**Sistemas Operativos**

1ª Série de Exercícios



Docente:

Prof. João Pedro Patriarca

Alunos:

Ricardo Cacheira, Nº31743

Rafael Pereira, Nº40680

1. **Considere uma MMU com espaço de endereçamento virtual de 256 GiBytes e 3 níveis de tabelas de páginas, com a dimensão máxima de uma página e com entradas de 4 bytes. Responda às questões justificando devidamente as suas respostas.**
2. **Qual o tamanho mínimo da página para suportar esta arquitetura? Considerando o espaço de endereçamento virtual igual ao espaço de endereçamento físico (RAM), quantos bits existem em cada PTE para especificar o endereço físico?**

Sabendo que o sizeof(PTE) = 4 (22) bytes, e que 256 GiBytes é 238 bytes, tendo páginas de um 1k byte chegamos a conclusão que seria necessários 4 níveis de tabelas de página, pois sobram 4 bits que teriam de ser indexados à 4 tabela.

|  |  |  |  |
| --- | --- | --- | --- |
| 8 | 8 | 8 | 10 |

10+8+8+8=34 bits -> 38-34 = 4 bits.

Por sua vez com páginas de 2k ficaríamos com 11 bits de offset e 9 bits divididos pelos 3 níveis ficando então a arquitetura com o tamanho mínimo da pagina para suportar a arquitetura.

|  |  |  |  |
| --- | --- | --- | --- |
| 9 | 9 | 9 | 11 |

11+9+9+9 = 38 bits.

Sendo o espaço de end. Virtual igual ao físico para sabermos o numero de bits da PTE que especificam o espaço de end. Físico apenas temos que subtrair o espaço de end. virtual com o offset: 38-11 = 27 bits.

1. **Considerando que são utilizados 6 bits para flags em cada PTE, indique quais as dimensões máximas de espaço de endereçamento virtual e físico mantendo a dimensão de página indicada na alínea anterior e o número de níveis indicado no enunciado. Indique as consequências de se introduzir um bit de controlo adicional em cada PTE.**

O espaço de endereçamento virtual máximo é o dado pelo enunciado visto que estamos a usar todos os 9 bits por nível, se por exemplo a tabela de 1ºnivel fosse menor com sete bits poderíamos usar mais 2 bits para endereçamento virtual.

2^38 = 256GB

O espaço de endereçamento físico máximo é dado pela subtração dos bits de endereçamento virtual com offset e a soma com os bits de flag da PTE dando 2^33 bytes que equivale a 8 GBytes.

1. **Comente a seguinte afirmação: “na arquitetura apresentada, cada acesso à memória, requerido por uma instrução de um programa em execução, implica sempre 4 acessos físicos, uma vez que existem 3 níveis de tabelas de páginas”.**

Confirma-se a veracidade da afirmação, visto que por cada acesso a memória, irá ocorrer um acesso a memória física por cada nível de tabelas de páginas, ou seja, 3 acessos a memória física. Depois dos 3 acessos irá ocorrer um ultimo que é o acesso à memória principal. Isto irá dar os quatro acessos a memória física.

1. **Considerando uma política de carregamento em que inicialmente apenas é carregada a página que contém o entry point de código e é alocada uma page frame para stack, faça um diagrama com o estado inicial da estrutura de páginas. Admita que o entry point corresponde ao endereço virtual 0x486140 e o stack pointer é iniciado com o valor 0x1fffffffff. Admita também que as page frames alocadas correspondem, respectivamente, aos endereços físicos 0x40000 e 0xC0000.**

|  |  |  |  |
| --- | --- | --- | --- |
| **9** | **9** | **9** | **11** |

Para primeiro endereço virtual 0x486140 temos:

|  |  |  |  |
| --- | --- | --- | --- |
| 0000 | 00000 0100 | 1000 0110 0 | 001 0100 0000 |

1. **Na gestão de memória dos sistemas operativos da família Windows, indique:**
2. **A motivação para a existência das listas standby e modified.**

A lista *Standby*, contem as páginas retiradas ao *working Set* de um processo, elas continuam na memoria principal até ser preciso uma página livre ou ela ser requisitada de novo (através de um *soft page fault*). Ela essencialmente é uma cache, pois é mais vantajoso recuperar uma página da memoria principal do que fazer uma leitura ao disco.

A lista *modified* difere da *Standby* pois antes de serem reutilizadas ou de o processo terminar são guardadas em memoria secundária.

1. **Porque razão, de uma forma geral, as listas Free e Zero indicam um valor de memória disponível consideravelmente inferior ao valor total de memória disponível no sistema?**

A razão pela qual indicam um valor de memoria inferior é devido a natureza destas listas pois contem paginas com zeros (Zero list) ou paginas que não estão sendo partilhadas entre processos e não são necessárias.

1. **Indique, pelo menos, duas funções da Windows API que possam levar páginas libertadas de um Working Set diretamente para a lista de páginas Free. Justifique a sua resposta**. decommitedMemorythreshhold
2. **Faça um diagrama de estados que apresente os estados por que passa uma page frame no Windows. Considere como estados iniciais e finais do ciclo de vida aqueles em que a page frame se encontra completamente dissociada de qualquer espaço de endereçamento (free ou zeroed). Dê nomes às transições entre estados, descrevendo cenários ilustrativos da sua ocorrência.**
3. **CODE**
4. **Para cada versão do Windows as bibliotecas dinâmicas de sistema (kernel32.dll, user32.dll, psapi.dll, etc) foram construídas para serem mapeadas em endereços bem definidos, sem conflitos (interseções) entre as várias bibliotecas.**
5. **Qual a vantagem deste procedimento?**

A vantagem deste procedimento é que torna mais eficiente a resolução dos símbolos indefinidos no código cliente aquando o carregamento da biblioteca, a indirecção fornece a partilha a secção .text.

1. **No caso de ocorrerem intersecções entre duas dll´s, que não de sistema, necessárias a determinado processo, como resolve o loader essa questão?**

O *loader* para resolver a questão da intersecção entre 2 *dll’s*, que não do sistema utiliza o processo de relocalização de *dll’s* o processo de rebase. A secção *reloc* contêm uma tabela de *RVA’s* (Relative Virtual Address) com referências para todos os endereçamentos relocalizados, isto tem penalidades a nível de tempo de carregamento e de recursos, recursos porque deixa de poder ser partilhada.

1. **CODE**