情報処理システム Ⅱ ・第 6 回

2025 年 5 月 16 日

配布資料の内容

6.1 仮想記憶システム

前回解説した OS による「アドレス空間の多重化」や「メモリ空間の拡大」を実現する仕組みを**仮想記憶システム**と呼びます。仮想記憶システムの実現方法にはいくつかありますが、ここでは最も一般的な**ページング方式**[1]と呼ばれている仮想記憶システムについて解説します。Windows や Linux など PC 向けのオペレーティングシステムカーネルの多くで、ページング方式の仮想記憶システムが採用されています。

ページ　ページング方式の仮想記憶システムでは、メモリ空間を、数 KiB から数 MiB 程度 (たとえば 4 KiB) の大きさの**ページ**と呼ばれる単位に分割して管理します。通常、ページの大きさは 2 の冪乗となるようにしますので、2 進表現されたアドレスの上位のビット列を調べるだけで、どのページに属するアドレスなのかが簡単に判別できます。たとえば、1 ページの大きさが 4 KiB で、48 bit の仮想アドレスと 32 bit の物理アドレスを使用している場合、$4096 = 2^{12}$ ですから、仮想アドレスの上位 36 bit や物理アドレスの上位 20 bit がページを識別するための番号 (ページ番号) となります。仮想アドレスでのページ番号を**仮想ページ番号**、物理アドレスでのページ番号を**物理ページ番号**と呼びます。仮想アドレスや物理アドレスの下位 12 bit は、それぞれ、その仮想ページや物理ページ内で、ページの先頭からアドレスがどれだけずれているかを表すことになりますが、これを**オフセット**と呼びます。

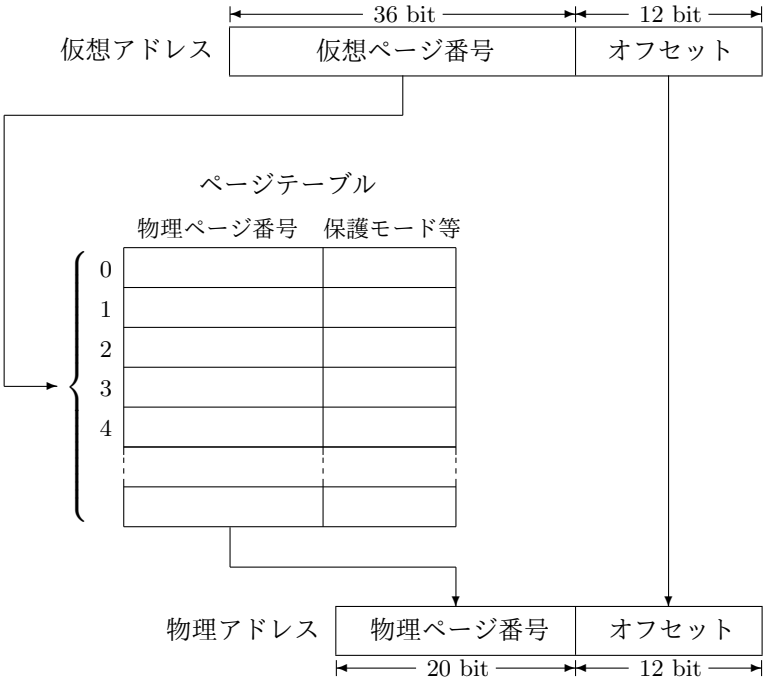

図 1: ページング方式によるアドレス変換

[1] **デマンドページング方式**とも呼ばれます。

ページテーブル　CPU がある仮想アドレスにアクセスしようとする際には、その仮想アドレスから仮想ページ番号を抽出し、**ページテーブル**と呼ばれる表を引きます (図 1)。ページテーブルには、仮想ページ番号ごとに、そのページの対応先となっている物理ページ番号が記録されていますので、このとき得られた物理ページ番号と仮想アドレスの下位 12 bit (オフセット) を図 1 のようにつなぎ合わせることで、CPU は物理アドレスを得ることができます。

CPU が ページング方式のアドレス変換で使用するページテーブルは、メモリ空間 (主記憶装置) 内に記憶されます。つまり、CPU がある仮想アドレスにアクセスする際には、ページテーブルを引くために、余計に主記憶装置にアクセスする必要が生じることになります。主記憶装置は CPU に比べるとかなり低速ですから、これは CPU の高速動作を阻害することになりますので、ページテーブルの (よく使われている) 一部は、CPU の内部に、そのコピー (一種のキャッシュメモリ) を置いておくことで高速化が図られるの普通です。このページテーブルに関する CPU 内のキャッシュメモリは TLB (Tanslation Lookaside Buffer) と呼ばれます。

