  11 | exit | exit | ready | run | exit | block | block | ready | new
  . . .

- Ficheiro *scheduler_complexo.out* adiciona ao output *scheduler_simples.out* o print da memória em todas as linhas/instantes

Entrega

No Moodle e devera ser um .zip com os números de aluno no nome do ficheiro, ex "l23455_l33212_l4444.zip" e deverá conter o código fonte do trabalho assim como um relatório em PDF.

Trabalho pode ser individual ou em grupo até 3 pessoas no máximo.

Data de Entrega:

20 de Maio às 23h59

Discussões:

Nas aulas práticas da última semana de aulas (dia 21 e 23 de Maio)