メモ

ページフォルト　ページジング方式のアドレス変換は、CPU がすべてハードウェア的に行いますので、それ自体に、OS (カーネル) が直接関与することはありません[2]。OS の仕事は、ページテーブルを適切に設定しておくことです。ページテーブルには仮想ページ番号ごとにその対応先の物理ページ番号を記録しておきますが、対応先の物理ページが存在しないという情報を記録しておくこともできます。対応する物理ページが存在しない仮想ページに CPU がアクセスしようすると、アドレス変換が失敗し、ソフトウェア割り込み (例外) が発生します。これを**ページフォルト**と呼びます。ページフォルトが発生すると、あらかじめ OS (カーネル) が設定しておいた手続き (機械語プログラム) が実行されて、そこで OS は、ページテーブルの更新等、メモリ管理に必要な処理を行うことができます。

メモ

[2]CPU 内の TLB 内にアドレス変換に必要な情報がないとき、ソフトウェア割り込み (例外) が発生し、その処理をカーネルが行うことはあります。

OS によるメモリ空間の拡大　短い時間だけをとって見ると、各プロセスは、いくつかの特定の仮想ページ (に属するアドレス) にしかアクセスしていないのが普通です (メモリアクセスの空間的局所性)。この性質を利用して OS (カーネル) はメモリ空間の拡大を実現することができます。プロセスが最近頻繁にアクセスしている (一部の) 仮想ページだけに物理ページを割り当てておき、あまりアクセスされていない仮想ページの内容は、ハードディスクや SSD などの補助記憶装置に退避しておきます (図 2)。補助記憶装置の方に記憶されている仮想ページへのアクセスが起こると、適当な物理ページをその仮想ページに割り当て直してから、補助記憶装置に退避してあったデータをその物理ページに戻し、それからユーザプロセスの仮想ページへのアクセスを続行させます。

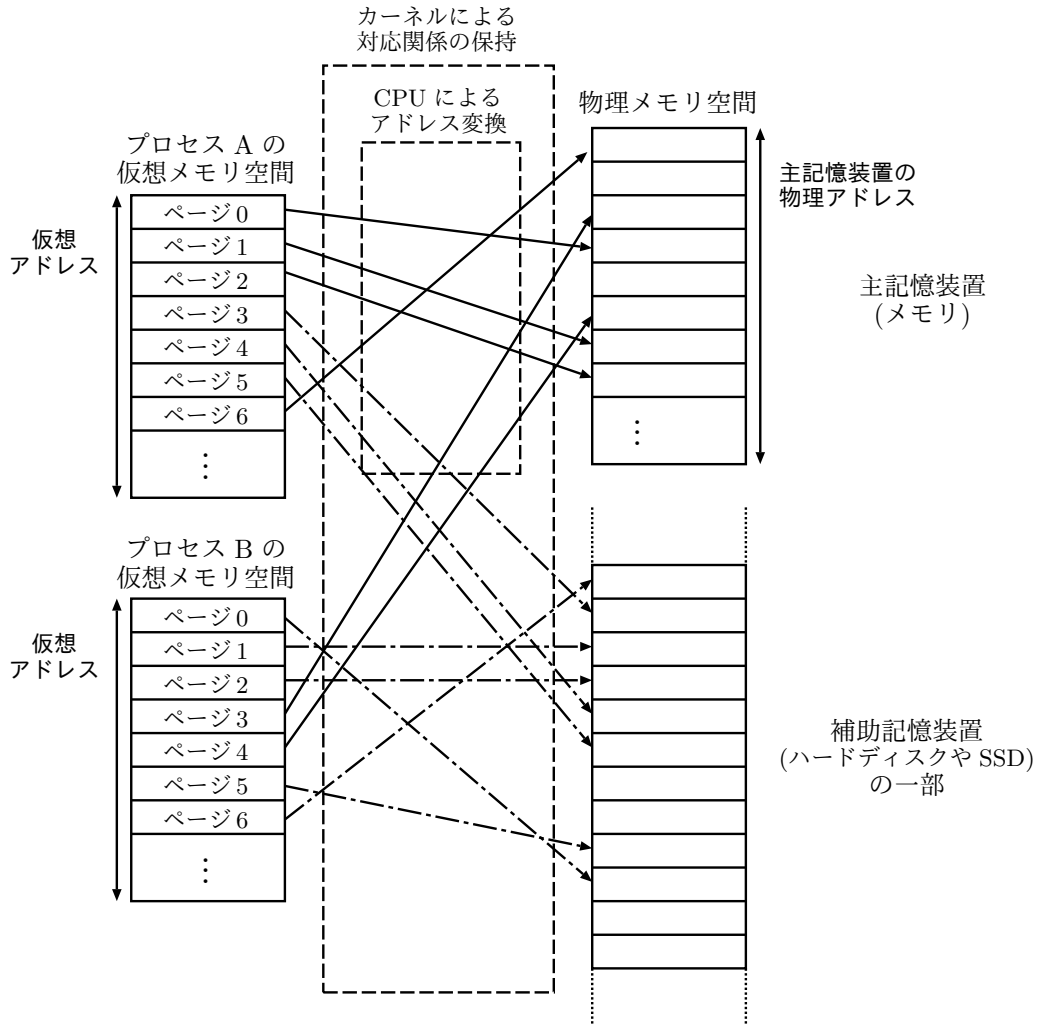

図 2: ページング方式による仮想記憶の仕組み

メモ

ページイン、ページアウト　物理的なメモリ (物理ページ) が割り当てられていない仮想アドレスに CPU がアクセスすると、ページフォルトが発生します。ページフォルトが発生すると、そのメモリアクセスはとりあえず棚上げされて、その時点の実行コンテクストが保存され、CPU は特権モードに移行して、カーネルプログラムによる例外処理が行われます。カーネルは、ページフォルトが発生したページアドレスを調べて、補助記憶装置に退避してあるそのページのデータを主記憶装置 (メモリ) の空きページにコピーし、その物理ページをページフォルトが発生した仮想ページに割り当てます。主記憶装置に空きページがない場合は、適当な方法で仮想ページを選択し、その内容を補助記憶装置に退避して空きページを作ってから同じことを行います。このようにして、ページフォルトが発生した仮想ページに無事主記憶装置の物理ページが割り当てられると、カーネルは、保存しておいた実行コンテクストを復帰して、棚上げされていたメモリアクセスを CPU に続行させます。

補助記憶装置に退避されていたページが主記憶装置のページに復帰することを**ページイン**、逆に、主記憶装置に割り当てられていたページが補助記憶装置に退避されることを**ページアウト**と呼びます[3]。

メモ

ページアウトするページの選択　新たな物理ページが必要となったにもかかわらず空いている物理ページがない場合には、ページアウトすべき仮想ページを選択する必要が出てきます。ページアウトされた仮想ページへのアクセスが近い将来に発生してしまうと、またページフォルトが発生して、ページアウトやページインが必要となってしまいますので、しばらくはアクセスされないであろうページを選択できると好都合です。ページアウトする仮想ページの選択方法には、以下のように様々なものがあります。

ランダム ページアウトするページをランダムに選択する。

FIFO (First In First Out) 最も遠い過去に物理ページが割り当てられた (ページインされた) 仮想ページを選択する。

クロック 基本的には FIFO でページアウトするが、過去一定時間 (たとえば 10ms) 内に使用されたページのページアウトは後回しにする。

NFU (Not Freqeuntly Used) ページが使われた回数が最も少ないものを優先してページア

[3]それぞれ、**スワップイン**、**スワップアウト**と呼ぶこともあります。

ウトする。実際には、最近使われた回数をより重要視するエージング法と呼ばれる方式が採用されます。

LRU (Least Recently Used) 最も遠い過去に使われたページ (つまり、最も長い間アクセスされていないページ) を選択する。

メモリ (ページ) アクセスの時間的局所性から、ランダムや FIFO よりもクロックが、また、クロックや (エージング法を取り入れた) NFU よりも LRU の方が、ページフォルトの発生が少なくなることが期待できますが、LRU では CPU のメモリアクセスの度に、ページのアクセス順の記録を更新しないといけないため、実際には、クロックを改良した方法やエージング法を取り入れた NFU などが採用されます。

メモ

スワップファイルとスワップ領域　ページアウトしたデータはハードディスクや SSD などの補助記憶装置へ退避されますが、この際、補助記憶装置上に構成されたファイルシステム中の特定のファイルが使用されることもあれば、補助記憶装置中の (ファイルシステムなどの用途には使用されていない) 専用の領域が使用されることもあります。前者を**スワップファイル** (swap file)、後者を**スワップ領域** (swap area) と呼びます[4]。

メモ

[4]Windows では通常スワップファイルが使用されます。Linux では、旧来スワップ領域を使用するのが標準的でしたが、最近ではスワップファイルを使用することも多くなっています。

スラッシング　ページインやページアウトが起こると、ハードディスクや SSD などの補助記憶装置へのアクセスが発生します。補助記憶装置 (特にハードディスク) は主記憶装置 (メモリ) に比べると非常に低速ですから、ページインやページアウトの処理には非常に時間が掛かってしまい、その間、ページフォルトを起したプロセスの実行は中断されてしまいます。その計算機に搭載されている主記憶装置 (メモリ) の容量に対して、実行中のプロセスが必要としているメモリの量が大きければ大きい程、また、各プロセスがアクセスするアドレスのばらつき具合が大きければ大きい程、ページインやページアウトの頻度が高まり、計算機システムが行わなければならない余計な仕事が増えて、その分、各プロセスが本来やるべき仕事が滞ることになります。ひどい場合は、計算機はページインやページアウトに伴う補助記憶装置の読み書きの処理に掛かり切りになり、本来の仕事はほとんど進行しないといった状態になります。このような状態は**スラッシング** (thrashing) と呼ばれます。

情報処理システム Ⅱ ・第 6 回・終わり